République Algerienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane Mira de Béjaïa

Faculté des Sciences Exactes

Département d'Informatique

Mémoire de Master Professionnel

Thème

Simulation du Routage dans les Réseaux de Capteurs Sans Fils

Présenté par :

TALEB Salim HOCINE Ahmed

Soutenu devant le jury composé de :

Présidente	Mme KHALED Hayette
Examinatrice	Melle CHELOUAH Lila
Directrice	Mme GHANEM Souhila
Co-Directrice	Melle TIAB Amal

Promotion 2015 – 2016

Remerciements

Nous tenons à remercier vivement Melle **TIAB Amal** et Mme **GHANEM Souhila** pour nous avoir honorés par leur encadrement, pour leur disponibilité, leurs orientations, leurs précieux conseils et leurs encouragements qui nous ont permis de mener à bien ce travail.

Nous tenons à remercier chacun des membres du jury pour l'intérêt qu'ils ont porté à notre mémoire en acceptant d'examiner notre travail et de l'enrichir par leurs propositions.

Un merci particulier à nos parents,nos familles et amis pour leur amour, leurs sacrifices et leurs patiences.

Nos remerciements s'étendent à tous nos enseignants et les membres du département d'informatique de l'université **ABDERRAHMANE MIRA**. Ainsi qu'à tous ceux et celles qui ont contribué de prés ou de loin à l'accomplissement de ce travail.

Dédicaces

A nos parents, pour leurs sacrifices déployés à notre égard,

pour leur patience, leur amour et leur confiance. Qu'ils trouvent dans ce modeste travail, le témoignage de notre profonde affection et de notre attachement indéfectible ; nulle dédicace ne puisse exprimer ce qu'on leur doit.

A nos frères et sœurs et tous nos amis pour chaque mot reçu,

chaque geste d'amitié, à chaque main tendue et pour toute attention témoignée.

Table des Matières

Table des figures

Liste des tableaux

Liste des Abréviations

AODV Ad-hoc On-demand Distance Vector

API Application Programming Interface

CBR Constant Bit Rate

CPU Central Processing Unit

CSMA Carrier Sense Multiple Access

DCSE Department of Computer Science and Engineering

EAR Eavesdrop And Register

ED Energie Disponible

EED End-to-End Deplay

EN Energie Nécessaire

FTP File Transfer Protocol

GAF Geographic Adaptive Fidelity

GEAR Geographical and Energy Aware Routing

GPS Global Positioning System

GPU Graphics Processing Unit

IDE Integrated Devlopment Environment

LEACH Low-Energy Adaptive Clustering Hierarchy

LLC Logical Link Control

MAC Media Access Control

NS Network Simulator

OSI Open Systems Interconnection

OTCL Object Tools Command Langage

QoS Quality of Service

RCSFs Réseaux de Capteurs Sans Fils

SAR Sequential Assignement Routing

SMACS Self-organizing Medium Access Control for Sensor networks

SMP Sensor Management Protocol

SPIN Sensor Protocols for Information via Negotiation

TADAP Task Assignement and Data Advertisement Protocol

TCL Tools Command Langage

TCP Transmission Control Protocol

TDMA Time Division Multiple Acess

UDP User Datagram Protocol

WSN Wireless Sensors Network

Introduction Générale

Aujourd'hui, les réseaux de capteurs sans fil sont de plus en plus populaires du fait de leur facilité de déploiement et de la multifonctionnalité des nœuds capteurs. Ces réseaux jouent un rôle crucial au sein des réseaux informatiques et ont été utilisés pour une variété d'applications telles que la santé, la poursuite de cibles et la surveillance de l'environnement.

En raison de la restriction de la portée de communication et de la forte densité des nœuds capteurs, le transfert de paquets dans les RCSFs est généralement réalisé par une transmission de données multi-sauts. Par conséquent, le routage dans les RCSFs a été considéré comme un domaine de recherche important au cours des dernières décennies.

Généralement, le routage permet d'établir une route de plus court chemin en terme de distance ou de délai entre deux nœuds source et destination. Le but du protocole de routage est de trouver la meilleure route selon les critères souhaités (délai, taux de perte, quantité de bande passante, ...).

Cette question a motivé la communauté scientifique pour développer des protocoles de routage plus appropriés aux RCSFs et qui offrent des gains de gestion tels que l'amélioration de la fiabilité de transmission des données, une tolérance aux fautes et un contrôle de congestion.

Cependant, il est très important d'étudier le comportement de ses protocoles et d'évaluer leurs performances avant leur déploiement sur les RCSFs, afin de comprendre et d'améliorer les éventuels problèmes qui pourraient affecter ces réseaux. En effet, il existe trois approches d'évaluation des performances : les modèles analytiques, les simulations et les mesures sur un

système réel.

Notre travail entre dans le cadre du projet de fin d'études et s'inscrit dans le contexte de la simulation du routage dans les RCSFs. Tout d'abord, nous avons élaboré un état de l'art des RCSFs et des techniques de routage existantes. Nous avons également parcouru et critiqué certains simulateurs présentés dans la littérature et ce qui nous a menés à nous concentrer sur un tout nouveau simulateur nommé CupCarbon. En effet, le simulateur CupCarbon fait partie du projet de recherche PERSEPTEUR supporté par l'Agence Nationale de Recherche ANR sous la référence ANR-14-CE24-0017-01. Son objectif étant de concevoir, visualiser, déboguer et valider des algorithmes distribués pour la surveillance et la collecte de données sur l'environnement, et de créer des scénarios environnementaux tels que les incendies, les gaz, les mobiles. Non seulement il peut aider visuellement à expliquer les concepts de base des RCSFs et leur fonctionnement ; mais il peut également aider les utilisateurs à tester leur topologies sans fil et leur protocoles, etc...

La suite de ce mémoire sera organisée comme suit :

- Le chapitre 1, présente un aperçu général sur les RCSFs : leurs architectures, leurs principales caractéristiques, leurs domaines d'application ainsi que les contraintes de conception d'un tel type de réseau.
- Le chapitre 2 est consacré à la problématique du routage dans les RCSFs. Les facteurs influents sur la conception des protocoles de routage, les différentes classes de ces protocoles.
- Le chapitre 3, dresse un état de l'art des simulateurs présentés dans la littérature. Nous présentons d'abord quelques simulateurs utilisés, en donnant les avantages et les inconvénients de chacun d'eux, ensuite nous effectuons une comparaison entre eux.
- Le chapitre 4, présente le simulateur CupCarbon, son architecture et son installation, l'implémentation d'un RCSF et enfin la simulation du protocole de routage LEACH.
- Enfin, notre travail s'achève par une conclusion générale résumant les grands points qui ont été abordés dans ce mémoire.

1

Généralités sur les RCSFs

Introduction

Les réseaux de capteurs sans fils ont été classés parmi les technologies les plus importantes du siècle. Ces réseaux sont constitués d'un grand nombre de micro-capteurs capables de récolter et de transmettre des données de manière autonome. Les micro-capteurs possèdent généralement de faibles capacités de calcul, de mémoire et d'énergie, ce qui a influencé une grande partie des recherches du domaine tel que le routage.

Dans ce chapitre, nous allons présenter les réseaux de capteurs sans fil : leurs architectures et leurs applications. Nous allons discuter également les principaux facteurs et contraintes qui influencent la conception des réseaux de capteurs sans fil. Par la suite, nous décrirons la problématique de la consommation d'énergie dans les réseaux de capteurs sans fil. Finalement, nous décrirons l'architecture de communication dans les réseaux de capteurs sans fil.

1.1 Définition d'un capteur sans fil

Un capteur sans fil est un petit dispositif électronique capable de mesurer une valeur physique environnementale et de la communiquer à un centre de contrôle via une station de base [1]. Ils sont capables de surveiller une grande variété de phénomènes ambiants notamment : la température, l'humidité, le mouvement des véhicules, la pression, le taux de bruits, la présence ou l'absence de certains types d'objets, et d'autres caractéristiques, tel que la vitesse, la direction et le volume d'un objet donné.

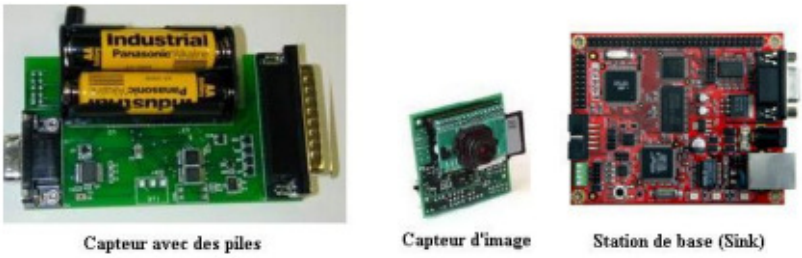

FIGURE 1.1 – Quelques capteurs existants sur le marché

1.2 Définition d'un réseau de capteurs sans fil

Un réseau de capteurs se définit comme un ensemble de capteurs connectés entre eux où chaque capteur étant muni d'un émetteur-récepteur [2]. Les réseaux de capteurs sans fil sont considérés comme un type spécial des réseaux Ad Hoc où l'infrastructure fixe de communication et l'administration centralisée sont absentes et les nœuds jouent, à la fois, le rôle des hôtes et des routeurs [3]. Chaque nœud est capable de surveiller son environnement et de réagir en cas de besoin en envoyant l'information collectée à un ou plusieurs points de collecte, à l'aide d'une connexion sans fil [1].

1.3 Architecture Des réseaux de capteurs sans fil [4]

Un nœud capteur est composé de trois unités principales :

- l'unité d'acquisition : l'unité d'acquisition est composée d'un capteur qui va obtenir des mesures numériques sur les paramètres environnementaux et d'un convertisseur Analogique/Numérique qui va convertir l'information relevée et la transmettre à l'unité de traitement.

- l'unité de traitement : l'unité de traitement est composée de deux interfaces, une interface pour l'unité d'acquisition et une interface pour l'unité de transmission. Cette unité est également composée d'un processeur et d'un système d'exploitation spécifique. Elle acquiert les informations en provenance de l'unité d'acquisition et les envoie à l'unité de transmission.

- l'unité de transmission : l'unité de transmission est responsable de toutes les émissions et réceptions de données via un support de communication radio.

Ces trois unités sont alimentées par une batterie comme la montre la figure ci-dessous :

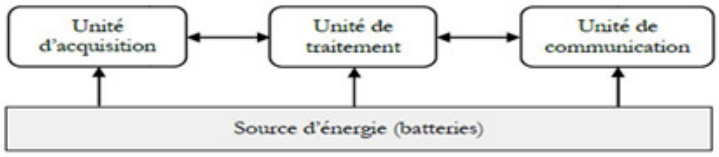

FIGURE 1.2 – Architecture d'un capteur sans fil. [5]

1.4 Caractéristique des réseaux de capteur sans fil [6]

Les principales caractéristiques des réseaux de capteurs sans fil se résument dans ce qui suit :

1.4.1 Système d'exploitation TinyOS

TinyOS est un système d'exploitation Open Source pour les réseaux des capteurs, conçu par l'université américaine de BERKELEY. Le caractère open source permet à ce système d'être régulièrement enrichie par une multitude d'utilisateurs. Sa conception a été entièrement réalisée en NesC, langage orienté composant syntaxiquement proche du C. Il respecte une architecture basée sur une association de composants, réduisant ainsi la taille du code nécessaire à sa mise en place. Cela s'inscrit dans le respect des contraintes de mémoires qu'observent les capteurs, pourvus de ressources très limités dues à leur miniaturisation.

Pour autant, la bibliothèque des composants de TinyOS est particulièrement complète, puisqu'on y retrouve des protocoles réseaux, des pilotes de capteurs, et des outils d'acquisition de données. Un programme s'exécutant sur TinyOS est constitué d'une sélection de composants systèmes et de composants développés spécifiquement pour l'application à laquelle il sera destiné (mesure de température, du taux d'humidité. . .).

TinyOS s'appuie sur un fonctionnement évènementiel, c'est à dire qu'il ne devient actif qu'à l'apparition de certains évènements, par exemple l'arrivée d'un message radio. Le reste du temps, le capteur se trouve en état de veille, garantissant une durée de vie maximale

connaissant les faibles ressources énergétiques des capteurs. Ce type de fonctionnement permet une meilleure adaptation à la nature aléatoire de la communication sans fil entre capteurs.

TinyOS est basé sur quatre grandes propriétés qui font que ce système d'exploitation, s'adapte particulièrement bien aux systèmes à faible ressources :

- **Evénementiel :** Le fonctionnement d'un système basé sur TinyOS s'appuie sur la gestion des évènements qui se déclenche. Ainsi, l'activation de tâches, leur interruption ou encore la mise en veille du capteur s'effectue à l'apparition d'évènements, ceux-ci ayant la plus forte priorité. Ce fonctionnement évènementiel (event driven) s'oppose au fonctionnement dit temporel (time driven), où les actions du système sont gérées par une horloge donnée.
- **Non 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.
- **Pas de 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. TinyOS se situe au-delà de ce second type, car il n'est pas prévu pour avoir un fonctionnement temps réel.
- **Consommation d'énergie :** TinyOS a été conçu pour réduire au maximum la consommation en énergie du capteur. Ainsi, lorsqu'aucune tâche n'est pas active, il se met automatiquement en veille.

1.4.2 Énergie de capteur sans fil

Les capteurs sans fils sont des éléments indépendants les uns des autres, comme leur nom l'indique. Par conséquent, ils doivent également disposer d'une alimentation autonome. Leur durée de vie est limitée par la durée de vie de leur batterie.

Cette contrainte forte a une influence majeure sur l'ensemble des techniques, mises en place pour le déploiement de tels réseaux. Un effet majeur de cette limitation énergétique est la limitation maximale des transmissions par voie hertzienne, très coûteuses. Il est donc primordial d'effectuer tant que possible le traitement de l'information localement au niveau du nœud. L'enjeu est donc d'étendre la durée de vie du système et sa robustesse, en cas de chute de certains nœuds seulement. Les problématiques sont donc très éloignées de celles des réseaux classiques, telle la maximisation du débit.

Dans les réseaux de capteur sans fils, il faut assurer une consommation répartie de l'énergie au sein du réseau. Cet énergie et consommé par les diverses fonctionnalités de réseaux qui sont donc par ordre décroissant de consommation d'énergie :

- Radio (Communication) .
- Protocoles (MAC, routage).
- CPU (calcul, agrégation).
- Acquisition.

1.5 Domaines d'application des réseaux de capteurs [6]

1.5.1 Orientées événements (event driven)

Comme exemple de ce type d'applications, nous pouvons citer la détection de feu de forêt ou la détection d'un tremblement de terre ou encore la surveillance de la qualité de l'air dans une région donnée. Dans ce type d'applications, Les nœuds capteurs doivent pouvoir détecter un événement qui pourrait arriver à tout moment dans n'importe quel endroit de la zone à surveiller. Ainsi, les nœuds capteurs doivent collecter les informations de façon continue et réagir immédiatement à des changements brusques des valeurs captées. Aucun contrôle périodique par le puits n'est exigé et le traitement du signal dans les nœuds capteurs est simple : chaque nœud capteur doit comparer la donnée mesurée à une valeur seuil donnée. Dès dépassement de cette valeur seuil, le nœud capteur envoie la donnée au puits. En outre, ce genre d'applications exige :

- Une forte densité des nœuds capteurs pour que la détection de l'Événement soit assurée

avec une bonne probabilité. Ce qui implique que le réseau doit avoir une couverture suffisante.

- Une bonne couverture de captage du nœud capteur qui dépend, généralement, du type d'événement à détecter.
- une localisation exacte de la position du nœud capteur dans laquelle l'événement est senti. Notons que des algorithmes de localisation distribués pourraient être utilisés à cette fin et ainsi, l'information peut être reçue par le puits avec géolocalisation.
- une bonne connectivité du réseau, liée à la couverture de transmission, afin que les protocoles de communication permettent à l'alarme envoyée par un nœud capteur d'arriver au puits avec une haute probabilité et dans le plus bref délai.

1.5.2 Orientées temps (time driven)

Les applications de ce type visent souvent à évaluer périodiquement un phénomène physique donné. Telle que l'application de surveillance de la pression atmosphérique d'une large zone ou les variations de température au sol dans un petit site volcanique. Ce sont des applications qui réagissent par rapport aux périodes. Les capteurs prennent un échantillon de l'environnement et le transmettent moyennant un protocole de communication approprié au puits final. Pour que les échantillons puissent être reçus par le puits avec une probabilité donnée et que l'évolution du phénomène observé soit suivie à la trace, il faut que la fréquence d'échantillonnage soit bien choisie, que la connectivité du réseau soit gardée sous contrôle et les protocoles de communication doivent être conçus pour que l'erreur d'évaluation de processus soit maintenue sous un seuil donné. Les nœuds capteurs utilisés dans ce type d'applications peuvent se mettre en veille pendant les périodes d'inactivité et n'enclenchent leur dispositif de capture qu'à des instants particuliers (contrainte d'économie d'énergie).

1.5.3 Applications orientées requêtes (query driven)

Dans ce cas, un capteur envoi de l'information uniquement suite à une demande explicite de la station de base. Cette classe d'application est destinée aux applications adaptées à l'utilisateur. Ce dernier peut requérir des informations à partir de certaines régions dans le réseau ou interroger les capteurs pour acquérir des mesures d'intérêts. Dans ce cas, des connaissances

sur la topologie du réseau et l'emplacement des capteurs sont nécessaires.

1.5.4 Hybrides

Ce type d'applications résulte des deux précédents types. Comme exemple, nous trouvons, l'application domotique. Où nous pouvons évaluer la température d'une chambre (orientée événements) ou/et surveiller un nouveau-né. (Time driven).

1.6 Exemples de domaine d'application des réseaux de capteurs

1.6.1 Application militaire

Comme pour de nombreuses autres technologies, le domaine militaire a été le moteur initial pour le développement des réseaux de capteurs. Le déploiement rapide, le coût réduit, l'auto-organisation et la tolérance aux pannes des réseaux de capteurs sont des caractéristiques qui font de ce type de réseaux un outil appréciable dans un tel domaine. Actuellement, les RCSFs peuvent être une partie intégrante dans le commandement, le contrôle, la communication, la surveillance, la reconnaissance, etc.

1.6.2 Application médical

Les réseaux de capteurs sont également largement répandus dans le domaine médical. Cette classe inclut des applications comme : fournir une interface d'aide pour les handicapés, collecter des informations physiologiques humaines de meilleure qualité, facilitant ainsi le diagnostic de certaines maladies, surveiller en permanence les malades et les médecins à l'intérieur de l'hôpital [7].

1.6.3 Applications agricoles

Dans le domaine de l'agriculture, les capteurs peuvent être utilisés pour réagir convenablement aux changements climatiques par exemple le processus d'irrigation lors de la détection de zones sèches dans un champ agricole.

1.6.4 Applications de sécurité

L'application des réseaux de capteurs dans le domaine de la sécurité peut diminuer considérablement les dépenses financières consacrées à la sécurisation des lieux et des êtres humains. Ainsi, l'intégration des capteurs dans de grandes structures telles que les ponts ou les bâtiments aidera à détecter les fissures et les altérations dans la structure suite à un séisme ou au vieillissement de la structure [3].

1.7 Facteurs et contraintes des RCSF [6]

La conception et la réalisation des réseaux de capteurs sans fil sont influencées par plusieurs paramètres. Ces facteurs servent comme directives pour le développement des algorithmes et protocoles utilisés dans les RCSFs.

1.7.1 Durée de vie du réseau

C'est l'intervalle de temps qui sépare l'instant de déploiement du réseau de l'instant où l'énergie du premier nœud s'épuise. Selon l'application, la durée de vie exigée pour un réseau peut varier entre quelques heures et plusieurs années.

