République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université Mouloud Mammeri de Tizi-Ouzou

Faculté de génie électrique et informatique

Département informatique

Mémoire de fin d'étude en vue de l'obtention du diplôme de Master en informatique

Spécialité RMSE

Thème :

Déploiement d'un réseau de capteurs sans fil en technologie ZigBee

Proposé par :

Mr Mehammed DAOUI.

Réalisé par :

RABIA Fatima Mélissa

TAZIBT Celia Yasmine

Année 2014/2015

Remerciements

Nous remercions d'abord Dieu tout puissant de nous avoir donné la force nécessaire de mener à bien ce travail.

Au terme de ce travail nous tenons à remercier notre promoteur ***Mr DAOUI*** *de nous avoir pris sous son aile pour la réalisation de ce travail grâce à ses précieux conseils et à son aide incommensurable.*

Nous tenons aussi à remercier le professeur Mustapha LALAM de nous faire l'honneur de présider notre jury.

Notre gratitude va aussi à Mr M.LAGHROUCHE et Mme R.AOUDJIT pour avoir accepté de juger ce travail.

Dédicaces

On dédie ce modeste travail à toute personne ayant contribué de près ou de loin à l'élaboration de ce travail et plus particulièrement :

A nos chers parents qui ont toujours été là pour nous,

A tous les membres de notre famille grands et petits veuillez trouver dans ce modeste travail l'expression de notre affection,

A mon très cher Midou qui m'a tenue main forte et m'a toujours soutenue,

A Youcef qui a toujours cru en moi et m'a encouragée.

A notre chère amie Lilia avec qui nous avons partagé ce cursus

A Lynda, Sedik et Lyes pour les échanges et partages d'expérience durant l'élaboration de ce mémoire, à Mayssa et Mira qui m'ont encouragé

Et à tous ceux qui nous ont aidés et soutenues tout au long de ce parcours.

Célia et Mélissa

Table des matières

Introduction générale.. 1

Chapitre I : Généralités sur les réseaux de capteurs sans fil (RCSF).

I.1 Introduction 3

I.2 Les réseaux de capteurs 3

I.2.1 Définition d'un nœud capteur 3

I.2.2 Architecture d'un nœud capteur 4

I.2.2.a Architecture matérielle 4

I.2.2.b Architecture logicielle 5

I.2.3 Définition et architecture d'un réseau de capteurs sans fil 7

I.2.3.a Définition d'un réseau de capteurs sans fil 7

I.2.3.b Les types d'architecture des RCSF 7

I.2.4 Les différentes topologies dans les RCSF 9

I.2.5 Facteurs de conception d'un réseau de capteurs sans fil 10

I.2.6 Communication dans les RCSF 11

I.2.6.a Pile protocolaire 11

I.2.6.b Standards de communication dans les RCSF 14

I.2.7 Domaines d'application dans les RCSF 16

I.2.8 Exemples d'équipements de réseaux de capteurs 17

I.2.8.a Les capteurs Crossbow 17

I.2.8.b Digi International(MaxStream) 19

I.3 Conclusion 21

Chapitre II : La norme ZigBee.

II.1 Introduction 22

II.2 Qu'est-ce que ZigBee ? 22

II.3 Historique 22

II.4 La norme IEEE 802.15.4 23

II.4.1 Architecture 23
II.4.2 Topologie 24
II.5 Architecture protocolaire de la norme IEEE 802.15.4/ZigBee 24
II.5.1 La couche physique (PHY) 25
II.5.2 La couche liaison de données (LNK) 26
II.5.3 La couche réseau (NWK) 28
II.5.4 La couche application (APL) 29
II.6 Les réseaux ZigBee 29
II.6.1 Les éléments du réseau ZigBee 29
II.6.2 La formation d'un réseau PAN ZigBee 30
II.6.3 La communication dans les réseaux ZigBee 32
II.7 Interface ZigBee (le module XBee) 34
II.7.1 Le module XBee 34
II.7.2 Caractéristiques des modules XBee 34
II.7.3 Communication avec le module XBee 36
II.7.4 Formation des réseaux XBee 37
II.7.5 Les protocoles d'interface série des modules XBee 38
II.7.6 Les modes de fonctionnement 39
II.8 Quelques applications ZigBee 41
II.8.1 Système de surveillance d'un patient 41
II.8.2 Système d'éclairage de rue sans fil intelligent 42
II.9 Conclusion 44

Chapitre III : Conception d'un réseau de capteurs sans fil en technologie ZigBee.

III.1 Introduction 45
III.2 Analyse et conception 45
III.3 Analyse 45
III.3.1 Objectifs 45
III.3.2 Analyse de l'architecture 46
III.3.3 Fonctionnement du réseau 47

III.4 Conception **49**
III.4.1 Conception matérielle **49**
III.4.1.1 Le nœud capteur **50**
III.4.1.2 Le nœud routeur **52**
III.4.1.3 Le nœud actionneur **53**
III.4.1.4 Le nœud coordinateur **54**
III.4.2 Conception logicielle **55**
III.4.2.1 Structure de l'application **55**
III.4.2.2 Comportement de l'application **56**
III.5 Domaines d'utilisation **57**
III.6 Conclusion **59**

Chapitre IV : Réalisation.

IV.1 Introduction **60**
IV.2 Outils de développement **60**
IV.3 Réalisation **63**
IV.3.1 Configuration des modules XBee **63**
IV.3.2 Implémentation de l'application de gestion du RCSF **65**
IV.4 Câblage et tests du réseau **69**
IV.4.1 Les composants **69**
IV.4.2 Tests de capture de données **70**
IV.4.3 Tests de fonctionnement des routeurs dans le réseau **75**
IV.5 Conclusion **81**
Conclusion générale et perspectives **82**

Références bibliographiques **83**

Annexes

Annexe I : Bibliothèques XBee **84**
Annexe II : Commandes AT **88**

Liste des figures

Chapitre I : Généralités sur les réseaux de capteurs sans fils.

Figure I.1 : Architecture d'un nœud capteur **4**

Figure I.2 : Architecture du système d'exploitation MANTIS-OS **6**

Figure I.3 : Architecture d'un réseau de capteurs **7**

Figure I.4 : Architecture plate des RCSF **8**

Figure I.5 : Architecture hiérarchique des RCSF **8**

Figure I.6 : Pile protocolaire des RCSF **11**

Figure I.7 : Capteur MicaZ **18**

Figure I.8 : Capteur TelosB **19**

Figure I.9 : Module XBee Série2 ..... **20**

Chapitre II : La norme ZigBee.

Figure II.1 : Différentes topologies du standard IEEE 802.15.4/ZigBee **24**

Figure II.2 : Pile protocolaire ZigBee **25**

Figure II.3 : Structure des trames **26**

Figure II.4 : Rejoindre un réseau ZigBee **32**

Figure II.5 : Brochage du module XBee **35**

Figure II.6 : Vue externe du module XBee série 2 ..... **36**

Figure II.7 : Vue interne du module XBee **36**

Figure II.8 : Structure d'une trame API **39**

Figure II.9 : Transmission de données **40**

Figure II.10 : Schéma fonctionnel du système (a) Coté patient, (b) Coté moniteur **42**

Figure II.11 : Schéma du lampadaire avec les capteurs **43**

Chapitre III : Conception d'un réseau de capteurs en technologie ZigBee.

Figure III.1 : Etapes de réalisation **46**

Figure III.2 : Architecture du réseau **47**

Figure III.3 : Interaction entre les nœuds du réseau **48**

Figure III.4 : fonctionnement du réseau **49**

Figure III.5 : Capteur de température LM35 ..... **50**

Figure III.6 : Branchement du nœud capteur au capteur de température **51**

Figure III.7 : Branchement du nœud capteur avec le capteur d'intrusion ILS **52**

Figure III.8 : Branchement du nœud routeur **53**

Figure III.9 : Branchement du nœud actionneur **53**

Figure III.10 : Station d'accueil USB pour XBee **54**

Figure III.11 : Branchement du nœud coordinateur au PC **54**

Figure III.12 : Structure de l'application **55**

Figure III.13 : Interaction entre les classes **57**

Chapitre IV: Réalisation.

Figure IV.1 : Architecture de la bibliothèque XBee **61**

Figure IV.2 : Ouverture du logiciel X-Ctu **63**

Figure IV.3 : Sélection du port de communication du coordinateur **64**

Figure IV.4 : Choix du firmware **64**

Figure IV.5 : Représentation graphique de notre réseau **69**

Figure IV.6 : Câblage du capteur ILS **70**

Figure IV.7 : Résultat d'exécution du programme avec interrupteur fermé **71**

Figure IV.8 : Ouverture de l'interrupteur ILS **71**

Figure IV.9 : Résultat d'exécution du programme avec interrupteur ouvert **72**

Figure IV.10 : Câblage du capteur LM35 ..... **73**

Figure IV.11 : Résultat d'exécution du programme de traitement de température..... **73**

Figure IV.12 : Résultats de l'exécution après atteinte du seuil **74**

Figure IV.13 : Action associée au dépassement du seuil de température prédéfini **75**

Figure IV.14 : Emplacement des modules dans le Laboratoire LARI..... **76**

Figure IV.15 : Table de routage du réseau **76**

Figure IV.16 : Architecture de notre réseau avec deux routeurs en série **77**

Figure IV.17 : Table de routage du réseau après débranchement du routeur **77**

Figure IV.18 : Vue graphique du réseau après débranchement du nœud routeur **78**

Figure IV.19 : Architecture du réseau avec deux routeurs en parallèle **79**

Figure IV.20 : Architecture du réseau après débranchement d'un routeur..... **79**

Liste des acronymes

- RCSF : Réseaux de Capteurs Sans Fil.
- WSN : Wireless Sensor Network.
- GPS : Global Positionning System.
- ADC : Analog to Digital Converter.
- NesC : Network embedded system C.
- FIFO : First In First Out.
- MANTIS OS : MultimodAl NeTworks of In-situ micro Sensor Operating System.
- RTOS : Real Time Operating System.
- UAV : Unmanned Air Vehicle.
- MAC : Media Access Control.
- TCP : Transmission Control Protocol.
- UDP : User Datagram Protocol.
- IRDA : Infrared Data Association.
- UWB : Ultra Wide Band.
- USB : Universal Serial Bus.
- IEEE : Institure of Electrical and Electronics Engineers.
- ISM : The industrial, scientific and medical.
- PAN : Personal Area Network.
- WPAN : Wireless Personal Area Network.
- RFD : Reduced Function Devices.
- FFD : Full Function Devices.
- SAP : Service Access Point.
- ED : Energie Detection.
- LQI : Link Quality Indication.
- ACK : ACKnowlodgement.
- MSB : Most Significant Bit.

Liste des acronymes

- LSB : Least Significant Bit.
- MPDU : MAC Protocol Data Unit.
- MHR : MAC HeadeR.
- MSDU : MAC Service Data Unit.
- MFR : MAC FouteR.
- CSMA-CA : Carrier Sense Multiple Access Colliонsion Avoidance.
- LLC : Logical Link Control.
- CRC : Cyclic Redundancy Check.
- APL : Application Support Layer.
- SSP : Security Service Provider.
- APS : APplication support Sub-layer.
- ZDO : Zigbee Device Object.
- CTS : Clear To Send.
- RTS : Ready To Send.
- LED : Light Emitting Diod.
- ILS : Interrupteur à Lames Sous-vides.
- JRE :JAVA Runtime Environment.
- JDK : Java Development Kit.
- IDE : Integrated Development Environment.
- LEACH : Low Energy Adaptative Clustering Hierarchy.

Introduction générale

Les progrès réalisés ces dernières décennies dans les domaines de l'électronique et des technologies de communication sans fil, ont permis de produire à un coût raisonnable des composants de quelques millimètres cubes de volume. De ce fait, un nouveau domaine de recherche s'est créé pour offrir des solutions économiquement intéressantes et faciles à déployer pour la surveillance à distance et au traitement des données dans les environnements complexes et distribués, l'une de ces solutions se présente sous le nom des « réseaux de capteurs sans fils ».

Le domaine des réseaux de capteurs sans fils (RCSF), aussi connu sous le sigle WSN (pour « Wireless Sensor Network »), est considéré comme l'une des technologies contemporaines les plus prometteuses. Un réseau de capteurs sans fil est constitué de nœuds (dits nœuds capteurs) de petite taille, à bas prix et visant une autonomie énergétique. Les nœuds capteurs ont des capacités de mesures, de stockage, de traitement des données et de communications sans fil.

Au sein d'un réseau de capteurs sans fil, les nœuds capteurs effectuent des mesures et collaborent pour les transmettre à des nœuds puits, qui sont des nœuds particuliers ayant des capacités importantes de stockage et de ressources énergétiques. Une fois arrivées dans le puits, les données issues des capteurs sont disponibles pour des applications centralisées. Les RCSF possèdent de nombreuses applications telles que des applications militaires, la surveillance de l'environnement et le suivi de biens ou de personnes, etc… Ces applications, bien que très différentes, présentent le besoin commun d'être constituées de réseaux composés de nœuds à faible débit, à très faible coût et à très faible consommation.

Les standards de communication pour les réseaux de capteurs sans fils doivent être conçus en respectant certains facteurs de conception (contraintes), entre autres le temps de latence faible ainsi qu'une consommation d'énergie très basse. D'où la nécessité de concevoir des normes sans fil capables de répondre à ces exigences. Parmi les standards les plus aptes à être exploités dans les réseaux de capteurs sans-fil se retrouvent *IRDA*, le *Bluetooth, UWB* et la *technologie ZigBee.*

L'objectif de notre travail consiste à déployer un réseau de capteurs sans fils se basant sur le standard de communication « ZigBee ». Le réseau ainsi déployé pourra servir à de nombreuses applications personnelles et industrielles. Pour cela nous avons utilisé son

interface matérielle connue sous le nom « XBee ». La norme ZigBee repose sur une solide fondation constituée par le standard IEEE 802.15.4. Utilisant une technologie simple spécifiquement étudiée pour minimiser la consommation en énergie globale du réseau, et permettant d'alimenter les nœuds par simples piles, si nécessaire. ZigBee garantit une grande qualité de service, par l'emploi de codes de détection d'erreurs, par la modulation utilisée (étalement de spectre), et par la capacité de changer de fréquence s'il y'a détection d'interférences nuisant aux communications. Avec un total de 27 canaux de communications possibles, partagés sur 3 bandes de fréquences (868 MHz, 915 MHz et 2,4 GHz), le protocole ZigBee/802.15.4 s'affirme comme un standard de couverture mondiale.

Le travail présenté dans ce mémoire est organisé en quatre chapitres :

Le **chapitre 1** donne un aperçu sur les réseaux de capteurs sans fils et leurs caractéristiques.

Le **chapitre 2** est consacré à l'étude du standard de communication sans fils ZigBee en présentant une de ses interfaces matérielle qui est le module « XBee ».

