Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Licence en Informatique

Thème

Développement d'une application orientée surveillance pour les réseaux de capteurs sous Contiki

Réalisé par :

- **Saidi Abdessamad**
- **Mamem Wafa**

Présenté le 27 Mai 2015 devant la commission d'examination composée de MM.

- *Mr BENMAMMAR Badr* *(Président)*
- *Mr MANA Mohamed* *(Examinateur)*
- *Mr LEHSAINI Mohamed* *(Encadreur)*
- *Melle CHIKH Asma* *(Invité)*

Année universitaire : 2014-2015

Remerciements

*Nous tenons à remercier tout d'abord **ALLAH** qui ma donné toutes ces années, la santé le courage la volonté pour réalisé ce travail. Le plus grand merci lui revient de nous avoir guidées vers le droit chemin de nous avoir aidées tout le long de nos années d'études et pour toutes ses grâces qui nous entourent.*

Nous remercions vivement notre encadreur Monsieur LEHSAINI Mohamed pour avoir accepté de nous encadrer et qui a toujours été à notre écoute et disponible tout au long de notre travail malgré ses charges ainsi que pour son aide et le temps qu'il a bien voulu nous consacré sans quoi ce mémoire n'aurait jamais vu le jour.

Nous tenons à remercier les membres du jury qui ont bien voulu accepter de valoriser ce travail.

Nous remercions tout particulièrement Mademoiselle CHIKH Asma qui nous a beaucoup aidées et soutenues pour réaliser ce laborieux travail, merci pour ça patience, ses conseils et ses encouragement prodigués tout le long de ce mémoire.

Nous adressons également nos sincères remerciements à nos famille : parents, frères, sœurs de nous avoir aidé à surmonter tous les obstacles et à nous forger à travers les difficultés vécues durant toute cette période de travail et qui nous ont toujours supportées moralement et financièrement pendant toutes nos longues années d'étude.

Nous n'oublieront pas aussi nos proches amis qui nous ont vivement soutenues et encouragées au cours de la réalisation de ce mémoire et nos remerciements à toutes les personnes qui ont contribué, de près comme de loin, à la réalisation de ce modeste travail.

A la fin nous tenons à remercier tous nos collègues d'étude, particulièrement notre promotion et pour tous ceux qui ne sont pas cités, qui sont si nombreux, on vous remercie de plus profond de nous-mêmes.

Merci à tous

Dédicace

Je tiens à exprimer ma profonde reconnaissance :

A DIEU, pour m'avoir donné la force dans les moments difficiles d'éditer ce mémoire.

A mes parents :

Ma mère, qui a œuvré pour ma réussite, de par son amour, son soutien, tous les sacrifices consentis et ses précieux conseils, pour toute son assistance et sa présence dans ma vie.

Mon père, qui peut être fier et trouver ici le résultat de longues années de sacrifices et de privations pour m'aider à avancer dans la vie. Puisse Dieu faire en sorte que ce travail porte son fruit. Merci pour les valeurs nobles, l'éducation et le soutient permanent venu de toi.

A mes sœurs et mon frère : Meriem, Sarah, Salima et Abderrazak qui m'ont assisté dans ces moments difficiles et m'ont servi d'exemple.

A toute la famille « SAIDI » et MAMEM ».

A mes cousins : Nabil, Amin, Hakim, Mourad, Mehdi, Yazid et surtout mon très cher Reda qui est comme un frère pour moi.

A mes amis et amies de par le monde qui n'ont cessé de m'encourager en particulier ma collègue de présent projet « MAMEM Wafa » qui n'a jamais maqué de travailler, du respect. Qui a toujours été la pour moi, je la remercie pour ça sincère amitié et confiance, elle n'a jamais cessé de m'encourager. Je lui souhaite tous le bonheur et joie sans oublié sa très cher sœur Wassila qui je lui souhaite une vie heureuse pleine de bonheur et de joie.

A mon professeur encadreur Mr LEHSAINI pour son aide et sa précieuse attention.

A Mademoiselle CHIKH Asma pour son aide et ses conseils, je vous souhaite la réussite dans ta vie précisément ton Doctorat.

SAIDI Abdessamad

Dédicace

A cœur vaillant rien d'impossible a conscience tranquille tout est accessible, quand il y a la soif d'apprendre, tout vient à point à qui sais attendre, quand il y a le souci de réaliser un dessein, tout devient facile pour arriver à nos fins, malgré les obstacles qui s'opposent, en dépit des difficultés qui s'interposent, les études sont avant tout. Notre unique et seul about, ils représentent la lumière de notre existence, l'étoile brillante de notre réjouissance.

Aujourd'hui, ici rassemblés auprès des jurys, nous prions dieu que cette soutenance fera signe de persévérance et que nous serions enchantés par notre travail honoré.

Je dédie cette mémoire :

A ma très chère mère qui représente pour moi le symbole de la bonté par excellence, la source de tendresse et l'exemple de dévouement qui n'a pas cessé de m'encourager et de prier pour moi. Ta prière et ta bénédiction m'ont été d'un grand secours pour mener à bien mes études. Aucune dédicace ne saurait être assez éloquente pour exprimer ce que tu mérites pour tous les sacrifices que tu n'a cessé de me donner depuis ma naissance, durant mon enfance et même l'age d'adulte. Tu as fait plus qu'une mère puisse faire pour que ses enfants suivent le bon chemin dans leur vie et leur étude.

A mon cher père aucune dédicace ne saurait exprimer l'amour, l'estime, le dévouement et le respect que j'ai toujours eu pour vous, rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien être. Ce travail est le fruit de tes sacrifices que tu as consentis pour mon éducation et ma formation.

A mon très cher frère Anas Mohammed qui m'est le père et la mère, je te souhaite un avenir plein de joie, de bonheur, de réussite et de sérénité.

A ma très chère sœur Wassila tu as toujours été présente pour les bons conseils, votre affection et votre soutien m'ont été d'un grand secours au long de ma vie professionnelle et personnelle.

A ma grand-mère maternelle que dieu la protège.

A toute la famille « MAMEM », « SEBAÂ », « SAIDI ».

Je remercie mes chères tantes maternelle et paternelle : Malika et Hayet pour leur soutiens de m'avoir aidées et de m'avoir épaulées dans les moments les plus dures.

A mon binôme Abdessamad avec qui j'ai partagé des belle années d'études et avec qui j'ai eu l'honneur de les terminés avec cette mémoire et merci pour tout ce que tu as fait pour moi, tu es toujours là quand j'en ai besoin, des garçons comme toi j'en connais pas tu es pour moi comme un frère.

A mon encadreur Monsieur LEHSAINI pour son excellence, de son accompagnement et la confiance qu'il nous a accordé.

A mademoiselle CHIKH Asma pour son aide et ses conseils, je vous souhaite la réussite pour ton Doctorat.

Sans oublier mes définîtes grand-mère paternelle et ma tante Chahida, j'aurai aimé qu'elles soient à côté de moi en ce moment là pour partager ma joie et ressentir les même émotions que moi parce qu'elles m'aimaient. Que Dieu les accueillent dans son vaste paradis.

A tous ceux qui ne sont chers ceux qui m'aiment et ceux que j'aime.

MAMEM Wafa

Table des matières

Liste des figures

Liste des tableaux

Introduction Générale

Introduction générale

Les progrès récents dans les domaines de la micro-électronique, les communications sans fil, et l'électronique numérique ont donné naissance à des dispositifs miniaturisés de faible coût, de faible puissance, dotés de capteurs capables de capturer différentes grandeurs environnementales et physiologiques [1]. Ces dispositifs sont appelés nœuds capteurs. Ils sont capables de collaborer ensemble pour former un réseau de capteurs sans fil.

Les réseaux de capteurs sans fil (Wireless Sensor Networks) sont des réseaux spontanés, constitués d'un grand nombre de nœuds déployés dans une zone d'intérêt en vue de collecter et de transmettre des données vers un ou plusieurs points de collecte appelé "puits" ou "station de base". Ces nœuds composant le réseau sont caractérisés généralement par une faible capacité de calcul, une mémoire dont la capacité de stockage est limitée à quelques koctets, une source d'énergie réduite (batterie), et un débit limité à quelques kbits/. Ils sont dispersés aléatoirement à travers une zone géographique, appelée "champ de captage" qui définit le terrain d'intérêt pour le phénomène observé. Les données collectées sont acheminées directement ou via un schéma de routage multi-sauts à un centre de contrôle distant. Ce dernier peut-être connecté à l'utilisateur du réseau via internet ou par satellite. Ainsi, l'usager peut adresser des requêtes aux autres nœuds du réseau, précisant le type de données requises et récolter les données environnementales captées par le biais du nœud puits.

Le champ d'applications des réseaux de capteurs est de plus en plus élargi grâce aux évolutions techniques que connaissent les domaines de l'électronique et des communications. Parmi ces évolutions, nous citons la miniaturisation des nœuds capteurs, le faible coût des capteurs, l'élargissement des gammes de capteurs disponibles (mouvement, température, physiologique…) et l'évolution des technologies des communications sans fil telles que Bluetooth [2] et Zigbee [3]. Cette évolution a permis aux réseaux de capteurs sans fil d'envahir une gamme variée de domaines d'applications telles que le domaine militaire, le domaine de la médecine, le domaine environnemental, le secteur industriel, etc.

Dans le cadre de notre projet, nous allons développer une application pour la mise en place d'un réseau de capteurs sous Contiki sur une plateforme de capteurs réels (TelosB) [4]. Le choix de la plateforme TelosB se justifie par la popularité de ce type de capteurs ainsi que la facilité de leur programmation. Cette application permet la collecte des grandeurs

environnementales selon deux scénarios. Dans le premier scénario, les capteurs envoient directement les données collectées à la station de base alors que dans le deuxième scénario un nœud relais est impliqué, auquel les nœuds envoient leurs données collectées et ce dernier les relaye vers la station de base.

Le manuscrit s'articule autour de trois chapitres :

Le premier chapitre de ce manuscrit présente des généralités sur les réseaux de capteurs sans fil, leurs caractéristiques et leurs domaines d'applications.

Le deuxième chapitre décrit les outils logiciels et matériels nécessaires au développement de l'application orientée surveillance.

Le troisième chapitre exploite les outils présentés dans le deuxième chapitre pour mettre en place l'application qui fait l'objet de projet de fin d'études.

Ce travail est terminé par une conclusion générale dans laquelle nous synthétisons l'objet de ce projet de fin d'études et nous présentons les références bibliographiques qui nous ont aidés à amener ce travail à une bonne fin.