1.7.2 Ressources limitées

En plus de l'énergie, les nœuds capteurs ont aussi une capacité de traitement et de mémoire limitée. En effet, les industriels veulent mettre en œuvre des capteurs simples, petits et peu coûteux qui peuvent être achetés en masse. De plus, les capteurs sont caractérisés par une bande passante limitée. Typiquement, le débit utilisé est de quelques dizaines de Kb/s. Le but de ce faible débit est de minimiser l'énergie consommée lors de transfert de données entre les nœuds. Un débit de transmission réduit n'est pas handicapant pour un réseau de capteurs où les fréquences de transmission ne sont pas importantes.

1.7.3 Facteur d'échelle

Le nombre de nœuds déployés pour une application peut atteindre des milliers. Dans ce cas, le réseau doit fonctionner avec des densités de capteurs très grandes. Un nombre aussi

important de nœuds engendre beaucoup de transmissions inter-nodales et nécessite que la station de base soit équipée de mémoire suffisante pour stocker les informations reçues.

1.7.4 Topologie dynamique

La topologie des réseaux de capteurs peut changer au cours du temps pour les raisons suivantes :

- Les nœuds capteurs peuvent être déployés dans des environnements hostiles (champ de bataille par exemple), la défaillance d'un nœud capteur est, donc très probable.
- Un nœud capteur peut devenir non opérationnel à cause de l'expiration de son énergie.
- Dans certaines applications, les nœuds capteurs et les stations de base sont mobiles.

1.7.5 Agrégation de données

Dans les RCSFs, les données produites par les nœuds capteurs voisins sont très corrélées spatialement et temporellement. Ceci peut engendrer la réception par la station de base d'informations redondantes. Réduire la quantité d'informations redondantes transmises par les capteurs permet de réduire la consommation d'énergie dans le réseau et ainsi d'améliorer sa durée de vie. L'une des techniques utilisée pour réduire la transmission d'informations redondantes est l'agrégation des données. Avec cette technique, les nœuds intermédiaires agrègent l'information reçue de plusieurs sources. Cette technique est connue aussi sous le nom de fusion de données.

1.8 Communication dans les réseaux de capteurs

Après le déploiement des nœuds capteurs sur une certaine zone de captage, ceux-ci commencent par la découverte de leurs voisins afin de construire la topologie de communication. Ainsi, ils deviennent capables d'accomplir les tâches qui leur sont affectées. Selon une communication multi-sauts, les capteurs sont chargés de collecter des données, les router vers un nœud particulier appelé nœud puits. Ce dernier analyse ces données et transmet à son tour l'information collectée à l'utilisateur via internet ou bien satellite. Comme l'indique la figure (2.3), l'ensemble de nœuds construisant le RCSFs est considéré comme étant un réseau

d'acquisition de données. Par contre, le réseau de distribution de données est composé des utilisateurs, et du réseau de communication : l'internet, et les satellites.

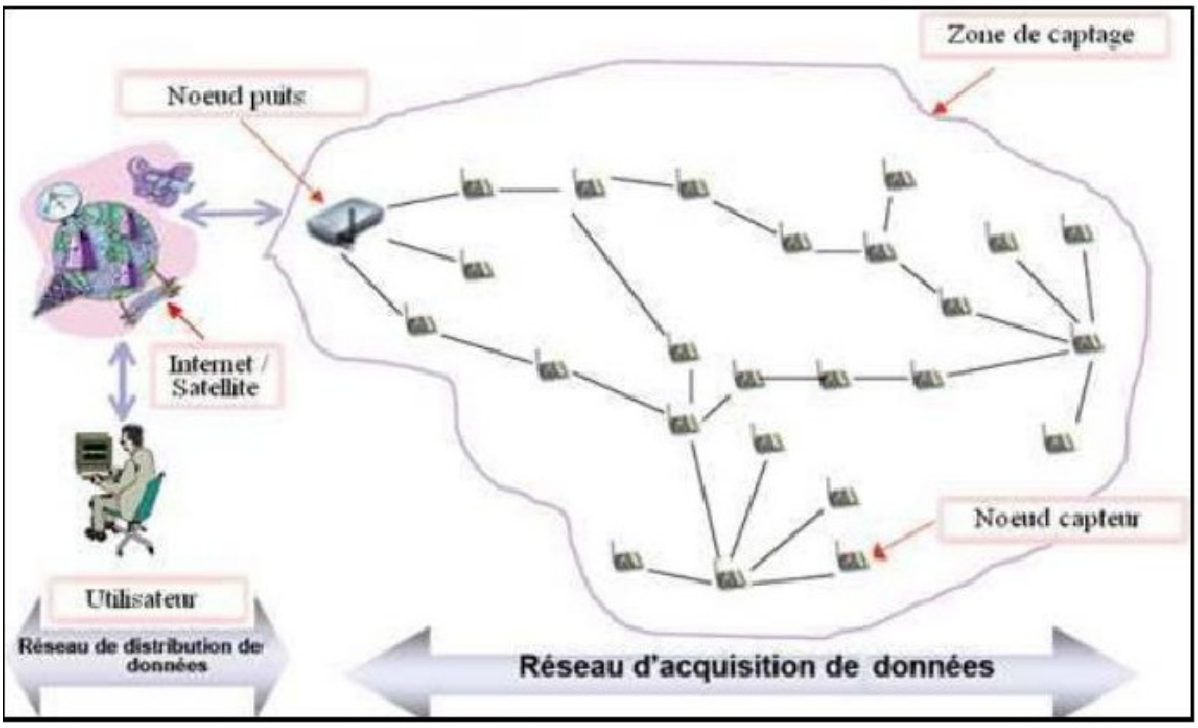

FIGURE 1.3 – Architecture de communication d'un RCSF.

Comme tous les types de réseaux, les RCSFs utilisent une architecture de communication en couches, ce sont les cinq premières couches du modèle OSI ; la couche physique, la couche liaison de données, la couche réseau, la couche transport et la couche application. Chaque couche a son propre rôle et ses propres protocoles pour atteindre son objectif. L'objectif d'un RCSF n'est pas seulement la communication elle même, mais il est soumis à de fortes contraintes énergétiques, par conséquence, d'autres unités doivent être ajouté afin de gérer la consommation d'énergie, la mobilité des nœuds et l'ordonnancement des tâches.

Ces plans de gestions d'énergie, de mobilité et de tâche aident les nœuds capteurs à coordonner les tâches et minimiser la consommation d'énergie. Ils sont donc nécessaires pour que les nœuds capteurs puissent collaborer ensemble, afin d'acheminer les données dans un réseau mobile et de partager les ressources entre eux en utilisant efficacement l'énergie disponible. Ainsi, le réseau peut prolonger sa durée de vie.

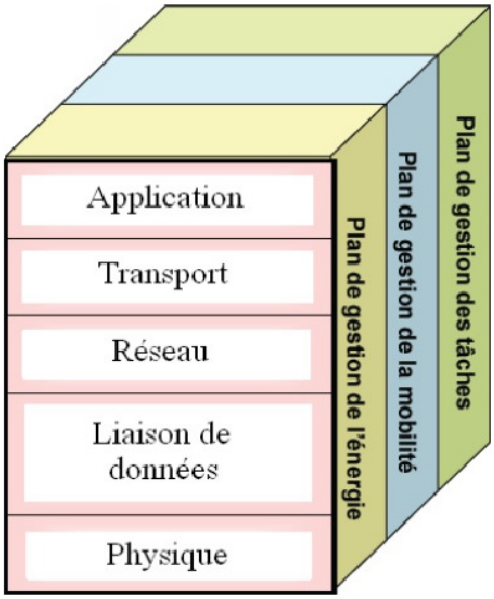

FIGURE 1.4 – Pile protocolaire dans les RCSF [5]

1.8.1 Couche application

Elle assure l'interface avec les applications. Il s'agit donc de la couche la plus proche des utilisateurs, gérée directement par les logiciels. Parmi les protocoles d'application, nous citons : SMP (Sensor Management Protocol) et TADAP (Task Assignement and Data Advertisement-Protocol).Elle apporte sa contribution dans les réseaux de capteurs sans fil en satisfaisant les différentes exigences du client, à travers des collaborations avec les autres couches de la pile protocolaire, dans le but de pouvoir proposer des services pour la gestion de l'énergie, la synchronisation, etc [8].

1.8.2 Couche transport

La couche transport fournit un service de communication de bout en bout, fiable pour l'application. Elle manipule la segmentation des grands paquets. Elle effectue le contrôle des flots de données de bout en bout afin d'éviter la surcharge du récepteur ou du réseau.Dans les RCSFs, la fiabilité de transmission n'est pas majeure. Ainsi, les erreurs et les pertes sont tolérées. Par

conséquent, un protocole de transport proche du protocole UDP et appelé UDP-Like(User Datagram Protocol Like) est utilisé. Cependant, comme le protocole de transport universel est TCP (Transmission Control Protocol), les RCSFs doivent donc posséder, lors d'une communication avec un réseau externe, une interface TCP-spliting pour vérifier la compatibilité entre ces deux réseaux communiquant [9].

1.8.3 Couche réseau

Elle s'occupe du routage de données fournies par la couche transport.Elle établit les routes entre les nœuds capteurs et le nœud puits et sélectionne le meilleur chemin en termes d'énergie, délai de transmission, débit, etc. Les protocoles de routage conçus pour les RCSFs sont différents de ceux conçus pour les réseaux Ad Hoc puisque les RCSFs sont différents selon plusieurs critères comme :

- l'absence d'adressage fixe des nœuds tout en utilisant un adressage basé-attribut.
- l'établissement des communications multi-sauts.
- l'établissement des routes liant plusieurs sources en une seule destination pour agréger des données similaires, etc.Parmi ces protocoles, nous citons : LEACH (Low-Energy Adaptive Clustering Hierarchy) et SAR (Sequential Assignement Routing).

1.8.4 Couche liaison de donnée

La couche de liaison de données est divisée en deux sous-couches : la sous-couche (LLC) et la sous-couche (MAC). La couche LLC fournit la plupart des mécanismes de gestion d'erreur. La couche MAC gère tous les accès au canal radio physique. De plus, elle établit une communication saut-par-saut entre les nœuds. C'est-à-dire, elle détermine les liens de communication entre eux dans une distance d'un seul saut. Comme l'environnement des réseaux de capteurs est bruyant et les nœuds peuvent être mobiles, la couche de liaison de données doit garantir une faible consommation d'énergie et minimiser les collisions entre les données diffusées par les nœuds voisins. Parmi les protocoles de liaison de données, nous citons : SMACS (Self-organizing MediumAccess Control for Sensor networks) et EAR (Eavesdrop And Register).

1.8.5 Couche physique

Elle permet de moduler les données et les acheminer dans le media physique tout en choisissant les bonnes fréquences. Elle est responsable de la sélection de fréquence, la génération de la fréquence porteuse, la détection du signal, la modulation/démodulation et le cryptage/décryptage des informations. La consommation d'énergie au niveau de la couche physique peut être affectée par l'environnement de l'application, le choix du type de la modulation ou la bande de fréquence utilisée. Il est avantageux en matière d'économie d'énergie que le concepteur de la couche physique choisisse une transmission à multi-sauts plutôt qu'une transmission directe qui nécessite une puissance de transmission très élevée.

1.8.6 Plan de gestion d'énergie

La couche de gestion d'énergie contrôle la manière d'utiliser l'énergie par le nœud capteur, et gère la consommation d'énergie selon le mode de fonctionnement employé (capture, calcul, et communication par radio). Par exemple pour éviter de recevoir des messages redondants, le nœud capteur change son mode en «Off » après une réception d'un message d'un de ses voisins. En outre un nœud capteur annonce à ses voisins qu'il a atteint un bas niveau d'énergie, par conséquence il ne va pas participer au routage des données, et l'énergie restante va être utilisée pour capter et détecter des tâches.

1.8.7 Plan de gestion de la mobilité

La couche de gestion de mobilité détecte et enregistre le mouvement/mobilité des nœuds capteurs. En utilisant ces positions, les nœuds capteurs peuvent connaitre qui sont leurs voisins. Parfois une auto-organisation des nœuds est nécessaire à cause de la destruction de quelques nœuds. Dans ce cas, la couche de gestion de mobilité doit être capable de faire changer la position des nœuds.

1.8.8 Plan de gestion des taches

La couche de gestion des tâches assure la coopération des efforts des nœuds capteurs, elle ordonnance les événements captés, et les tâches détectées dans une zone de capture spécifique. Par conséquent, les nœuds capteurs qui appartiennent à la même zone de capture ne sont pas

obligés d'effectuer les tâches de capture en même temps. Selon leur niveau d'énergie, quelques nœuds capteurs peuvent accomplir des tâches de capture mieux que d'autres.

conclusion

Bien qu'ils apportent de nombreux avantages, les réseaux de capteurs posent un certain nombre de défis scientifiques. En effet, la miniaturisation des batteries a entrainé un problème de limitation d'énergie. La consommation d'énergie représente ainsi un facteur majeur lors de la conception des réseaux de capteurs. De ce fait, les travaux doivent se porter sur l'élaboration de protocoles de communication minimisant la consommation énergétique. Les approches traditionnelles telles que les protocoles de routage développés pour les réseaux filaires ne sont pas adaptées aux réseaux de capteurs sans fil. C'est pourquoi de nouvelles approches de routage doivent être mises en place afin de tenir compte des différentes contraintes imposées par les réseaux de capteurs sans fil. Le chapitre suivant sera consacré au routage dans les réseaux de capteurs sans fil.

2

Routage dans les RCSFs

Introduction

Le routage dans les RCSFs consiste à acheminer les paquets de la source vers la destination (station de base). En effet, le principal objectif du routage est de trouver un chemin optimal minimisant la consommation d'énergie augmentant ainsi la durée de vie du réseau. De ce fait, les protocoles élaborés doivent assurer une consommation minimale d'énergie tout en maintenant le bon fonctionnement du réseau et sans dégrader ses performances.
Dans ce chapitre, nous présentons les principaux facteurs influents sur la conception des protocoles de routage et les critères servant à leur classification en citant quelques exemples de protocoles.

2.1 Les considérations de conception d'un protocole de routage dans les RCSFs

La mise en œuvre d'algorithmes de routage pour assurer la connexion des réseaux de capteurs sans fil doit tenir compte de certaines contraintes. La contrainte énergétique étant la plus importante, un protocole de routage pour réseaux de capteurs doit assurer un prolongement de la durée de vie du réseau ainsi que le maintien de sa connectivité. Cependant, d'autres facteurs doivent être pris en considération, parmi lesquels nous citons [10] :

2.1.1 Le déploiement des nœuds

Le déploiement des nœuds est spécifique à une application et peut affecter les performances du protocole de routage. Il peut être déterministe ou aléatoire. Dans le cas d'un déploiement déterministe, les nœuds sont placés manuellement et les données sont routées suivant des chemins prédéterminés. Dans le cas d'un déploiement aléatoire, les routes sont construites dynamiquement. Ainsi certaines régions du champ de captage peuvent bénéficier d'une meilleure connectivité par rapport à d'autres et les données captées dans ces régions peuvent être routées plus facilement.

2.1.2 Le modèle de livraison de données

Ce facteur définit la manière dont les données captées sont transmises à travers le réseau. Le modèle de transmission de données peut être classé dans l'une des catégories suivantes :

- Basé sur le temps (Time-Driven) : les nœuds envoient les données périodiquement suivant un taux de transmission prédéterminé.
- Basé sur les évènements (Event-Driven) : les nœuds capteurs envoient les paquets de données lors de la détection d'un événement.
- Basé sur les requêtes (Query-Driven) : les capteurs ne livrent les données que lorsqu'ils reçoivent une requête émise par le nœud puits.
- Hybride : combinaison des trois modèles précédents.

2.1.3 Tolérance aux pannes

La propriété de tolérance aux pannes est définie par l'aptitude du protocole de routage à maintenir ses fonctionnalités, en cas de panne de quelques nœuds. Le but de la tolérance aux pannes est d'éviter la faille totale du système malgré la présence de fautes dans un sous ensemble de ses composants élémentaires

2.1.4 L'hétérogénéité des nœuds

Généralement, les nœuds d'un RCSF sont homogènes et disposent des mêmes capacités de calcul, de stockage et de ressources énergétiques. Cependant, selon les besoins des applications, les nœuds peuvent avoir des rôles différents. Ainsi, selon la tâche assignée au capteur, les besoins en ressources de calcul, de stockage, de communication et d'énergie peuvent varier d'un nœud à un autre. Pour remédier à ce problème, une solution envisagée par certaines applications consiste à intégrer des nœuds spéciaux plus puissants que les autres et qui seront chargés d'effectuer les tâches les plus coûteuses en termes de ressources énergétiques. Cependant, l'intégration d'un ensemble de nœuds hétérogènes dans un seul réseau impose de nouvelles contraintes liées au routage de données. Par conséquent, la conception des protocoles de routage doit prendre en compte les différents types de nœuds, et les contraintes qui en résultent.

2.1.5 Consommation d'énergie

La consommation d'énergie s'effectue essentiellement au niveau de la capture, du traitement et de la transmission des données. Les études concernant les réseaux de capteurs ont montré que la phase de communication est la plus consommatrice en énergie. Vu que l'énergie des capteurs n'est pas renouvelable, les opérations de capture, de traitement et de routage doivent être correctement évaluées afin d'éviter l'épuisement prématuré de cette énergie et de prolonger la durée de vie du réseau [11]. Les techniques de conservation d'énergie lors de la communication et le calcul sont donc d'une importance majeure dans un réseau de capteurs sans fil. L'efficacité énergétique représente alors une métrique de performance significative car elle influence directement la durée de vie du réseau. Pour cela, au moment du développement d'un protocole de routage, les autres métriques telles que le délai de bout en bout peuvent être négligées au détriment du facteur de consommation d'énergie. Les protocoles de routage doivent donc utiliser des mécanismes efficaces en consommation d'énergie en choisissant par exemple les chemins les moins consommateurs en énergie et en réduisant le nombre de messages de contrôle.

2.1.6 Scalabilité

Le nombre de nœuds dans les réseaux de capteurs est souvent élevé (des centaines de nœuds voire même des milliers). Un protocole de routage doit assurer le bon fonctionnement du réseau quel que soit le nombre de nœuds.

2.1.7 La connectivité

La densité élevée des nœuds dans les réseaux de capteurs exclut la possibilité de l'isolement entre eux [12]. Par conséquent, chaque nœud doit être fortement connecté pour éviter son isolation. Cependant, leur dispersement parfois aléatoire fait qu'ils ne sont pas uniformément répartis. Par conséquent, certaines régions bénéficient d'une grande connectivité contrairement à d'autres. Les protocoles de routage mis en place doivent donc permettre l'auto-organisation des nœuds par rapport à la distribution aléatoire et à la topologie dynamique du réseau.

2.1.8 L'agrégation des données

L'agrégation de données est très importante dans les réseaux de capteurs sans fil. La densité importante des nœuds peut causer des redondances de données dans le réseau. Un nœud du réseau peut recevoir la même donnée à partir de plusieurs nœuds. La transmission de la même donnée par plusieurs capteurs implique alors une perte d'énergie inutile. L'agrégation de données peut être réalisée en supprimant les paquets redondants ou en choisissant le maximum ou le minimum ou encore la moyenne des valeurs perçues.

Lors de la conception d'un protocole de routage, il faut prévoir la ou les techniques d'agrégation et éventuellement choisir des nœuds concernés par ces opérations.

2.1.9 La mobilité

La mobilité des nœuds et/ou de la station de base est parfois nécessaire dans les réseaux de capteurs sans fil. C'est le cas pour des applications de type 'detection/tracking' où la cible change de position d'une façon continue. Le routage dans de telles applications devient très difficile, car il faut acheminer des paquets de et vers des nœuds mobiles. Dans ces applications, la stabilité des chemins devient un défi aussi important que la conservation d'énergie.