Dans le **chapitre 3** nous aborderons l'approche que nous avons adoptée pour l'analyse et la conception de notre projet de déploiement du réseau de capteurs sans fil. En présentant les démarches de conception matérielle et logicielle.

Dans le **Chapitre 4** nous allons décrire les étapes de réalisation de notre application de gestion de RCSF. Nous avons également présenté les différents tests effectués sur ce dernier ainsi que les résultats obtenus.

Chapitre I : Généralités sur les réseaux de capteurs sans fil.

I.1 Introduction :

Dans le milieu industriel et numérique il est devenu nécessaire, depuis quelques décennies, d'observer et de contrôler certains phénomènes physiques tels que la température, le degré de pollution, …etc. Ces observations sont parfois effectuées dans des environnements hostiles et inaccessibles.

Grâce aux récentes avancées technologiques dans le domaine des communications sans fil, et aussi grâce à la disponibilité de capteurs à faible coût munis de puissantes capacités de calcul et de perception, les réseaux de capteurs sans fil sont considérés comme le plus grand essor des avancements technologique du 20ème siècle.

Ce type de réseaux est amené à résoudre divers problèmes. Aujourd'hui on peut facilement envisager un large éventail d'application : surveillance environnementale, la sécurité, la santé, … etc.

Dans ce chapitre, nous allons présenter les réseaux de capteurs sans fil (RCSF). Nous allons d'abord présenter la définition d'un RCSF, ses caractéristiques et facteurs de conception. Ensuite, nous allons décrire son architecture et ses domaines d'application. Pour finir, nous décrirons son architecture protocolaire et ses standards de communication.

I.2 Les réseaux de capteurs :

Les récentes avancées dans les domaines des technologies sans-fil et électroniques ont permis le développement à faible coût de minuscules dispositifs consommant peu d'énergie appelés « nœuds capteurs ».

I.2.1 Définition d'un nœud capteur :

Un nœud capteur est un mini-composant ayant 3 fonctions principales : capter des données (de type son, vibration, lumière,...), calculer des informations à l'aide de ces valeurs collectées et les communiquer à travers un réseau de capteurs.

Les données captées par les nœuds sont acheminées grâce à un routage multi-saut à un nœud considéré comme un "point de collecte" appelé nœud-puits (ou sink). Ce dernier peut être connecté à l'utilisateur du réseau via Internet, un satellite ou un autre système. L'usager

peut adresser des requêtes aux autres nœuds du réseau, précisant le type de données requises pour récolter les données environnementales captées par le biais du nœud-puits **[5].**

I.2.2 Architecture d'un nœud capteur :

I.2.2.a Architecture matérielle :

Un nœud capteur est un ensemble de quatre unités de base : unité de captage, unité de traitement, unité de communication et l'unité de contrôle d'énergie. Des composants additionnels peuvent être ajoutés selon le domaine d'application, comme par exemple un système de localisation tel qu'un GPS (Global Positionning System), un générateur d'énergie ou un mobilisateur lui permettant de se déplacer. Ces éléments sont décrits dans la figure ci-dessous :

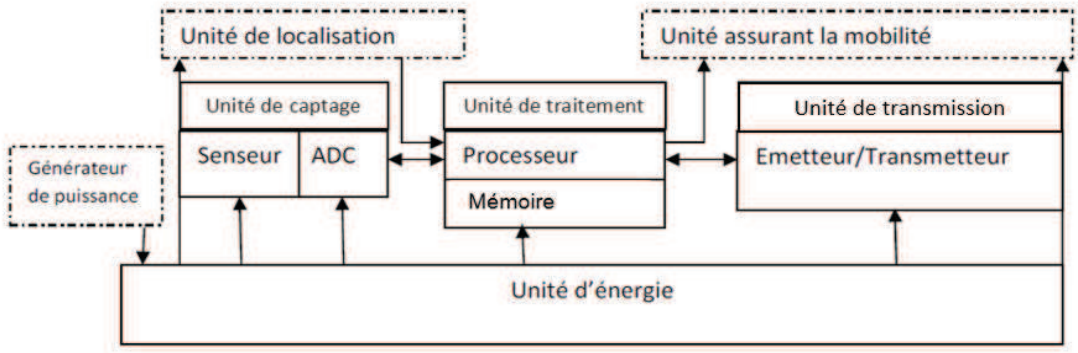

Figure I.1 : Architecture d'un nœud capteur.

- **Unité de captage (Sensing Unit):** elle est composée de deux sous unités, un dispositif de capture physique qui prélève l'information de l'environnement local, et un convertisseur analogique/numérique appelé ADC (Analog to Digital Converter) qui convertit l'information prélevée et la transmet à l'unité de traitement.
- **Unité de traitement (Processing Unit):** elle est composée d'un processeur et d'une unité de stockage de faible capacité intégrant un système d'exploitation spécifique (TinyOs par exemple). Cette unité possède deux interfaces, une interface pour l'unité de captage et une deuxième pour l'unité de transmission. Elle acquiert des informations en provenance de l'unité de captage et les envoie à l'unité de transmission et elle est chargée d'exécuter les protocoles de communication qui permettent de faire collaborer le nœud capteur avec les autres nœuds du réseau.

- **Unité de transmission (Transceiver Unit):** Elle effectue toutes les émissions et les réceptions de données sur un médium sans fil. Trois mediums de communication sans fil peuvent être utilisés : LASER, infrarouge et radiofréquences (RF). La RF reste le moyen le plus répandu pour la communication des capteurs.
- **Unité de contrôle d'énergie (Power Unit):** un capteur est muni d'une batterie pour alimenter tous ses composants. Cependant, à cause de sa taille réduite, la batterie dont il dispose est limitée et généralement irremplaçable. Pour cela l'énergie est la ressource la plus précieuse puisque elle influe directement sur la durée de vie des capteurs et donc d'un réseau de capteurs **[1].**

I.2.2.b Architecture logicielle (Les systèmes d'exploitation pour les capteurs):

Plusieurs systèmes d'exploitation pour capteur ont été implémentés, chacun offrant des fonctionnalités propres. Dans cette section, nous allons présenter différents système d'exploitation couramment utilisés lors de la programmation de capteurs : TinyOs, MANTIS OS et Free RTOS.

❖ TinyOs :

TinyOs est un système d'exploitation open source conçu pour les réseaux de capteur sans fil. Ils respectent une architecture basée sur une association de composants, réduisant la taille du code nécessaire à sa mise en place. Cela respecte la contrainte de mémoire des capteurs. Ce système d'exploitation est dédié aux capteurs dont la sortie date de 2000. Il supporte plusieurs plateformes dont MicaZ et TelosB.

Les applications pour TinyOs sont écrites en NesC (Network embedded system C), une extension de C. Il adopte une programmation basée évènements, c'est-à-dire que le système attend que les évènements se produisent. Un évènement peut être l'arrivée d'un paquet, la fin d'un timer ou une nouvelle donnée disponible.

Dans TinyOs chaque composant fournit des commandes et réagit à des évènements. La communication entre composants se fait grâce à des interfaces et l'ordonnancement se fait grâce à la politique FIFO.

❖ MANTIS OS : (MultimodAl NeTworks of In-situ micro Sensor)

C'est un système d'exploitation léger et multi-threads pour capteurs. Son architecture est montrée par la figure suivante :

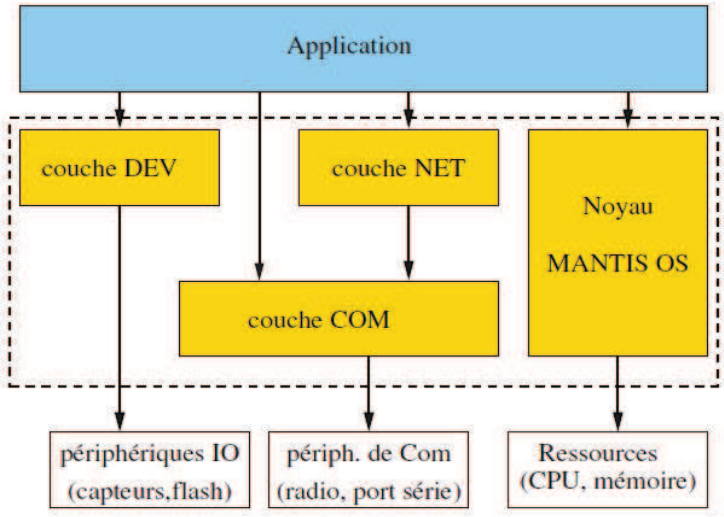

Figure I.2 : Architecture du système d'exploitation MANTIS OS

Il supporte plusieurs plateformes notamment MicaZ et TelosB. La programmation d'application sur MANTIS OS se fait en C. Les trois principaux composants sont :

- la couche DEV qui fournit un accès aux périphériques d'entrée-sortie.
- la couche Net qui gère entre autre la pile de communication.
- la couche COM qui permet d'accéder aux communications tels que le port série ou la RF.

❖ Free RTOS :

Free RTOS est un noyau temps réel pour système embarqué où l'ordonnancement des tâches se fait à intervalle réguliers ou lors d'évènements asynchrones. Une tâche est un processus avec son propre contexte d'exécution, il peut donc exister plusieurs tâches dans une application, chacune possédant une priorité préalablement définie.

La communication entre tâches se fait à l'aide de files, de sémaphores ou de mutex pour partager une ressource commune.

I.2.3 Définition et architecture d'un réseau de capteurs sans fil :

I.2.3.a Définition d'un réseau de capteurs sans fil :

Un **réseau de capteurs sans fil** (RCSF), ou Wireless Sensor Network (WSN) en anglais, est un réseau constitué d'un grand nombre de nœuds capteurs capables de récolter et de transmettre des données environnementales d'une manière autonome. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils peuvent être aléatoirement déployés dans une zone géographique, appelée *« champ de captage »* correspondant au terrain d'intérêt pour le phénomène capté. Les nœuds capteurs acheminent les données captées via la liaison RF vers un nœud considéré comme « point de collecte » appelé nœud puits (sink en anglais). Ce dernier peut être connecté à une machine via internet ou par satellite, et permet de collecter et stocker les informations issues des capteurs. La figure ci-dessous schématise l'architecture d'un RCSF**.**

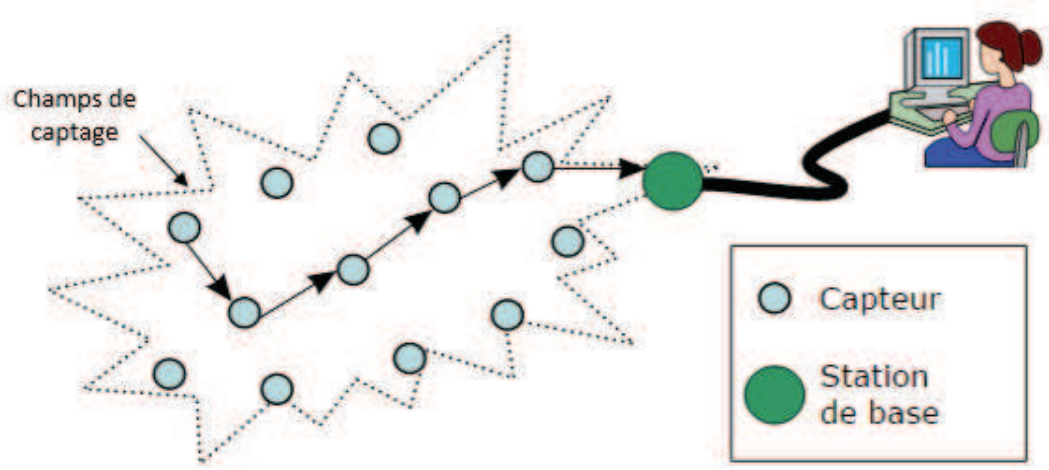

Figure I.3 : Architecture d'un réseau de capteurs.

I.2.3.b Les types d'architectures des RCSF :

- **Les réseaux de capteurs sans fil plat :** Dans la topologie plate d'un RCSF, l'ensemble des nœuds capteurs ont le même rôle et sont reliés entre eux dans le but de créer des chemins qui atteignent la station de base. La Figure ci-dessous montre un exemple d'une topologie plate d'un RCSF. Le nœud S désigne la station de base et les nœuds de couleur gris foncé désignent des capteurs. Les données sont routées d'un capteur à un autre afin d'arriver à la destination S.

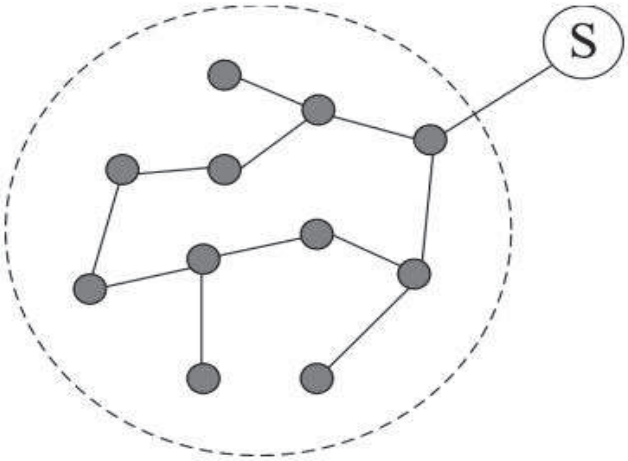

Figure I.4 : Architecture plate des RCSF

- **Les réseaux de capteurs sans fil hiérarchique :** Dans la topologie hiérarchique d'un RCSF, tous les nœuds capteurs n'ont pas le même rôle. Afin d'augmenter la scalabilité du système, les topologies hiérarchiques ont été introduites en divisant les nœuds en plusieurs niveaux de responsabilité. L'une des méthodes les plus employées est le clustering, où le réseau est partitionné en groupes appelés "clusters". Un cluster est constitué d'un chef (cluster-head) et de ses membres. Dans la figure qui suit, nous donnons un exemple de topologie hiérarchique :

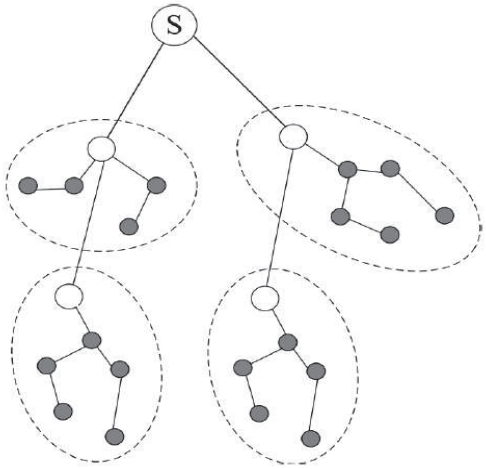

Figure I.5 : Architecture hiérarchique des RCSF.

I.2.4 Les différentes topologies dans les RCSF :

❖ La topologie en étoile :

Dans cette topologie une station de base peut envoyer ou recevoir des messages à un certain nombre de nœuds. Ces nœuds peuvent seulement envoyer un message à l'unique station de base ou en recevoir des messages.