Chapitre 1

Généralités sur les réseaux de capteurs sans fil

Chapitre 1
Généralités sur les réseaux de capteurs sans fil

1.1 Introduction

Les réseaux de capteurs sans fil (RCSF) se composent généralement d'un grand nombre de petits dispositifs, qui communiquent entre eux via des liens radio de faible portée pour le partage d'information et le traitement coopératif [5]. Ces dispositifs appelés nœuds capteurs, sont déployés selon des positions prédéterminées ou aléatoirement dans une zone d'intérêt pour surveiller des phénomènes divers. Après le déploiement initial, les nœuds capteurs s'auto-organisent en une infrastructure réseau appropriée, souvent en mode multi-sauts. Les données collectées par ces nœuds capteurs sont acheminées directement ou via un schéma de routage multi-sauts à un nœud considéré comme "point de collecte", appelé station de base. Cette dernière peut être connectée à une machine puissante via internet ou par satellite. En outre, les nœuds capteurs peuvent être interrogés par des requêtes à distance ou remontent périodiquement des données au centre contrôle et dans les applications sensibles les nœuds capteurs envoient l'information que si un événement pertinent survient.

Dans ce chapitre, nous présentons les RCSF, leurs caractéristiques et leurs domaines d'applications.

1.2 Qu'est-ce qu'un nœud capteur ?

Un nœud capteur sans fil est un petit dispositif électronique doté d'un ou plusieurs capteurs qui sont capables de mesurer une valeur physique environnementale (température, lumière, pression, etc.) ou physiologique (glycémie, tension, etc.), et de la communiquer à un centre de contrôle via une station de base.

Un capteur est capable de transformer une grandeur physique observée (température, humidité, etc.) en une grandeur utilisable (intensité électrique, position d'un flotteur). Il

possède au moins un transducteur dont le rôle est de convertir une grandeur physique en une autre tel qu'ADC[1].

Par ailleurs, grâce aux avancées technologiques, principalement dans le domaine de la miniaturisation, les capteurs sont devenus des éléments de très petite taille. Ils sont ainsi dotés de moyens leur permettant : de stocker les résultats de leurs observations, d'effectuer un certain nombre de traitements sur ces résultats et de les transmettre au monde réel via une communication sans fil.

1.2.1 Le capteur intelligent

Le terme capteur intelligent (smart sensor ou intelligent sensor) a été utilisé dans l'industrie des capteurs pour distinguer les capteurs qui ne fournissent pas seulement des mesures mais aussi une fonctionnalité aux mesures spécifiques. Par rapport, à un capteur classique, un capteur intelligent intègre de nombreux éléments électroniques additionnels, ainsi que des unités programmables et des aspects logiciels nécessaires au traitement de données, aux calculs et à la communication numérique. Il est donc caractérisé par sa capacité à effectuer une collecte de mesures, les traiter et les communiquer au monde extérieur. La figure illustre un exemple de capteur intelligent.

[1] ADC : Analog to Digital Converter (convertisseur analogique numérique)

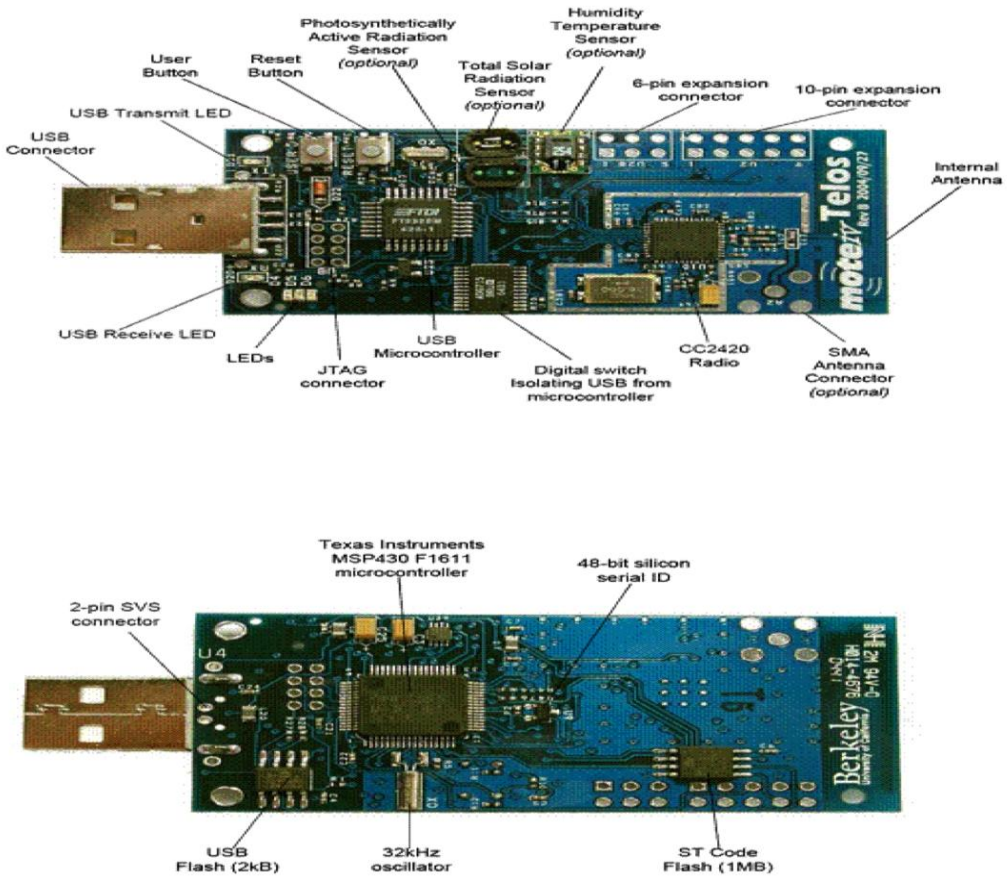

Figure 1: Anatomie d'un capteur TelosB

1.2.2 Les composants d'un nœud capteur

Un nœud capteur est composé de quatre unités de base [1] :

- **L'unité d'acquisition** : est composée d'un ou plusieurs capteurs et d'un convertisseur Analogique/Numérique (ADC). Les capteurs obtiennent des mesures sur les paramètres environnementaux sous forme analogique et l'ADC convertit ces données analogiques en données numériques compréhensibles par l'unité de traitement.
  - **L'unité de communication :** cette unité est responsable de toutes les émissions et les réceptions de données via un support de communication sans fil et une antenne.
  - **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 qui supporte un système d'exploitation spécifique tel que Contiki [6] et Tinyos [7]. Elle acquiert les informations en provenance de l'unité d'acquisition et les envoie à l'unité de transmission son rôle et de contrôler le bon fonctionnement des autre unités sur certains capteurs. Elle peut embarquer un système

d'exploitation pour faire fonctionner le nœud capteur. Elle peut aussi être couplée à une unité de stockage.

- **L'unité d'énergie :** un capteur est muni d'une source d'énergie, généralement une batterie, pour alimenter ses composants. Les batteries utilisées peuvent d'être rechargeables par l'énergie solaire ou non dans le cas où les capteurs sont déployés à l'abri du soleil comme dans les deux pôles (sud et nord). Souvent, dans les environnements hostiles il est impossible de recharger ou changer une batterie. Pour cela, l'énergie est considérée comme une ressource très précieuse dans les RCSF puisqu'elle influe directement sur la durée de vie des capteurs et donc d'un réseau de capteurs.

Un nœud capteur peut contenir également, suivant son domaine d'application, des modules supplémentaires tels qu'un système de localisation (GPS[2]), ou bien un système générateur d'énergie (cellule solaire). On peut même trouver des micro-capteurs, un peu plus volumineux, dotés d'un système mobilisateur chargé de déplacer le micro-capteur en cas de nécessité comme les micro-capteurs déplacés par des drones.

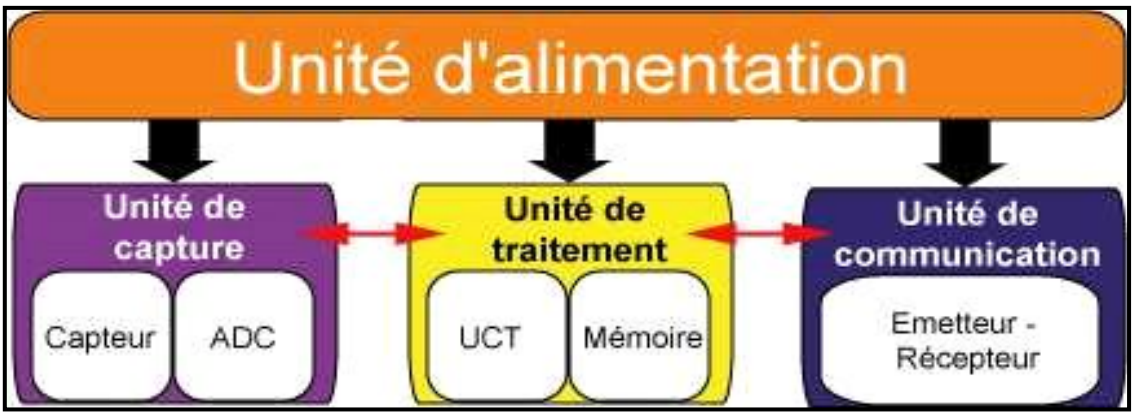

Figure 2: Architecture d'un nœud capteur

1.3 Les Réseaux de Capteurs Sans Fil (RCSF)

Un RCSF est constitué généralement d'un grand nombre de nœuds capteurs car ces derniers sont sujets à des pannes accidentelles ou intentionnelles. Ces nœuds communiquent entre eux selon une certaine topologie de réseau afin d'acheminer les informations à un centre de contrôle distant de la zone de leur déploiement. La mise en place d'un RCSF est soumise à plusieurs contraintes parmi lesquelles la conservation de l'énergie des nœuds capteurs, la

[2] GPS : Global Positioning System

fidélité de monitoring et la fidélité de routage. Néanmoins, la conservation de l'énergie et la tolérance aux pannes restent parmi les grands défis des RCSF. Dans cette optique plusieurs contributions ont été proposées dans la littérature. Ces contributions visent à minimiser la consommation d'énergie afin d'optimiser l'autonomie des nœuds qui constituent le réseau par suite garantir une longue durée de vie de réseau et assurer la fidélité de routage c'est-à-dire garantir l'acheminement de l'information vers la station de station à n'importe quel moment au cours du fonctionnement du réseau. La figure 3 résume un fonctionnement simple d'un RCSF.