2.1.10 La qualité de service

Certaines applications des réseaux de capteurs sans fil exigent que les données perçues doivent atteindre la station de base durant un temps limité pour que la donnée soit utile et acceptable. Dans ce cas, le temps de réponse constitue un paramètre très important dans le protocole de routage. Cependant, dans plusieurs applications, la conservation d'énergie, qui est directement liée à la durée de vie du réseau est considérée relativement plus importante que la qualité de service. Ainsi, le réseau peut être appelé à réduire la qualité de service des données envoyées afin d'augmenter la durée de vie du réseau.

2.2 Classification des approches de routage dans les réseaux de capteurs

Les protocoles de routage peuvent être classés selon plusieurs approches :

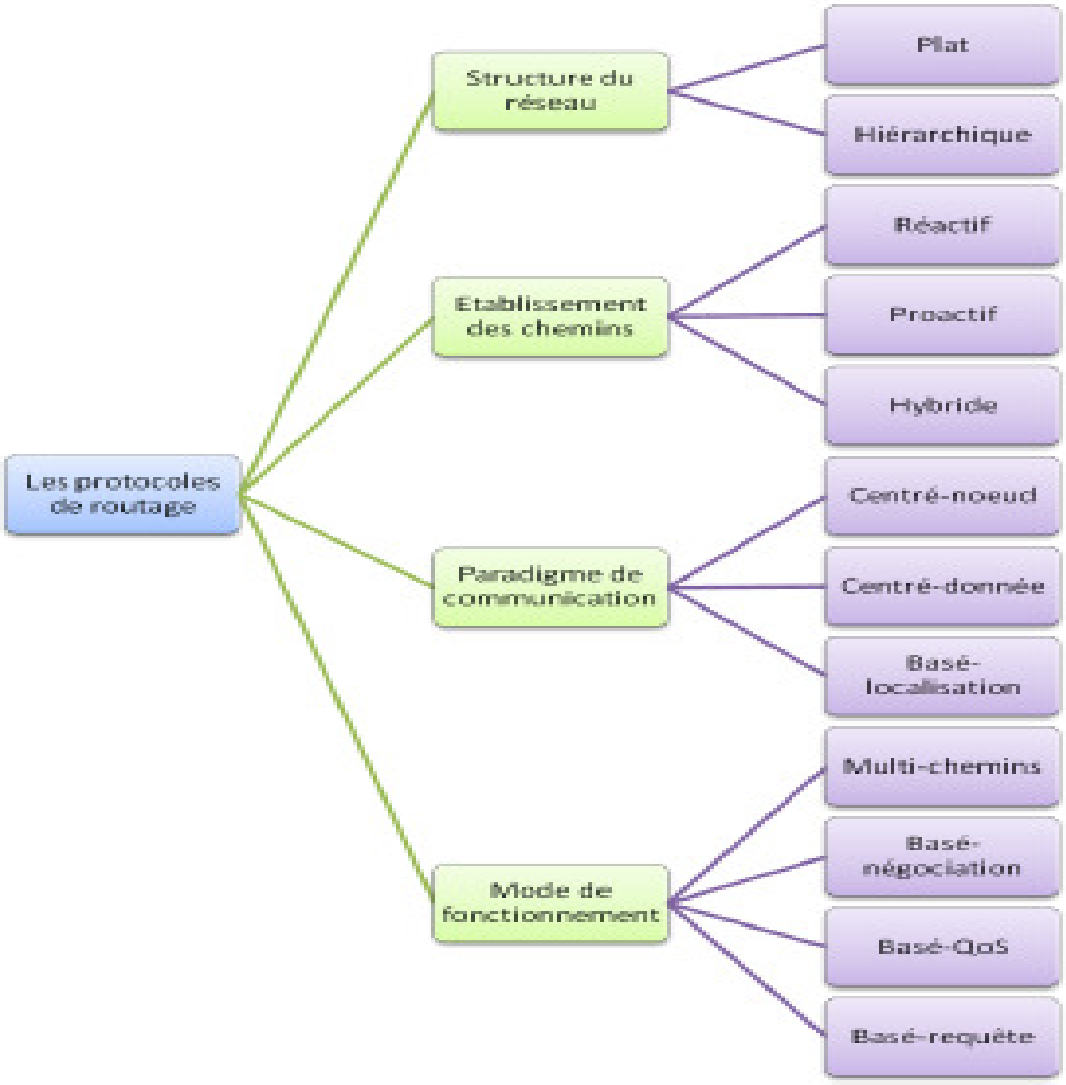

FIGURE 2.1 – Classification des protocoles de routage [13]

2.2.1 Classification selon la structure du réseau

La topologie détermine l'organisation des capteurs dans un RCSF. Globalement, il existe deux topologies dans les RCSFs : la topologie plate et la topologie hiérarchique

- Topologie plate : Dans une topologie plate tous les nœuds sont au même niveau et participent de la même manière au routage des données en adoptant un routage multi-sauts. Seul le nœud puits est chargé de la collecte des données issues des différents nœuds capteurs. Cette architecture est caractérisée par : un coût de maintien réduit, une grande tolérance aux pannes ainsi qu'une habilité à construire de nouveaux chemins suite aux changements de topologie. Cependant, Les nœuds proches de la station de base vont participer plus que les autres aux tâches de routage. De plus, cette architecture présente une faible scalabilité due au fonctionnement identique des nœuds et d'une manière distribuée nécessitant ainsi un grand nombre de messages de contrôle.

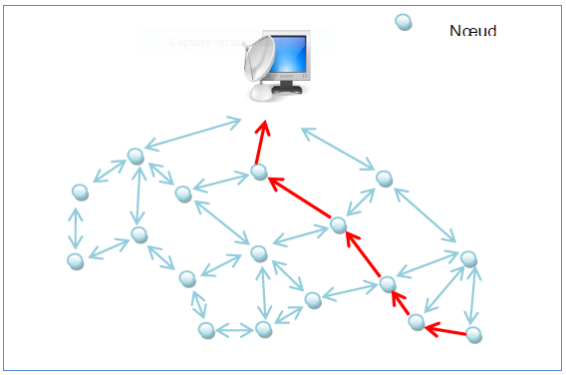

FIGURE 2.2 – Topologie Plate

- Topologie hiérarchique : Afin d'augmenter la scalabilité du système, les topologies hiérarchiques ont été introduites en répartissant les nœuds sur plusieurs niveaux de responsabilité. Le réseau est divisé en groupes, où chaque groupe est formé d'un ensemble de nœuds membres et d'un nœud responsable. Selon l'application, les membres peuvent être des voisins directs ou indirects du responsable du groupe. L'architecture hiérarchique présente de grands avantages tels que l'agrégation des données collectées ainsi qu'une grande scalabilité qui assure l'ajout de nouveaux nœuds sans dégrader les performances du réseau. Cependant, cette approche présente l'inconvénient de la surcharge des responsables qui induit un déséquilibre de la consommation d'énergie. Pour remédier à ce problème, les responsables peuvent être des capteurs spécifiques avec plus de ressources énergétiques et plus de capacités de traitement ou bien ils peuvent être élus dynamiquement et ainsi garantir

un équilibre de la consommation d'énergie et augmenter la tolérance aux pannes.

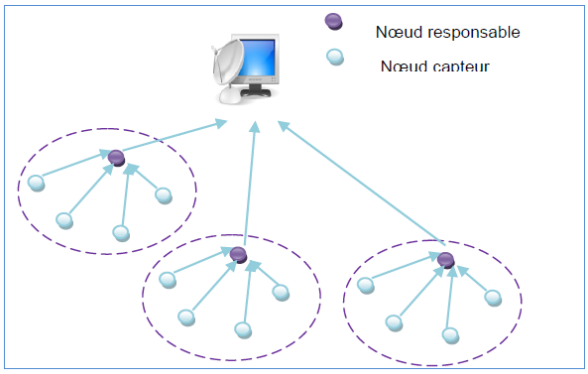

FIGURE 2.3 – Topologie hiérarchique

2.2.2 Classification selon l'établissement des chemins

Le mode d'établissement des chemins permet de classer les protocoles de routage en trois catégories : proactif, réactif ou hybride.

- Protocole proactif : Les protocoles de routage proactifs établissent au préalable les meilleures routes pour chaque nœud vers toutes les destinations possibles. Chaque nœud envoie périodiquement à tous ses voisins, sa table de routage contenant l'état de tous ses liens connus et permet ainsi, de garder une vision globale à jour de l'état du réseau. Dans le cas d'un réseau dense, les tables de routages deviennent vite volumineuses ce qui constitue un réel inconvénient. De plus, des routes peuvent être sauvegardées sans qu'elles ne soient utilisées.
- Protocole réactif : Dans un protocole réactif les routes sont créées à la demande. Lorsque le réseau a besoin d'une route, une procédure de découverte est lancée. Une fois la donnée routée, la route est immédiatement détruite. Cependant, le routage à la demande induit une lenteur à cause de la durée nécessaire à rechercher les chemins, ce qui peut dégrader les performances des applications interactives.
- Protocole hybride : Les protocoles hybrides combinent les deux idées des protocoles proactifs et réactifs. Ils utilisent des méthodes proactives pour établir les routes vers les plus proches voisins (par exemple le voisinage à deux ou à trois sauts) et des méthodes réactives au-delà de la zone de voisinage.

2.2.3 Classification selon le paradigme de communication

Le paradigme de communication détermine la manière dont les nœuds sont interrogés. Dans les RCSFs, nous pouvons distinguer trois paradigmes de communication [14] :

- Centré-nœuds : Utilisé dans les réseaux conventionnels où il est nécessaire de connaître et d'identifier les nœuds communicants, ce modèle ne reflète pas la vision des RCSFs quant à leurs applications. En effet, dans de tels réseaux la donnée transmise est plus importante que le nœud lui-même. Cependant, le paradigme centré-nœuds reste utilisé dans certaines applications de RCSFs nécessitant une interrogation individuelle des nœuds.
- Centré-données : Ce modèle est utilisé dans les réseaux où il n'existe pas un système d'identification global, toutes les communications sont identifiées par leurs données, l'interrogation et le routage doivent être réalisé en se basant sur cette propriété. Le réseau peut être vu comme une base de données distribuée où les nœuds forment des tables virtuelles alimentées par les données captées. Le protocole Directed Diffusion [15] est considéré comme l'un des protocoles de référence dans le routage centré-données.
- Basé-localisation : Cette approche est utilisée dans les applications où il est plus intéressant d'interroger le système en se basant sur la localisation des nœuds et où on peut tirer profit des positions des nœuds pour prendre des décisions qui minimisent le nombre de messages transmis pendant le routage. Cependant, ce type de mécanismes nécessite le déploiement d'une solution de positionnement, dont le degré de précision requis dépend de l'application ciblée. Une solution simple est l'utilisation d'un système de localisation global ou GPS mais ce genre de système reste trop coûteux pour un RCSF. Néanmoins, d'autres méthodes de localisation et de positionnement des capteurs ont été développées telles que la triangulation et la localisation par centroïdes [16]. Nous pouvons citer les protocoles GAF [17] et GEAR [18] comme exemples de protocoles basés-localisation.

2.2.4 Classification selon le mode de fonctionnement du protocole

Le mode de fonctionnement définit la manière avec laquelle les données sont propagées dans le réseau. Selon ce critère, les réseaux de capteurs sans fils peuvent être regroupés en quatre catégories.

- Routage multi-chemins (Multipath) : Les protocoles basés sur ce type de routage utilisent des chemins multiples au lieu d'un seul dans le but d'assurer une bonne fiabilité et d'augmenter les performances du réseau. Des chemins alternatifs sont créés entre la source et la destination et sont maintenus grâce à l'envoi périodique de messages de contrôle. Malgré leur grande tolérance aux pannes, ces protocoles requièrent plus de ressources énergétiques et plus de messages de contrôle.
- Routage basé sur les requêtes : Dans ce type de protocole, le puits propage des requêtes vers les nœuds capteurs. Ces derniers ayant des données à transmettre, répondent en émettant les données via le chemin inverse des requêtes. Les deux protocoles : Directed Diffusion [19] et Rumour Routing [20] se basent sur ce principe.
- Routage basé sur la négociation : Les protocoles basés sur la négociation utilisent des descripteurs de données de haut niveau (métadonnées) afin de décrire la donnée avant de l'émettre. Ainsi, le récepteur éventuel peut décider de recevoir ou pas le paquet. Cette procédure garantit que seules les informations utiles seront transmises et élimine la redondance des données. Le protocole SPIN [21] appartient à la classe des protocoles de routage basés sur la négociation.

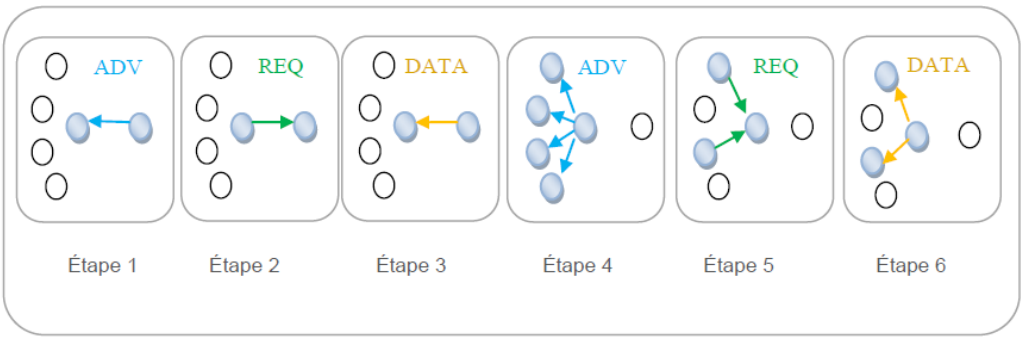

FIGURE 2.4 – Illustration du protocole basé négociation SPIN

- Routage basé-QoS : Les protocoles de routage basés-QoS sont utilisés dans les applications qui ont des exigences temps-réel. Par exemple, dans le domaine de la sécurité, la détection d'intrusion doit être acheminée au plus bref délai vers le nœud puits. Ce type de protocoles essaye de répondre à quelques exigences de qualité de service (délai de transmission ou niveau de fiabilité) et doit faire l'équilibre avec la consommation d'énergie. Le protocole SPEED [22] est l'un des premiers protocoles géographiques basé sur la qualité de service.

2.3 Métriques d'efficacité des protocoles de routage

Un calcul de métrique est un algorithme qui traite le coût associé chaque chemin de routage. Les protocoles de routages permettent aux nœuds de comparer les métriques calculées afin de déterminer les routes optimales à emprunter.
Nous définissons dans cette section les métriques commune utilisées pour mesurer l'efficacité des protocoles de routages. Si la métrique est optimale, le protocole de routage considère que la probabilité pour que les données arrivent intacte et dans un minimum de temps au nœud puit, à travers certains nœud intermédiaire, est grande. Plusieurs métriques peuvent affecter le routage en termes d'énergie, délai, longueur du chemin... etc. Les métriques de routage peuvent être classées en deux classes : métrique applicable à toutes les architectures des RCSFs et métriques significatives seulement pour l'environnement soumis à des contraintes d'énergie.

2.3.1 Métriques applicables à toutes les architectures des RCSFs

2.3.1.1 Le nombre de saut « Hop-count »

Les protocoles de routages utilisent cette métrique pour minimiser le nombre de sauts pendant le routage [23]. L'idée est de calculer le nombre de nœuds intermédiaire pouvant être traversées lors d'une transmission d'un paquet du nœud source vers le nœud puits. La route choisie est celle qui contient un nombre minimum de sauts.

2.3.1.2 La perte de paquets

Les protocoles de routage utilisent cette métrique dans le but de minimiser le nombre de paquets de données perdus lors du transfert depuis une source vers une destination pendant le routage [23]. L'idée est de calculer le taux de paquets perdus et de paquets émis transitant dans le réseau. Autrement dit, on calcule le nombre de paquets perdus sur le nombre de paquets transmis lors d'une transmission. Dans le cas où le taux de perte de paquets est élevé, il est nécessaire de mettre en place un mécanisme permettant de minimiser cette perte.

2.3.1.3 Le temps de traverser un saut « Per hop round trip time »

Cette métrique mesure le temps d'aller-retour des requêtes envoyés aux nœuds voisins. Elle peut être calculée en ayant un nœud qui va envoyer des paquets de requête avec une estampille à l'un des voisins avec un intervalle de temps. Quand le reçoit le paquet, il le transmet de nouveau à l'expéditeur. En comparant l'estampille avec la durée du retour, la qualité du lien peut être évaluée. Naturellement, les résultats de ce test peuvent être altérés par le temps d'attente ou la charge sur les deux nœuds [23].

2.3.1.4 Le délai de bout-en-bout « EED »

L'EED (End-toEnd Deplay) est le temps moyen nécessaire pour qu'un paquet de données soit acheminé à partir de la source vers la destination [23]. Cette technique est parmi les métriques les plus connues dans les réseaux sans fil. Les protocoles de routages l'utilisent pour minimiser le temps de propagation des paquets de données échangées pendant le routage.

2.3.2 Métriques significatives seulement pour les environnements soumis à des contraintes d'énergie

Les protocoles de routage utilisent cet ensemble de métrique pour minimiser la consommation d'énergie pendent le routage [23]. L'idée est de calculer l'énergie disponible(ED) pour chaque nœud du réseau et l'énergie nécessaire(EN) pour les transmissions des paquets entre une paire de nœuds. Les routes entre les nœuds et le puits sont établies et

chacune d'elle est caractérisée par la somme des ED des nœuds qui la constituent et par la somme des EN des liaisons qui la constituent. La consommation d'énergie suit plusieurs approches :

2.3.2.1 La notion du coût « Cost Awarness »

Cost Awareness représente une technique pour minimiser la consommation d'énergie dans le routage, dans laquelle la durée de vie d'un nœud doit être prolongée au maximum. Les choix de l'opération de routage que le nœud fera sont une fonction relative à son de batterie restante. Afin d'utiliser cost awarenessen tant que métrique, On doit calculer la quantité d'énergie consommé pour chaque route imposé au réseau. Plus la consommation d'énergie est minime plus les tâches de routages peuvent être accomplie par le réseau avant qu'il soit défaillant [23]. Autrement dit, la route choisie est celle caractérisée par la plus petite somme des énergies nécessaire.

2.3.2.2 La notion du puissance « Power Awareness »

Power Awareness représente une technique pour minimiser la consommation d'énergie. Elle essaye de réduire au maximum l'énergie totale dépensée lors de l'envoi d'un message depuis sa source à sa destination. Afin d'utiliser « Power Awareness » en tant que métrique, on doit attribuer un poids, basé sur la distance, sur chaque saut possible entre les nœuds du réseau. La route choisie est celle caractérisée par la somme des énergies disponible la plus élevée.

2.3.2.3 La notion du coût puissance

Cette métrique est la combinaison des deux métriques précédentes. Elle vise à réduire au maximum l'énergie consommée dans tout le réseau et en même temps elle évite qu'un nœud ait une quantité d'énergie limitée [23]. La route choisie est celle caractérisée par la puissance des énergies nécessaire et la plus grande somme des énergies disponible.

2.3.2.4 Le temps du premier nœud à mourir

Cette métrique détermine le temps auquel le premier nœud épuise complètement son énergie [23]. Elle n'est pas concernée par la défaillance d'un nœud dû à des raisons techniques.

2.3.2.5 Le temps du dernier nœud à mourir

C'est l'opposé exact de la métrique précédente. Elle enregistre le temps ou le dernier nœud du réseau a consommé toute son énergie [23]. En d'autre terme, cette métrique mesure la durée de vie du réseau.

Conclusion

La communication est très importante dans un réseau de capteurs, puisqu'elle permet aux nœuds de coopérer entre eux et d'acheminer les données vers leurs destinations. L'efficacité en consommation d'énergie représente une métrique de performance significative qui influence directement la durée de vie du réseau en entier.
Dans ce chapitre nous avons présenté les considérations de conception d'un protocole de routage, ensuite nous avons vu les classifications des approches de routage dans les RCSFs, et enfin nous avons présenté les différentes métriques d'efficacité des protocoles de routage. Nous nous intéressons dans le chapitre qui suit aux outils de simulation des RCSFs.