L'avantage de cette topologie est sa simplicité, sa capacité à minimiser la consommation d'énergie des nœuds et la minimisation de la latence de communication entre les nœuds et la station de base. Son inconvénient est que la station de base n'est pas robuste puisque tout le réseau est géré par un seul nœud.

❖ La topologie en toile ou grille (Mesh network) :

Dans ce type de topologie, n'importe quel nœud peut envoyer à n'importe quel autre nœud dans le réseau qui est dans sa portée de transmission. Ceci est appelé la communication multi-sauts, dans laquelle, si un nœud veut transmettre un message à un autre nœud qui est en dehors de sa portée de transmission, il utilise un nœud intermédiaire pour envoyer son message au nœud destinataire. L'avantage de cette topologie est la possibilité du passage à l'échelle, la redondance et la tolérance aux fautes. L'inconvénient de cette topologie est la consommation d'énergie dans la communication multi-sauts et la latence qui sont créés par le passage des messages entre nœuds avant d'arriver à la station de base.

❖ La topologie hybride :

Une topologie hybride entre celle en étoile et en grille fournit des communications réseau robustes et diverses, en assurant la minimisation d'énergie dans les réseaux de capteurs. Dans ce type de topologie, les nœuds capteurs à faible puissance ne routent pas les messages, mais il y a d'autres nœuds qui ont la possibilité de le faire. En général, ces nœuds ont une puissance élevée.

I.2.5 Facteurs de conception d'un réseau de capteurs sans fil :

Les réseaux de capteurs sont influencés par plusieurs facteurs de conception, qui représentent la base de la conception des algorithmes et protocoles utilisés. Ces facteurs sont représentés ci-dessous :

- **Tolérance aux pannes :**

  Les nœuds capteurs peuvent tomber en panne à cause de l'absence d'énergie ou des interactions externes (dommage physique, interférences environnementales) qui causent leur disfonctionnement.

  La propriété de tolérance aux pannes est définie par l'habilité du réseau à maintenir ses fonctionnalités sans interruptions provoquées par la panne des capteurs. Elle vise donc à minimiser l'influence de ces pannes sur la tâche globale du réseau. **[2]**

- **Scalabilité :**

  Le réseau de capteurs doit être scalable c'est-à-dire être capable d'accepter un très grand nombre de nœuds tout en permettant l'augmentation de ce nombre et la densité de ce dernier dans une région (pouvant dépasser 20 nœuds/m^3).

- **Coût de fabrication :**

  Le coût de production d'un seul micro-capteur est très important pour l'évaluation du cout global du réseau, si ce dernier est supérieur à celui nécessaire pour le déploiement des capteurs classiques, l'utilisation de cette nouvelle technologie ne serait pas rentable. Par conséquent, réduire le coût de production jusqu'à moins de 1$ par nœud est un objectif important pour la faisabilité de la solution de réseaux de capteurs sans fil.

- **Topologie du réseau :**

  Il faut que les nœuds capteurs soient capables d'adapter leur fonctionnement afin de maintenir la topologie souhaitée.

  On distingue généralement trois phases dans la mise en place et l'évolution d'un réseau :

  - Phase de déploiement : les nœuds capteurs sont soit lancés en masse soit placés un à un dans des champs de captage. Il faut alors que ceux-ci s'organisent de manière autonome.
  - Phase de post déploiement : durant cette phase la topologie du réseau peut être soumise à des changements dus à des modifications de la position des nœuds ou bien à des pannes.
  - Phase de redéploiement : l'ajout de nouveaux capteurs dans un réseau existant implique aussi la mise à jour de la topologie.

- **Consommation d'énergie :**

  L'économie d'énergie est une des problématiques majeures dans les réseaux de capteurs. En effet, la recharge des ressources énergétiques est souvent trop couteuse est parfois impossible. Il faut donc que les capteurs économisent l'énergie afin de pouvoir maximiser la durée de vie du réseau **[4]**.

I.2.6 Communication dans les RCSF :

I.2.6.a Pile Protocolaire :

Les réseaux de capteurs sans fil utilisent une pile protocolaire représentée dans la **figure I.6**. Ces protocoles doivent prendre en compte des contraintes de conception propres aux RCSF vu dans la section **I.2.5**. Cette pile protocolaire utilisée par la station de base ainsi que tous les autres capteurs du réseau comprend 5 couches (la couche application, la couche transport, la couche réseau, la couche liaison de données et la couche physique) ainsi que 3 plans (plan de gestion des tâches, plan de gestion de la mobilité et plan gestion de l'énergie. Ces derniers sont accessibles par toutes les couches de la pile.

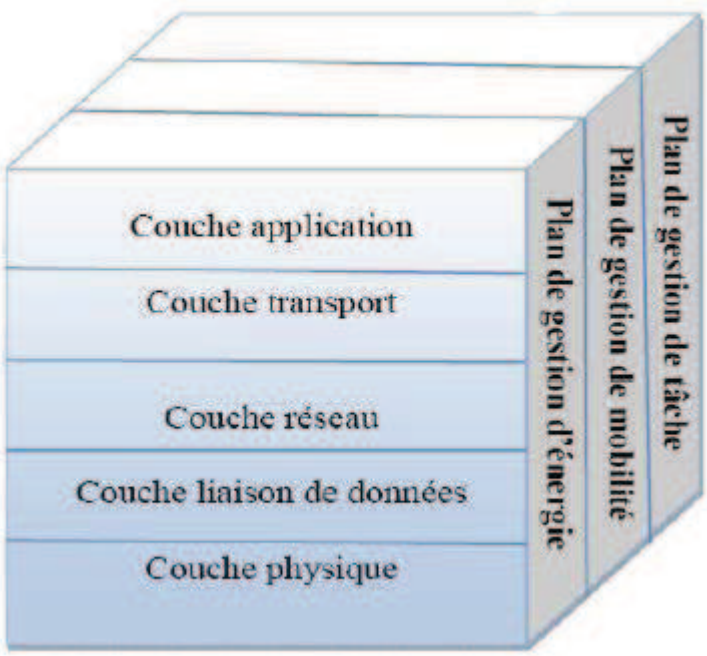

Figure I.6 : Pile Protocolaire des RCSF

➢ Couche physique :

Elle est responsable de la sélection de fréquence, de la génération de la fréquence de porteuse, de la détection du signal, de la modulation/démodulation et du cryptage/décryptage des données. La consommation d'énergie au niveau de cette couche peut être affectée par le choix du schéma de modulation/démodulation ou la bande de fréquence utilisée.

➢ Couche liaison de données :

Le protocole MAC (Media Access Control) de cette couche assure la gestion de l'accès au support physique. Cette couche permet de montrer comment les données sont expédiées entre deux nœuds. Elle est responsable du multiplexage des flots de données, de la détection de la trame de données, du contrôle d'erreur et de l'accès au média
. Elle assure de manière fiable une liaison point-à-point et point-multipoint dans les réseaux de communication.

➢ Couche réseau :

Les nœuds capteurs sont répartis de manière dense et aléatoire dans un champ de captage pour observer un phénomène. Pour permettre la communication dans le réseau, des protocoles de routage spéciaux basés sur la communication multi-sauts sont nécessaire entre les nœuds capteurs et le nœud puits du réseau.

La conception de la couche réseau dans un réseau de capteur doit être guidée par les principes suivants :

- L'efficacité en consommation d'énergie est une considération importante.
- Les réseaux de capteurs sont généralement centrés données (Data Centric).
- Un réseau de capteur idéal doit disposer d'un moyen de localisation.

➢ Couche transport :

Cette couche constitue une interface entre la couche application et la couche réseau. Les principales fonctionnalités de cette couche sont :

- Le multiplexage/démultiplexage des messages entre les applications et la couche réseau.

- Le contrôle haut niveau de données.
- Découpage des données en paquets.

Le rôle de cette couche intervient essentiellement lorsque le réseau de capteurs sera accédé via internet ou d'autres réseaux externes. Les mécanismes de communication du protocole TCP ne sont pas compatibles avec les RCSF à cause de l'énergie et la mémoire limitées des nœuds capteurs. De ce fait, la communication entre les utilisateurs et le nœud puits sera via les protocoles TCP ou UDP, et les communications entre le nœud puits et les nœuds capteurs du réseau peuvent être via le protocole UDP.

➢ Couche application :

La couche application constitue l'ensemble des applications implémentées sur un réseau de capteurs. Ces applications doivent fournir des mécanismes permettant à l'utilisateur d'interagir avec le réseau de capteurs à travers différentes interfaces, et éventuellement par l'intermédiaire d'un réseau étendu (par exemple : internet). Il s'agit donc du niveau le plus proche des utilisateurs, géré directement par les logiciels **[1].**

➢ Plan de gestion d'énergie :

Il contrôle l'utilisation de la batterie. Par exemple, après la réception d'un message, le capteur éteint son récepteur afin d'éviter la duplication des messages déjà reçus. En outre, si le niveau d'énergie devient bas, le nœud diffuse à ses voisins une alerte les informant qu'il ne peut pas participer au routage. L'énergie restante est réservée au captage **[2].**

➢ Plan de gestion de la mobilité :

Puisque les nœuds peuvent être mobiles, un système de gestion de mobilité doit exister. Un tel système doit être capable d'enregistrer les mouvements du nœud afin de l'aider à se localiser.

➢ Plan de gestion de tâche :

Le niveau de gestion des tâches assure l'équilibrage et la distribution des tâches sur les différents nœuds du réseau, afin d'assurer un travail coopératif et efficace en matière de consommation d'énergie, et par conséquent, prolonger la durée de vie du réseau.

I.2.6.b Standards de communication dans les RCSF :

Les standards de communication pour les réseaux de capteurs sans fil doivent être conçus en respectant les facteurs de conception (contraintes) cités dans la section I.2.5, entre autres le temps de latence faible ainsi qu'une consommation d'énergie très basse. D'où la nécessité de concevoir des normes sans fil capables de répondre à ces exigences. Parmi les standards les plus aptes à être exploités dans les réseaux de capteurs sans-fil se retrouvent *IRDA*, le *Bluetooth, UWB (Ultra Wide Band)* et la *technologie ZigBee.*

- L'*IRDA* qui est un acronyme de *Infrared Data Association*, communément appelé infrarouge, est un moyen de communication qui utilise comme médium la lumière infrarouge pour transmettre des informations. Le standard initial, normalisé en 1994, offrait un débit de 115 Kbit/s qui a abouti en 1999 à une extension allant jusqu'à 16 Mbits/s. La principale caractéristique de l'IRDA qui est aussi son principal inconvénient est que deux périphériques utilisant cette technologie doivent être en ligne de vue pour pouvoir communiquer. En effet, les transmetteurs IRDA étant directifs avec un angle d'environ 15°, ils doivent être bien orientés pour communiquer. Cette contrainte de fonctionnement pour cette technologie qui a une faible consommation électrique fait qu'au niveau sécurité elle limite les possibilités d'interception du signal. Au début des années 1990, l'IRDA était très utilisé pour les téléphones portables, les ordinateurs et d'autres périphériques mais avec l'apparition de Bluetooth, la donne a changé.

- Le *Bluetooth*, ce standard de communication créé en 1994 par Ericsson, a été défini à la base pour remplacer les câbles et combler les lacunes de l'IRDA qui était populaire à l'époque. Il a été standardisé sous la norme IEEE 802.15.1 et a comme but la création et le maintien de réseaux à portée personnelle, PAN (Personal Area Network). Un tel réseau est utilisé pour le transfert de données à bas débit à faible distance entre appareils compatibles. Malheureusement, le grand défaut de cette technique est sa trop grande consommation d'énergie et ne peut donc pas être utilisée par des capteurs qui sont alimentés par une batterie et qui idéalement devraient fonctionner durant plusieurs années.

- *L'UWB,* est un acronyme de *Ultra Wide Band* ou encore *Ultra large bande passante* en français, a été standardisé sous la norme IEEE 802.15.3. *C'*est une technologie sans fil conçue pour faire communiquer des périphériques grand public sur une courte distance avec un haut débit, tout en considérant une des principales contraintes des réseaux de capteurs sans fil qui est la consommation d'énergie. Cette technologie, qui consomme peu d'énergie, est basée sur une technique d'étalement de spectre permettant de transmettre des données sur un très large spectre en un temps très court. Ce profil fait qu'elle est bien adaptée pour le transport de données nécessitant un haut débit tel que le multimédia. Cependant, le coût des composants et la concurrence d'autres technologies comme l'USB sans fil ou encore Bluetooth fait qu'elle est en perte de vitesse. Cela se traduit par le départ d'un des promoteurs, Intel en novembre 2008 et l'arrêt en Mars 2009 de WiMedia. Néanmoins, il faut noter que les briques de l'UWB se retrouveront dans les spécifications des normes de l'USB sans fil et Bluetooth 3.0. En effet, la norme Bluetooth 3.0.encore appelée Next-Gen Bluetooth ou Bluetooth UWB aura sa couche matérielle basée sur l'UWB permettant d'atteindre un débit de 480 Mbits/s **[3].**

Face à ces différentes technologies, une s'est démarquée considérablement par rapport aux autres pour les réseaux de capteurs : IEEE 802.15.4/ZigBee.

- Le *ZigBee* combiné avec IEEE 802.15.4 offre des caractéristiques répondant encore mieux aux besoins des réseaux de capteurs en termes d'économies d'énergie. Ce standard spécifie les couches basses, MAC et physique, pour les RCSF. Comme l'IEEE ne définit que la couche MAC et la couche physique, un groupe d'entreprises appelé la ZigBee Alliance a spécifié les couches hautes, c'est-à-dire la couche réseau et la couche application pour ce standard allant du routage à l'application, ce qui a donné naissance au protocole ZigBee. Un faible débit, une faible portée et une faible consommation énergétique sont les principales caractéristiques d'un réseau de capteurs classique : c'est pour cette raison que dans notre étude, nous allons nous intéresser à cette technologie sans fil qui utilise, comme Bluetooth et WIFI, la bande ISM de 2,4 GHz et offre un débit de 250Kbit/s. Cette technologie fera l'objet d'une étude plus complète dans le chapitre suivant.

I.2.7 Domaines d'application dans les RCSF :

La diminution de taille et de coût des micro-capteurs, l'élargissement de la gamme des types de capteurs disponibles (thermique, optique, vibrations…) et l'évolution des supports de communication sans fil utilisé, permettent aux réseaux de capteurs d'intégrer plusieurs domaine d'application, tel que le domaine militaire, l'environnement, le domaine médical et l'industrie.

❖ Applications militaires :

L'exploitation militaire est l'une des principales applications des réseaux de capteurs.

Dans ce contexte, l'emploi des réseaux de capteurs peut aller des surveillances de routine des périmètres, jusqu'à assister des attaques aériennes ou terrestres et conduire des opérations d'espionnage.