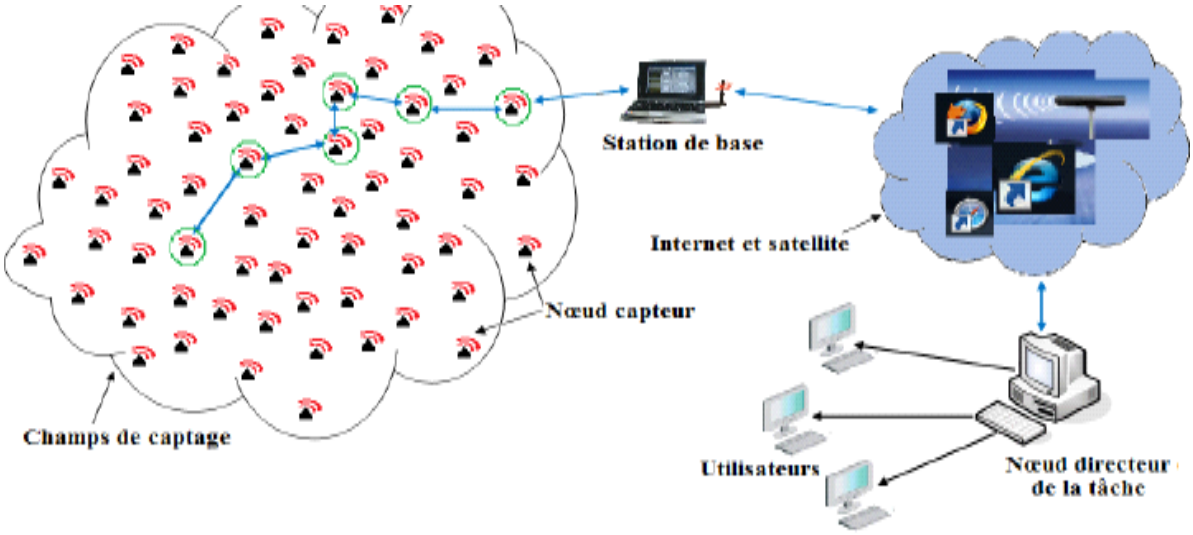

Figure 3: Exemple d'un réseau de capteurs

1.3.1 La topologie d'un réseau de capteurs

Un RCSF est composé d'un ensemble de nœuds capteurs qui s'occupent de collecter les données des capteurs et de les transmettre à l'utilisateur via l'internet ou par satellite. L'acheminement des données dans les RCSF se fait selon plusieurs schémas de routage qui sont sous-jacents à des topologies. Il existe plusieurs topologies pour les réseaux de capteurs :

- **Topologie en étoile :** la topologie en étoile est un système uni-saut c'est-à-dire les nœuds terminaux communiquent directement avec le nœud coordinateur (station de base). Cette topologie est simple et elle demande une faible consommation d'énergie, mais la vulnérabilité de la station base pourrait bloquer tout le réseau. En outre, cette topologie exige que la distance entre les nœuds et la station de base soit limitée.
- **Topologie en toile :** la topologie en toile est un système multi-sauts. La communication entre les nœuds et la station de base pourrait être directe ou via un schéma de routage

multi-sauts. Chaque nœud a plusieurs chemins pour envoyer des données. Cette topologie permet le passage à l'échelle et la tolérance aux pannes, mais elle demande une consommation d'énergie plus importante.

- **Topologie hybride :** la topologie hybride est une combinaison des deux topologies (étoile et toile). Les stations de base forment une topologie en toile et les nœuds autour d'elles sont en topologie étoile. Elle assure la minimisation d'énergie dans les réseaux de capteurs.

1.3.2 Les types de RCSF

Les RCSF peuvent être classifiés selon la manière de collecte de l'information

- **RCSF de poursuite** : Ces réseaux sont généralement développés par l'armée, ils peuvent servir à surveiller toutes les activités d'une zone stratégique ou d'accès difficile, ainsi on pourra détecter les agents chimiques, biologiques ou des radiations avant des troupes. On peut aussi penser à des capteurs embarqués sur les soldats pour faciliter leur guidage et le contrôle de leur position depuis la base militaire. Ces RCSF sont liés aux applications orientées événement.
- **RCSF de collecte des données d'environnement** : Dans ce type de réseaux, il existe différents types de capteurs (capteurs hétérogènes) qui peuvent avoir plusieurs fonctionnalités. Ce type de réseaux nécessite généralement un flux de données faible, une durée de vie importante. Il sert à la collecte périodique des données environnementales puis leur transmission vers la station de base. Ces RCSF sont liés aux applications orientées surveillance.
- **RCSF de surveillance et sécurité** : La différence entre ce type de réseaux et celui de collecte d'environnement et que les nœuds ne transmettent pas l'ensemble de données collectées mais seulement des rapports concernant une violation de la sécurité. Ce sont généralement des nœuds fixes qui contrôlent d'une façon continue la détection d'une anomalie dans le fonctionnement d'un système. Par exemple, les altérations dans la structure d'un bâtiment, suite à un séisme, pourraient être détectées par des capteurs intégrés dans les murs ou dans le béton.

1.4 Contraintes de conception des RCSF

Les principaux facteurs et contraintes influençant l'architecture des réseaux de capteurs peuvent être résumés comme suit :

- **La tolérance aux pannes :** Certains nœuds peuvent générer des erreurs ou ne plus fonctionner à cause d'un manque d'énergie, un problème physique ou une interférence. Ces anomalies ne doivent pas affecter le reste du réseau, c''est le principe de la tolérance aux pannes. Cette contrainte mesure la capacité de maintenir les fonctionnalités du réseau sans interruption dues à une panne intervenue sur un ou plusieurs capteurs.
- **Passage à l'échelle :** La plupart des protocoles conçus pour RCSF ont été établis pour des réseaux dont le nombre de nœuds est moyen. Cependant, les performances de ces protocoles risquent de se dégrader lorsque le nombre de nœuds augmente. Ce principe est appelé le passage à l'échelle. D'où on dit qu'un protocole s'adapte au passage à l'échelle si et seulement si quand le nombre de nœuds augmente dans le réseau, les performances de ce dernier ne doivent pas se dégrader.
- **Le coût de production :** Souvent, les réseaux de capteurs sont composés d'un très grand nombre de nœuds. Le prix d'un est critique afin de pouvoir construire un réseau de surveillance traditionnel. Actuellement, un nœud ne coûte souvent pas beaucoup plus que 1$.
- **Environnement :** Les nœuds capteurs doivent être conçus d'une manière à résister aux sévères conditions de l'environnement : forte chaleur, pluie, humidité …
- **Les medias de transmission :** Dans un réseau de capteurs, les nœuds sont reliés par une architecture sans fil. Pour permettre des opérations sur ces réseaux dans le monde entier, le média de, transmission doit être normé. On utilise le plus souvent l'infrarouge, le Bluetooth et les communications radio Zigbee.
- **La consommation d'énergie :** Les capteurs sont généralement alimentés par des batteries et dans certaines applications ils sont déployés dans des zones hostiles où l'intervention humaine est presque impossible pour toute opération de maintenance. De ce fait, le remplacement de la batterie est impossible. Par ailleurs, la durée de vie d'un capteur dépend grandement de la durée de vie de la batterie. Cette énergie est consommée par les différentes unités du capteur afin de réaliser les tâches de captage, traitement de données et de communication. Cette dernière est l'opération qui consomme le plus d'énergie, c'est pour cette raison que les recherches actuelles se concentrent principalement sur les moyens de réduire cette consommation pour garantir une longue durée de vie aux réseaux déployés.

- **Les contraintes matérielles :** La principale contrainte matérielle est la taille du capteur. La petite taille des capteurs ne permet de le doter par d'autres unités telles que un grand nombre de batteries et un mobilisateur.
- **La topologie de réseau :** Le déploiement d'un grand nombre de nœuds nécessite une maintenance de la topologie. Cette maintenance consiste en trois phrases :
  - Déploiement.
  - Post-déploiement.
  - Redéploiement de nœuds additionnels.

1.5 Domaines d'applications des RCSF

La taille de plus en plus réduite des micro-capteurs, leur coût de plus en plus faible, la large gamme des types de capteurs disponibles (thermique, optique, ect) ainsi que le support de communication sans fil utilisé, permettent aux réseaux de capteurs d'envahir rapidement plusieurs domaines d'applications. En effet, ces réseaux se révèlent très utiles dans de nombreuses applications lorsqu'il s'agit de collecter et de traiter des informations provenant de l'environnement. Parmi ces domaines, nous citons les domaines : militaire, l'environnemental, domestique et médical.

1.5.1 Applications militaires

Les applications militaires ont souvent été au centre de nombreux développements technologiques. En effet, dans le cadre de nombreux projets lancés par le PYNTAGONE tel que ARPANET, INTERNET a vu le jour. Puis, dans le cadre d'un autre projet les réseaux ad-hoc ont été conçus et les réseaux de capteurs étaient le fruit des recherches dans le cadre du projet WINS de DARPA[3] (Defense Advanced Research Project Agency). Dans le domaine militaire, cette technologie peut impulser de nouvelles stratégies de communication ou encore servir à détecter des dispositifs nucléaires et les dépister, à surveiller les activités d'ennemies, ou à analyser un endroit stratégique et difficile d'accès avant un déploiement de troupes.

[3] DARPA (Defense Advanced Research Project Agency)

Figure 4: Les réseaux de capteurs: Application militaire

1.5.2 Applications médicales

Le champ de contrôle de santé représente un grand marché pour les réseaux de capteurs sans fil qui a tendance à croître très rapidement. Ces RCSF peuvent être utilisés pour assurer une surveillance permanente des organes vitaux de l'être humain telle que la survaillance de la glycémie, la détection du cancer à une étape précoce, …grâce à des micro-capteurs qui pourront être avalés ou implantés sous la peau. Ils peuvent aussi faciliter le diagnostic de quelques maladies en effectuant des mesures physiologiques telle que la tension artérielle, le rythme cardiaque, … à l'aide des capteurs ayant chacun une tâche bien particulière.

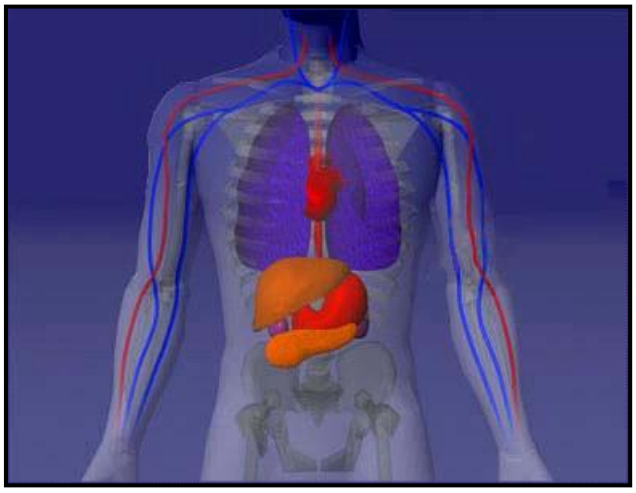

Figure 5: Un réseau de capteurs dans le domaine médical

1.5.3 Applications environnementales

Le contrôle des paramètres environnementaux par les réseaux de capteurs peut donner naissance à plusieurs applications. Par exemple, le déploiement des thermo-capteurs dans une forêt peut aider à détecter un éventuel début de feu et par suite faciliter la lutte contre les feux de forêts avant leur propagation. Le déploiement des capteurs chimiques dans les milieux urbains peut aider à détecter la pollution et analyser la qualité de l'air. De même leur déploiement dans les sites industriels empêche les risque industriels tels que la fuite de produits toxiques (gaz, produits chimiques, éléments radioactifs, pétrole, ect.)

Figure 6: Application environnementale

1.5.4 Applications domestiques

En plaçant sur le plafond ou dans le mur des capteurs on peut économiser l'énergie en gérant l'éclairage ou le chauffage en fonction de la localisation des personnes.