3

Etat de l'art des simulateurs dans les RCSFs

Introduction

La simulation constitue actuellement l'outil le plus pratique pour évaluer le comportement d'un système complexe dont la formalisation à l'aide de méthodes analytiques est difficile .Pour tester les performances d'un réseau mobile on a souvent recours à la simulation. En effet il serait trop coûteux, voire impossible, de mettre en place un réseau à des fins de test pour certains critères. Par exemple, tester des applications sur des réseaux de grande envergure n'est possible en réalité que si l'on dispose de moyens matériels importants. Cependant, dans le cadre d'une simulation, il suffit de changer les paramètres de simulation correspondant à la taille du réseau. Plusieurs simulateurs pour réseaux sans fil ont été proposés, dans ce qui suit nous vous présenteront quelques simulateurs leurs avantages et leurs inconvénients .

3.1 Définition d'un simulateur

Nous appelons simulateur un programme qui met en œuvre un modèle de simulation par événements discrets. La tache première d'un simulateur est d'assurer que la chronologie des événements soit respectée. A chaque occurrence d'un événement, les actions qui sont associées à celui-ci sont exécutées [24].

Les applications de la simulation sont innombrables. Parmi les domaines dans lesquels elle est le plus utilisée, on s'intéresse à la simulation des réseaux de capteurs sans fil. L'un des simulateurs de ce type est « Glomosim » développé en utilisant le langage PARSEC.

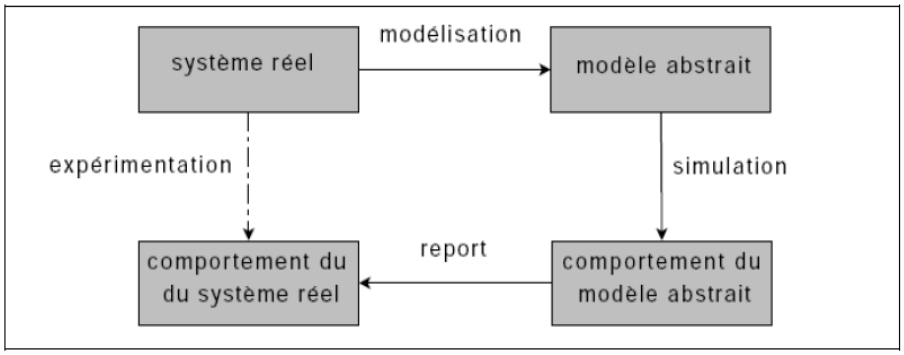

FIGURE 3.1 – Modélisation d'un système

3.1.1 Avantage et inconvénients de la simulation [25]

3.1.1.1 Avantages

- Observations des états du système.
- Etudes des points de fonctionnement d'un système.
- Etudes de l'impact des variables sur les performances du système.
- Etude d'un système sans les contraintes matérielle

3.1.1.2 Inconvénients

- La conception de modèles peut nécessiter des compétences spéciales.
- Résultats pas forcément généralisable.

3.2 Architecture d'un simulateur dans un RCSF

Il y a eu un changement énorme dans l'architecture des simulateurs de réseaux de capteurs sans fils à partir des temps traditionnels jusqu'aux années en cours. De nos jours, les simulateurs comprennent des composants nouveaux et mis à jour en termes d'énergie, consommation d'énergie et d'interface graphique de l'utilisateur. Le simulateur de RCSFs est toujours une application spécifique, parce que les environnements dans lesquels les capteurs sont déployés varient d'un état à l'autre.

Les simulateurs de réseaux de capteur sans fil comprennent les modules suivants : [26] [27]

- **Evénement** : l'événement est une classe basse abstraite qui fournit la fonctionnalité de base pour tous les événements. Il contient le temps ou un événement devrait se déroulé.
- **Medium** : définit la manière de communication entre les Nœuds et détermine le travail de ces derniers.

- **Environnement** : détermine les propriétés physiques d'un modèle. En termes de RCSFs, l'environnement physique incluent : lumière, température, humidité, bruit, secteur optique, zone de couverture, champ magnétique etc.

- **Nœud** : il représente un seul nœud dans le réseau de capteurs, et un RCSF est composé de différents ensembles de nœuds qui communiquent entre eux par l'intermédiaire d'un protocole de routage. Le module nœud contient à la fois des propriétés matérielles et logicielles de nœud, comprend diverses propriétés telles que : le processeur, l'émetteur-récepteur, capteur, source d'énergie, protocole de routage et les applications.

- **Transceiver** : Ce module couvre essentiellement l'émetteur-récepteur du matériel de nœud de capteur. Il couvre les états de l'émetteur-récepteur, et génère des événements pour le début et la fin de chaque signal qu'il transmet. Tous ces événements sont échangés avec instance de module moyen.

- **Protocole physique** : est la couche de plus bas niveau de la pile de réseau, il est principalement mis en œuvre dans le matériel de l'émetteur-récepteur. La couche physique fournit les services suivants :(modification de l'état d'émetteur-récepteur, transporteur de détection, envoi et réception de paquets, détecter l'énergie à la réception des paquets, changement de canal sur la couche physique, qui supporte plusieurs canaux).

- **Protocole MAC** : la couche MAC se trouve entre la couche physique et le routage de couche de la pile réseau, et elle est principalement mis en œuvre sur le logiciel d'exploitation dans le processeur de nœud. Afin d'assurer la fiabilité et l'efficacité du RCSF, les travaux majeurs se font par la couche MAC. La couche MAC est responsable des services suivants :(les politiques d'accès au canal,la programmation, la Gestion des erreurs et l'envoi et la réception de paquets entre les nœuds).

- **Protocoles de routage** :les protocoles de routage comprennent des diverses techniques de routage pour envoyer des données entre les nœuds de capteurs et stations de base. Les protocoles de routage en termes de réseaux de capteurs peuvent être classés comme suit :(sur la base du fonctionnement et Applications, sur la base de style de participation ou sur la base de la structure du réseau).

- **Couche d'application** : la couche d'application comprend différents protocoles de couche d'application qui exécutent diverses applications de réseaux de capteurs tels que la diffusion de la requête, le nœud de localisation, la synchronisation temporelle et la sécurité du réseau. Cette couche est également responsable de la gestion du trafic des nœuds de capteurs et fournit un support logiciel pour diverses applications qui traduisent les données sous une forme compréhensible ou transmettent des requêtes pour obtenir certaines informations.

3.3 Comparaison de quelques simulateurs existants

3.3.1 Le simulateur NS-2

- **Présentation**

Le simulateur du réseau NS-2 est un outil logiciel de simulation de réseaux informatiques. Il est principalement bâti avec les idées de la conception par objets, de réutilisation du code et de modularité. NS-2 est écrit en C++ et utilise le langage OTCL (Object Tools Command Langage) dérivé de TCL. A travers OTCL, l'utilisateur décrit les conditions de la simulation : la topologie du réseau, les caractéristiques des liens physiques, les protocoles utilisés, les communications qui ont lieu. La simulation doit d'abord être saisie sous forme de fichier que NS-2 va utiliser pour produire un fichier contenant les résultats. Mais l'utilisation de l'Otcl permet aussi à l'utilisateur de créer ses propres procédures (par exemple s'il souhaite enregistrer dans un fichier l'évolution d'une variable caractéristique du réseau au cours du temps). Il contient les fonctionnalités nécessaires à l'étude des algorithmes de routage unicast ou multicast, des protocoles de transport, de réservation, des services intégrés, des protocoles d'application. De plus le simulateur possède déjà une palette de systèmes de transmission, d'ordonnanceurs et de politiques de gestion de files d'attente pour effectuer des études de contrôle de congestion. [28] Le Simulateur se compose d'une interface de programmation en TCL et d'un noyau écrit en C++ dans lequel la plupart des protocoles réseaux ont été implémentés :

- Traffic parreto, ON/OFF, CBR, FTP, telnet, etc.
- Couche Transport TCP, UDP

- Couche Réseaux IP, routage dans les réseaux ad-hoc (aodv, dsr,dsdv, tora, amodv), routage dans les réseaux filaire (Link state, Distance vector), les réseaux multicast, IntServ, DiffServ .
- Couche MAC, CSMA, CDMA, 802, X, Token ring, MPLS, liens satellite, etc. . . .

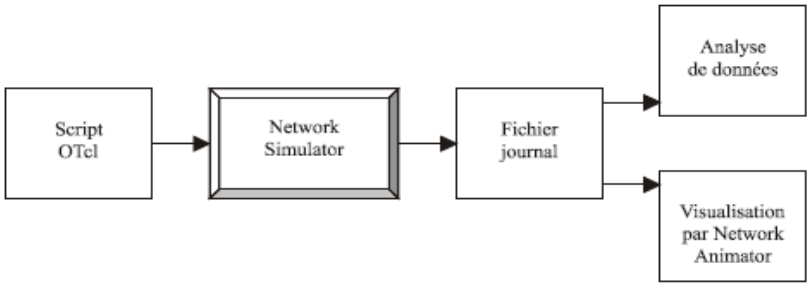

FIGURE 3.2 – Simulation sous NS

- **Avantages**

- Un logiciel de simulation multicouche.
- Un outil complètement libre pour plusieurs plateformes.
- Possibilité d'ajouter des composants à la demande.
- Développement orienté objet.
- Du fait de sa popularité, de nombreux protocoles sont à priori disponibles pour NS-2.

- **Inconvénients**

- la modélisation dans NS-2 reste une tâche complexe : il n'y a pas d'interface graphique.
- Une forte technicité est requise pour utiliser ce simulateur

3.3.2 Le simulateur OMNeT++

- **Présentation**

OMNet++ IDE (integrated devlopment environment) est basé sur la plateforme Eclipse. C'est un environnement open source qui fournit des outils pour la création et les configurations des modèles de réseaux (les fichiers NED et INI) et des outils pour l'exécution d'un lot de programme ainsi que pour l'analyse des résultats de simulation. OMNeT++ semble être le meilleur parmi les solutions open source et freeware. OMNeT++ semble séduire de plus en plus la communauté scientifique et un nombre croissant de modèle sont disponible.

- **Avantages**

- Architecture modulaire permettant l'intégration de nouveau modèle ;
- Utilisation du C++ (et récemment du C#) pour le développement du noyau ;
- Les classes de base du simulateur peuvent être étendue et personnalisées ;
- Conception de modèle rapprochant de la réalité ;
- La mise en route avec ce simulateur est assez simple grâce à une conception claire du simulateur ;
- Il fournit également une puissante bibliothèque d'interfaces graphiques pour l'animation et la gestion du débogage ;

- **Inconvénients**

- Peu de modèles pour les réseaux sans fil ;
- Description des modèles en langage NED ;
- il y a un manque cruel de protocoles disponibles dans la bibliothèque comparé à d'autres simulateurs. [29]

3.3.3 Le simulateur J-Sim

- **Présentation**

J-Sim a été créé par le DCSE (Department of Computer Science and Engineering) de l'université West Bohemia de Pilsen, en République Tchèque. Il est entièrement gratuit. J-Sim permet de simuler des réseaux de l'ordre de 1000 nœuds. Le passage à l'échelle peut toutefois être amélioré. Le simulateur utilise quasi indifféremment deux langages :

Java et TCL (Tool Command Langage). L'architecture et le code sont suffisamment bien structurés pour permettre une prise en main relativement rapide. L'analyse des résultats est aisée et son architecture très modulable. De plus il permet d'utiliser n'importe quelle application Java comme générateur de trafic.

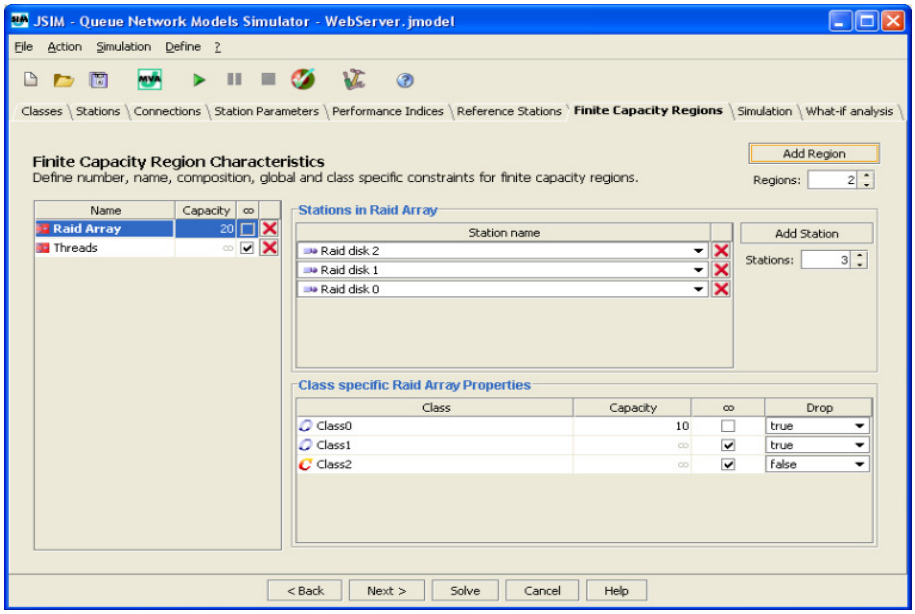

FIGURE 3.3 – Interface graphique de J-Sim

- **Avantages**

- Les modèles de J-Sim ont une bonne réutilisabilité et interchangeabilité, ce qui facilite l'installation de simulation.
- Il contient un grand nombre de protocoles, ce simulateur peut également prendre en charge des diffusions de données, les routages et les simulations de localisation dans les RCSFs par des modèles de détail dans les protocoles de J-Sim. J-Sim peut aussi simuler les chaînes de radio et la consommation d'énergie dans les RCSFs.
- J-Sim fournit une bibliothèque de GUI, ce qui peut aider les utilisateurs à tracer et déboguer des programmes. La plate-forme indépendante facilite aux utilisateurs de choisir des composants spécifiques pour résoudre les problèmes individuels.
- En comparaison avec le NS-2, J-Sim peut simuler un grand nombre de nœuds de capteurs, environ 500, et il peut économiser beaucoup de tailles de mémoire.

- **Inconvénients**

- Le temps d'exécution est beaucoup plus long que celui de NS-2. Parce que J-Sim n'a pas été conçu pour simuler les RCSFs
- La conception inhérente de J-Sim rend l'ajout de nouveaux protocoles et composants plus difficile pour les utilisateurs.

3.3.4 Le simulateur Prowler

- **Présentation**

Prowler est un simulateur de réseaux sans fil conçu pour fonctionner sous l'environnement Matlab. C'est un simulateur, écrit à l'origine pour simuler des grains de BerkeleyMICA, et il est extensible également pour des plates-formes plus générales. Prowler est implémenté et mis en application sous le langage de Matlab (m-File) ce qui rend le code de simulation direct, par exemple, protocole de routage. Les avantages gagnés de l'environnement de Matlab sont le prototypage facile des applications, l'intégration des différents algorithmes d'optimisation, une interface graphique et une capacité de visualisation. [30] Le simulateur à un mode déterministe qui peut être testé par exemple pour les codes d'application et un mode probabiliste pour des simulations bas-niveau. L'échelle prowler est conçue pour simuler un nombre arbitraire de nœuds capteur avec tout type d'application, il s'adapte aussi au changement dynamique de la topologie du réseau. Prowler est capable de simuler la transmission par radio, la propagation, la réception comprenant des collisions dans les réseaux de radio ad hoc, et l'opération de la couche MAC. Les modèles par radio sont basés sur des calculs de force du signal spécifiques combinés avec les erreurs aléatoires [30].

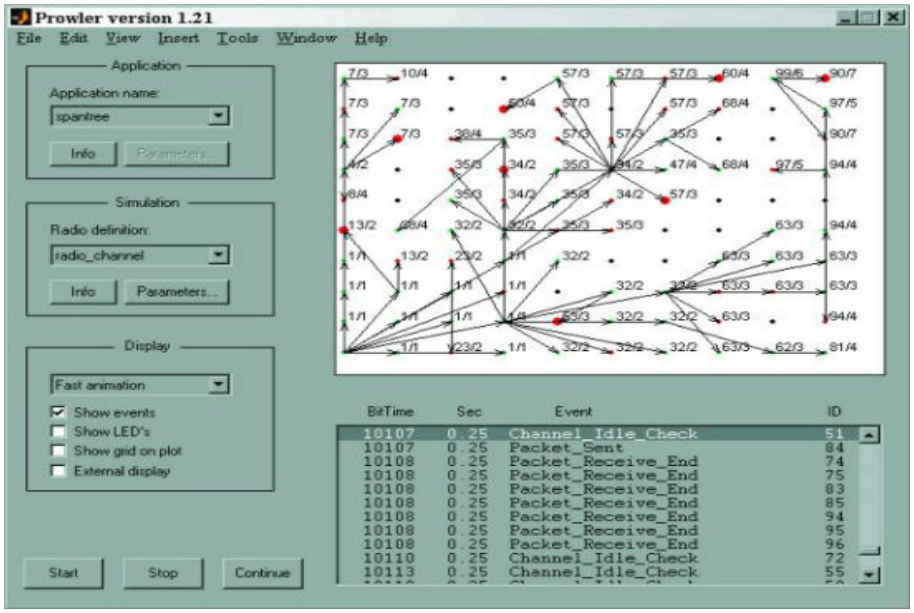

FIGURE 3.4 – Interface graphique de Prowler

- **Avantages**

- Grace au plug-in, prowler a une bonne extensibilité ;
- Il peut être utilisé pour analyser l'énergie des protocoles de routage dans les RCSFs, le taux de livraison et l'analyse du retard du paquet.

- **Inconvénients**

- Seul un guide utilisateur est disponible sur le simulateur Prowler, par contre la documentation est actuellement indisponible ;
- Prowler ne fournit pas plusieurs modèles de propagation radioélectrique, seulement deux ont été mis en application ;
- Aucune modélisation détaillée d'antenne n'est inclus dans les paquets de Prowler ;
- Prowler ne prend pas en compte la modélisation de l'énergie des nœuds capteur, et a besoin d'une extension pour cela.

3.3.5 Le simulateur TOSSIM

- **Présentation**

TOSSIM est le simulateur de TinyOS. C'est un simulateur/émulateur à événements discrets de RCSFs accompagnant le système d'exploitation embarqué TinyOS [31]. Il est répandu pour la disponibilité de son code ainsi pour sa nature comme logiciel libre. TOSSIM peut fonctionner sous Linux ou Windows. Son grand avantage par rapport à beaucoup d'autres simulateurs de réseaux de capteurs est qu'il supporte différents types de nœuds de RCSFs. En effet, TOSSIM peut simuler le fonctionnement de mica, imote, mica2, mica2-dot, micaz, telos, telos-hc08, telosa, telosb et tmote. En plus, il peut simuler simultanément un nombre très grand de nœuds. Les simulations avec TOSSIM nous donnent une idée sur le fonctionnement du réseau, les émission/réceptions, les liaisons radio entre les nœuds, les messages d'erreurs ... etc.
Ce qui nous intéresse le plus dans ce simulateur est son extension PowerTOSSIM. Cette dernière permet de simuler la consommation dans chaque périphérique au niveau de chaque nœud du réseau.

- **Avantages**

- TOSSIM permet de simuler fidèlement le comportement d'un réseau de capteurs
- L'affichage des évènements et des messages de débogage pour chaque capteur mais aussi simultanément pour l'ensemble des capteurs [32]
- L'interface graphique TinyViz, qui permet la visualisation des échanges radios, conjointement aux messages de débogage, ce qui permet, à chaque instant de la simulation, d'avoir une vue globale de l'activité du réseau étudié
- TinyViz offre la possibilité de ralentir la simulation par un délai, afin d'observer le déroulement des évènements, ce qui est très intéressant lorsque le réseau est surchargé de messages.
- Le modèle open-source avec la documentation en ligne gratuite économise le Coût de la simulation.