❖ Applications environnementales :

Les réseaux de capteurs peuvent être utilisés pour surveiller les changements environnementaux. Ils servent à déterminer les valeurs de certains paramètres à un endroit donné, comme par exemple : la température, la pression atmosphérique, etc. En dispersant des nœuds capteurs dans la nature, on peut détecter des événements tels que des feux de forêts, des tempêtes ou des inondations. Ceci permet une intervention beaucoup plus rapide et efficace des secours. Avec les réseaux de capteurs on peut contrôler la pollution, par exemple en déposant des capteurs au-dessus d'un emplacement industriel pour détecter et surveiller des fuites de gaz ou de produits chimiques.

❖ Applications médicales :

Les capteurs peuvent être implantés dans le corps humain pour contrôler les problèmes médicaux comme le cancer et pour aider les patients à maintenir leur santé. Ils peuvent être utilisés aussi pour surveiller les patients et l'avancement de leurs états dans un hôpital. En outre, en implantant sous la peau des mini capteurs vidéo, on peut recevoir des images en temps réel d'une partie du corps sans aucune chirurgie et pendant environ 24h. On peut ainsi surveiller la progression d'une maladie ou la reconstruction d'un muscle. Un projet actuel consiste à créer une rétine artificielle composée de 100 micro-capteurs pour corriger la vue.

❖ Applications domestiques :

Les réseaux de capteurs peuvent également être utilisés dans la domotique et l'environnement intelligent. Ils jouent un rôle essentiel dans les grandes usines et les entrepôts en surveillant les changements climatiques. Par exemple, des capteurs peuvent être utilisés pour contrôler les vibrations susceptibles d'endommager la structure d'un bâtiment. Comme exemple d'application domestique un réseau de capteurs a été déployé dans un campus universitaire, permettant aux différentes machines (serveurs, imprimantes, etc) de tous les départements de communiquer ensemble.

❖ Autres applications :

Parmi les autres applications des réseaux de capteurs, on a : l'agriculture, des capteurs sont incorporés dans la terre pour donner des informations sur l'état du champ. La protection des barrages pourrait être accomplie en y introduisant des capteurs. La détection prompte de fuites d'eau permettrait d'éviter des dégâts. Les êtres humains sont conscients des risques et attaques qui les menacent. Du coup, ils mettent à disposition toutes les ressources humaines et financières nécessaires pour leur sécurité. Grâce aux réseaux de capteurs, les entreprises pourraient offrir une meilleure qualité de service tout en réduisant leurs coûts. Les réseaux de capteurs peuvent également être utilisés pour : surveiller l'infrastructure, lutter contre le terrorisme, contrôler le trafic, détecter des intrusions (en plaçant, à différents points stratégiques, des capteurs, on peut ainsi prévenir des cambriolages), contrôler les stocks (savoir le lieu, la quantité, la forme de tous les produits, contrôler leur flux, …etc.), l'urbanisme, l'ingénierie civile (surveillance des structures, les capteurs peuvent être placés dans les ponts afin de détecter et de signaler les faiblesses structurelles, dans les réservoirs d'eau pour détecter les matières dangereuses), le recouvrement des catastrophes (par exemple chercher des signes de vie après un tremblement de terre), et beaucoup d'autres applications qui rendent notre entourage plus intelligent.

I.2.8 Exemples d'équipements de réseaux de capteurs :

Il existe dans le monde plusieurs fabricants de capteurs pour les réseaux de capteurs, dont Crossbow et Digi dont nous allons citer des exemples :

I.2.8.a Les capteurs Crossbow :

Nous allons citer en exemple deux capteurs MicaZ et TelosB :

❖ MicaZ :

Le module MicaZ est constitué d'une mémoire flash de 512 kbytes permettant de stocker plus de 100 000 mesures, d'une mémoire programme de 128 kbytes ainsi que d'une RAM de 4Kbytes. Ces capteurs ont un processeur « ATMEL AT-Mega 128L » d'une vitesse de 16 MHz. Il fonctionne avec deux piles AA, soit un voltage de 2.7V. Ils communiquent sur la bande de fréquence ISM entre 2.4 et 2.48 GHz. Les systèmes d'exploitation supportés pour ces modules sont TinyOs, SOS, MantisOS et ManoRK.

La figure suivante montre un capteur MicaZ :

Figure I.7 : Capteur MicaZ.

❖ TelosB :

Ces capteurs ont un processeur « TI MS430 » d'une vitesse de 8MHz. Les modules sont dotés d'une RAM de 10Kbytes, d'un espace programme de 48Kbytes et d'une mémoire flash de 256Kbytes.

Ces modules fonctionnent avec un voltage entre 1.8 et 3.6V et a un débit de transmission de 250kb/s. Ils communiquent sur une bande de fréquence ISM qui va de 2.4 to 2.4835 GHz. Ils supportent plusieurs systèmes d'exploitation entre autres TinyOs et SOS.

La figure suivante montre un capteur TelosB :

Figure I.8 : Capteur TelosB.

I.2.8.b Digi International (MaxStream) :

Il existe plusieurs types des modules fabriqués par Digi, nous allons citer le module XBee Serie2 (que nous allons utiliser lors de la réalisation de notre travail).

- **XBee Serie2 :**

Ces modules ont un microcontrôleur fait par Digi qui fonctionne avec un protocole ZigBee. Ils se basent sur le standard IEEE 802.15.4.

Les principales caractéristiques de ces capteurs sont :

- Fréquence : ISM 2.4 GHz
- Portée : 10-100 m
- Débit : 250Kb/s
- Voltage : 3.3 V
- Communication : doté d'une communication série pour le coordinateur avec l'ordinateur et RF pour la communication avec les autres capteurs.
- Topologie de réseau : maillé, point à point, point à multipoint.

La figure suivante montre un module XBee Série 2 :

Figure I.9 : Module XBee Série2.

I.3 Conclusion :

Dans ce chapitre, nous avons vu les concepts généraux liés aux réseaux de capteurs sans fil. Parmi ces concepts, nous avons identifié les contraintes de conception de ces réseaux et leurs domaines d'application.

Nous avons aussi abordé les protocoles de communication dans ces réseaux, dont ZigBee qui fera le sujet de notre deuxième chapitre.

Chapitre II : La norme ZigBee

II.1 Introduction :

Comme nous l'avons vu dans le chapitre précédent, les réseaux de capteurs sans fils constituent un des domaines de recherche les plus actifs actuellement. Ils se révèlent utiles dans de nombreuses applications pour collecter et traiter des informations provenant des nœuds capteurs. Le travail de définition d'un protocole s'impose alors. Il faut choisir les fréquences de communication, les mécanismes réseaux, les couches applicatives, qui garantiront la fiabilité, la robustesse, et la longévité du système qui conviendra à cet environnement.

Issu de l'alliance entre l'organisme IEEE et un groupe d'industriels, le protocole ZigBee basé sur le standard 802.15.4, se présente comme l'un des meilleurs candidats en raison de ses caractéristiques techniques, comme la technologie sans-fil idéale pour les applications de contrôle et de surveillance en milieux industriels et résidentiels.

Dans ce chapitre nous allons présenter la norme ZigBee qui est basée sur la norme IEEE 802.15.4, en présentant un bref historique de sa création, son architecture et ses différentes topologies. Nous allons aussi présenter une interface matérielle très connue de la norme ZigBee, le module « XBee ».

II.2 Qu'est-ce que ZigBee :

ZigBee est un protocole de haut niveau permettant la communication radios, à consommation réduite, basée sur la norme IEEE 802.15.4 pour les réseaux à dimension personnelle (Wireless Personal Area Networks : WPANs).

Cette technologie a pour but de fournir les mêmes services que la technologie Bluetooth, tout en étant moins chère et plus simple, avec une plus longue portée et une moindre consommation d'énergie.

II.3 Historique :

1998 : Dès l'arrivée des technologies sans fil Wi-Fi et Bluetooth les premières ébauches de réseaux de type ZigBee firent leur apparition dans le cadre d'applications où les technologies précédentes n'étaient pas utilisables. La technologie Bluetooth a beaucoup inspiré le protocole ZigBee.

2003 : Le standard IEEE 802.15.4 est lancé. Et peu après cela, Philips abandonne la ZigBee Aliance,

2004 : Au mois d'**Octobre** Le nombre d'inscriptions à la ZigBee Aliance devient assez suffisant pour continuer les recherches. Le **14 décembre 2004** ils présentent les premières spécifications de ZigBee.

2005 : A la date du **13 juin,** la ZigBee Aliance publie les premières spécifications officielles de la version ZigBee 1.0 qui sont désormais disponibles **[6].**

II.4 La norme IEEE 802.15.4 :

II.4.1 Architecture :

Un PAN (Personnal Area Network) est composé d'un ensemble de nœuds ayant pour rôle de collecter et de transmettre l'information vers un organe central du réseau, qui sera en charge de traiter les informations ou de jouer le rôle de passerelle. Ce dernier a pour nom « puits » ou « coordinateur ».

La norme IEEE 802.15.4 décrit les couches basses d'un réseau PAN, elle propose une couche physique et une couche liaison de données adaptées aux applications à faible débit dont l'autonomie énergétique est une contrainte forte.

IEEE 802.15.4 définit deux types d'entités pouvant participer à un réseau: les **RFD** *(Reduced Function Devices)* et les **FFD** *(Full Function Devices)* :

- Les **RFD** sont des nœuds aux fonctionnalités réduites, très économes en énergie, et pouvant être connectés à des capteurs ou à des actionneurs, il est considéré comme un dispositif d'extrémité (end device).
- Les **FFD** sont des nœuds ayant la capacité de router ou de coordonner le réseau. Dans chaque réseau IEEE 802.15.4, l'un des **FFD** est le coordinateur du réseau. Le rôle du coordinateur est entre autres d'initier la formation de la topologie du réseau.

Pour communiquer sur un même réseau, un **FFD** (au moins) et des **RFD** doivent utiliser le même canal physique parmi ceux définis selon la bande de fréquence choisie. Le **FFD** peut dialoguer avec des **RFD** et des **FFD**, tandis que le **RFD** dialogue avec un **FFD** uniquement.

II.4.2 Topologie :

Le standard IEEE 802.15.4 supporte plusieurs types de topologies : La topologie en étoile, La topologie point à point (peer to peer) et la topologie en arbre (cluster-tree).

- **La topologie en étoile :** Cette topologie impose que les communications s'établissent directement et uniquement entre le coordinateur et les autres nœuds, c'est-à-dire que tous les nœuds sont à portée radio du coordinateur.
- **La topologie point à point** : Chaque nœud pourra communiquer avec n'importe quel autre nœud du réseau grâce à la collaboration des nœuds intermédiaires sollicités afin de relayer les paquets jusqu'à la destination, pour cela il faudra ajouter un protocole de routage.
- **La topologie cluster-tree :** Cette topologie n'est qu'un cas particulier de la topologie précédente. Dans cette dernière, le réseau est hiérarchisé : les nœuds sont rassemblés en cluster et dans chaque cluster il y'a un « cluster head ».

La figure suivante montre ces différentes topologies :

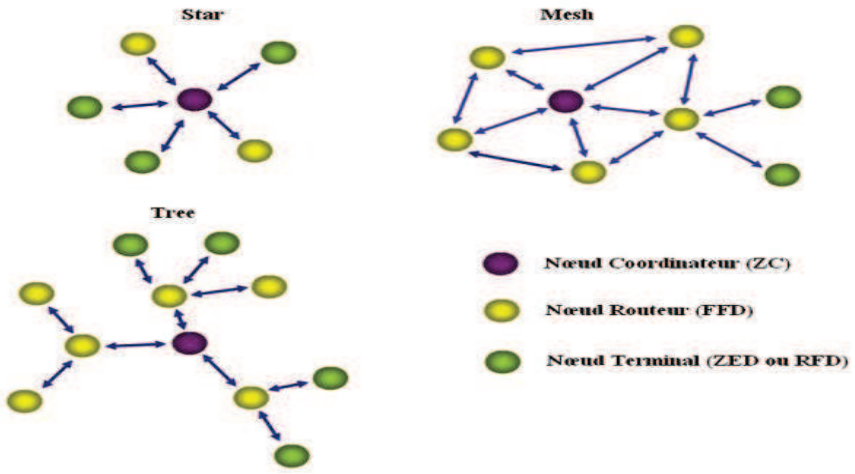

Figure II.1 : Différentes topologies du standard IEEE 802.15.4

II.5 Architecture protocolaire de la norme IEEE 802.15.4/ZigBee :

L'architecture ZigBee est composée de 4 couches sur les 7 du modèle OSI : la couche Physique (PHY), la couche Liaison (LNK), la couche Réseau (NWK) et la couche Application (APL). Entre ces couches se trouve les points d'accès aux services (SAP Service Access Point), ces SAP offrent les API (Application Programming Interface) pour permettre

aux couches de communiquer tout en isolant le travail interne à chacune des couches. ZigBee utilise deux types de SAP par couche : un pour les données, et un pour le management.

Les deux couches inférieures (PHY et LNK) sont définies par les spécifications de l'IEEE 802.15.4 et les couches supérieures sont définies par la norme ZigBee [7].

La figure suivante présente l'architecture protocolaire de ZigBee :

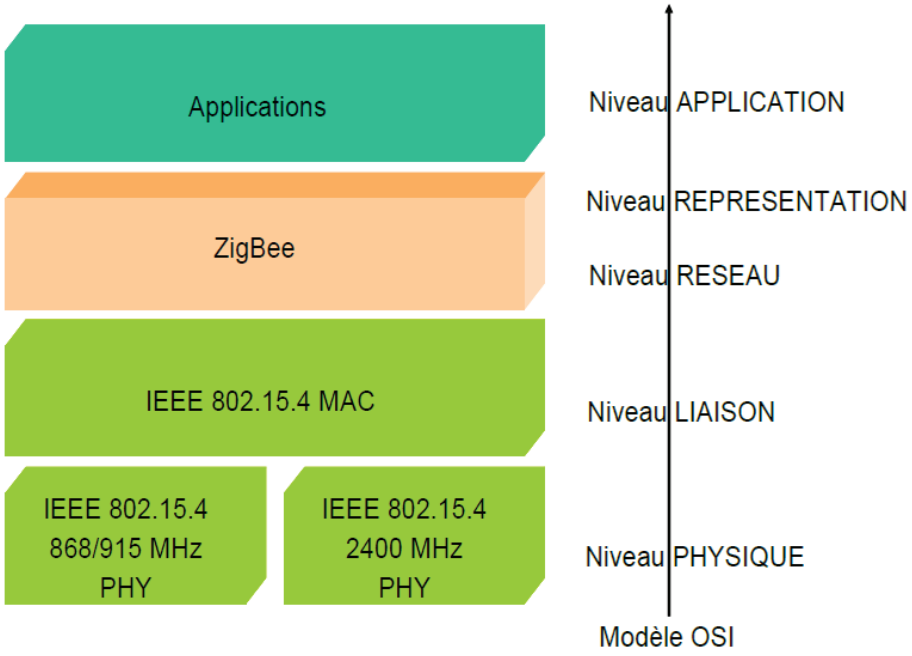

Figure II.2 : Pile Protocolaire ZigBee.