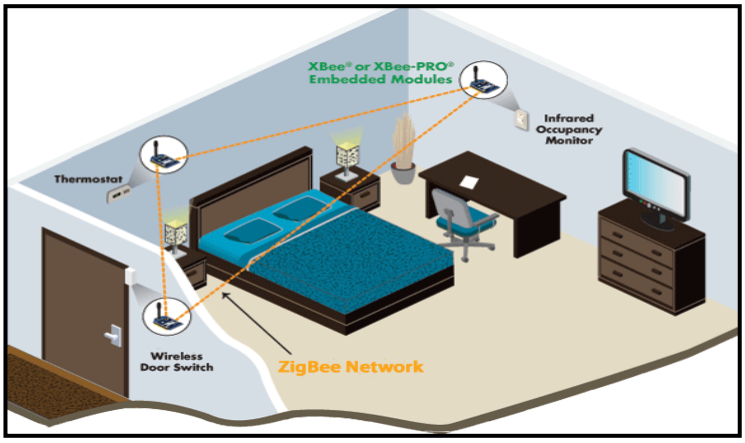

Figure 7: Application domotique

1.6 Communications dans les RCSF

1.6.1 Architecture de communication basée sur le modèle OSI

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

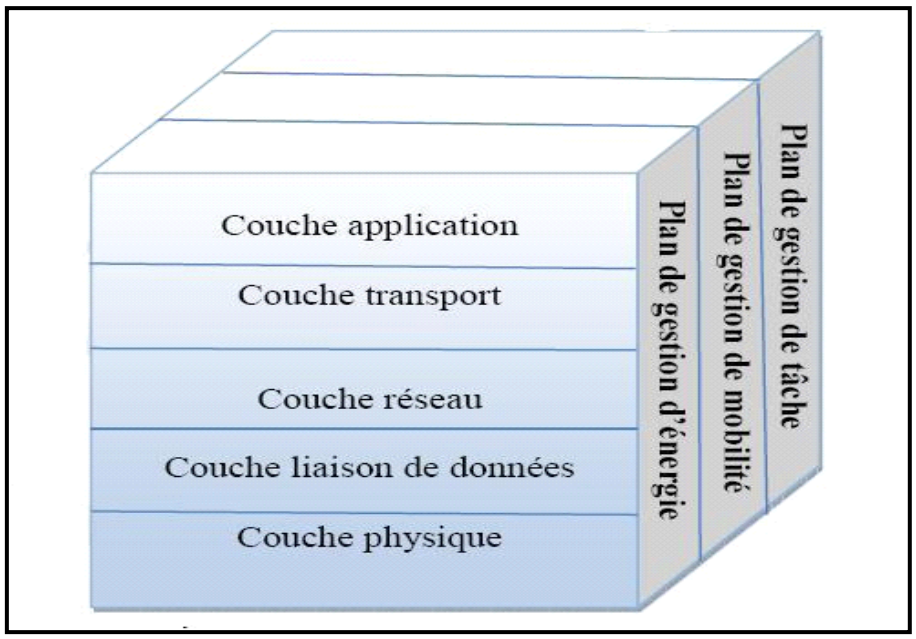

Figure 8: Pile protocolaire dans les RCSF

Le rôle de chacune de ces couches est résumé comme suit :

- **Couche physique :** matériels pour envoyer et recevoir les données.
- **Couche liaison de données :** gestion des liaisons entre les nœuds et la station de base, contrôle d'erreurs.
- **Couche réseau :** routage et transmission de données.
- **Couche transport :** transport de données, contrôle de flux.
- **Couche application :** interface pour les applications au haut niveau.
- **Plan de gestion d'énergie :** contrôle l'utilisation d'énergie.
- **Plan de gestion de mobilité :** gestion des mouvements des nœuds.
- **Plan de gestion de tâche :** balance les tâches entre les nœuds afin d'économiser de l'énergie.

1.6.2 Technologie de la communication dans les RCSF

a) Bluetooth / IEEE [2]

Bluetooth est une spécification de l'industrie des télécommunications. Elle utilise une technique radio courte distance destinée à simplifier les connections entre les appareils électronique. Malheureusement, un grand défaut de cette technologie est sa trop grande consommation d'énergie. En plus, elle ne peut pas être utilisée pour les RCSF denses.

b) ZigBee / IEEE [3]

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

1.7 Conclusion

Dans ce chapitre, nous avons présenté en premier lieu quelques généralités sur les réseaux de capteur sans fil, à savoir la description d'un réseau de capteur sans fil, leur son architecture et sa topologie. En outre, nous avons présenté leurs principaux domaines d'applications et les défis de conception matérielle et logicielle.

Cette présentation nous aide à développer une application orientée surveillance sous Contiki en connaissant la spécificité des réseaux de capteurs sans fil.

Chapitre 2
Outils de développement pour les réseaux de capteurs

Chapitre 2
Outils de développement pour les réseaux de capteurs

2.1 Introduction

Les réseaux de capteurs sont considérés parmi les systèmes à ressources limitées. De ce fait, les outils logiciels conçus aux ordinateurs ne sont plus adaptables à ce type de systèmes. Il y a des outils logiciels légers qui sont dédiés spécialement aux réseaux de capteurs que ce soient des systèmes d'exploitation ou des langages de programmation. Dans ce contexte, plusieurs systèmes d'exploitation et des langages ont vu le jour pour exploiter les réseaux de capteurs.

Dans ce chapitre, on présente les systèmes d'exploitation pour les réseaux de capteurs en se focalisant sur Contiki qui est considéré comme un système d'exploitation complet et réputé. Puis, on présente le simulateur Cooja qui fonctionne à base de Contiki.

2.2 Outils logiciels

Les capteurs sont caractérisés par un processeur de faible fréquence de calcul, une mémoire réduite et une ressource d'énergie très limitée. De ce fait, il s'avère nécessaire de concevoir des logiciels légers qui s'adaptent à leurs faibles performances. Parmi ces logiciels, le système d'exploitation qui est l'élément fondamental pour faire fonctionner le capteur.

Les systèmes d'exploitation embarqués généralistes multitâches ont été développés à l'origine pour une plateforme PC et par la suite ils ont été modifiés ou adaptés à des systèmes embarqués. Ces systèmes d'exploitation sont trop généralistes pour être suffisamment efficaces, et les changements de contexte et les mécanismes de synchronisation entre processus génèrent un surcoût inacceptable pour des nœuds du réseau de capteur sans fil, à la

fois en termes de coût processeur que d'occupation mémoire. Pour résoudre ce problème, des systèmes logiciels ont été créés pour cibler particulièrement les applications de réseaux de capteurs.

Il existe plusieurs systèmes d'exploitation qui ont été conçus pour les RCSF, parmi lesquels nous citons Contiki, MANTIS et TinyOS. Dans notre projet nous avons travaillé avec le système d'exploitation Contiki qu'on le présente dans ce qui suit. Puis on présente le simulateur Cooja qui fonctionne sous contiki.

1.2.1 Le système d'exploitation Contiki

a) Présentation de Contiki

Contiki est un système d'exploitation open source qui fonctionne sur les microcontrôleurs de faible puissance et permet de développer des applications qui font une utilisation efficace du matériel tout en fournissant une communication standardisée sans fil de faible puissance pour une gamme de plateformes des microcontrôleurs tels que le TI MSP430 et Atmel AVR, qui sont utilisés dans les familles de Telos, de Tmote et de Mica.

Les propriétés suivantes, ont permis l'évolution de Contiki :

1. **Normes Internet :** Contiki fournit des communications Internet de faible puissance. Contiki et il prend en charge IPv6 et IPv4 avec des normes sans fil de faibles puissances: 6LoWPAN, RPL, CoAP.
2. **Développement rapide :** Avec Contiki, le développement est facile et rapide : Les applications à base de Contiki sont écrites en C standard. En outre, ces applications peuvent être émulées avec le simulateur Cooja avant d'être injectées sur des capteurs réels. Il existe aussi l'environnement Instant Contiki qui fournit un environnement de développement.
3. **Une sélection du matériel :** Contiki fonctionne sur une gamme d'appareils sans fil de faible puissance tels que la famille des Mica et la famille des Telos.
4. **Communauté active :** Contiki est développé par une équipe mondiale de développeurs avec des contributions d'Atmel, Cisco, ETH, Redwire LLC, SAP, Thingsquare, et beaucoup d'autres, menés par Adam Dunkels de Thingsquare.
5. **Logiciel Open Source :** Contiki est un logiciel open source : Contiki peut être utilisé librement tant dans les systèmes commerciaux et non commerciaux et le code source complet est disponible.

6. **Support commercial :** Contiki fournit à la fois le soutien de la communauté des développeurs Contiki et support commercial.

Le tableau suivant récapitule l'évolution de Contiki au cours du temps.

Tableau 1: Evolution de Contiki

Année	Evolution de Contiki
2003	Présentation par Adam Dunkels de FULL TCP/IP sur une architecture 8 bits.
2004	Contiki présenté par Dunkels et al. à Tampa en Floride USA, lors de la 29ième conférence annuelle à l'occasion du premier atelier de l'IEEE sur les capteurs en réseau.
Décembre 2007	Contiki 2.1
Juillet 2008	Contiki 2.2 Collaboration de Cisco système pour implémenter l'open source uIPv6. Pour mettre en place d'un RCSF sur IPv6 .
Juin 2009	Contiki 2.3 ContikiSec : Une version de Contiki supportant la sécurité.
Février 2010	Contiki 2.4.
Septembre 2011	Contiki 2.5. Low-Power CoAP pour Contiki. Low-Power sans fil IPv6 avec Contiki RPL. Contiki MAC, mécanisme qui permet de mettre les capteurs en veille 99% du temps dans le but de réduire la consommation d'énergie.
Juillet 2012	Contiki 2.6.
Septembre 2012	Dans le but de développer un ensemble de produits logiciels basés sur Contiki Österlind et al. fondent une société nommée Thingsquare qui fournit de logiciels open source destinés à l'Internet des objets.
Octobre 2012	Thingsquare commercialise des plateformes dédiées au marché de la domotique, de l'automatisation des bâtiments et de l'éclairage intelligent.

b) Les caractéristiques de Contiki

Contiki se caractérise par :

- **Allocation de mémoire :** Contiki est conçu pour les systèmes légers. L'espace mémoire utilisé par le système d'exploitation et par l'application doit être suffisamment faible pour être contenu dans la mémoire du capteur. Une configuration typique de Contiki (le noyau et le chargeur de programmes) consomme 2 Koctets de RAM et 40 Koctets de ROM.