- **Inconvénients**

- Deux capteurs ne peuvent pas exécuter deux applications différentes. En effet, il serait plus intéressant de distinguer les capteurs maîtres des capteurs esclave, en leur demandant d'exécuter des codes différents. [32]

- TOSSIM est conçu pour simuler des comportements et des applications de TinyOS, et il n'a pas été conçu pour simuler les indicateurs de performance des autres nouveaux protocoles. Donc TOSSIM ne peut pas simuler correctement les questions de la consommation d'énergie dans les RCSFs.

3.3.6 Le simulateur SENS

- **Présentation**

SENS [33] est un simulateur à base de composants personnalisable pour les applications de WSN. Il se compose d'éléments interchangeables et extensibles pour les applications, la communication réseau, et l'environnement physique. Dans SENS, chaque nœud est divisé en quatre composantes principales :

- Application : simule l'application logicielle du nœud de détection ;
- Network : traite les paquets entrants et sortants ;
- Physique : lit les informations détectées ;
- Environnement : c'est les caractéristiques de propagation du réseau.

SENS définit trois modèles de réseau qui peuvent être utilisés. Le premier transmet des paquets avec succès à tous les voisins, le second livre avec un risque de perte basé sur une probabilité fixe, et le troisième considère le risque de collision au niveau de chaque nœud. La composante physique comprend le matériel non-réseau pour le capteur comme le pouvoir, capteurs et actionneurs. A un niveau inférieur, le composant de l'environnement modélise les phénomènes physiques et la mise en page. Le modèle de mise en page comprend différents types de surfaces, chaque radio affectant et la propagation du son d'une manière différente.

- **Avantages**
  - Les utilisateurs peuvent choisir entre différents environnements spécifiques à l'application avec des caractéristiques différentes de propagation du signal.
  - Le code source de SENS peut être porté directement dans les nœuds de capteurs réels, ce qui permet la portabilité des applications.

- Elle fournit un module de puissance pour le développement fiable des applications.

- **Inconvénients**

  - SENS est moins personnalisable que beaucoup d'autres simulateurs, ne fourni aucune chance de modifier le protocole MAC, ainsi que d'autres protocoles de réseau de bas niveau

  - SENS utilise l'un des modèles environnementaux les plus sophistiqués et met en œuvre une bonne utilisation des capteurs. Cependant, le seul phénomène mesurable c'est le son.

3.3.7 Le simulateur Glomossim

- **Présentation**

GloMoSim pour Global Mobile Information System Simulator [34] est un simulateur conçu à l'origine pour la simulation de réseaux mobiles à grande échelle. Il est construit à partir du langage Parsec. Le Parsec pour Parrallel Simulation Environment for Complex System est un langage de simulation parallèle dérivé du C. C'est un C auquel sont rajoutées des fonctions d'envoi, de réception de message et de gestion de timer [35]. L'élément de base dans le Parsec est l'entité. GloMoSim profite de cet aspect. Il est conçu suivant une conception modulaire et hiérarchique, dans laquelle, un ensemble de nœuds sont agrégés au sein d'une seule entité Parsec : partition. De même, la pile de protocole est agrégée dans une seule entité de façon très fidèle au concept de structuration en couche du modèle Transport Control Protocol/Internet Protocol (TCP/IP) [36]. Ces deux techniques d'agrégation lui confèrent une très bonne scalabilité (support pour la simulation de milliers de nœuds).

Les interactions inter-couches se font très simplement par l'utilisation d'APIs (Application Programming Interface) réutilisables et extensibles (Figure1.15). Quant aux autres interactions entre les nœuds et la gestion de timer, elles sont assurées par le moteur de simulation. GloMoSim constitue l'un des simulateurs les plus riches, avec des modèles très évolués au niveau des couches basses, auxquels nous nous intéressons. Il se vend en

version commerciale sous le nom de QualNet [37] qui propose dans sa version 4.5 une implémentation de la pile protocolaire du 802.15.4.

- **Avantages**

- Glomosim est un simulateur doté d'une grande portabilité (Sun Solaris, Linux, Windows) et sa licence est gratuite pour les universitaires.
- Il est capable de simuler un réseau purement sans fil, avec tous les protocoles de routage que cela inclut (AODV, DSR, algorithme de BellmanFord (routage par vecteur de distance), ODMRP20, WRP21, FSR22, ...). Dans le futur, des nouvelles versions pourront simuler à la fois un réseau filaire et un réseau hybride. La plupart des systèmes réseaux de Glomosim sont construits en utilisant une approche basée sur l'architecture à sept couches du modèle OSI.
- L'intégration de modules supplémentaires ne nécessite pas de comprendre le fonctionnement du noyau : il suffit juste de savoir utiliser précisément Parsec

- **Inconvénients**

- Son apprentissage peut s'avérer être très difficile.
- Indisponibilité de nouveaux protocoles.
- GloMoSim soutient actuellement des protocoles pour un réseau purement sans fil. À l'avenir, les développeurs prévoient d'ajouter la fonctionnalité pour simuler les réseaux câblés et les réseaux hybrides.
- GloMoSim est limité aux réseaux IP en raison du niveau bas de conception. Par conséquent, il souffre des mêmes problèmes que Ns-2, les formats de paquets, les modèles d'énergie et Les protocoles MAC ne sont pas représentatifs à ceux utilisés dans les RCSFs.

3.3.8 TIKTAK [38]

- **Présentation**

Le simulateur TikTak peut simuler avec précision un nœud de matériel, et l'exécution se fait par l'interprétation de tout le code du processeur embarqué (code d'utilisateur et protocole de communication), au niveau du cycle précis. Plusieurs nœuds peuvent être émulés à ce niveau

alors qu'ils interagissent les uns et les autres par la présence d'un cadre de communication qui émule la couche physique du réseau, y compris l'atténuation du signal et les interférences / collisions entre les nœuds. En même temps, cet émulateur peut émuler un nœud à un niveau supérieur d'abstraction, reposant sur la même pile de protocoles, qui est dans ce cas compilé et exécuté nativement sur la machine hôte (nous les appellerons les nœuds natifs) et qui gère des centaines de fois plus rapidement que celui qu'on a interprété. De cette façon, un système complexe peut être émulé complètement on se fondant sur la précision de l'émulateur matériel pour un nombre limité de nœuds et de la vitesse des nœuds natifs.

La partie centrale du simulateur est un cadre de communication comme indiqué dans la figure ci-dessous qui agit en tant que serveur d'interconnexion (nous l'appellerons le PHY-serveur). Sa principale fonction c'est qu'il est le support de communication de données entre les nœuds par le biais de connexions TCP / IP, simulant la couche physique du RCSF. Le PHY-serveur est un serveur TCP / IP qui écoute les paquets en provenance des nœuds. Chaque nœud (matériel émulé et nœuds natifs) crée une liaison TCP / IP avec PHY-serveur dans le cadre de ses routines d'initialisation, en utilisant la connexion pour la communication au serveur central durant toute la simulation. Enfin, l'environnement de simulation interagit avec l'utilisateur via une interface graphique. L'application de l'interface graphique, qui se connecte directement au PHY-serveur, est utilisée pour configurer la structure du RCSFs et les propriétés de chaque nœud.

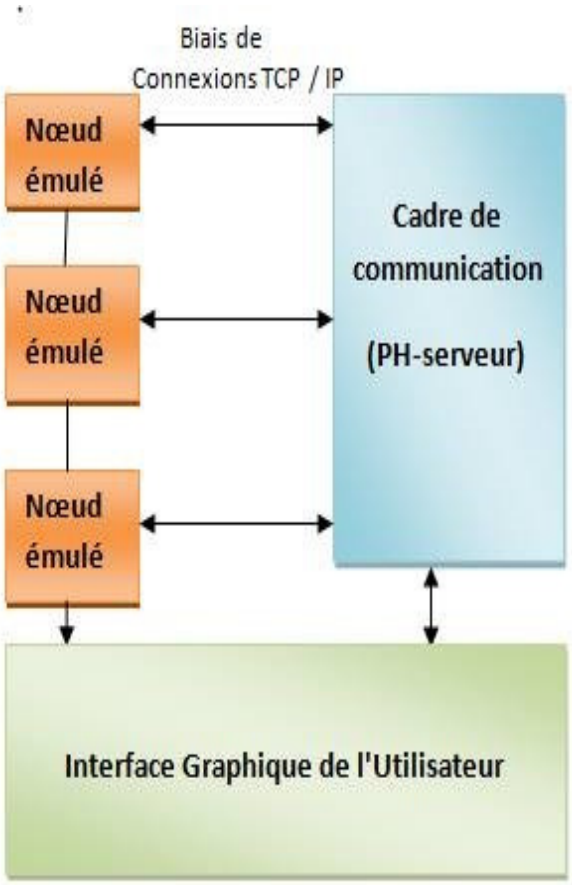

FIGURE 3.5 – Schéma fonctionnel du simulateur TIKTAK

Les nœuds peuvent être divisés à deux niveaux d'abstraction :

- **nœud émulé par le matériel** :Le nœud matériel est un émulateur cycle-précis d'un système inclus, composé de microcontrôleur à 8 bits et d'émetteur-récepteur sans fil. Du point de vue d'environnement de simulation le nœud matériel émulé est un processus indépendant dans l'ordinateur central qui agit l'un sur l'autre avec le simulateur par des connexions de TCP/IP. Les connexions sont employées pour transporter des paquets de RCSF mais également pour commander et configurer chaque nœud.
- **nœud natif** :le nœud natif a été créé à des fins de performance, il est un objet abstrait composé d'une pile de protocole open source Zigbee intégré, compilé directement pour la machine (Linux).

- **Avantages**

La principale caractéristique de ce simulateur est la capacité de simuler le programme et la pile de latence, en raison de l'émulation des nœuds bas niveau du matériel, et de permettre le test et le débogage des codes intégrés comme dans la demande finale. Au même temps, l'émulation au niveau du protocole permet d'augmenter la vitesse de simulation lorsque la précision temporelle est moins stricte.

3.3.9 Tableau comparatif

Une comparaison des simulateurs présentés par rapport aux critères d'évaluation définit est présentée dans la table 3.1 :

- **La mobilité :**Possibilité de paramétrer le mouvement d'une machine. On peut ainsi étudier le comportement du réseau quand un ordinateur devient mobile (topologie dynamique).
- **Le modèle de propagation de la radio :** La propagation des ondes radio constitue un aspect important de la fiabilité. Les ondes radio sont soumises à la diffraction, la réfraction, et de la dispersion. Parmi les simulateurs existants, aucun ne permet l'implémentation de ces trois propriétés de la propagation radio. Les modèles de propagation existants sont basés soit sur des modèles statistiques ou bien des modèles de réfraction partielle.

- **Le modèle énergétique :** La consommation d'énergie joue un rôle très important dans l'estimation de la durée de vie des réseaux (réseaux de capteurs). Concevoir un modèle de consommation d'énergie est une nécessité pour les simulateurs des réseaux de capteurs sans fil. Ces modèles doivent prendre en compte tous les composants qui ont besoin de l'énergie (de la batterie) pour fonctionner, pour créer pour chaque composant son propre modèle.
- **Parallélisme et distribution :** Le parallélisme fait référence à l'exécution simultanée de différentes instructions du même programme. Il est utilisé pour accélérer les simulations. Le parallélisme est une technologie pertinente pour la simulation des réseaux câblés. La distribution est définie comme étant la répartition des données des programmes sur des ordinateurs distincts. Elle est principalement utilisée pour le passage à l'échelle et/ou pour permettre le parallélisme.
- **Interface :**Possibilité de paramétrage de la simulation (entrée) et détails dans les résultats obtenus (sortie).
- **Passage à l'échelle ou scalabilité :** Capacité à gérer les changements de topologie, d'échelle (agrandissement, réduction).
- **La plateforme d'exécution :**Caractérise la machine abstraite sur la (les) quel (les) l'outil est compatible (système d'exploitation et compilateur).
- **Le type de licence :**Il définit, pour chaque entité (personne physique ou morale), comment se procurer d'une version de l'outil en toute légalité.

Simulateurs	NS-2	Omnet++	J-Sim	Tossim	Prowler	Glomosim	Sens	Tiktak
Architecture	orienté objet	modules orienté objet	modules	modules	___	modules	modules	
Mobilité	oui	oui	oui	oui	oui	oui	oui	oui
Parallélisme	non	MPI/PVM	RMI	non	___	SMP	___	oui
Modéle d'énergie	___	oui	oui	___	oui	___	___	___
Modéle radio	oui	oui	___	oui	oui	oui	oui	oui
Passage a l'échelle	oui	oui	oui	oui	oui	oui	oui	oui
Interface	C++ /OGTL	C++	Java	C++ /python	Matlab	Parsec (C)	C++	C++
Licence	Gratuit	Gratuit pour les universitaires et pour toute utilisation non lucrative	Gratuit	Gratuit	Gratuit pour les universitaires	Gratuit	Gratuit	
Plate-forme	Windows Unix (Linux, Solaris)	Windows (Cygwin) Unix	Java	Windows (Cygwin) Unix	Windows, Linux	Windows, Linux, Mac	Windows, Linux	Linux

TABLE 3.1 – Tableau comparatif

Conclusion

Plusieurs simulateurs pour les réseaux sans fil existent et présentent différents modèles et caractéristiques. Le choix d'un simulateur doit être dicté par les exigences des protocoles. Comme les simulateurs permettent de traiter les réseaux en totalité, il est bien pratique de les utiliser surtout qu'ils rendent leur surveillance plus facile. En outre, comme les expérimentations sont décrites comme des scénarios de fichiers, ces derniers sont évidemment reproductibles.

Le choix du simulateur est donc une question délicate, et la réponse dépend en grande partie sur le besoin d'utilisation. Les simulateurs doivent présenter plusieurs propriétés pour l'amélioration de leur précision, leur rapidité, leur scalabilité, leur facilité d'utilisation. Nous allons présenté dans le chapitre suivant un tout nouveau simulateur nommé CupCarbon et nous allons voir ses avantages et ses inconvénients.. . .

4

Simulation du routage avec Cupcarbon

Introduction

Ce présent chapitre représente un guide d'utilisation du nouveau simulateur CupCarbon dédié aux RCSFs. En effet, nous présenterons ce simulateur a travers la description de son architecture ainsi que les étapes de son installation. Nous allons voir comment implémenter un réseau de capteurs en détails. Nous allons également éxecuter une simulation du protocole de routage LEACH et enfin nous allons voir comment génerer les résultats de simulation et créer les graphes dans CupCarbon.

4.1 Présentation du simulateur Cupcarbon

CupCarbon [39] [40] est un simulateur de réseaux de capteurs multi-agents et à événements discrets (ED) basé sur la géolocalisation. Il permet de concevoir et de simuler des réseaux sur une carte géographique de type OpenStreetMap. Pour ce faire, CupCarbon fournit un ensemble d'objets facilement manipulables et configurables. L'utilisation des systèmes multi-agents permet d'optimiser le temps de simulation en parallélisant les différents objets et événements. CupCarbon est composé de trois blocs principaux : le simulateur de l'environnement multi-agents, le simulateur online de réseaux sans fil (WiSeN) et le simulateur offline de réseaux sans fil (SimBox).

Le simulateur de l'environnement multi-agents permet de simuler des objets tels que les capteurs et les variations de leurs paramètres (exemple : les variations paramétriques de leur portée), les déplacements d'un mobile (exemple : voiture, bus, objets volants [41], etc.), des phénomènes environnementaux (feux, gaz, etc.). Le simulateur WiSeN [39] permet de simuler un réseau de capteurs sans fil de sorte à se rapprocher le plus possible de la réalité. Son but est plutôt pédagogique afin d'offrir la possibilité de visualiser les différents phénomènes liés à la simulation. Il permet aussi de calculer la consommation énergétique en se basant sur un algorithme à événements discrets. Les événements s'exécutent en pingpong à base de deux threads. Un thread génère un événement puis il se met en attente tout en réveillant l'autre thread. Le deuxième thread calcul l'énergie liée à cet événement puis il se met en attente et il réveille le premier thread et ainsi de suite. Le package SimBox quant à lui englobe trois algorithmes de simulation. Un algorithme classique pas à pas, un algorithme à événements

discrets s'exécutant sur le CPU et un algorithme à événements discrets s'exécutant sur GPU.

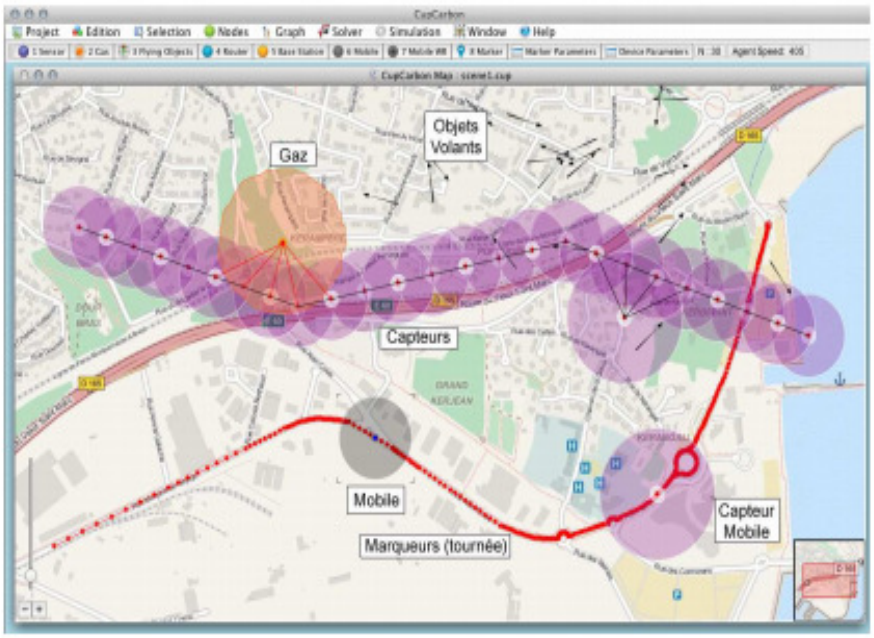

FIGURE 4.1 – L'interface principale du simulateur Cupcarbon

4.2 Description architecturale du simulateur Cupcarbon

Cupcarbon est un outil développé en Java. Sa structure générale est composée de deux niveaux. Le premier niveau concerne les modules utilisés pour mettre en place une simulation. Le deuxième niveau concerne la simulation proprement dit. La Figure suivante montre les différents modules de Cupcarbon.

- Module des Agents : il inclut les dispositifs et les événements nécessaires pour le maquettage d'un réseau et pour préparer et configurer le simulateur.
- Module OpenStreetMap : il permet de concevoir les réseaux de capteurs sur une carte de type OSM.
- Module simulateur WiSeN : il permet de simuler un réseau de capteurs sans fil. Il est connecté au simulateur des agents.

- Module Solveur : il intègre des algorithmes d'optimisation tel que l'algorithme de recouvrement, de coloration, etc. L'algorithme de recouvrement permet par exemple de trouver le minimum de capteurs qui détectent un ensemble de cibles données [42].