II.5.1 La couche physique (PHY) :

La norme IEEE 802.15.4 supporte 3 bandes ISM (Industrial Scientific Medical) de **868 MHz**(Usage :Europe, Débit : 20Kb/s, 1 canal), **915 MHz**(Usage :Amérique, Débit :40Kb/s, 10 Canaux) et **2.4 GHz**(Usage :Mondial, Débit :250Kb/s, 16 Canaux).

Cette couche (PHY) traduit simplement les trames en bits qu'elle peut transmettre et recevoir par transmission radio.

Le niveau physique gère les fonctionnalités suivantes :

- Activation/désactivation de l'interface radio.
- Détection de l'énergie dans le canal. Choix du canal radio (ED).
- Qualité de la liaison radio (LQI).
- Evaluation du canal pour la mise en œuvre du protocole d'accès CSMA.
- Emission/réception des paquets dans le canal radio.

II.5.2 La couche liaison de données (LNK) :

De façon très similaire au modèle définit par le standard IEEE 802. L'IEEE 802.15.4 a un niveau liaison de données qui comprend une sous-couche d'accès au medium (MAC) et une sous-couche de convergence (LLC) **[8]**.

❖ **Structure des trames :**

La norme IEEE 802.15.4 définit 4 types de trames :

- **Trame de données** (Data Frame) : transfert de données.
- **Trame d'acquittement** (Acknowledgement Frame) : notée ACK ce sont des trames de confirmation de données bien reçues.
- **Trame beacon** (Beacon Frame) : émise par le coordinateur de réseau en mode beacon.
- **Trame de commande MAC** (MAC Command Frame) : pour le contrôle des nœuds.

La figure suivante représente la structure des trames **[6]**:

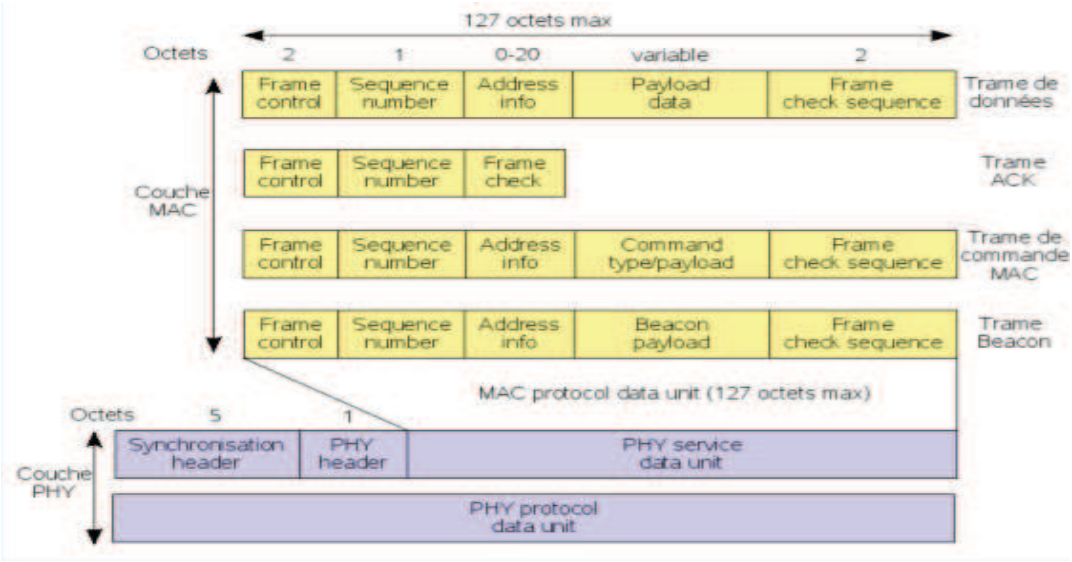

Figure II.3 : Structure des trames.

Note :

La structure de la trame MAC est flexible pour s'adapter aux différentes applications. Elle est définit comme suit :

MPDU (MAC Protocol Data Unit) = **MHR** (MAC HeadeR) + **MSDU** (MAC Service Data Unit) + **MFR** (MAC FooteR)

❖ **MHR (MAC HeadeR) :**

- Le champ Frame Control indique type de trame MAC et spécifie le format du champ adresse
- Le champ Sequence Number assure l'ordre à la réception et permet l'acquittement des trames MAC
- Le champ adresse est variable de 0 à 20 octets en fonction du type trame :

❖ **MSDU (MAC Service Data Unit) :**

- La trame de données permet une charge utile jusqu'à 104 octets.

❖ **MFR (MAC FooteR) :**

- Le FCS (Frame Check Sequence) assure que la trame est transmise sans erreur.
- CRC sur 16 bits

❖ La sous-couche MAC :

Cette sous-couche gère les accès aux medium, il existe ainsi deux modes d'accès au canal, un mode non coordonné (CSMA-CA) et un mode coordonné (beacon mode).

✓ **Le mode non coordonné (CSMA-CA) :**

On utilise une communication avec une politique d'accès au medium de type CSMA-CA. Dans ce mode, il n'y a pas d'émission de beacon donc pas de synchronisation entre les différents nœuds du réseau. Dans ce cas, le nœud voulant émettre écoute le réseau, s'il est encombré, la transmission est différée pour éviter les collisions. Dans le cas contraire, si le média est libre, la station peut émettre.

✓ **Le mode coordonné (mode beacon) :**

Ce mode n'est disponible que pour les topologies en étoile.

Le coordinateur du réseau envoie périodiquement des trames balise (beacon) sur lesquelles peuvent se synchroniser les nœuds et s'en servir comme relais. Ce mode de fonctionnement permet de meilleures performances énergétiques, car une fois l'information transmise au coordinateur, le nœud émetteur peut se mettre en veille. Puisque les données en attente sont stockées dans la mémoire du coordinateur, alors un nœud peut choisir de se réveiller et demander alors ces données en attente.

❖ La sous-couche LLC [9]:

La sous-couche LLC joue les rôles suivants :

1. La vérification de l'intégrité des données reçues avant la remise à la couche supérieure, par exemple en utilisant un Code de Redondance Cyclique (CRC) et un mécanisme d'acquittement.

2. Le contrôle de flux, afin d'éviter la saturation des tampons de réception et la perte éventuelle de données ou les débordements de mémoire.

3. La convergence d'adressage, c'est-à-dire la correspondance qu'il faut effectuer entre les adresses de niveau 3 et les adresses de niveau 2.

II.5.3 La couche réseau (NWK) :

La couche réseau est responsable de la topologie (construction et maintenance), de la gestion de l'adressage, du routage et de la sécurité. La couche réseau propose aussi des services aux couches supérieures (applications).

❖ **Adressage :**

Pour être identifié au sein d'un réseau, chaque nœud ZigBee possède une adresse MAC unique composée de 64 bits. Un coordinateur du réseau ZigBee est capable d'attribuer des adresses raccourcies locales limitées à 16 bits à chaque nœud rejoignant le réseau.

❖ **Routage :**

Du fait de la mémoire très limitée des nœuds, le routage ZigBee suit le principe de base suivant :

– si la table de routage contient une entrée qui correspond au routage demandé, il faut router le paquet selon cette entrée ;

– si elle ne contient aucune entrée et si la mémoire libre le permet, il faut lancer le processus de découverte de route ;

– sinon, il faut router le paquet selon le routage hiérarchique en arbre (Tree Routing).

Algorithmes de routage :

1. **Algorithme de routage à la demande :** L'algorithme de routage à la demande proposé par la ZigBee Alliance est proche d'AODV, et il convient aux topologies peer to peer. Le principe est simple: un nœud voulant envoyer un paquet devra d'abord vérifier sa table de routage. Si

l'adresse y figure, le nœud envoie directement le paquet. Dans le cas contraire où l'entrée n'existe pas, le nœud devra alors faire une demande de route (Route Request) vers ses voisins. Ces derniers relayeront cette requête dans tout le réseau jusqu'à la destination qui répondra à son tour par un paquet Route Reply en unicast vers l'initiateur de la requête de route.

2. **Algorithme de routage hiérarchique (Tree clustering) :** Le cas particulier de la topologie en arbre permet de simplifier grandement le routage puisque, dans le cadre de cette topologie hiérarchique, l'adressage est fonction de la topologie. Il suffit donc d'observer l'adresse du nœud de destination pour déterminer à quel cluster head envoyer le paquet.

II.5.4 La couche application (APL) :

La couche applicative est la couche de niveau le plus haut. C'est elle qui 7. La couche APL (Application Support Layer) est associée à plusieurs entités protocolaires :

- **le module SSP (Security Service Provider)** qui peut gérer les fonctions de sécurité (authentification, cryptage) ;
- **le module APS (Application Support Sub-Layer)** qui est un soutien aux applications pour la mise en liaison des dispositifs et les services de messagerie ;
- **le module ZDO (ZigBee Device Object)** qui permet la découverte des dispositifs et des services, et de définir le rôle d'un dispositif dans le réseau.

II.6 Les réseaux ZigBee :

II.6.1 Les éléments du réseau ZigBee :

Les réseaux ZigBee sont appelés réseaux à dimension personnelle (PAN). Chaque réseau contient un identificateur de 16 bits appelés PAN ID.

ZigBee définit trois différents types d'entités de réseau : **le coordinateur**, **le routeur** et **l'équipement d'extrémité (end device) [10]**.

- **Le coordinateur :** Il est responsable de la sélection du canal de communication et du PAN ID. Le coordinateur initie la formation du réseau, et une fois établit, il peut permettre à des routeurs ou à des end devices de rejoindre le réseau. Le coordinateur peut aussi transmettre et recevoir des

données RF, et acheminer ces données à travers le réseau. Les coordinateurs doivent être alimentés continuellement et ne peuvent donc pas passer en mode veille.

- **Le routeur :** le routeur doit joindre le PAN ZigBee avant qu'il puisse fonctionner. Il peut permettre à d'autres routeurs ou à des end devices de rejoindre le réseau. Le routeur peut aussi transmettre et recevoir des données en RF et router les paquets de données à travers le PAN. Puisque le routeur participe au routage de données et peut permettre à des équipements de joindre le réseau, il ne peut pas passer en mode veille et doit être alimenté continuellement.
- **End Device :** Cet équipement doit joindre le réseau de la même façon que le routeur. Cependant, le end device ne peut pas autoriser d'autres équipements à rejoindre le PAN, et ne participe pas au routage de données à travers le réseau. L'équipement d'extrémité peut transmettre et recevoir des données en RF. Cet équipement peut passer en mode veille. Puisque ce dispositif peut dormir, le routeur ou le coordinateur qui a permis à l'équipement d'extrémité de joindre le réseau doit collecter tous les paquets de données destinés à ce dernier, et les enregistrer jusqu'à ce que l'équipement en question se réveille et puisse les recevoir. Le routeur ou le coordinateur qui a permis à l'équipement d'extrémité de joindre le réseau est connu comme étant son parent, et donc le dispositif est son enfant.

II.6.2 Formation d'un réseau PAN ZigBee :

Un PAN ZigBee consiste en un coordinateur et un ou plusieurs routeur(s) et/ou end device. Un PAN ZigBee est créé lorsque le coordinateur sélectionne un canal de communication et un identificateur de réseau (PAN ID). Après que le coordinateur ait mis en marche le PAN, ce dernier peut permettre aux routeurs et aux end devices de rejoindre le réseau, ils héritent donc du même PAN ID que le coordinateur.

Quand un routeur ou un end device rejoint un PAN il reçoit une adresse réseau de 16 bits, et peut ainsi transmettre ou recevoir des données dans le réseau.

La formation du réseau ZigBee est résumée dans les trois étapes suivantes :

❖ **Initiation d'un réseau :**

Puisque le coordinateur est responsable d'initier le réseau, alors tous les réseaux ZigBee doivent avoir un coordinateur présent au préalable. Pour initier le PAN, le coordinateur fait une série de scan pour découvrir le niveau d'activité des RF dans les différents canaux de communication (scan d'énergie) et ainsi découvrir les réseaux voisins.

Quand le coordinateur opère pour la première fois, il lance un « scan énergie » dans plusieurs canaux pour détecter leurs fréquences. Les canaux ayant un niveau d'énergie excessivement élevé, sont éliminés de la liste des canaux potentiellement utilisables.

Lorsque le scan d'énergie prend fin, le coordinateur balaye les canaux restants à la recherche d'autres PANs existants. Pour cela, le coordinateur envoie un broadcast ''beacon request''. Les coordinateurs et les routeurs voisins répondent en envoyant une trame ''beacon''. La trame ''beacon'' contient des informations à propos du PAN comme le PAN ID. Ce scan est appelé « beacon scan ».

Après que le coordinateur ait fini le scan du PAN et le scan d'énergie, il balaye les trames beacons reçus en vue de choisir un canal et un PAN ID non utilisé. Après cela, le coordinateur peut autoriser des routeurs ou des « end devices » à rejoindre le réseau.

❖ **Rejoindre un réseau :**

Une fois le choix fait entre les différents réseaux disponibles, pour rejoindre le réseau, un nœud sélectionne un père parmi les nœuds dont il a connaissance, et demande à la couche MAC d'initier une procédure d'association. Il obtient ainsi un identifiant auprès de son père (adresse de 16 bits) qui sera utilisé pour toutes les communications dans le réseau.

Quand un routeur veut rejoindre un réseau, il doit chercher et joindre un PANID. Pour ce faire, il envoi des « beacon request » dans de multiples canaux pour localiser les réseaux voisins.

Les routeurs et les coordinateurs à proximité répondent en transmettant des « beacon » pour indiquer leur canal et PANID

Le routeur scanne ces canaux pour déterminer à quel dispositif il pourrait s'associer. Si un PAN valide est trouvé parmi un des « beacon » reçu,

le routeur envoi un « join request » au dispositif qui a envoyé le « beacon ». Si le dispositif lui permet l'association, le routeur reçoit un « join confirmation ».

Le routeur peut ainsi router les données et permettre à d'autres dispositifs de rejoindre le réseau.

Les « end devices » suivent le même processus que les routeurs pour rejoindre un réseau. En revanche, les « end Devices » ne peuvent pas router des données, et ne peuvent communiquer qu'avec leur parent.