- **Consommation d'énergie :** L'énergie est souvent apportée par une batterie qui est généralement non rechargeable. Les recherches scientifiques explorent les possibilités de réduire la consommation d'énergie dans capteurs. Le module radio qui est chargé de la transmission et la réception, est l'élément qui consomme plus d'énergie. Pour cela, le module radio devrait être mis en mode actif que s'il a des données à transmettre ou à recevoir sinon il devrait passer en mode veille. Cependant, lorsque le module radio est mis en mode veille, le capteur ne reçoit pas les messages qui lui sont destinés. En outre, un réveil périodique risque d'être inutile, et donc de consommer de l'énergie de façon inefficace. Pour gérer cette problématique, Contiki propose par défaut ContikiMAC, un mécanisme conçu pour rester en communication avec le réseau efficacement, tout en permettant la mise hors tension du module radio 99% du temps. D'autres techniques permettent de limiter la consommation telle que le compactage des données transférer, le pré-calcul (afin de ne transmettre que les données réellement utiles), mais aussi une optimisation du routage.
- **Portabilité :** la portabilité consiste à adapter le système d'exploitation aux différents types de capteurs, selon les éléments électroniques les constituant. Contiki est complètement écrit en langage C. Ce langage de programmation est le langage le plus répondu pour la programmation des systèmes. La portabilité de Contiki concerne les plateformes suivantes : Z1, WISMOTE, IRIS, MICAZ, SKY, TELOS.
- **Interopérabilité :** l'interopérabilité d'un capteur est le fait de pouvoir communiqué avec les capteurs gérés par un système d'exploitation différent. Adams Dunkels de l'équipe scientifique suédoise présente dès 2003 uIP et IwIP permettant d'implémenter le protocole IP sur les systèmes limités en ressources tels que les capteurs. Jusque là, les capteurs utilisaient des protocoles de communications propriétaires ou alors des adaptations d'IP permettant le fonctionnement des applications mais sans offrir toutes les fonctionnalités du protocole IP. Dès la présentation de Contiki en 2004 uIP et IwIP étaient disponibles. De se fait, les applications exécutées sur Contiki pouvaient dialoguer vers n'importe quel matériel supportant le protocole IP. L'arrivée de IPv6 est uIPv6 sur Contiki apporte une nouvelle interopérabilité vers les matériels supportant ce protocole. Le support de 6LoWPAN permet à Contiki de communiquer avec les matériels via un réseau sans fil suivant la norme 802.15.4. Contiki est réputé pour être un système d'exploitation robuste et mature, fournissant IPv4 et IPv6 pour les réseaux de capteur sans fil. Selon une étude publiée en 2011, comprenant des tests d'interopérabilité entre des capteurs sous Contiki et

d'autre sous TinyOS, l'interopérabilité est bien au rendez-vous, mais des efforts sont à déployer pour mesurer et améliorer les performances de la couche réseau.

c) L'architecture de Contiki

Contiki est constitué d'un noyau, des bibliothèques, d'un ordonnanceur et d'un gestionnaire de processus. Comme tout système d'exploitation, son rôle est de gérer les ressources physiques telles que le processeur, la mémoire, les périphériques (d'entreé/sortie). Il fournit ensuite aux applications informatiques des interfaces permettant d'utiliser ces ressources. Conçu pour les modules de capteur sans fil miniatures, il occupe peu d'espace en mémoire et permet une consommation d'énergie très faible.

Pour économiser la mémoire, tout en fournissant un bon contrôle de flux dans le code, Contiki utilise un mécanisme appelé Protothreads qui est est un compromis entre la programmation événementielle et la programmation multi-threads. Contiki gère les standards 6LoWPAN, RPL, CoAP. Le système de fichiers Coffee permet des opérations sur des fichiers stockés sur une mémoire flash externe.

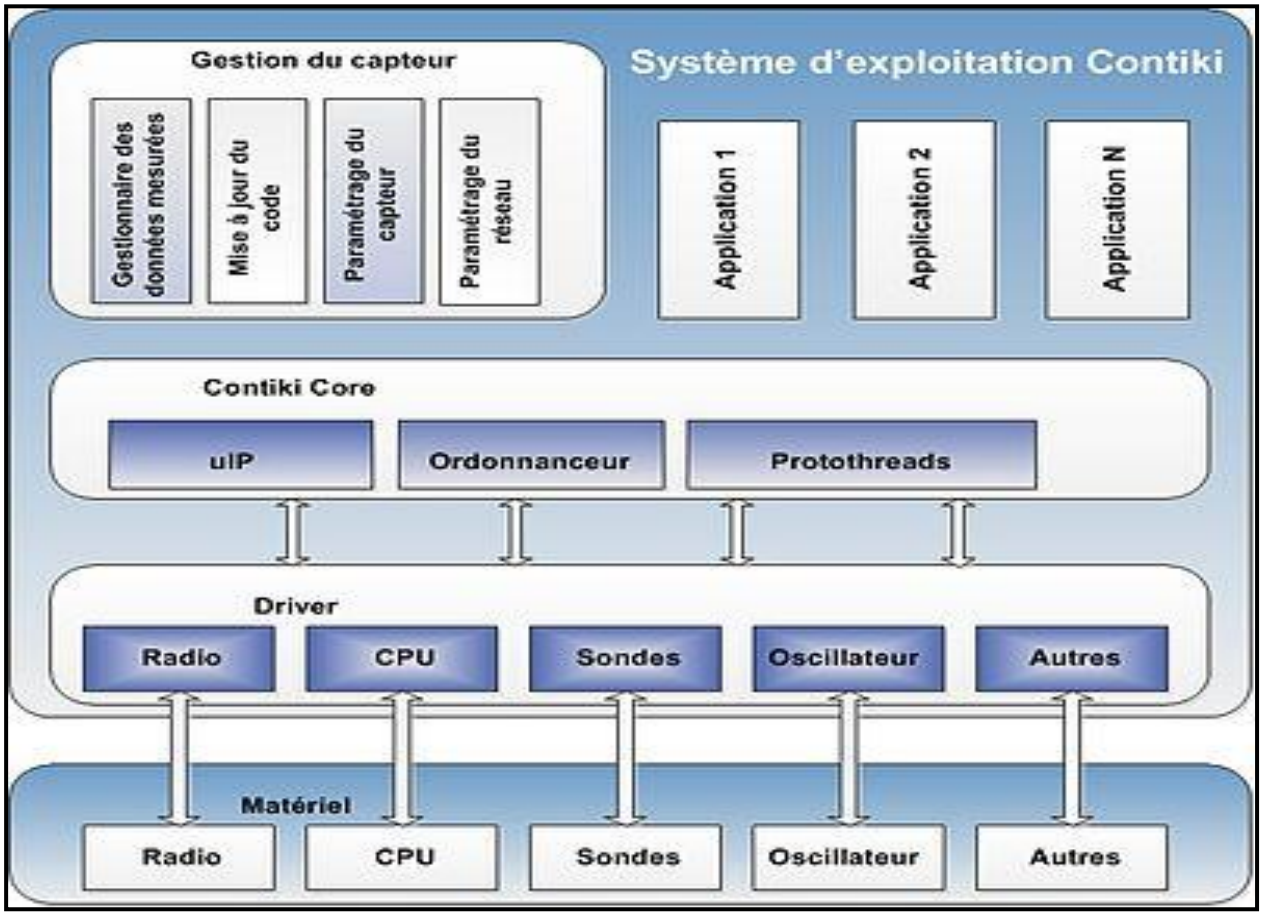

Figure 9: Architecture de Contiki

d) La communication dans Contiki

Contiki offre deux types de connectivité :

- La couche uIP, orientée Internet, offre les services essentiels du protocole IP mais nécessite plus de ressources que Rime.
- La couche Rime permet un dialogue avec les capteurs voisins pour établir des schémas de routage.

1. **La couche uIP :** Contiki fournit une implémentation de la pile du protocole TCP/IP pour les petits microcontrôleurs de 8 bits. uIP n'exige pas de ses pairs pour avoir une pile complète de protocoles, mais il peut communiquer avec des pairs exécutant une pile légère similaire. La mise en œuvre uIP est écrite en C et elle a l'ensemble minimal de fonctionnalités nécessaires pour une pile complète TCP/IP. uIP ne peut soutenir qu'une interface réseau, et il prend en charge TCP, UDP, ICMP, et des protocoles IP.

L'implémentation traditionnelle de TCP/IP a exigé plus de ressources à la fois en termes de taille du code et utilisation de la mémoire pour être utile dans les petits systèmes 8 ou 16 bits. La taille du code de quelques centaines de Koctets de RAM et les exigences de plusieurs centaines de Koctets ont rendu impossible pour s'adapter à la pile TCP/IP complète dans les systèmes avec quelques dizaines de Koctets de RAM.

La pile uIP peut être exécutée en tant que tâche dans un système multitâche, ou comme un programme principal dans un système multitâche unique. Dans les deux cas, la boucle principale de contrôle fait deux choses à plusieurs reprises :

- Vérifier si un paquet est arrivé à partir du réseau.
- Vérifier si un délai périodique s'est produit.

- **La couche Rime :** D'autre part, Contiki supporte la couche Rime qui est une autre pile de protocoles en couches pour la communication en réseau. Rime régit au principe du meilleur effort et la transmission fiable.

La pile Rime supporte à la fois les primitives de communication à un saut ou multi-sauts. Les primitives pour les communications multi-sauts ne précisent pas comment les paquets sont acheminés à travers le réseau. Comme le paquet est envoyé sur le réseau, l'application ou le protocole de la couche supérieure est invoqué au niveau de chaque nœud pour qu'il choisisse le nœud voisin qui relaye le paquet. Il est ainsi possible de mettre en œuvre des protocoles de routage arbitraires au dessus des primitives à bande multiples.

Les protocoles ou les applications s'exécutant au dessus de la pile Rime peuvent mettre en œuvre des protocoles additionnels qui ne sont pas dans la pile Rime. Si un protocole ou application s'exécute au-dessus de la pile Rime, il aura besoin de primitives de communication qui ne sont pas actuellement dans la pile Rime.

1.2.2 Le simulateur Cooja

Un émulateur est un sous ensemble de simulateurs qui permet d'analyser les codes destinés à être embarqués dans la plateforme ciblée. Il permet d'émuler de manière réaliste le comportement du logiciel embarqué. Il existe plusieurs types d'émulateurs et ils sont souvent dédiés à des plateformes ou logiciels spécifiques. Par exemple ceux qui permettent de simuler des nœuds qui ont leur propre système d'exploitation, tel que TOSSIM [9] pour les nœuds qui utilisent TINYOS, ou COOJA [10] pour les nœuds qui utilisent CONTIKI, etc.

Pour développer les programmes au sein de Contiki, le système met à disposition un simulateur appelé Cooja. Le logiciel permet d'émuler des nœuds et de charger un programme compilé. Ceci est particulièrement utile pour tester les programmes avant de les mettre dans la mémoire flash des nœuds réels, puisque le logiciel simule les conditions d'exécution et de mémoire de la plateforme TI MSP430. Les données collectées provenant de la station de base via sa sortie standard peuvent être enregistrées dans des fichiers ou lues par des logiciels qui peuvent par la suite traiter et présenter les données à l'utilisateur. On peut par exemple citer l'application collect-view intégrée dans Contiki qui permet de visualiser les valeurs des capteurs et des données de supervision du réseau. Parmi les capteurs supportés à ce jour par Cooja sont : exp4538, z1, wismote, sky, Micaz, jcreate, sentilla-usb …

Il est très utile pour tester des applications de réseaux de capteurs écrites pour la plateforme Contiki sans avoir à les mettre en matériel réel de réseau de capteurs. Ainsi, chaque fois que nous écrivons un programme sous Contiki, nous essayons de le faire fonctionner sur le simulateur Cooja, puis nous pouvons le flasher sur des capteurs réels.