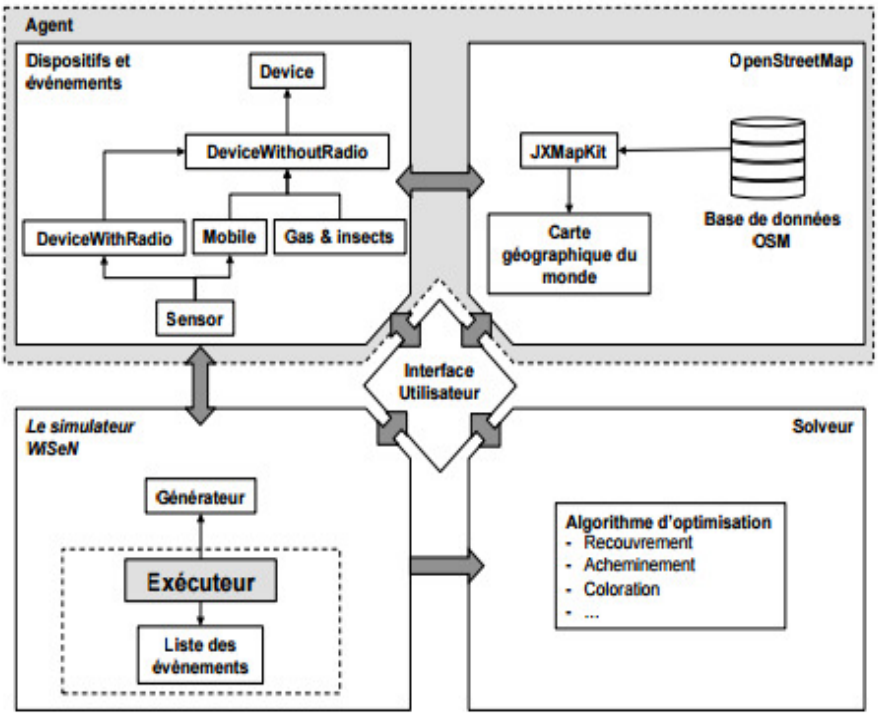

FIGURE 4.2 – L'architecture du simulateur CupCarbon

Le générateur communique avec les agents et en particulier avec les capteurs. Il lance chaque capteur. Ensuite, chaque capteur exécute son script de communication et se déplace selon son fichier gps associé. Les scripts permettent de générer des événements qui seront lancés par le générateur et qui seront récupérés par l'exécuteur afin de les exécuter dans l'ordre de leur date de création.

Durant la simulation, un agent commence à générer des événements décrits dans les fichiers de script de communication et ceux de mobilité associés. Ils seront ensuite communiqués à l'exécuteur afin de les trier pour une phase de traitement suivante.

Tant que la condition d'arrêt n'est pas vérifiée par le générateur, il générera un seul événement à la fois à chaque fois que l'exécuteur le déverrouille en appelant le sémaphore de

synchronisation. Le générateur déverrouille l'exécuteur et entre en état d'attente jusqu'au prochain appel de l'exécuteur.

L'exécuteur est un module indépendant qui s'exécute en parallèle avec les agents. Il organise les événements générés par le générateur afin de les exécuter. Il est composé d'une liste d'événements et d'un algorithme de gestion et de traitement.

L'exécuteur est lancé juste après son initialisation. Ensuite, il s'exécutera en tâche de fond en collectant à chaque itération les prochains événements. Si l'agent générant les événements possède l'énergie nécessaire alors l'événement sera exécuté, sinon, l'exécuteur passe au prochain événement de la liste. Après son exécution, il met à jour tous les agents associés à l'événement, tel que la réception d'un message d'un capteur et l'ajout de cette action au fichier log. L'exécuteur indique l'événement au générateur de sorte à pouvoir générer et ajouter le prochain événement dans la liste de l'exécuteur. Ainsi, il bascule à l'état de repos tant que la liste est utilisée par le générateur (la liste des événements est une ressource critique). L'exécuteur répète ce cycle jusqu'à ce que la condition d'arrêt soit vérifiée. La simulation globale sera finie lorsque l'une des conditions suivantes est vérifiée :

- s'il n'y a pas d'événement à exécuter dans la liste de l'exécuteur
- si le temps de simulation est dépassé
- si un arrêt est forcé par l'utilisateur

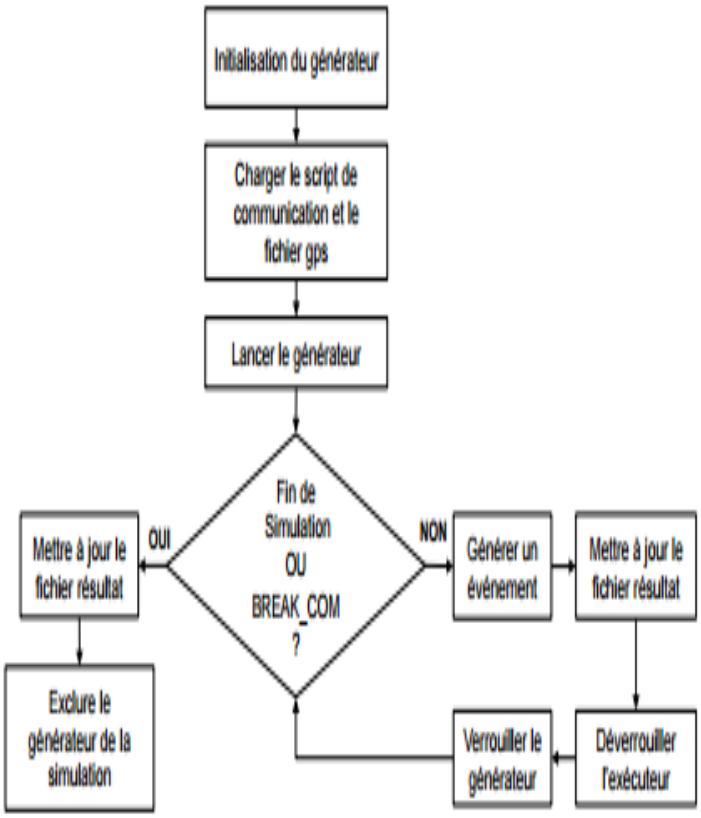

FIGURE 4.3 – Le comportement d'un générateur

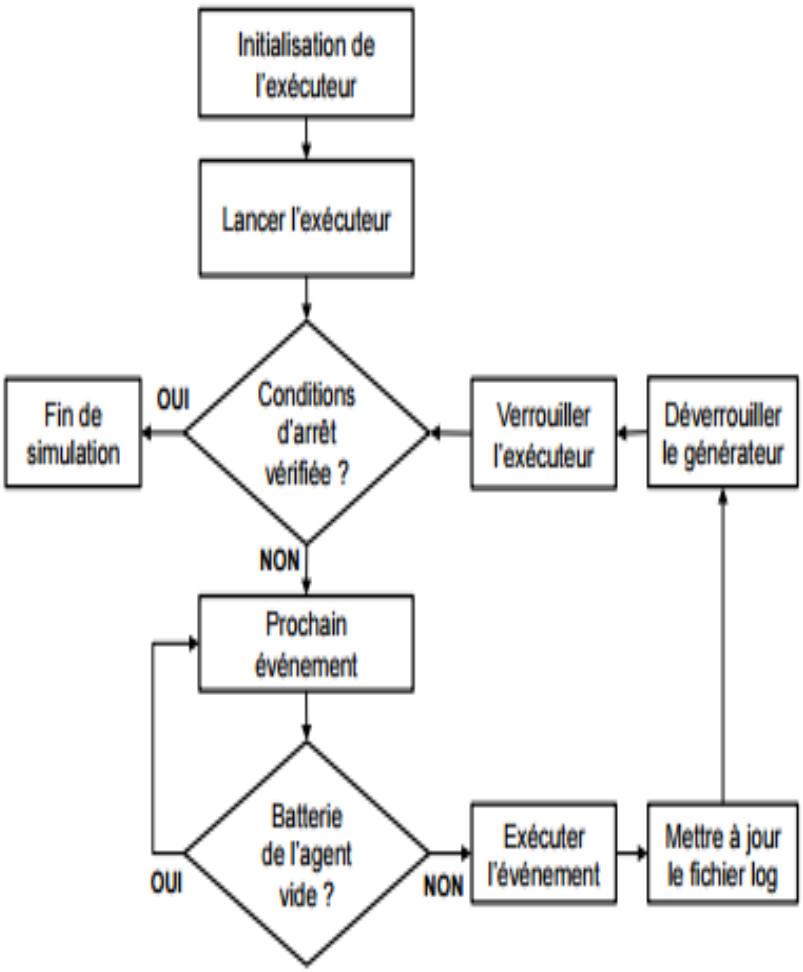

FIGURE 4.4 – Le comportement d'un exécuteur

4.3 Installation du simulateur Cupcarbon

Pour installer Cupcarbon, il suffit de télécharger gratuitement le fichier compressé Cup-Carbon U-One 2.8.5 sur le site officiel du simulateur www.cupcarbon.com, et d'avoir Java préalablement installé, puis on extrait le fichier, après on lance l'application en cliquant sur l'icône CupCarbon.

FIGURE 4.5 – Fichier compressé à télécharger gratuitement [40]

4.4 Implémentation d'un réseau avec le simulateur Cupcarbon [40]

Dans cette section nous allons montrer comment utiliser CupCarbon afin d'implémenté un réseau de capteurs maquette sur OpenStreetMap.

4.4.1 Lancement de Cupcarbon

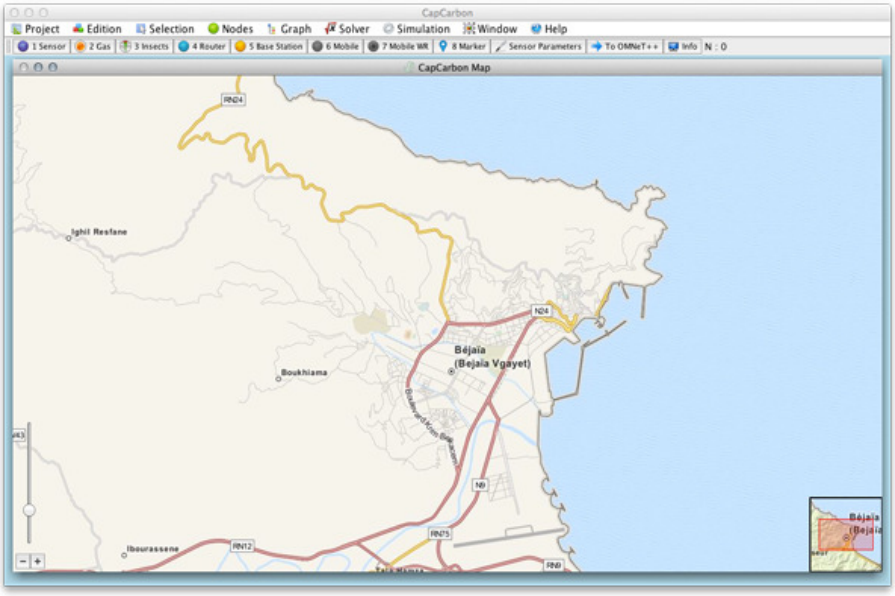

FIGURE 4.6 – Lancement de Cupcarbon

4.4.2 Création d'un nouveau projet

Avant d'ajouter des objets, vous devez créer un nouveau projet (Project -> New Project). Le projet doit être enregistré (Projet -> Enregistrer le projet). Il est également possible de créer un nouveau projet en commençant par le même environnement que celui en cours (Projet -> Nouveau projet à partir du courant).

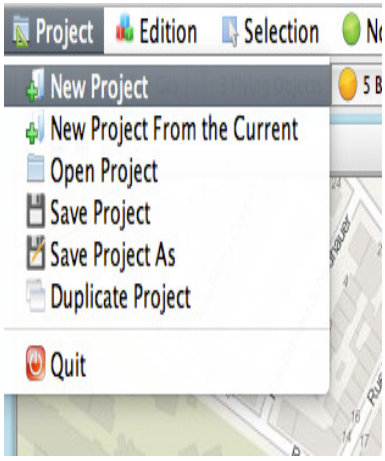

FIGURE 4.7 – Création d'un nouveau projet

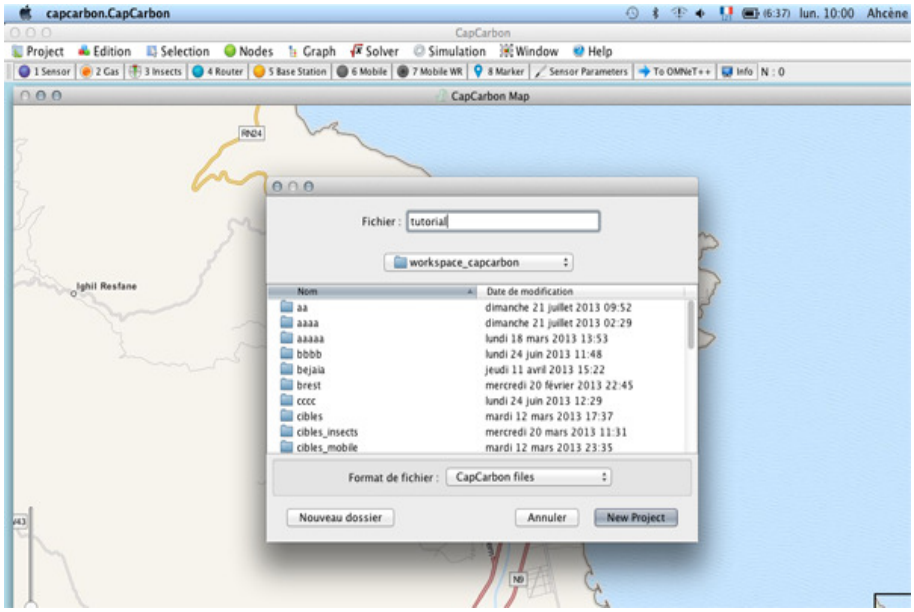

FIGURE 4.8 – Définition du Workspace

4.4.3 Création d'une topologie et des capteurs

On peut ajouter un capteur en cliquant sur le bouton Sensor directement à partir de la barre d'outils, puis en cliquant sur l'espace du travail.

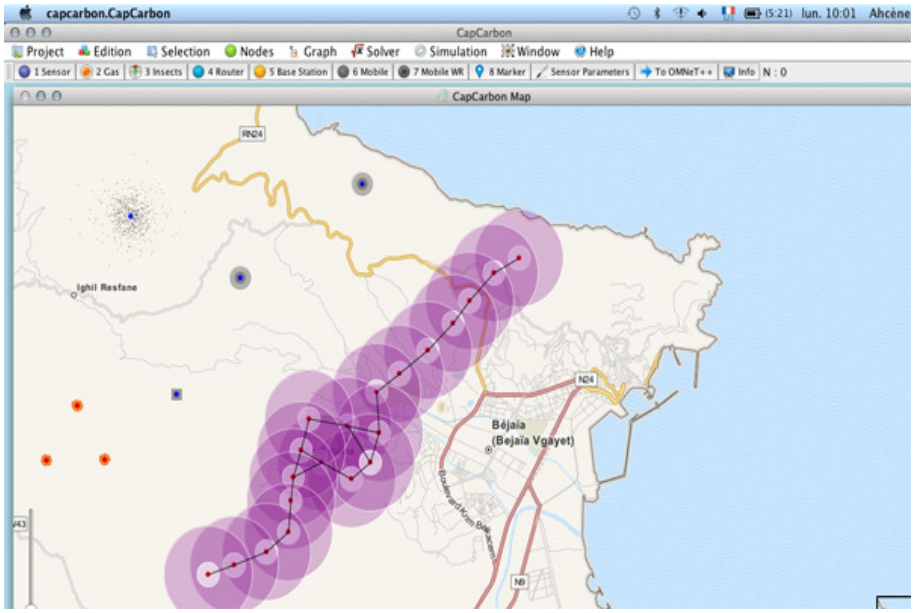

FIGURE 4.9 – Ajout de capteurs

4.4.4 Configuration du réseau et création des scripts

On clique sur le bouton Simulation de barre d'outils, en suite on clique sur Communication Script.

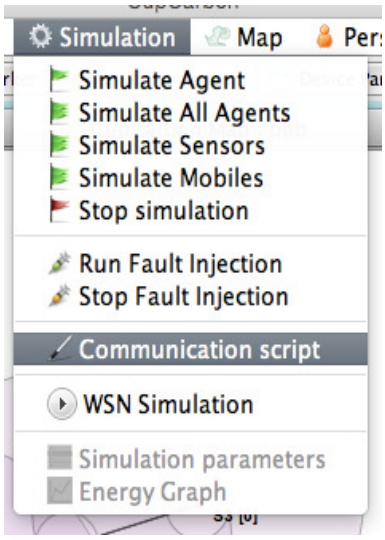

FIGURE 4.10 – Création d'un script

4.4.4.1 Création du script de l'émetteur

Ajouter le script suivant et entrez le nom du fichier : script1 et enregistrez-le.

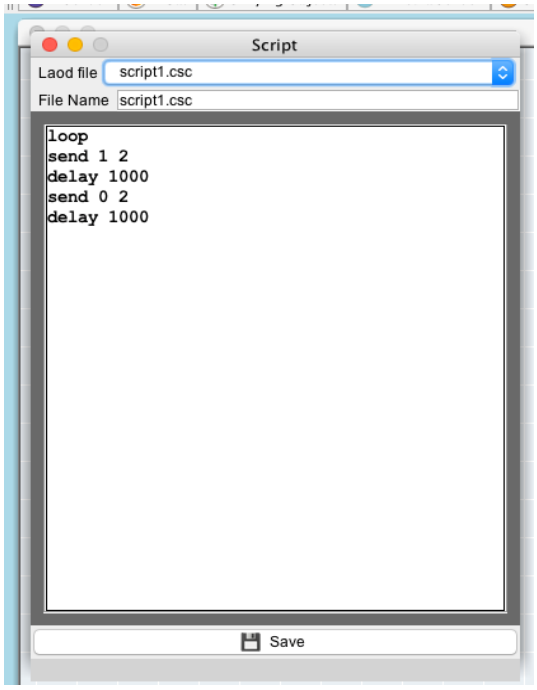

FIGURE 4.11 – Script de l'émetteur

4.4.4.2 Création du script de router

Ajouter le script suivant et entrez le nom du fichier : script2 et enregistrez-le.

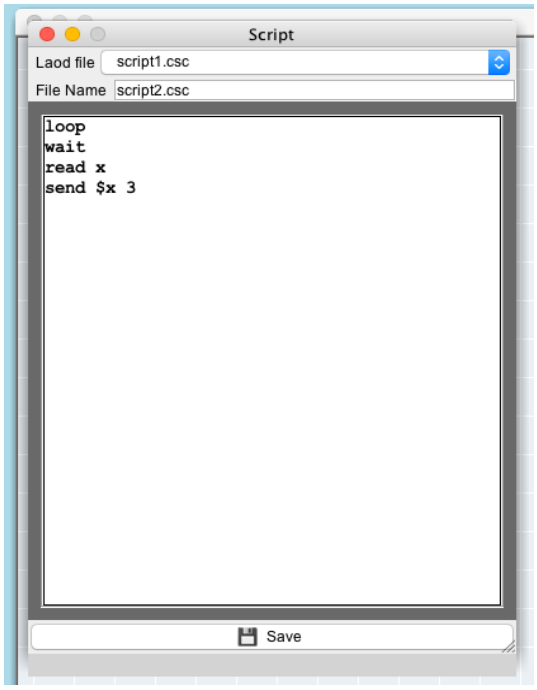

FIGURE 4.12 – Script de router

4.4.4.3 Création du script de coordinateur

Ajouter le script suivant et entrez le nom du fichier : script3 et enregistrez-le.

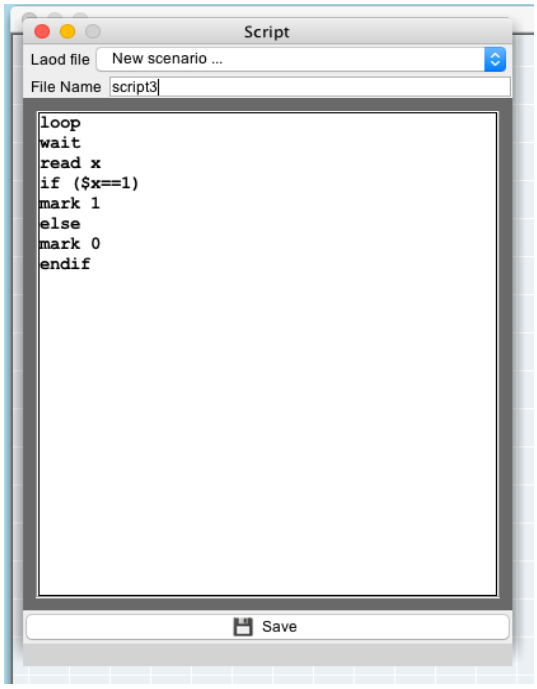

FIGURE 4.13 – Script de coordinateur

4.4.5 Affectation des scripts

Attribuer le script de chaque capteur.