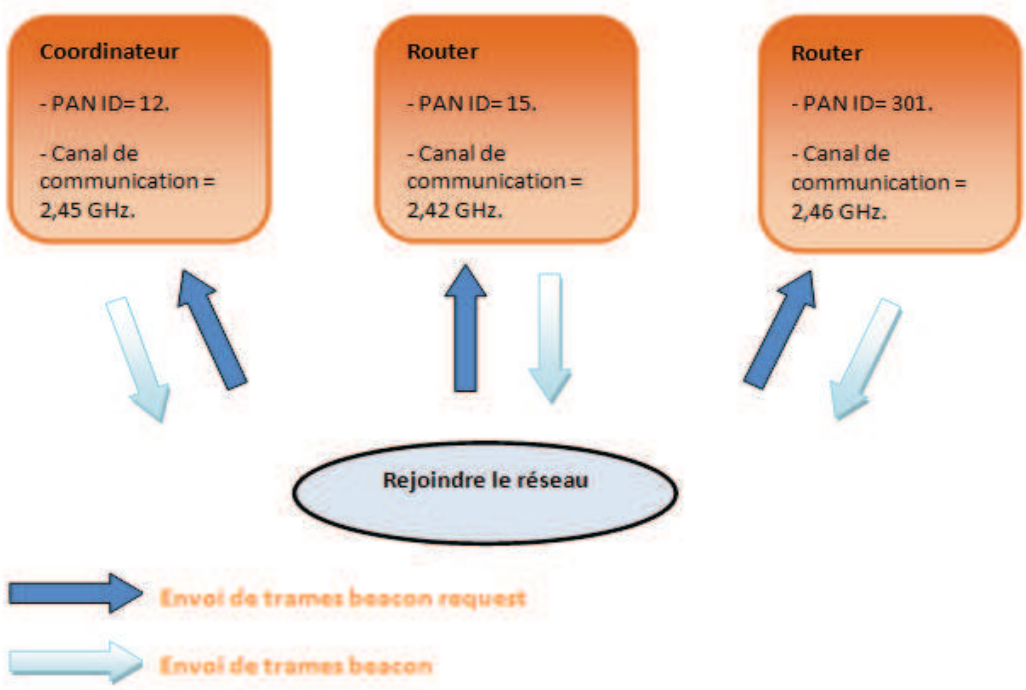

Figure II.4 : Rejoindre un réseau ZigBee.

II.6.3 La communication dans les réseaux ZigBee :

ZigBee supporte deux types d'adressage : l'adressage des équipements (device addressing) et l'adressage de couche application (application layer addressing). L'adressage des équipements spécifie l'adresse de destination des paquets, et l'adressage de couche application indique une application particulière de destination.

❖ **Adressage des équipements ZigBee :** la norme IEEE 802.15.4 sur laquelle s'appuie le protocole ZigBee spécifie deux types d'adressage :

- **Adressage réseau de 16 bits :** une adresse de 16 bits est assignée à un nœud lorsqu'il rejoint le réseau. Cette adresse est unique mais elle n'est pas statique, c'est-à-dire que cette adresse peut changer. Ce changement peut être

dû à deux raisons : la première est que si un end device n'arrive pas à communiquer avec son parent il sera forcé de quitter le réseau pour rejoindre un nouveau parent. La seconde est que si un équipement change de type, il est obligé de quitter le réseau et de le rejoindre par la suite en ayant un nouveau type.

- **Adressage réseau de 64 bits :** Chaque nœud contient une adresse unique de 64 bits qui l'identifie de manière permanente.

❖ **Adressage de la couche application :** les couches d'application ZigBee définissent deux entités : les endpoints et les identificateurs de cluster (Cluster ID) qui sont utilisés pour adresser des services ou des applications spécifiques dans un nœud.

Un endpoint est une tâche ou application distincte qui fonctionne sur un nœud ZigBee de façon similaire au port TCP. Chaque équipement ZigBee peut avoir un ou plusieurs endpoint(s). Un cluster ID définit une fonction ou une action spécifique sur un nœud.

❖ **Transmission de données :**

Tous les paquets de données sont adressés en utilisant les deux types d'adressage précédents **device addressing** et **application layer addressing**. La donnée peut être envoyée en utilisant une transmission broadcast, multicast ou unicast.

- **Transmission broadcast :**

Les transmissions broadcast au sein du protocole ZigBee sont prévues pour être propagées dans l'intégralité du réseau afin que tous les nœuds reçoivent cette transmission. Pour cela, tous les équipements qui reçoivent les broadcasts, retransmettront chaque paquet reçu trois fois.

- **Transmission multicast :**

Les transmissions multicast opèrent de la même manière que les broadcasts, mais les paquets de données ne sont destinés qu'à un groupe de nœuds non pas à tous les nœuds du réseau.

- **Transmission unicast :**

Les transmissions ZigBee unicast utilisent l'adresse de 16 bits pour atteindre l'équipement de destination. Toutefois, seule l'adresse de 64 bits du dispositif est permanente. Pour cette raison, les équipements ZigBee peuvent utiliser l'adresse de découverte de réseau pour identifier l'adresse de l'équipement (16 bits) qui correspond

à une adresse connue de 64 bits. Après que l'adresse de 16 bits soit connue, la route vers l'équipement de destination doit être établie.

Pour la mise en place des réseaux ZigBee vus précédemment, on utilise des composants XBee qu'on étudiera dans section II.7 suivante.

II.7 Interface ZigBee (Le module XBee) :

II.7.1 Le module XBee :

Les modules XBee sont des tranceivers radio au standard ZigBee/IEEE 802.15.4 qui sont spécialement conçus pour la réalisation de système de communication au sein d'un réseau de capteurs sans fil.

Les modules opèrent sur la bande ISM de 2.4GHz. Ces modules peuvent être utilisés avec plusieurs modes de fonctionnement qui sont accessibles via commandes AT. Ces commandes sont listées dans l'annexe II.

II.7.2 Caractéristiques des modules XBee :

- **Communication RF :**
  - Les modules XBee présentent une puissance de sortie RF de 10mW.
  - La portée est de 30 mètres en intérieur et de 100 mètres en extérieur.
  - Le débit est de 250Kbps.
- **Gestion réseaux :**
  - Il est possible de choisir son canal ainsi que l'adresse ZigBee du module en adressage court (sur 16 bits).
  - Les topologies supportées sont celles offertes par la norme ZigBee.
- **Consommation :**
  - Le module est alimenté sous une tension comprise entre 2.8v et 3.4v, la consommation est de 45mA en émission et de 50mA en réception.
- **Brochage :**

Le tableau suivant décrit les fonctions des 20 broches présentes sur le module :

Broche	Nom	Direction	Description
1	VCC	-	Alimentation
2	DOUT	Out	Sortie UART
3	DIN/CONFIG	In	Entrée UART
4	DIO8	InOut	Entrée/Sortie digitale 8
5	RESET	-	Reset
6	PWM0/RSSI/DIO10	Out	Sortie PWM0/Indication puissance Rx/ E/S digitale 10
7	PWM1	Out	Sortie PWM1
8	Réservé	-	-
9	DTR*/SLEEP_RQ/DI8	In	Contrôle Sleep/Entrée digitale 8
10	GND	-	Ground
11	AD4/DIO4	InOut	Entrée analogique 4 ou E/S digitale 4
12	CTS/DIO7	Inout	Clear To Send/ E/S digitale 7
13	ON/SLEEP	Out	Indicateur état
14	VREF	-	Tension de référence pour conversion
15	Associate/AD5/DIO5	Inout	Indication association/Entrée analogique 5 ou E/S digitale 5
16	RTS/AD6/DIO6	Inout	Reday To Send/Entrée analogique 6 ou E/S digitale 6
17	AD3/DIO3	Inout	Entrée analogique 3 ou E/S digitale 3
18	AD2/DIO2	Inout	Entrée analogique 2 ou E/S digitale 2
19	AD1/DIO1	Inout	Entrée analogique 1 ou E/S digitale 1
20	AD0/DIO0	Inout	Entrée analogique 0 ou E/S digitale 0

Figure II.5 : Brochage du module XBee

La figure suivante montre une vue sur les différentes fonctions des broches du module XBee Série 2 :

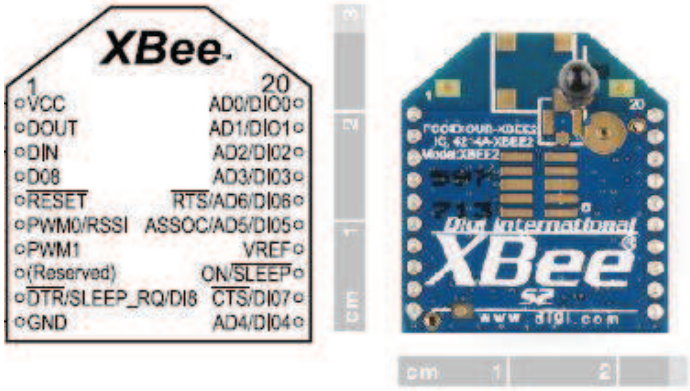

Figure II.6 : Vue externe du module XBee Série 2.

II.7.3 Communication avec le module XBee :

Le module peut être connecté à n'importe quel système embarqué (microcontrôleur, …etc) possédant une interface série.

La communication entre les modules XBee nécessite au minimum de câbler les broches d'alimentation (VCC et GND) et les signaux DIN et DOUT pour respectivement les données entrantes et sortantes via le module.

En Effet, le processeur envoie ses données via le port Din (pin3) en mode série asynchrone. Chaque paquet est constitué d'un bit start de niveau bas (0 logique), suivi de 8 bits de données (le LSB en premier), et enfin un bit stop de niveau haut (1 logique).

La figure suivante présente une vue interne du module XBee :

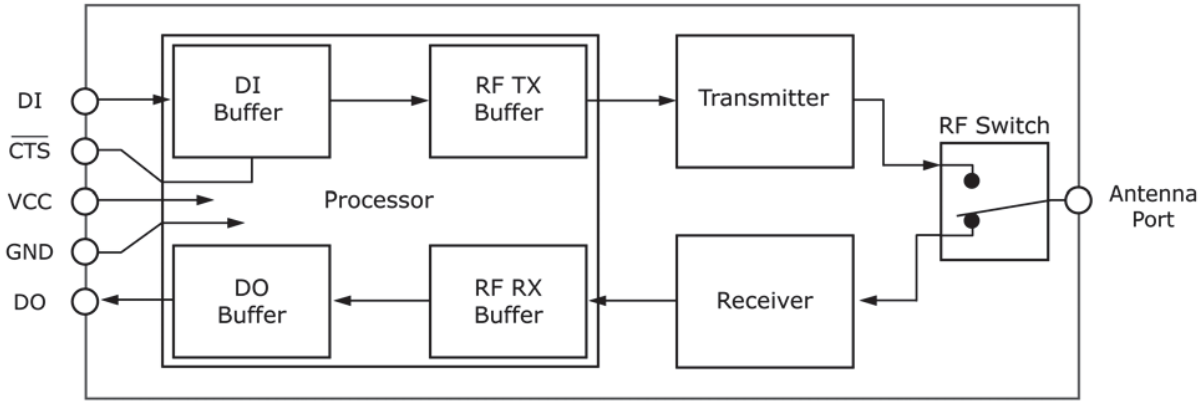

Figure II.7 : Vue interne du module XBee.

Etant connecté à un circuit présentant une liaison série asynchrone, le module utilise le « buffer DI » pour stocker les données transmises par ce circuit via le port DI. Ce flux de

données est contrôlé par le signal $\overline{CTS}$ de la manière suivante : lorsque le buffer DI ne dispose que d'un espace libre de 138 bits, $\overline{CTS}$ est mis à 1 afin de signaler au circuit d'arrêter l'envoi de données. Il est remis à 0 que lorsque le buffer DI dispose de 276 bits d'espace mémoire libre.

Par ailleurs, le buffer DO est utilisé pour stocker les données envoyées par un autre module XBee par exemple. Lorsque le buffer DO atteint sa capacité maximale, toute donnée envoyée par voie RF est perdue. Ce flux est également contrôlé par le signal $\overline{RTS}$: lorsqu'il est au niveau haut, les données restent stockées dans le buffer DO et elles ne sont transmises via le port DO que lorsqu'il est niveau bas.

II.7.4 Formation des réseaux XBee [10]:

- **Créer un réseau XBee :**

Pour former le réseau XBee, le coordinateur doit sélectionner un canal de communication inutilisé et un PAN ID, comme indiquer dans la section II.5.2. Pour cela, le coordinateur effectue le balayage d'énergie « scan énergie » sur tous les canaux spécifiés par le paramètre SC (Scan Channel). Le temps du scan de chaque canal est déterminé par le paramètre SD (Scan Duration).

Une fois que le scan d'énergie ait été accompli, le coordinateur procède au « beacon scan » en envoyant la trame « beacon request » sur chacun des canaux spécifiés par le paramètre SC, et il attend les réponses des voisins (coordinateurs et routeurs) en écoutant les trames « beacons ». L'information provenue du scan énergie et du scan beacon est utilisée pour sélectionner un canal inutilisé et un PAN ID. Si le paramètre ID (PAN ID) est à la valeur 0*FFFF, le coordinateur sélectionnera un PAN ID aléatoire. Autrement, le coordinateur opérera dans le PAN ID spécifié par le paramètre ID.

Après la sélection du canal de communication et du PAN ID, le coordinateur peut permettre à des nœuds de se joindre à lui pendant une durée qui est basée sur le paramètre NJ (Node Join).

- **Rejoindre un réseau XBee :**

Avant qu'un équipement puisse participer à la formation du réseau XBee, il doit localiser un coordinateur ou un routeur qui lui permettra de rejoindre le réseau. Pour cela il

suit la même procédure que celle suivit par le coordinateur pour créer le réseau (procédure basée sur les paramètres SC, SD, NJ et ID).

Si le paramètre ID de ces équipements est à 0xFFFF, l'équipement va rejoindre un autre équipement dans n'importe quel PAN ID.

Après que les équipements aient rejoint le réseau, les routeurs pourront participer au routage de données dans le réseau et pourront aussi permettre à d'autres équipements de rejoindre le réseau à leur tour. En revanche, les end devices n'ont pas la capacité d'autoriser d'autres dispositifs à rejoindre le réseau.

II.7.5 Les protocoles d'interface série des modules XBee :

Les modules XBee supportent deux interfaces série (Transparent et API).

❖ **Mode transparent:**

Dans le mode transparent le module agit comme une ligne série. Toutes les données reçues sur Din sont transmises via RF, et toutes les données reçues par RF sont transmises via Dout.

❖ **Mode API :**

Dans le mode API, toutes les données entrantes (sur la broche Din) sont mises sous forme de trames qui définissent des opérations ou des événements sur le module.

- Les trames reçues par Din (pin 3) comprennent :
  - La trame de transmission de données RF
  - La trame de commande (équivalent à la commande AT)
- Les trames transmises par Dout (pin 2) comprennent :
  - La trame de données RF reçue
  - Réponse à la commande
  - Les notifications d'évènements

Le mode API requière que la communication avec le module soit faite à travers une interface structurée (la donnée est transmise dans une trame dans un ordre bien définit).

❖ **Spécification des trames API :**

Il existe deux types de modes API, et tous deux sont activables grâce à la commande AP.

AP=1 : Mode API.

AP=2 : Mode API avec caractères ignorés.