Cooja est écrit en langage java et a une structure très modulaire qui lui permet d'être étendue avec des plugins sur mesure pour elle. La figure illustre l'interface Cooja.

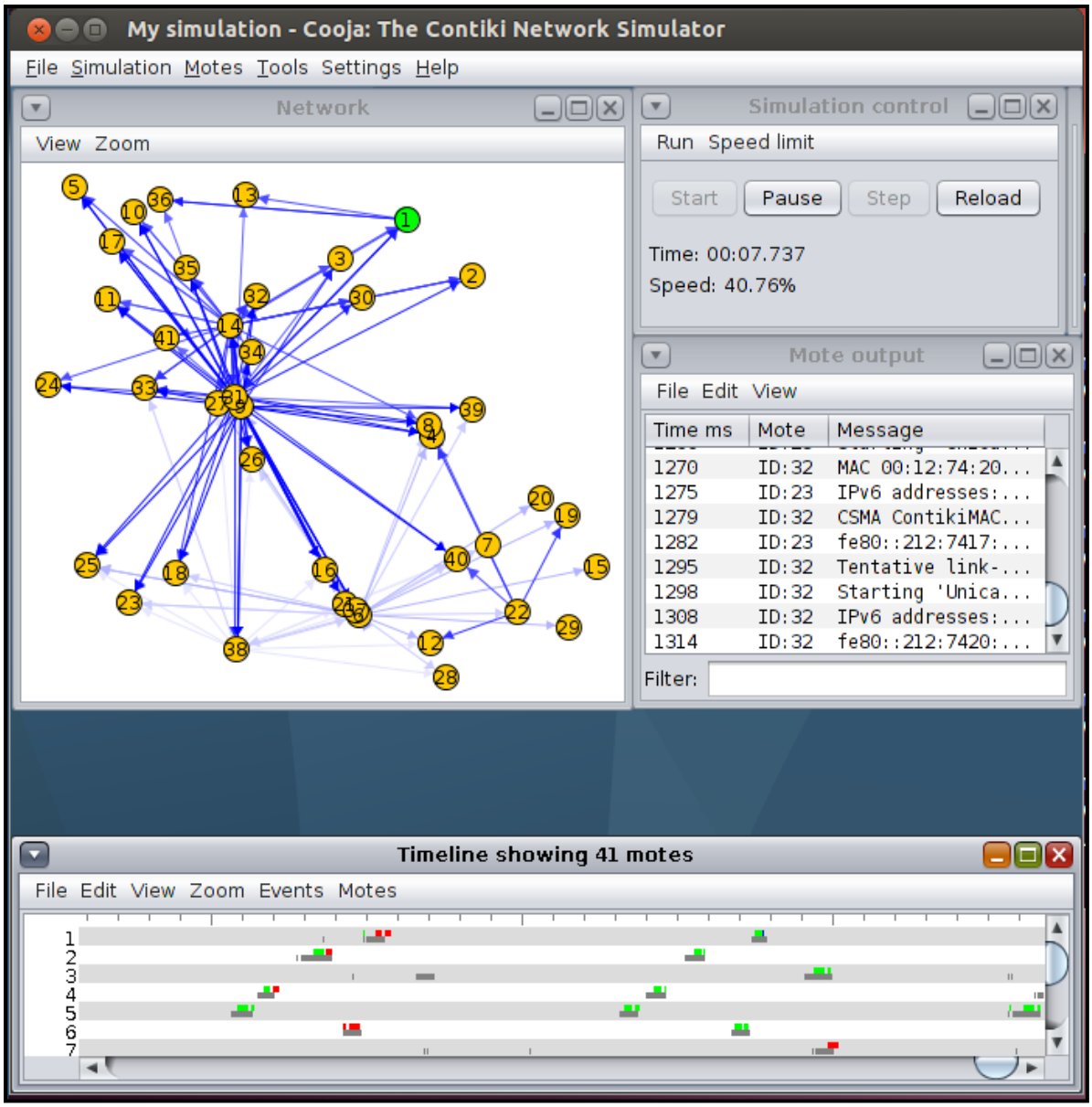

Figure 10: Interface graphique de Cooja

1.2.3 Les modules en langage C pour la programmation des capteurs

Un corps d'un programme Contiki basic est composé comme suite :

```
#include <contiki.h>
PROCESS (hello_world_process, " hello world process ") ;
AUTOSTAR_PROCESSES (&hello_world_process) ;
PROCESS_THREAD(hello_world_process, ev, data)
{
  PROCESS_BEGIN() ;
  Printf("Hello, world !\n") ;
```

```
    PROCESS_END() ;
}
```

Commentaires du code :

- La 1ère ligne inclut les en-têtes (header) C du Contiki. Cette ligne est nécessaire pour inclure les librairies et les définitions de Contiki.
- La 2ème ligne déclare le processus Contiki. Ce processus possède un variable nommée (hello_world_process) et une chaîne de caractères nommée (hello world process). La chaine de caractères est utilisée pour le débogage et comme une commande pour le Shell de Contiki.
- La 3ème ligne informe le système Contiki que le programme hello_world_process doit se lancer automatiquement lors du démarrage du système.
- La 4ème ligne définit le processus hello_world_process. Les arguments ev et data dans la macro « PROCESS_THREAD () » sont des variables représentant le nombre d'évènements et les données que le processus peut recevoir.
- Le processus lui-même est définit entre les deux macros : PROCESS_BEGIN () et PROCESS_END (). Dans ce cas, le processus se limite à afficher le message « hello, word ».

Il existe des programmes Contiki complexe composés par :

- Des structures de données.
- Defines.
- Des fonctions.
- Des variables.

Le tableau ci-dessous présente quelques modules utilisés pour la programmation des capteurs.

Tableau 2 : Différentes fonctions pour programmer un capteur

Modules	Indications
PROCESS_BEGIN ()	Cette macro définit le début d'un processus, il doit toujours apparaître dans une définition PROCESSUS fil ().
PROCESS_END ()	Cette macro définit la fin d'un processus. Il doit apparaître dans un PROCESS_THREAD (). Le processus se ferme lorsque la macro PROCESS_END est atteinte.
PROCESS_WAIT_EVENT ()	Attend un évènement à être affecté à ce processus. Cette macro bloque le processus en cours d'exécution jusqu'à ce que le processus reçoive un évènement.
PROCESS_THREAD (name, ev, data)	Cette macro est utilisée pour définir le corps (protothread) d'un processus. Le processus est appelé chaque fois qu'un évènement se produit dans le système.
PROCESS_YIELD_UNTIL (c)	Cédez le processus en cours jusqu'à ce qu'une condition se produise. Cette macro est différente de PROCESS_WAIT_UNTIL () (C : la condition d'attente).
Void* uip_appdata	Pointeur vers les données d'applications dans la mémoire tampon de paquets. Ce pointeur pointe vers les données d'applications quand l'application est appelée. Si l'application souhaite envoyer des données, elle peut utiliser cet espace pour écrire les données avant d'appeler uip_send ().
Struct uip_conn* uip_conn	Pointeur vers la connexion TCP courante. Le pointeur uip_conn peut-être utilisé pour accéder à la connexion TCP courante.
Struct uip_udp_conn	Représentation d'une connexion uIP UDP.
Uint16_t uip_len	Le uip_len variable globale contient la longueur du paquet dans le tampon uip_buf. Lorsque le pilote du périphérique de réseau appelle la fonction d'entrée uIP, uip_len doit être réglé sur la longueur du paquet dans le tampon d'uip_buf. Lors de l'envoi des paquets, le pilote de périphérique doit utiliser le contenu de la variable uip_len pour déterminer la longueur du paquet sortant.
Uip_datalen ()	La longueur de toutes les données d'entrée qui sont actuellement disponibles dans le tampon uip_appdata. La fonction test uip_data () doit d'abord être utilisée pour vérifier s'il y a des données disponibles.
Uip_new_data ()	Est-ce qu'il y a des nouvelles données entrantes disponibles ? Ce test est positif s'il y a des nouvelles

	données pour l'application présente au niveau du pointeur uip_appdata. La taille des données est disponible via la variable uip_len.
#define UIP_APPDATA_SIZE	Cette macro détient la taille disponible pour les données utilisateur dans le tampon uip_buf. La macro est destinée à être utilisée pour le contrôle de la taille de données utilisateur disponible.
#define UIP_STAT (s)	Les statistiques uIP TCP/IP. S : la variable dans la quelle les statistiques uIP TCP/IP sont réunis.
Void uip_init (void)	Fonction d'initialisation uIP. Cette fonction doit être appelée au démarrage jusqu'à l'initialisation de la pile uIP TCP/IP.
Struct uip_udp_conn* uip_udp_new (const uip_ipaddr_t* ripaddr, u16_t rport)	Configurer une nouvelle connexion UDP. Cette fonction met en place une nouvelle connexion UDP. La fonction attribue automatiquement un port local utilisé pour la nouvelle connexion. Cependant, un autre port peut être choisi à l'aide de l'appel uip_udp_blin (), après la fonction uip_udp_new () a été appelée.
Void uip_send (const void* data, int len)	Envoyer des données sur la connexion en cours. Cette fonction permet d'envoyer un seul segment de données TCP. Seules les demandes qui ont été invoquées par uIP pour le traitement de l'évènement. La quantité de données qui est en fait envoyée après un appel de cette fonction est déterminée par la quantité maximum de données TCP. uIP recadre automatiquement les données de sorte que seule la quantité appropriée de données sont envoyées. La fonction uip_mss () peut être utilisée pour interroger uIP pour la quantité de données qui seront effectivement envoyé. Data : un pointeur de données qui doit être envoyé. Len : le nombre maximal d'octets de données à envoyer.
#define udp_bind (conn, port)	Cette fonction est liée à une connexion et à un port UDP local spécifié. Lorsque la connexion est créée avec udp_new (), elle obtiendra un numéro de port local automatiquement. Si l'application a besoin de lier la connexion à un port local spécifié, cette fonction doit être utilisée. Le numéro de port doit être prévu dans l'ordre des octets du réseau pour une conversion avec UIP_HTONS () est généralement nécessaire. Conn : un pointeur vers la connexion UDP qui d'être liée. Port : le numéro de port dans l'ordre des octets du

	réseau auquel est liée la connexion.
#define uip_udp_send (len)	Envoyer un datagramme UDP de longueur len sur la connexion en cours. Cette fonction ne peut être appelée en réponse à un évènement UDP. Les données doivent être présentes dans le tampon uip_buf, à l'endroit indiqué par le pointeur uip_appdata.
#define uip_ip6addr (addr, addr0, addr1, addr2, addr3, addr4, addr5, addr6, addr7)	Cette fonction construit une adresse IPv6 de 8 mots de 16 bits.
Struct uip_ds6_nbr	Une entrée dans le cache de nbr.
Struc uip_ds6_addr	Structure d'adresse unicast.
Struc uip_ds6_aaddr	Structure d'adresse anycast.
#define DEBUG 0	Option de compression ajoutée à UDP, ne supporte actuellement que deux ports compressés
#define MAC_MAX_PAYLOAD 120	Pointeur dans la mémoire tampon de Rime
Void clock_int (void)	Cette fonction initialise la bibliothèque d'horloge et doit être appelée à partir de la fonction main () du système.
Clock_time_t clock_time (void)	Cette fonction retourne la heure de l'horloge du système. L'heure actuelle, mesurée dans le système ticks. Référencé par minuterie expiré (), minuterie restart (), et réglage de la minuterie ().