Sélectionnez un capteur, puis cliquez sur Device parameters dans le menu Nodes.

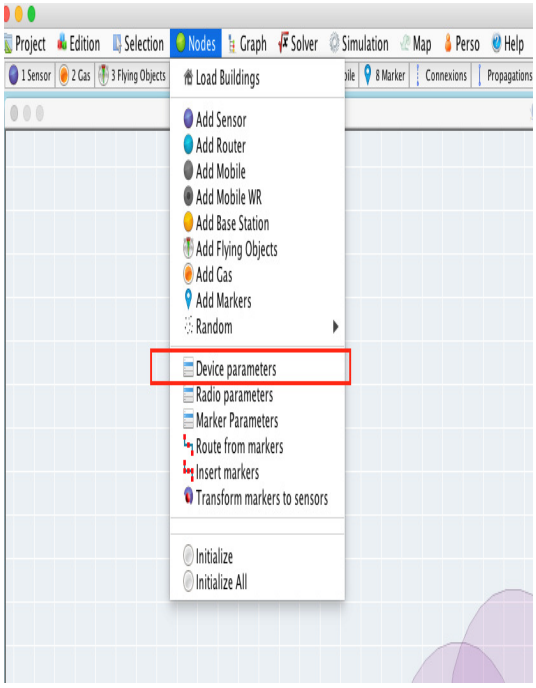

FIGURE 4.14 – Menu Nodes

Une fois la fenêtre Device parameters est ouvert, cliquez sur Entrée pour afficher ses paramètres.

Sélectionnez le script correspondant dans la liste du fichier de script, puis cliquez sur le bouton (flèche bleue) dans la partie droite.

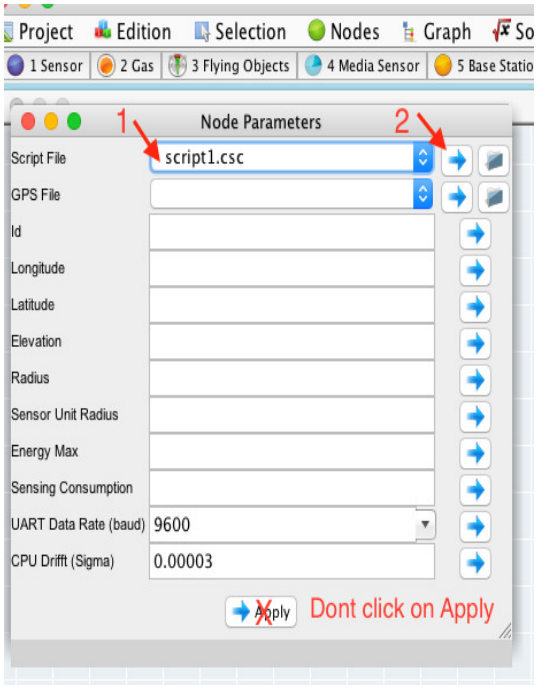

FIGURE 4.15 – La fenêtre Device parameters

4.4.6 Simulation [40]

Cliquez sur le menu simulation puis cliquez sur WSN Simulation pour obtenir la fenêtre suivante :

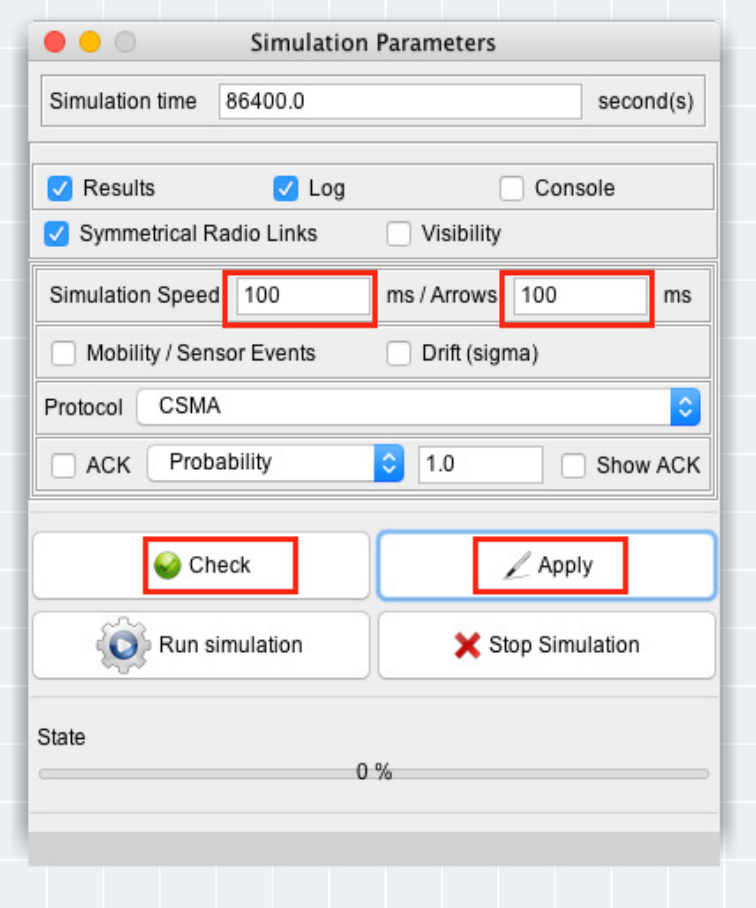

FIGURE 4.16 – La Fenêtre de Simulation

Cliquez sur le bouton Check, le message Ready to simulate doit être obtenu. S'il n'est pas le cas, vérifiez que tous les capteurs ont leur script. Cela peut être fait en vérifiant également le centre de chaque nœud de capteur. Si elle est rouge, cela signifie qu'un script est affecté. En sélectionnant un nœud et en cliquant sur r, vous pouvez obtenir des informations supplémentaires.

Cliquez sur le bouton Apply. Il est possible de cliquer sur le bouton de simulation de course, mais la visualisation n'est pas optimale. Pour obtenir la meilleure visualisation, après avoir cliqué sur Apply, fermez la fenêtre Simulation. Ensuite, cliquez sur le bouton de course sur la barre d'outils (le bouton avec le triangle vert). Pour arrêter la simulation, cliquez sur le bouton avec le rectangle rouge.

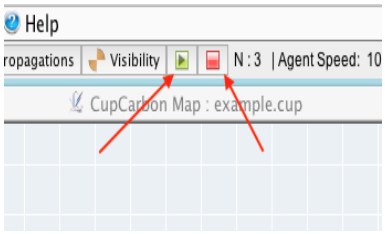

FIGURE 4.17 – Le bouton de simulation

4.4.7 Les résultats de la simulation

Le dossier du projet est composé des dossiers suivants :

- Config : Il contient des informations à propos du projet (les objets contenus).
- gps : Il contient les chemins créés pour les nœuds mobiles.
- logs : il contient le dossier de notation (tous les détails de la simulation).
- Results : il contient le dossier wisen_simulation.csv (ou wisen_simulation_mob.csv dans le cas de la simulation avec mobilité) qui contient les résultats en termes d'énergie consommée pour chaque nœud capteur. Le type du dossier est un csv qui peut être ouvert par l'Excel afin de pouvoir tracer des graphes.
- Scripts : Il contient les codes de SenScript qui sont créés par l'utilisateur.

FIGURE 4.18 – Le dossier du projet

4.5 Simulation du protocole de routage LEACH

4.5.1 Principe de fonctionnement du protocole Leach

LEACH (Low-Energy Adaptive Clustering Hierarchy) est un protocole de routage hiérarchique, employant un procédé de regroupement qui divise le réseau en deux niveaux : cluster-heads et les nœuds. Le protocole se déroule en rounds.

LEACH arrange les nœuds dans le réseau en petits groupes et choisit l'un d'entre eux comme cluster-head. Le nœud détecte sa cible et envoie ensuite les informations pertinentes à son cluster-head. Ensuite, le cluster-head comprime les informations reçues de tous les nœuds et les envoie à la station de base. Les nœuds sélectionnés en tant que cluster-head drainent plus d'énergie par rapport aux autres nœuds, comme il est nécessaire d'envoyer des données à la station de base qui peut être situé loin. Ainsi LEACH utilise la rotation aléatoire des nœuds nécessaires aux cluster-heads pour répartir uniformément la consommation d'énergie dans le réseau.

Les opérations LEACH peuvent être divisées en deux phases :

Phase de construction :

Le but de cette phase est la construction des clusters en choisissant les chefs et en établissant la politique d'accès au média au sein de chaque groupe. Cette phase commence par la prise de décision locale pour devenir cluster-head. Chaque nœud n choisit un nombre aléatoire, si ce nombre est inférieur à une valeur T (n), le nœud devient cluster-head. T (n) est défini comme suit :

$$T(n) = \begin{cases} \frac{P}{1-P\times(r \mod \frac{1}{p})} & \text{si } n \in G \\ 0 & \text{sinon} \end{cases} \tag{4.1}$$

- P : pourcentage désiré de cluster-heads pendant un round.
- r : numéro du round.
- G : l'ensemble des nœuds qui n'ont pas été élu cluster-heads pendant les 1/P rounds précédents.

Par la suite, chaque nœud qui s'est élu cluster-head émet un message de notification. Les nœuds membres récoltent les messages de notification, et décident leur appartenance à un cluster. La décision est basée sur l'amplitude du signal reçu : le cluster-head ayant le signal le plus fort est choisi (i.e. le plus proche). En cas d'égalité, un chef aléatoire est choisi. Chaque membre informe son chef de sa décision. Toutes les communications précédentes étant faite dans une topologie plate, la méthode CSMA doit être employée. Par la suite, les communications au sein d'un cluster peuvent être faites avec la méthode TDMA. Pour cela, chaque chef établie un Schedule TDMA pour ses membres, en indiquant pour chaque nœud son slot d'émission. Ce Schedule est envoyé aux membres.

Phase de communication :

En utilisant le Schedule TDMA, les membres émettent leurs données captées pendant leurs propres slots. Cela leur permet d'éteindre leur interface de communication en dehors de leurs slots réservés, afin d'économiser leur énergie. Ces informations sont

ensuite agrégées, pour être transmises au collecteur (sink). Cette communication, entre un cluster-head et le collecteur, se fait d'une manière directe, i.e. : le cluster-head adapte son émetteur radio afin d'atteindre directement le collecteur.

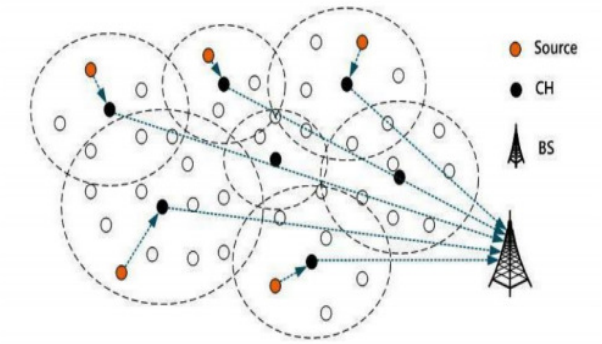

FIGURE 4.19 – Architecture du routage hiérarchique LEACH

4.5.2 Simulation

Comme nous l'avons expliqué précédemment le protocole Leach divise le réseau en deux groupes, les clusters-Head et les nœuds. Donc la première étape du protocole consiste à élire les clusters-Head. La figure ci-dessous montre les clusters-Head choisis pour le premier round.

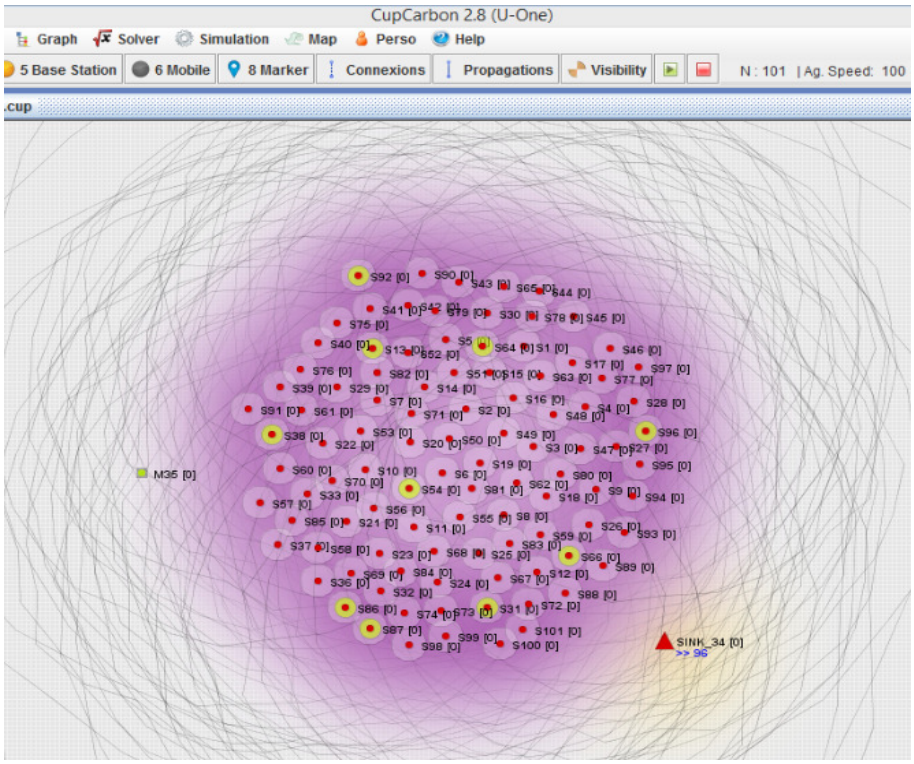

FIGURE 4.20 – Election des cluster-Head

Ensuite les cluster-Head émettent des messages de notification aux nœuds voisins afin de déterminer les membres du cluster. La figure ci-dessous montre cette étape.

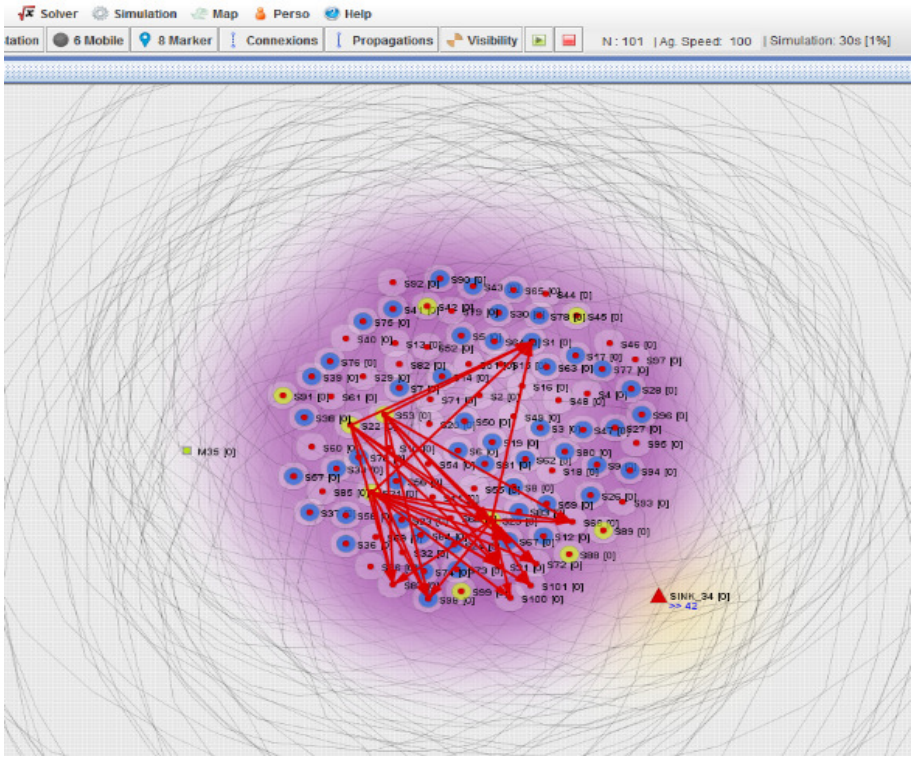

FIGURE 4.21 – Message de notification envoyé par le cluster-Head

A la réception du message de notification par les nœuds, ces derniers décident selon la puissance du signal d'amplitude du message leur appartenance à un cluster.

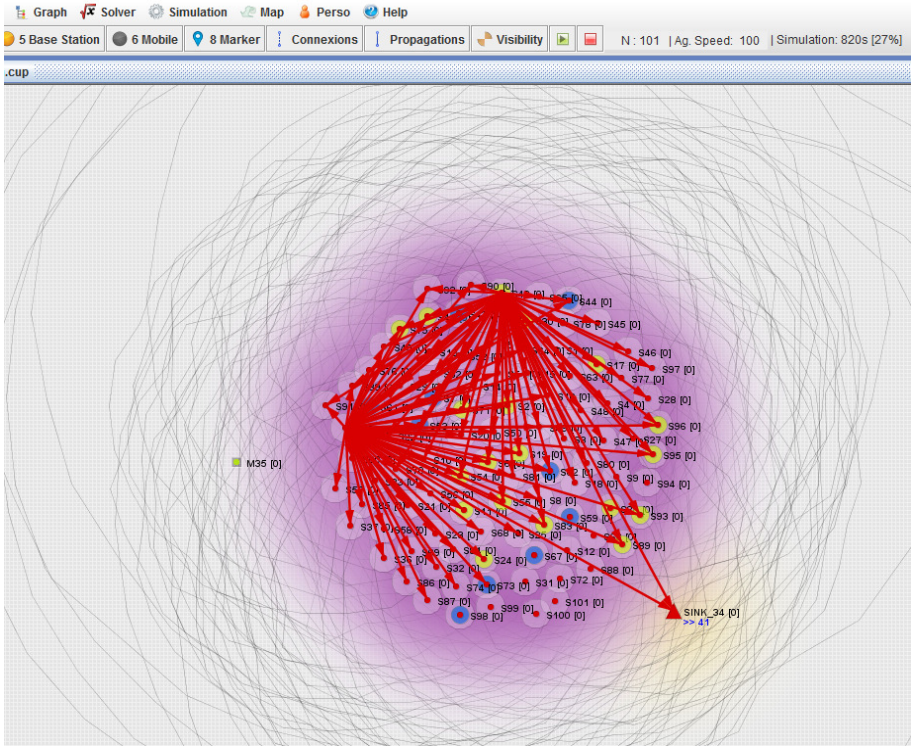

FIGURE 4.22 – Message d'appartenance à un cluster

Ensuite le cluster-Head établie un Schedule TDMA pour les membres de son cluster en indiquant pour chaque nœud son flot d'émission.

En fin vient l'étape de communication entre le cluster-Head et le collecteur(Sink), et cela se fait par l'envoie des données captées par les nœuds au cluster-Head qui va ensuite les envoyer à son tour à la station de base. La figure ci-dessous montre cette étape.