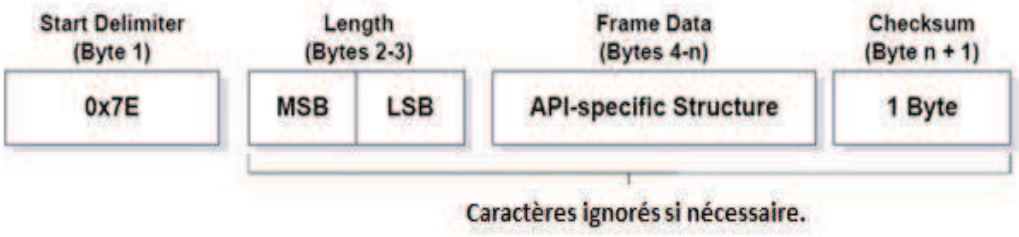

Figure II.8 : Structure d'une trame API.

- **Start Delimiter :** Ce champ est toujours égal à 0x7E.
- **Length :** Le champ de longueur a une valeur de deux octets qui spécifie le nombre d'octets qui seront contenus dans le champ de données de trame. Il ne comprend pas le champ de somme de contrôle(Checksum).
- **Frame Data :** Le contenu de ce champ est composé de l'identifiant de l'API et l'identifiant des données spécifiques de l'API. En fonction de l'identificateur de l'API (également appelé API de type de trame), le contenu des données spécifiques change.
- **Checksum :** Pour tester l'intégrité des données, le checksum (somme de contrôle) est calculé. Pour calculer le checksum, il faut additionner tous les bytes(sans les champs délimiteur de début et longueur) et ne garder que les 8 bits de poids faible, et les soustraire à 0xFF.
- **Caractères ignorés :** (Si AP=2), lors de la transmission ou la réception de données, certaines valeurs de données doivent être ignorées afin qu'elle n'interfère pas avec le séquençage de trame de données. Pour ignorer des caractères, il faut insérer 0x7D suivit des 8 bits à ignorés mit en XOR avec 0x20.

II.7.6 Les modes de fonctionnement :

Le module XBee peut être dans l'un des modes suivants :

❖ Mode scrutation :

Quand le module n'envoie pas et ne reçoit pas de données, il se met en mode scrutation. Dans le mode attente, le module attend des données valides à transmettre ou à recevoir.

Le module passe aux autres modes dans les cas suivants :

- Mode transmission : Lorsque des données se trouvant dans le ***« Serial Receive Buffer »*** sont prête à être envoyée.
- Mode réception : Lorsque des données sont reçues par l'antenne.
- Mode veille : sur les « End Device » Uniquement.
- Mode commande : Lorsqu'une trame de mode commande est envoyée.

❖ Mode transmission :

Le diagramme de la figure suivante montre les états du module au fil de la transmission de données :

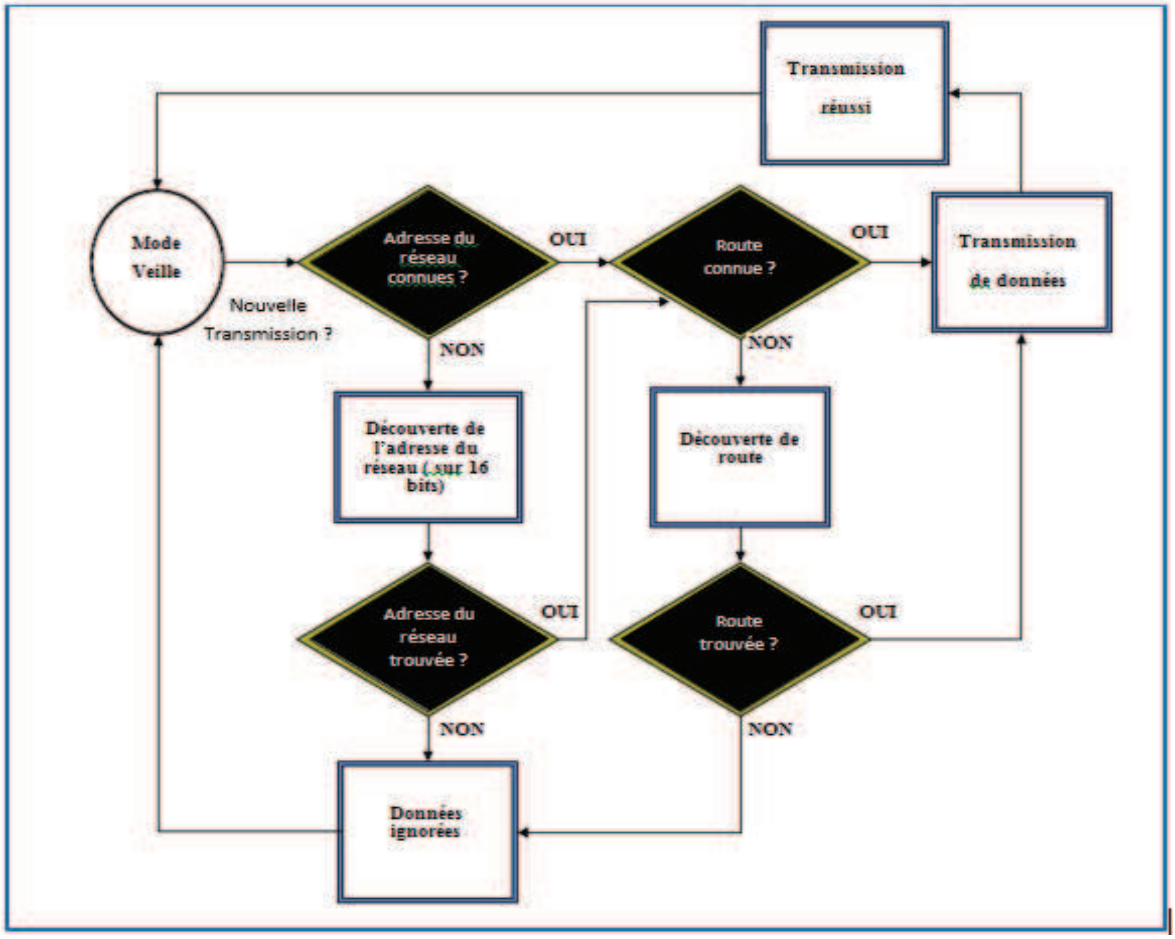

Figure II.9 : transmission de données**.**

❖ **Mode réception :**

Lorsqu'un paquet est correctement reçu et que son adresse correspond au paramètre MY (l'adresse source de 16 bits) du module, les données sont mises dans le ***« Serial Transmit Buffer »***.

❖ **Mode commande :**

Le module se met dans ce mode lorsqu'on a besoin de lire ou de modifier les paramètres du module. Dans ce mode toutes données reçues sont interprétées comme étant des commandes.

❖ **Mode veille :**

Le mode veille n'est supporté que dans les « End Device », car les routeurs et les coordinateurs participent au routage de données et qu'ils sont supposés être alimentés par secteur.

II.8 Quelques applications ZigBee :

II.8.1 Système de surveillance d'un patient :

Ce travail présenté dans [11] est illustré par la **figure II.10** et présente un réseau de capteurs sans fil (WSN) pour surveiller les conditions physiologiques d'un patient en continu en utilisant la technologie ZigBee. Les conditions physiologiques du patient sont surveillées par des capteurs et la sortie de ces capteurs est transmise via ZigBee à un moniteur à distance sans fil. Ce moniteur est constitué d'un module ZigBee et d'un ordinateur personnel (PC). Bien que Bluetooth soit meilleur que ZigBee pour le débit transmission, ce dernier convient mieux grâce à sa faible consommation d'énergie.

La première fonction du système est que les capteurs sans fil sont utilisés pour mesurer la fréquence cardiaque, la température et une éventuelle chute du patient. La seconde procédure du système est de mesurer le niveau de solution saline dans la bouteille.

Les résultats sont envoyés par l'intermédiaire du module ZigBee à un ordinateur hôte qui stocke les données. Les valeurs peuvent ensuite être affichées sur une interface graphique. Quand la personne est médicalement en détresse, une alarme peut être générée. Le système permet d'avoir un rapport continu sur l'état actuel de l'individu.

❖ **Description fonctionnelle du système :**

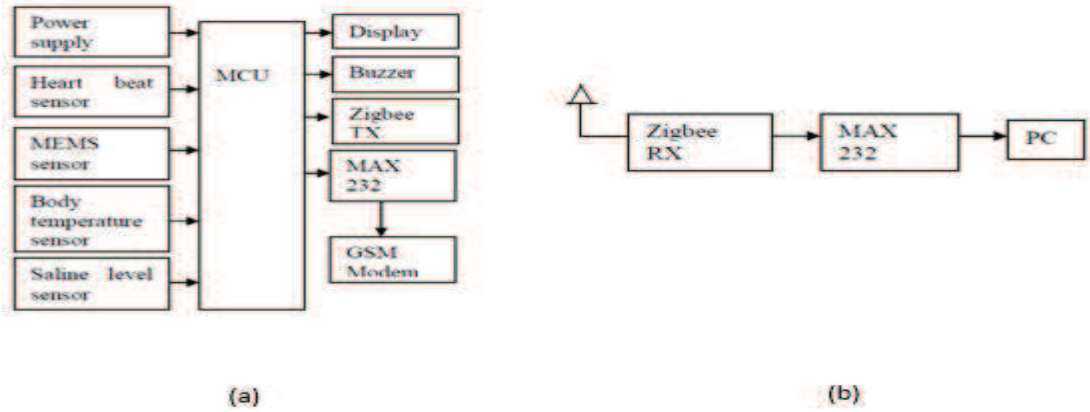

Figure II.10 : Schéma fonctionnel du système (a) Coté patient, (b) Coté moniteur.

Le système proposé se compose de quatre capteurs : un capteur de température, capteur de fréquence cardiaque, un capteur MEMS (pour détecter la chute) et le capteur de niveau de solution saline.

II.8.2 Système d'éclairage de rue sans fil intelligent :

Ce travail présenté dans [12] est décrit par la **figure II.11** et propose un éclairage de rue innovant sans fil avec un système de gestion et un rendement optimisé. L'éclairage est réalisé en fonction de la luminosité ambiante et de la présence des passants. La communication utilise des dispositifs sans fil basés sur la technologie ZigBee. Il utilise de nombreux capteurs de pour garantir le fonctionnement optimal du système; les informations sont transmises à une borne de commande utilisée pour vérifier l'état des lampes et à prendre des mesures appropriées en cas d'échec. Le système permet une gestion efficace d'énergie.

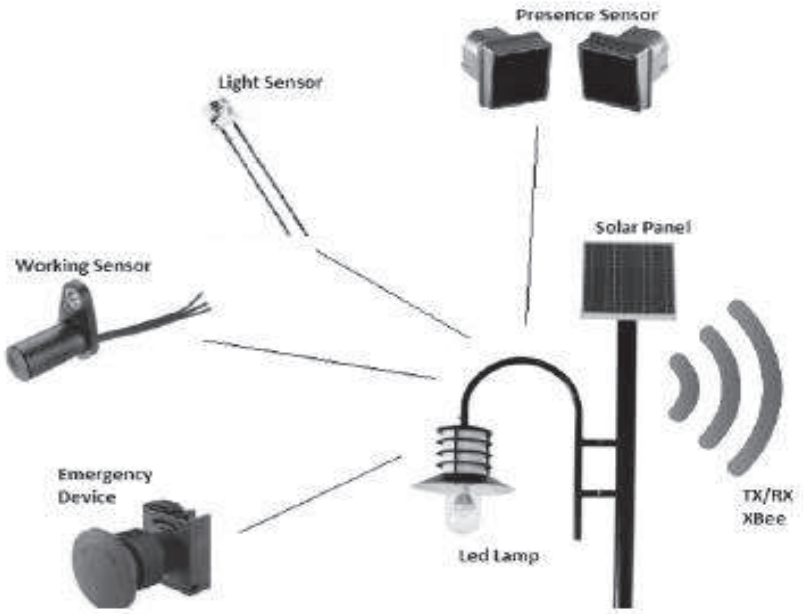

Figure II.11 : Schéma du lampadaire avec les capteurs.

II.9 Conclusion :

Dans ce chapitre, nous avons présenté le protocole ZigBee qui s'appuie sur la norme IEEE 802.15.4. Nous avons ainsi abordé les principaux points de cette norme, notamment son architecture, ses topologies et les éléments qui forment les réseaux ZigBee.

Par la suite, nous avons expliqué la manière dont les composants ZigBee communiquent en présentant l'implémentation de son protocole à travers son interface XBee.

Au final, nous avons cité quelques exemples d'application utilisant la technologie ZigBee.

Dans le chapitre qui suit, nous allons nous concentrer sur la description de notre approche pour la mise en place d'un réseau de capteur en utilisant le module XBee.

Chapitre III : Conception d'un réseau de capteur sans fil sous la technologie ZigBee.

III.1 Introduction :

Dans les chapitres précédents, nous avons présenté les réseaux de capteurs sans fil ainsi que le protocole ZigBee. Nous avons vu que ces réseaux ont pour finalité d'être mis en place dans un lieu donné permettant d'automatiser des mesures environnementales et/ou de contrôler l'environnement dans diverses applications. Dans le cadre de notre projet, nous avons déployé un réseau de capteurs sans fils en utilisant les modules XBee. Ce travail constitue une architecture de base qui pourra servir dans différents domaines, notamment dans la domotique, les domaines industriel et médical.

Dans ce chapitre, nous aborderons l'approche que nous avons adoptée pour l'analyse et la conception de notre projet. En premier lieu, nous présenterons l'architecture sur laquelle il se base. Ensuite, nous présenterons notre démarche de conception matérielle et logicielle. Pour finir, nous donnerons quelques exemples de domaines dans lesquels notre travail pourrait être exploité.

III.2 Analyse et conception :

Afin de mener à bien notre travail, nous suivrons une méthode d'analyse et de conception qui aura pour rôle la formalisation des étapes préliminaires du développement de notre système.

La phase d'analyse permet de lister les objectifs et de ressortir les besoins matériels et logiciels. La phase de conception quant à elle permet de décrire de manière plus ou moins précise le fonctionnement futur du réseau afin de faciliter sa réalisation.

III.3 Analyse:

III.3.1 Objectifs :

Le protocole ZigBee se distingue des autres technologies sans-fil par sa simplicité d'implémentation ce qui fait de lui, grâce à ses performances, le standard de communication idéal pour les réseaux de capteurs sans fil. Ces derniers offrent trois fonctionnalités principales ; la capture, la communication et l'action.

- **La capture** consiste à prélever des grandeurs numériques ou analogiques d'un champ de captage. Celle-ci est effectuée par des nœuds capteurs.

- **La communication** consiste quant à elle à transmettre les données captées à travers le réseau. Elle est réalisée par des nœuds routeurs.
- **L'action** est effectuée par le nœud actionneur qui exécute une tâche particulière en réponse à un événement ou une commande donnés.

Notre objectif est donc de créer un réseau qui exploite les fonctionnalités sus-citées en utilisant des modules XBee fonctionnant sous la norme ZigBee. Pour cela, nous allons suivre les différentes étapes illustrées dans **figure III.1**.