En plus de tout ça le fichier système Makefile est utilisé dans le système de construction Contiki qui est composé d'un certain nombre de makefiles. Ceux-ci sont :

- Makefile : le makefile du projet, situé dans le répertoire du projet.
- Makefile.include : Contiki makefile l'echelle du système, situé à la racine de la source Contiki arbre.

2.3 Conclusion

Dans ce chapitre, nous avons présenté les outils logiciels nécessaires pour développer une application sous contiki. Nous avons détaillé les spécificités du langage Contiki pour tirer profit des ses avantages. Puis nous avons présenté comment se fait la programmation d'un capteur sous contiki. Le chapitre suivant décrit en détails l'application que nous avons développée.

Chapitre 3

Implémentation

Chapitre 3

Implémentation

3.1 Introduction

Dans ce chapitre nous détaillons le fonctionnement de l'application que nous avons développée. Nous commençons la première partie par préciser l'environnement et les outils utilisés dans le développement. Ensuite nous présentons une vue globale sur le fonctionnement de l'application.

3.2 Environnement de travail

L'implémentation de notre application a nécessité l'utilisation de différents outils matériels qui sont les capteurs Telosb ainsi que des outils logiciels bien spécifiques aux réseaux de capteurs sans fil, tels que le langage C, ContikiOS qui est un système d'exploitation dédié aux équipements à ressources limitées, le simulateur Cooja qui est fourni par ContikiOS et Java pour une meilleure présentation des résultats recueillis par les capteurs.

3.2.1 Matériels à disposition

Dans notre implémentation, nous avons utilisé des capteurs de type TelosB. Ce capteur a été conçu par l'université de Californie à Berkeley. Il est principalement composé d'un microcontrôleur MSP430, d'une puce radio TI-CC2420 (permettant l'émission et la réception), d'un capteur de température, d'un capteur d'humidité, d'un capteur de luminosité, d'un autre pour l'infrarouge, d'un bouton paramétrable, d'un bouton reset, de trois LEDs, et d'un port USB. Pour fonctionner, un mote a besoin de deux piles de 1.5V, sa durée de vie est donc limitée lors d'un déploiement. En outre, il possède 10KB de mémoire vive et seulement 48 KB de mémoire flash.

A noter que dans le réseau de capteurs sans fil que nous souhaitons déployer, chaque mote communique avec ses voisins via le protocole de communication 802.15.4 sur une bande de fréquence 2.4 GHz.

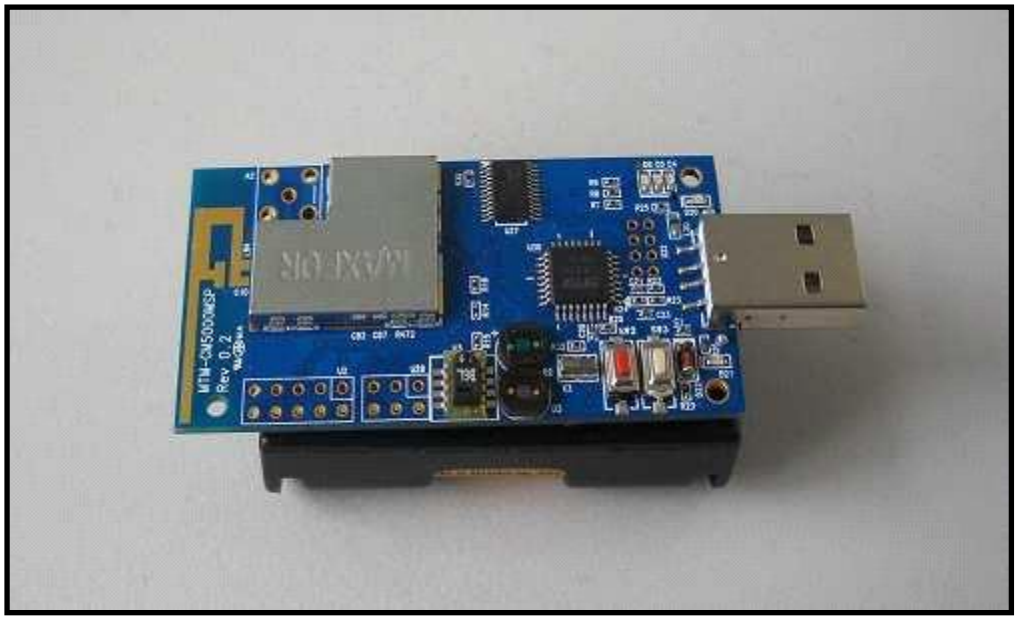

Figure 11: Un capteur TelosB

3.2.2 Logiciels utilisés

Dans cette partie, nous présentons les outils logiciels nécessaires pour le développement de notre application. Pour cela, nous avons utilisé Ubuntu 14.04 LTS, Contiki 2.7, langage « C » pour programmer les capteurs, Java pour réaliser l'interface graphique grâce à sa librairie Swing et aussi parce que c'est le seul langage dont le code exécutable est portable et Netbeans IDE 8.0.2.

3.3 Mise en place de la plateforme

La mise en place de la plateforme s'articule autour de deux parties : l'installation logicielle et l'installation matérielle.

3.3.1 Installation logicielle

La phase installation est la plus délicate. En effet, on a commencé par installer ContikiOS et le microcontrôleur MSP430 pour les capteurs de type Telosb. La procédure d'installation de ContikiOS se déroule en plusieurs étapes qu'on va la détailler. Puis on a installé Netbeans au niveau du poste de contrôle pour communiquer avec les capteurs et faire visualiser les valeurs des grandeurs remontées à la station de base.

- L'installation de ContikiOS :
  - Télécharger contiki-2.7.zip à partir :

http://downloads.sourceforge.net/project/contiki/Contiki/Contiki%202.7/contiki-2.7.zip

- Décompressez le fichier. Pour les fins de ce guide, il est supposé que le fichier est décompressé dans / home / user / :

  unzip contiki-2.7.zip

- Renommez le dossier de contiki-2.7 à contiki :

  mv contiki-2.7 contiki

- Installez tous les paquets nécessaires pour compiler et d'exécuter ContikiOS. Vous pouvez exécuter la commande suivante pour installer tous les paquets pour de multiples plates-formes:

  sudo apt-get install build-essential binutils-msp430 gcc-msp430 msp430-libc msp430mcu mspdebug binutils-avr gcc-avr gdb-avr avr-libc avrdude openjdk-7-jdk openjdk-7-jre ant libncurses5-dev

3.3.2 Installation matérielle

Une fois l'installation logicielle est terminée, il a fallu installer le matériel. Le protocole de communication entre les différents nœuds du réseau de capteurs est l'UDP

Notre réseau de capteurs est composé de trois types de nœuds (voir Figure) :

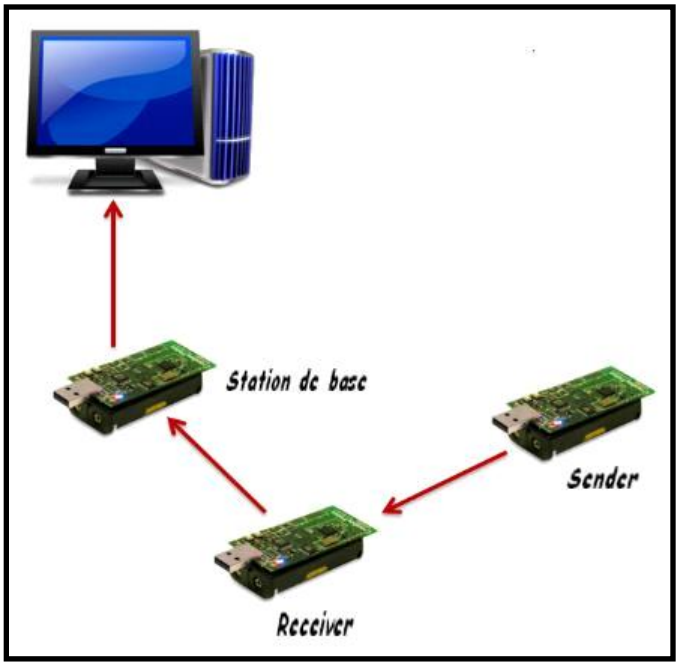

Figure 12: Architecture de l'application

- « **UDP client** » : ils sont fixes et ont pour tâche l'acquisition de la température et leurs envois au nœud « **UDP server mobile** ».
- « **UDP server mobile** » : c'est un nœud mobile, son rôle c'est de recevoir toutes les données émises par les UDP client et les renvoyer à son tour au nœud « **UDP station de base** ».
- « **UDP station de base** » : connecté à la station de base PC Fujitsu, Asus, il permet de faire le lien entre le réseau de capteur et la station de base.

3.4 Les étapes d'exécution de notre l'application

Cette section du chapitre décrit l'expérimentation de notre application qui permet l'acquisition de la température.

Après une période d'apprentissage du système d'exploitation Contiki et du langage C, nous sommes passés à la mise en œuvre, au codage et à l'implémentation de notre application d'acquisition de température sur les plateformes TelosB.

Dans notre cas on a des capteurs clients, un serveur et une station de base flashés respectivement par des programmes en langage C test-client.c, test-server.c et test-base.c. En plus, nous avons ajouté quatre interfaces graphiques programmées en Java à l'aide de NetBeans IDE pour une meilleure visibilité des résultats sensés par les capteurs.

3.4.1 Quelques exemples d'exécution

La figure 13 représente la fenêtre qui sert à s'authentifier, qui prend comme login : mamem ou saidi et comme mot de passe : license.