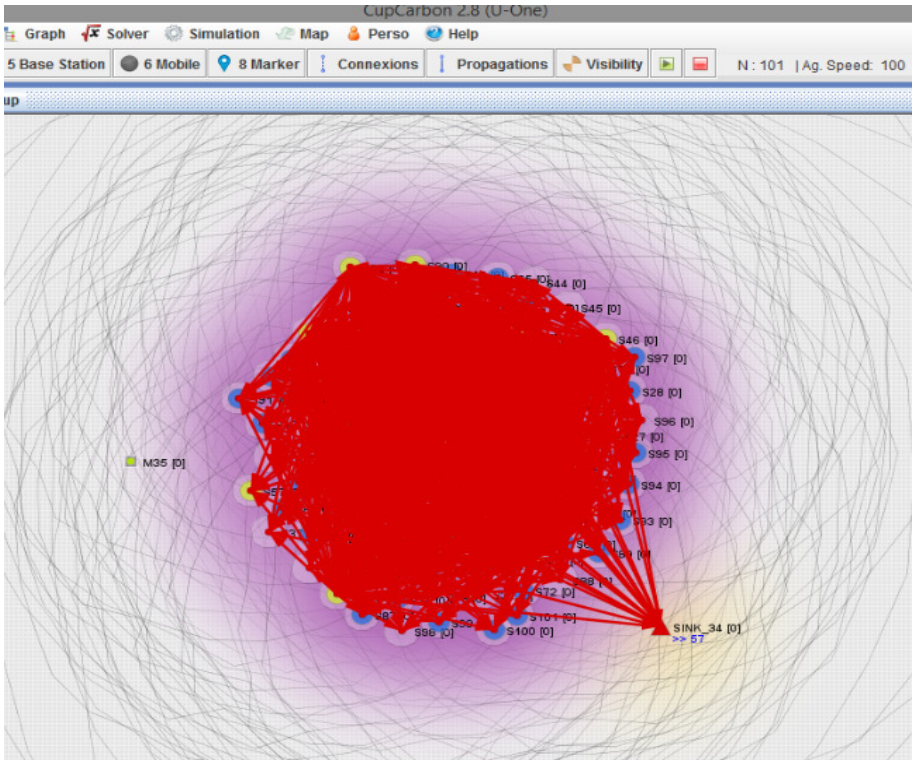

FIGURE 4.23 – Communication entre les cluster-Head et la station de base

4.5.3 Interprétation des résultats de la simulation de LEACH

D'après la simulation on constate que le protocole LEACH minimise le nombre de messages échangés entre les nœuds et la station de base. Le graphe ci-dessous montre le nombre de messages échangés on augmentant le nombre de nœuds déployés.

- **l'impacte du nombre de nœuds sur le nombre de messages échangés**

  le graphe suivant montre l'impacte du nombre de nœuds sur le nombre de messages échangés dans un RCSF.

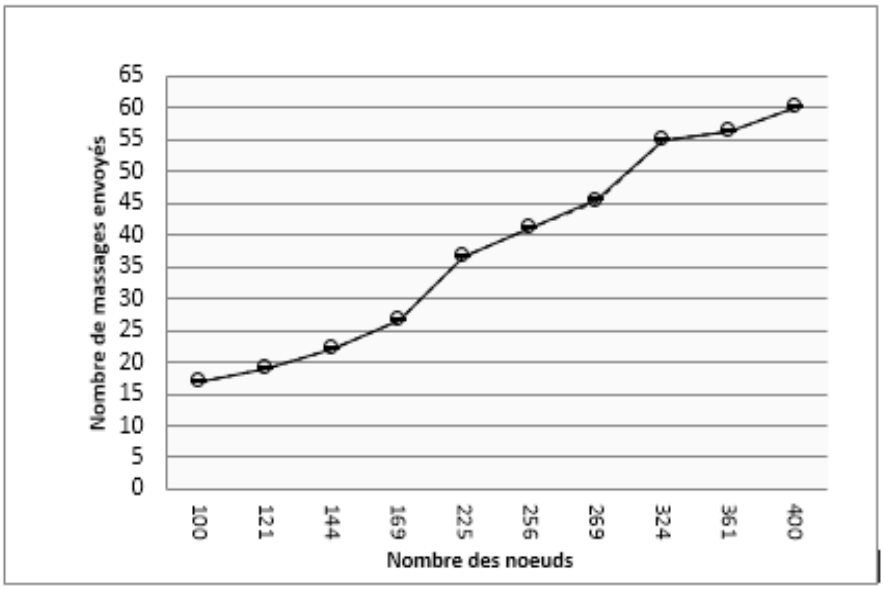

FIGURE 4.24 – Nombre de messages échangés

- **La consommation d'énergie**

Le graphe suivant représente la consommation d'énergie d'un nœud capteur actif pendant une période de temps.

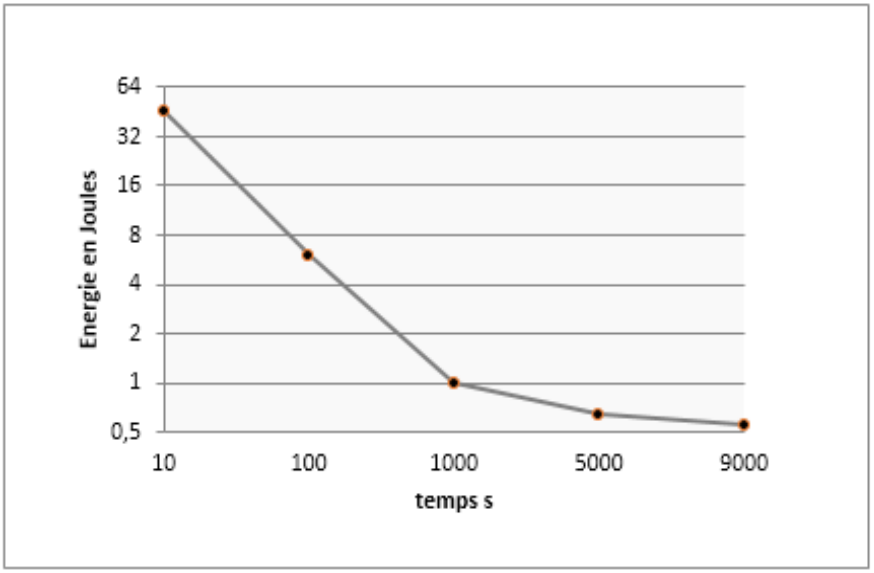

FIGURE 4.25 – Consommation d'énergie d'un capteur pendant une période de temps

Conclusion

Nous avons présenté dans ce chapitre le simulateur des réseaux de capteurs sans fils, Cupcarbon. Parmis ses avantages, Cupcarbon permit de concevoir les réseaux de capteurs sans fils grâce au module OpenStreetMap, ces réseaux peuvent inclure les capteurs et d'autres éléments tels que les mobiles, le gaz, les incendies... etc, qui sont faciles à utiliser et à configurer, en plus de ça, Cupcarbon peut exécuter deux types de simulations : La simulation multi-agent est utilisée pour paralléliser le comportement de chaque capteur pour le rendre indépendant, et La simulation à événement discret est utilisée pour simuler les communications entre agents et en particulier entre les capteurs. L'objectif principal de ce simulateur est éducatif. Il démontrera l'utilisation des capteurs sans fil dans presque les mêmes conditions que dans le monde réel. Le simulateur peut également être utilisé pour calculer le diagramme d'énergie de chaque capteur.

Conclusion Générale

Les Réseaux de Capteurs Sans Fil (RCSFs) sont une nouvelle technologie qui a surgit après les grands progrès technologiques concernant le développement des capteurs intelligents, des processeurs puissants et des protocoles de communications sans fil. Ce type de réseau composé de certaines ou de milliers d'éléments, a pour but la collecte de données de l'environnement, leur traitement et leur dissémination vers le monde extérieur.

Des propriétés comme flexibilité, hautes capacités de captage, cout réduit, installation rapide sont les caractéristiques qui ont permis à cette nouvelle technologie d'avoir de nouveaux domaines d'applications multiples. Ce large étendu d'applications fera de cette technologie émergeante une partie intégrale de nos vies futures.

Cependant, ces réseaux sont limités dans leurs performances et leurs interactions avec l'environnement et ils posent plusieurs défis à relever, notamment le défi du routage. En effet, plusieurs protocoles de routage ont été proposés dans la littérature afin d'améliorer les performances des RCSFs et de répondre aux besoins des applications considérées. En outre, plusieurs classifications de ces protocoles ont été proposées. Dans ce contexte, plusieurs outils de simulation ont été développés pour évaluer les performances de ses nouveaux protocoles.

Nous nous sommes intéressés dans ce mémoire à la simulation des protocoles de routage dans les RCSFs. Notre objectif étant de déterminer le simulateur le mieux adapté aux RCSFs. Pour cela, nous avons d'abord présenté un état de l'art détaillé sur les réseaux de capteurs sans fil où nous avons étudié plusieurs aspects liés à ses réseaux, ce qui nous a permis de relever les principales caractéristiques et contraintes des RCSFs. Par la suite, nous avons

présenté la problématique du routage dans les RCSFs. En passant en revue les considérations de conception d'un protocole de routage et la classification des approches de routages dans les RCSFs. Nous avons également dressé un état de l'art des différents outils de simulation des RCSFs tout en se focalisant sur les avantages et inconvénients de chacun d'eux. Enfin, nous avons présenté le nouveau simulateur de réseaux de capteur sans fil Cupcarbon. Par ailleurs, afin de voir de plus près le fonctionnement de ce simulateur, nous avons exécuté le protocole LEACH dans ce dernier.

Nous avons constaté que le simulateur Cupcarbon est simple à utiliser, il permet de concevoir les RCSFs grâce au module OpenStreetMap, ces réseaux peuvent inclure les capteurs et d'autres éléments tels que les mobiles, le gaz, les incendies. . . etc, qui sont faciles à utiliser et à configurer,en plus de ça, Cupcarbon peut exécuter deux types de simulations : La simulation multi-agent est utilisée pour paralléliser le comportement de chaque capteur pour le rendre indépendant, et La simulation à événement discret est utilisée pour simuler les communications entre agents et en particulier entre les capteurs. Néanmoins, ce nouveau simulateur est toujours en cours de développement. Ce qui signifie qu'il y'a des modules incomplets, principalement au niveau du routage étant donné qu'il existe un seul protocole de routage implémenté (LEACH), ce qui limite son utilisation.

Bibliographie

[1] Y. YASER, *Routage pour la gestion de l'énergie dans les réseaux de capteurs sans fil.* PhD thesis, Université de Haute Alsace Faculté des Sciences et Techniques, 2011.

[2] G. PUJOLLE, *Les réseaux.* Eyrolles, 2008.

[3] L. KHELLADI and N. BADACHE, *Réseaux de capteurs : Etat de l'art.* Rapport de recherche, Laboratoire LSI USTHB, 2004.

[4] I. AKYILDIZ, W. SU, Y. SANKARASUBRAMANIAM, and E. CAYIRCI, *Wireless sensor networks : A Survey.* Georgia Institute of Technology, 2011.

[5] BERKLEY, "Brainy buildings." `http://coe.berkeley.edu/labnotes/0701brainybuildings.html`. Connecté ; dernière visite le 02 Mars 2016.

[6] ACADEMIA, "état de l'art sur les réseaux de capteurs sans fils." `www.academia.edu/`. Connecté ; dernière visite le 02 Mars 2016.

[7] W. BECHKIT, *Un nouveau protocole de routage avec conservation d'énergie dans les réseaux de capteurs sans fil.* Mémoire pour l'obtention du diplôme d'ingénieur d'état en informatique, École nationale Supérieure d'Informatique, 2009.

[8] I. AKYILDIZ, W.SU, Y.SANKARASUBRAMANIAM, and E.CAYIRCI, *A Survey on Sensor Networks.* Georgia Institute of Technology, 2002.

[9] M. ILYAS and I. MAHGOUB, *Handbook of sensor networks : compact wireless and wired sensing systems.* CRC Press, 2004.

[10] J. AL-KARAKI and A. KAMAL, *Routing Techniques in Wireless Sensor Networks : A Survey.* IEEE wireless communication, 2004.

[11] N. LASLA, *La gestion de clés dans les réseaux de capteurs sans-fil.* Mémoire de magistère, Institut National de formation en Informatique INI, Algérie, 2007.

[12] D. NICULESCU, *Topics In Ad-Hoc Networks : Communication Paradigms for Sensor Networks.* NEC Laboratories America, IEEE Communications Magazine, Mars 2005.

[13] C. INTANAGONWIWAT, R. GOVINDAN, and D. ESTRIN, *Directed Diffusion : a scalable and robust communication paradigm for sensor networks.* In MOBICOM, 2000.

[14] N.BULUSU, J.HEIDEMANN, and D.ESTRIN, *GPS-less Low Cost Outdoor Localization For Very Small Devices.* University of Southern California / Information Sciences Institute Marina del Rey, October 2000.

[15] Y. XU, J. HEIDEMANN, and D. ESTRIN, *Geography informed Energy Conservation for Ad Hoc Routing.* 2001.

[16] Y.YU, R.GOVINDAN, and D.ESTRIN, *Geographical and Energy Aware Routing : A Recursive Data Dissemination Protocol for Wireless Sensor Networks.* 2001.

[17] C. INTANAGONWIWAT, R. GOVINDAN, and D. ESTRIN, *Directed Diffusion : a scalable and robust communication paradigm for sensor networks.* In MOBICOM, 2000.

[18] W. HEINZELMAN, A. CHANDRAKASAN, and H.BALAKRISHNAN, *Energy-Efficient Communication Protocol for Wireless Microsensor Networks.* Proceedings of the 33rd Hawaii International Conference on System Sciences, 2000.

[19] D. BRAGINSKY and D. ESTRIN, *Rumor Routing Algorithm for Sensor Networks.* 1st Workshop on Sensor Networks and Apps, Atlanta, GA, 2002.

[20] W. HEIZELMAN, J. KULIK, and H. BALAKRISHNAN, *Adaptive Protocols for Information Dissemination in Wireless Sensor Networks.* ACM MobiCom '99, Seattle, WA, 1999.

[21] T. HE, J.A.STANKOVIC, C. LU, and T. ABDELZAHER, *SPEED : A Stateless Protocol for Real-Time Communication in Sensor Networks.* International Conference on Distributed Computing Systems, 2003.

[22] S. BOULFEKHAR, *Approche de minimisation de l'énergie dans les réseaux de capteurs.* Mémoire pour l'obtention du grade de magistère en informatique, Université de A/Mira Bejaïa, 2006.

[23] B. ASMA, *Routage multi chemin avec Qos dans les réseaux de capteur sans fil.* Université de A/Mira Bejaïa, 2013.

[24] P.-J. ERARD and P. DÉGUÉNON, *Simulation par évènements discrets.* presses polytechnique et universitaire Romandes, 1996.

[25] B. KARIMA, *Les simulateurs réseaux Technologie réseau.* PhD thesis, Université de Bejaia Faculté des Sciences et Techniques, 2013/2014.

[26] SUNDANI, LI, DEVABHAKTUNI, ALAM, and B. and, *Wireless sensor network simulators a survey and comparisons.* International Journal of Computer Networks, 2011.

[27] E. LOPEZ, V. ALONSO, M. SALA, P. MARINO, and GARCUA-HARO, *Simulation tools for wireless sensor networks.* InProceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems, 2005 July.

[28] P. ANELLI and E. HORLAIT, *NS-2 : Principes de conception et d'utilisation.*

[29] N. BOUZIANE, "Les simulateurs 2013." `www.usthb.dz/`. Connecté ; dernière visite le 15 Avril 2016.

[30] "Prowler network simulator." `http://www.isis.vanderbilt.edu/`. Connecté ; dernière visite le 20 Avril 2016.

[31] P. LEVIS and N. LEE, *TOSSIM : A Simulator for TinyOS Networks.* 17 September 2003.

[32] F. ABDELFATAH, *Développement d'une bibliothèque de capteur sans fil.* université Montpellier 2, avril 2008.

[33] S. SUNDRESH, W. KIM, and G. AGHA, *SENS : A Sensor Network Environment.* université Montpellier 2, avril 2008.

[34] J. NUEVO, *A Comprehensible GloMoSim, Tutorial Contents.* Universite du Quebec, 2004.

[35] M. TAKAI, J. MARTIN, R. MEYER, B. PARK, and H. Y. SONG, *PARSEC User Manual.* Parallel Computing, 1999.

[36] G. PUJOLLE, *Cours Reseaux et Telecoms.* Eyrolles, Sept 2009.

[37] *QualNet 5.0 Programmer's Guide Scalable Network Technologies.* S.N. Technologies, octobre 2008.

[38] R. B. REESE, "A zigbee-subset/ieee 802.15.4 multi-platform protocol stack v0.2.6." `http://www.ecemsstate.edu/`. Connecté ; dernière visite le 25 Avril 2016.

[39] K. MEHDI, M. LOUNIS, A. BOUNCEUR, and T. KECHADI, *Cupcarbon : A multi-agent and discrete event wireless sensor network design and simulation tool.* IEEE 7th International Conference on Simulation Tools and Techniques (SIMUTools'14), Lisbon, Portugal, March 17-19 2014.

[40] `http://www.cupcarbon.com`. Connecté ; dernière visite le 12 avril 2016.

[41] M. LOUNIS, K. MEHDI, and A. BOUNCEUR, *A cupcarbon tool for simulating destructive insect movements. 1st IEEE International Conference on Information and Communication Technologies for Disaster Management.* (ICT-DM'14), Algiers, Algeria, March 24-25 2014.

[42] N. SAADI, A. BOUNCEUR, and B. POTTIER, *Modélisation et simulation pour la résolution du problème de couverture de cibles mobiles dans un réseau de capteurs sans fil.* 8ème Colloque du GDR SoC-SiP, Paris, France, 11-13 Juin 2014.

Résumé

Ces dernières années, il est devenu nécessaire, d'observer et de contrôler certains phénomènes physiques. Ceci a été rendu possible grâce à l'apparition des Réseaux de Capteurs Sans Fils (RCSFs). Ces derniers suscitent un grand intérêt vu les nombreux avantages qu'ils apportent mais souffrent, néanmoins, de plusieurs contraintes liées aux caractéristiques inhérentes à ce type de réseau. Par instance, il est impératif de mettre en place un protocole de routage efficace en énergie et qui prenne en compte les contraintes imposées par ces réseaux. Dans ce contexte, de nombreux protocole de routage pour les RCSFs ont vu le jour. Par conséquent, de nouveaux outils de simulation sont nécessaires pour évaluer les performances de ses nouveaux protocoles.

Notre travail vise donc à comparer différents simulateurs présentés dans la littérature, afin d'exposer leurs avantages et leurs inconvénients. Ceci pour savoir lequel est le mieux adapté au routage dans les RCSFs. Un intérêt particulier a été porté sur le nouveau simulateur Cupcarbon, qui est une plate-forme pour la conception des villes intelligentes et des RCSFs. Il fournit des avantages qui le rendent important par rapport aux autres simulateurs classiques des RCSFs. Nous avons donc, présenté un guide d'utilisation du simulateur Cupcarbon, Allant de l'installation jusqu'à la simulation du protocole de routage LEACH.
Mots clés : RCSFs, Routage, Simulation, Cupcarbon.

Abstract

These last years, it became necessary, to observe and control some physical phenomena. This was made possible thanks to the appearance of the Wireless Sensors Networks (WSNs). These latter attracted a great interest considering the many advantages they bring but suffer, nevertheless, several constraints related to the characteristics inherent in this type of network. For instance, it is imperative to set up an energy efficient routing protocol which takes into account the constraints imposed by these networks. In this context, many routing protocol for WSNs were proposed. Consequently, new tools for simulation are necessary to evaluate the performances of these new protocols.

Our work aims at comparing various simulators presented in the literature, in order to expose their advantages and their disadvantages. This to know which is more adapted to simulate routing in WSNs. A particular interest was related to the new simulator Cupcarbon, which is a platform for the design of the intelligent cities and WSNs. It provides advantages which make it significant compared to the other traditional simulators of WSNs. We then, presented a user guide of the simulator Cupcarbon, from the installation to the simulation of the routing protocol LEACH.
Keywords : Wireless Sensors Networks, Routing, Simulator, Cupcarbon.