Figure 13: Interface d'authentification

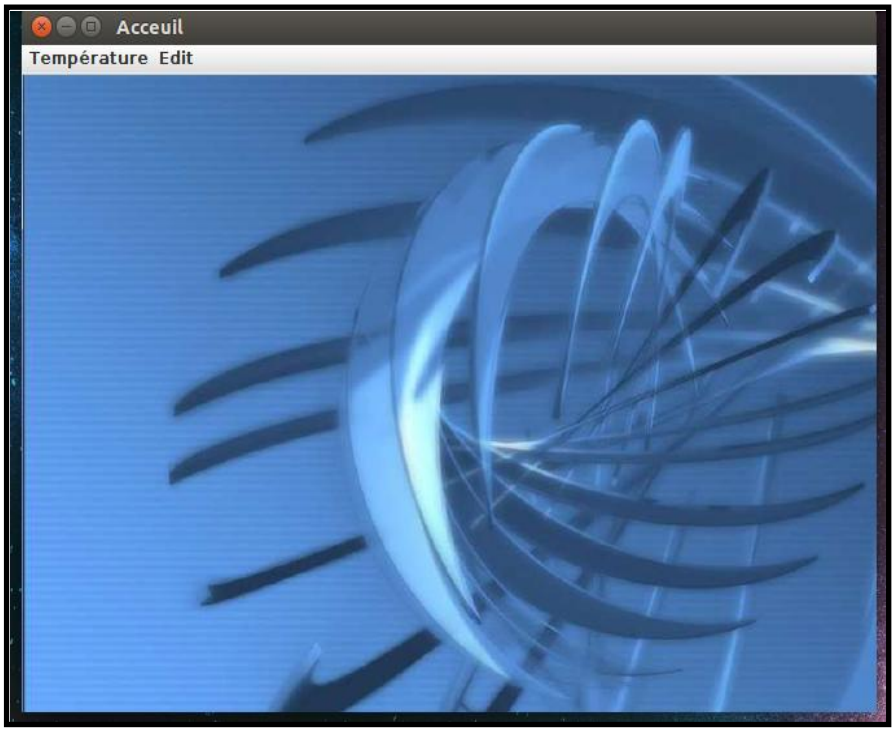

Figure 14: Fenêtre d'accueil

La figure ci-dessus c'est une simple fenêtre qui contient une barre de menu composée par des sous-menus. Un menu Température contient un sous-menu température qui mène à l'interface graphique Température et le menu Edit qui a le sous-menu fermer qui sert à fermer la fenêtre. Les sous menus sont exécutés par des raccourcis shift + F, ctrl + T respectivement.

La fenêtre ci-dessous contient les informations qui concernent les capteurs et la température sensée. Ces données sont représentées dans un tableau de quatre colonnes et dix lignes.

- La première colonne représente le numéro de relevé.
- La deuxième est l'identifiant du capteur qui envoie la température à la station de base.
- La troisième colonne contient la température repérée par les capteurs.
- La dernière nous informe la date et l'heure de l'envoie des données.

Figure 15: Collecte de données par les capteurs

La dernière fenêtre permet de représenter la température repérée par les capteurs dans un graphe (courbe) en fonction du temps.

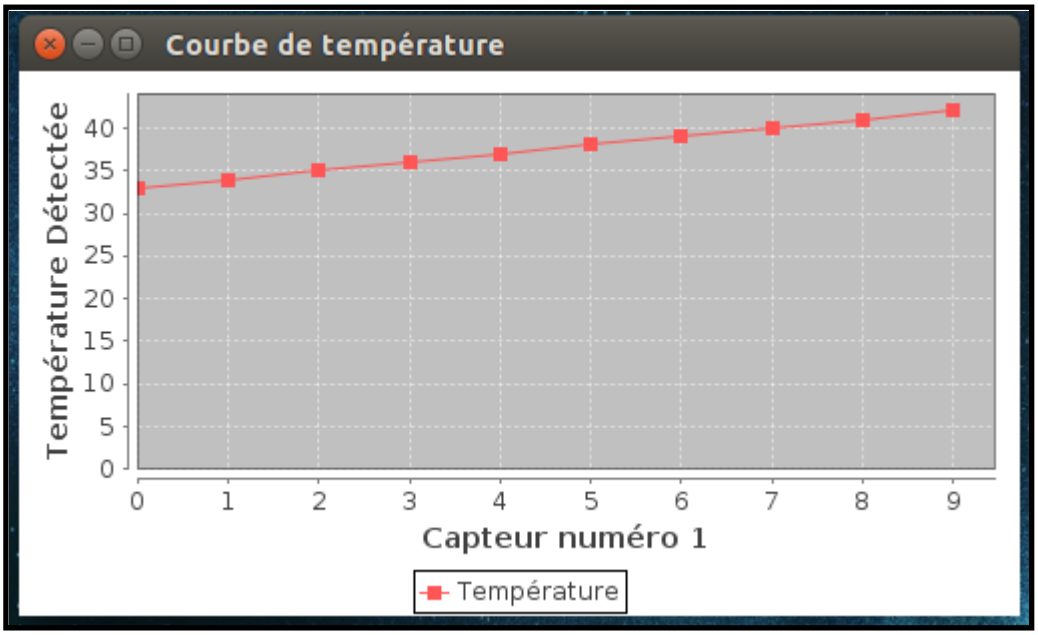

Figure 16: Représentation par une courbe les données collectées

3.5 Conclusion

Dans ce chapitre nous avons implémenté notre application et présenté les outils qu'on a utilisé (outils logiciels et matériels) comme on a montrée la méthode de leurs installations.

Globalement nous avons présentée la démarche à suivre pour réaliser une application qui permet la collecte de la température dans les réseaux de capteurs sans fil sous Contiki et on a terminé par présenter quelques exemples d'exécution de l'application.

Conclusion Générale

Conclusion générale

Les réseaux de capteurs sans fil sont une nouvelle technologie qui a surgi après les grands progrès technologiques concernant le développement des capteurs intelligents, des processeurs puissants et des protocoles de communication sans fil. Ce type de réseau composé de centaines ou de milliers d'éléments, a pour but la collecte de données de l'environnement, leur traitement et leur dissémination vers le monde extérieur.

Notre travail consiste à mettre en œuvre une application orientée surveillance basée sur les RCSF. Cette application est constituée de quatre parties: une partie qui est embarquée sur des nœuds capteurs (Senders) du réseau permettant la mesure de la température, une deuxième qui embarquée sur le reste des nœuds (Relais) chargés de recevoir les messages envoyés par les Senders et les acheminer vers la troisième partie qui est embarquée sur la station de base. Cette dernière permet la collecte des messages envoyés et leur acheminement vers le PC, et une quatrième partie qui est installée sur le PC qui est chargée de visualiser ces données sur interface.

Dans le cadre de ce projet, nous avons utilisé Contiki comme système d'exploitation, le langage C pour programmer les capteurs et le langage Java pour générer des interfaces ergonomiques.

Bibliographie

Bibliographie

[1] I.F. Akyildiz et al., Wireless sensor networks: a survey, Computer Networks 38 (4) (2002) 393–422.

[2] Y. Zhang and H. Xiao, “Bluetooth-based sensor networks for remotely monitoring the physiological signals of a patient,” IEEE Transactions on Information Technology in Biomedicine, vol. 13, no. 6, pp. 1040–1048, November 2009.

[3] D.-M. Han and J.-H. Lim, “Smart home energy management system using ieee 802.15.4 and zigbee,” IEEE Transactions on Consumer Electronics, vol. 56, no. 3, pp. 1403–1410, August 2010.

[4] C. Mettu. (2011) Telosb datasheet @ONLINE. [Online]. Available : https://fr.scribd.com/doc/68138250/Telosb-Datasheet-t

[5] M. Lehsaini, “Diffusion et Couverture basées sur le clustering dans les réseaux de capteurs”, Thèse de Doctorat, Université de Tlemcen et Université de Franche-Comté, 2009.

[6] A. Dunkels, B. Gronvall, and T. Voigt. ”Contiki a Lightweight and Flexible Operating System for Tiny Networked Sensors”, In Proceedings of the 9th Annual IEEE International Conference on Local Computer Networks, Washington, DC, USA, 2004.

[7] http ://www.tinyos.net/tinyos-1.x/doc/nesc/ref.pdf.

[8] B. Malli. (2011, May) Micaz datasheet @ONLINE. [Online]. Available : https://fr.scribd.com/doc/56641260/Micaz-Datasheet

[9] R. Nath. “A TOSSIM based implementation and analysis of collection tree protocol in wireless sensor networks”. International Conference on Communications and Signal Processing (ICCSP), 2013.

[10] P. Kugler, P. Nordhus, and B. Eskofier. “Shimmer, Cooja and Contiki : A New Toolset for the Simulation of On-node Signal Processing Algorithms”, IEEE International Conference on Body Sensor Networks (BSN), 2013.

Résumé

Les progrès technologiques réalisés ces dernières années ont permis le développement de nouveaux types de capteurs dotés de moyens de communication sans fil, peu onéreux et pouvant être configurés pour former des réseaux autonomes. Les limites imposées sont la limitation des capacités de traitement, de stockage et surtout d'énergie. Le rôle d'un capteur est la surveillance de la zone géographique et de l'émission de l'alerte quand il y a un évènement anormal (incendie, tremblement de terre, éruption volcanique). Dans ce mémoire nous avons développé une application qui permet l'acquisition de la température en utilisant le système d'exploitation Contiki et une plateforme réels de capteurs TelosB. Notre réseau de capteurs est composé de trois types de capteurs, les capteurs clients qui vont capter la température et l'envoyer à la station de base (deuxième type) via un nœud relai (troisième type). Puis nous avons visualisé leurs captures via une interface Java.

Abstract

Technological advanced during the last few years allowed the development of new and sheap sensors equiped with wireless communication which can be configured to form autonomous networks. The imposed constraints are limited capacity of processing , storage and especialy energy. The role of a sensor is monitoring the geographical area and the isuance of the alert when there is an abnormal event (fire, earthquake, volcanic ereption). In this w work we developped an application that allows the acquisition of the temperature via TelosB platforms using a Contiki operating system. The deployed sensor network consists of three types of sensors, « Sender » sensor that will capture temperature and sent to the base station (second type) via a relay node (third type). Then, we presented the results using a java interface.

ملخص

التكنولوجيا المتقدمة خلال السنوات القليلة الماضية سمحت بتطوير أجهزة استشعار جديدة ورخيصة مجهزة للاتصالات اللاسلكية التي يمكن تهيئتها لتشكيل شبكات مستقلة. هذه الشبكات مقيدة من حيث محدودية قدرة المعالجة، التخزين وخاصة الطاقة. دور جهاز الاستشعار هو مراقبة المناطق الجغرافية و إصدار تنبيه في حالة حدوث حدث غير طبيعي (الحرائق الزلازل وثوران البراكين). في هذه المذكرة نقدم تطبيقا يسمح بالتقاط درجة الحرارة بواسطة أجهزة استشعار ونظام تشغيل صنع خصيصا لشبكات الاستشعار اللاسلكية. تتكون شبكتنا اللاسلكية على ثلاثة أنواع من أجهزة الاستشعار أجهزة من النوع الأول دوره في الشبكة هو التقاط درجة الحرارة وإرسالها إلى المحطة الأساسية (النوع الثاني) عبر جهاز استشعار وسطي (النوع الثالث) وعرض النتائج في واجهة المستخدم الرسمية .