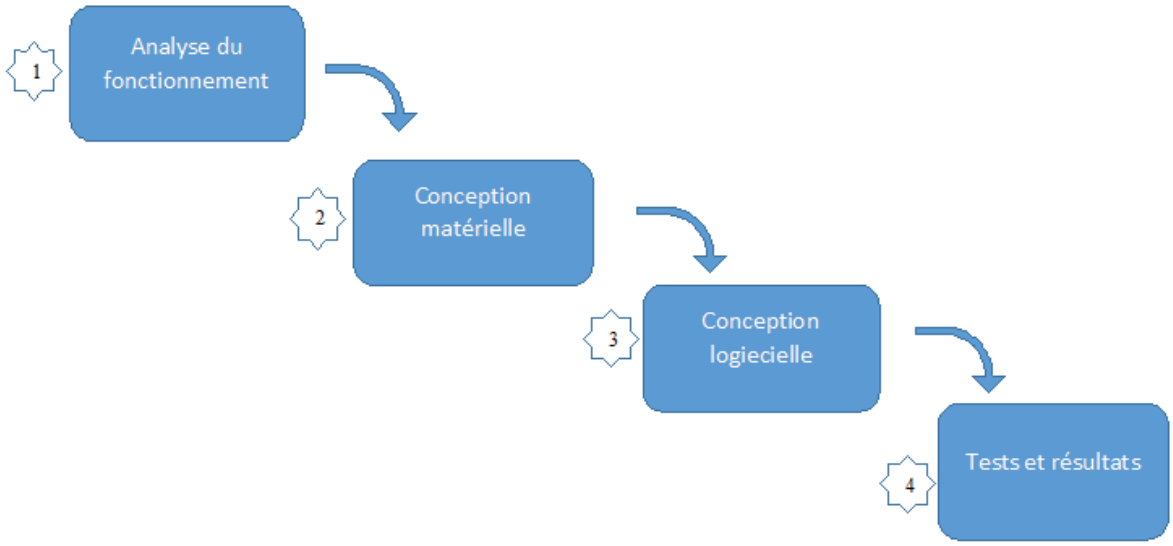

Figure III.1 : Etapes de réalisation.

Au cours de ce chapitre nous allons développer les trois premiers points, à savoir l'analyse du fonctionnement, la conception matérielle et la conception logicielle. Le dernier point fera le sujet de notre prochain chapitre.

III.3.2 Analyse de l'architecture:

La figure suivante donne un aperçu de notre réseau :

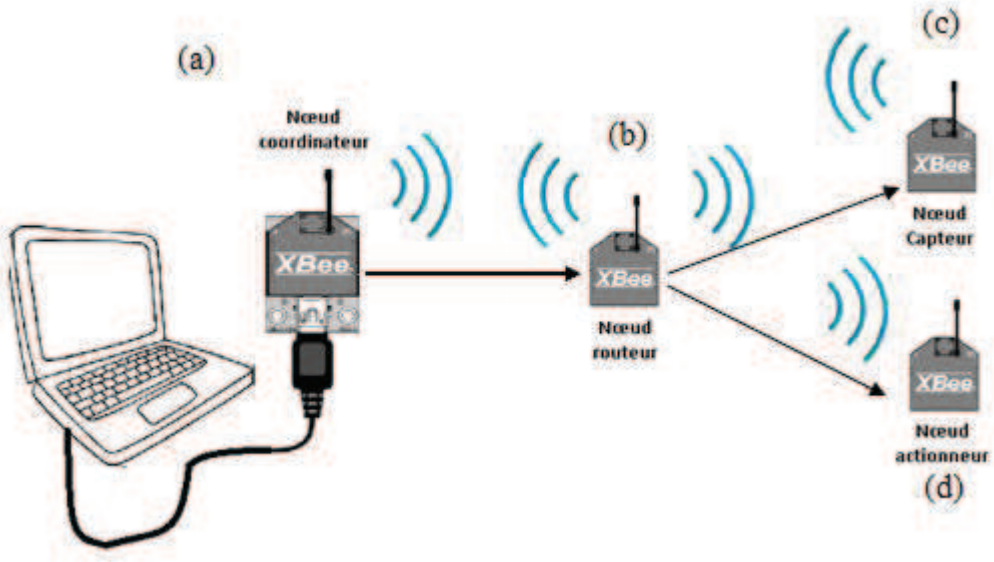

Figure III.2 : Architecture du réseau.

Comme on peut le voir dans la figure III.2, notre réseau est constitué de quatre éléments qui forment une topologie en arbre.

(a) : Module XBee configuré en mode coordinateur + Station d'accueil + PC.
(b) : Module XBee configuré en mode routeur.
(c) : Module XBee configuré en mode end device + capteurs.
(d) : Module XBee configuré en mode end device + actionneurs.

Le nœud coordinateur **(a)** est directement relié au PC afin d'initier le réseau. Le nœud routeur **(b)** s'associe au coordinateur qui devient de fait son nœud père. Les nœuds capteur **(c)** et actionneur **(d)** prennent quant à eux le routeur comme père. Cette architecture fait que toute donnée ou commande circulant dans le réseau doit impérativement transiter par le routeur.

III.3.3 Fonctionnement du réseau :

Lors de l'exécution de notre application de gestion du RCSF, une connexion série est établie avec le coordinateur lequel choisit un canal de communication et un PAN ID.

Les autres nœuds (routeur, capteur et actionneur) rejoignent ensuite le réseau avec une adresse de 16 bits assignée par le coordinateur et partagent le même PAN ID que ce dernier. Une fois que tous les nœuds ont été rajoutés au même réseau, ils auront la possibilité de

communiquer entre eux. L'interaction entre les différents composants de notre réseau est décrite dans la **figure III.3**.

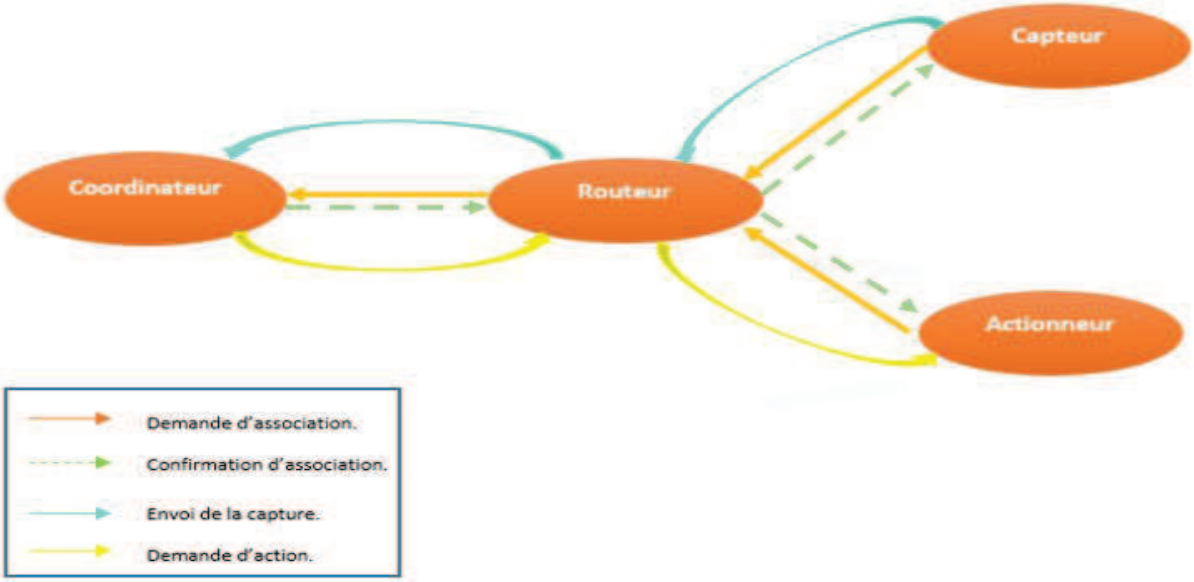

Figure III.3 : Interaction entre les nœuds du réseau.

Deux sortes de capteurs ont été utilisés. Un capteur analogique comme un capteur de température LM35 par exemple, et un capteur digital comme un détecteur d'ouverture par exemple.

Le nœud capteur prélève la température et l'achemine vers le sink. Ensuite, une fois reçue par le nœud coordinateur, cette dernière est traitée puis affichée. Si la température franchit un certain seuil, le coordinateur enverra une requête au nœud actionneur s'exprimant par l'allumage de la LED (Light Emitting Diod). De la même façon, si le capteur intercepte l'ouverture de la porte, l'information est envoyée au nœud puits qui déclenchera l'allumage de la LED.

Le fonctionnement de notre réseau est résumé dans l'organigramme de la **figure III.4**.

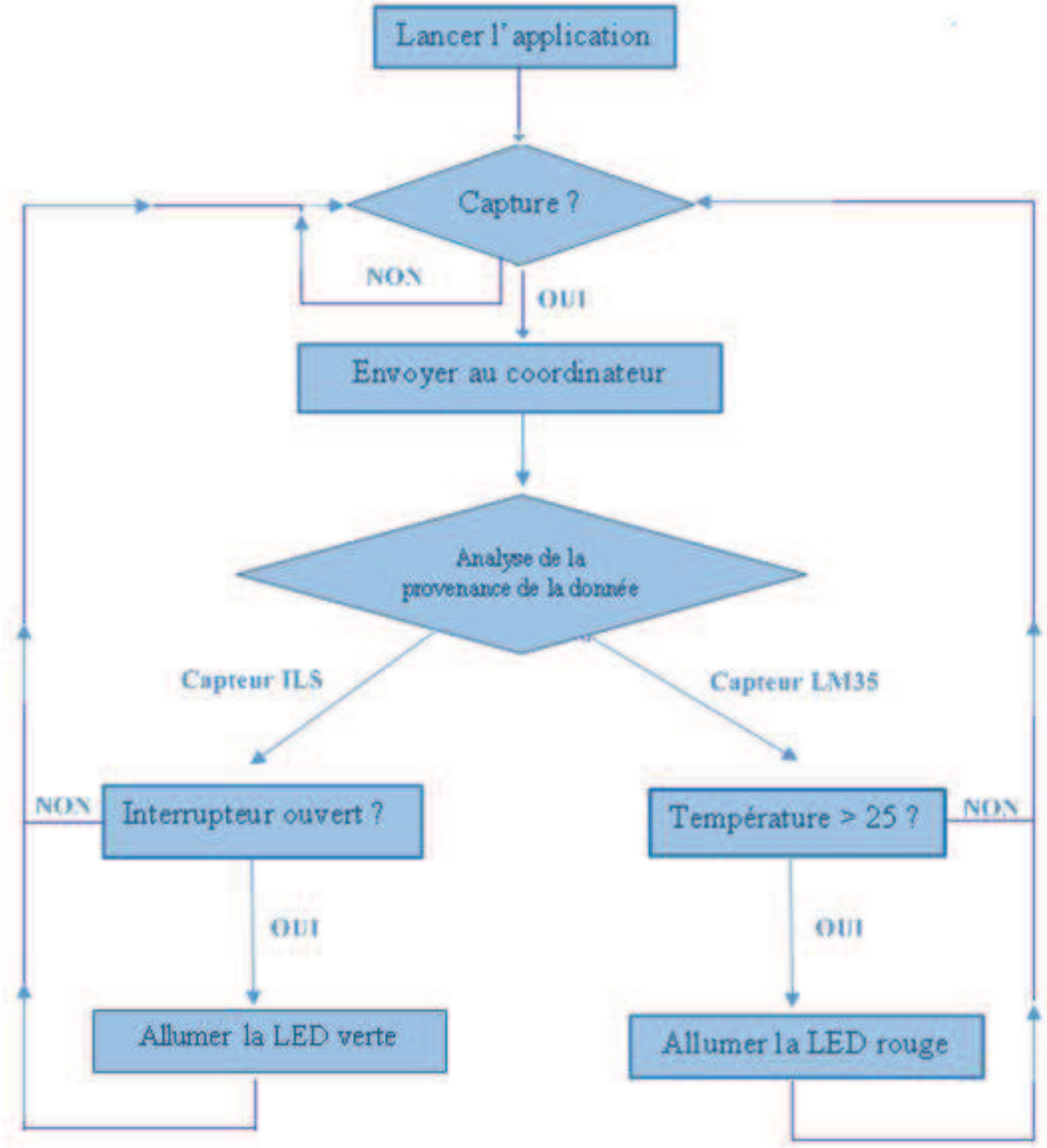

Figure III.4 : Fonctionnement du réseau.

II.4 Conception :

III.4.1 Conception matérielle :

Afin de concevoir le réseau décrit précédemment, nous aurons besoin d'un ensemble de dispositifs que nous détaillerons dans les prochains points.

III.4.1.1 Le nœud capteur :

- **La capture analogique :**

Nous avons utilisé la capture analogique pour mesurer la température. Le capteur de température est un dispositif qui permet de transformer une grandeur physique en une grandeur électrique (tension ou courant). Nous avons choisi le capteur LM35 illustré dans la **figure III.5**.

Décliné en plusieurs versions, le LM35 est un circuit intégré calibré en usine pour être utilisé comme capteur de température de précision. Sa principale particularité est que sa tension de sortie 'Vout' est linéairement proportionnelle à la température exprimée en degrés Celsius. Plus exactement, la tension de sortie 'Vout' augmente de 10 mV chaque fois que la température augmente de 1°C. Il doit être alimenté avec une tension comprise entre 3V et 20V, *3.3V* dans notre cas. La sortie de capture 'Vout' sera reliée à la pin20 de notre nœud capteur (le XBee).

Température en degré Celsius = Vout/10.

Figure III.5 : Le capteur de température LM35.

Le schéma de câblage du capteur à notre module est illustré par la **figure III.6** :

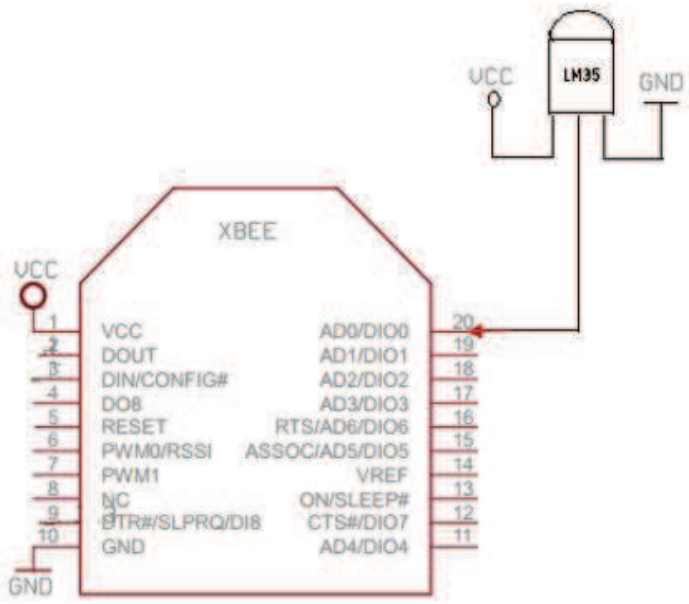

Figure III.6 : Branchement du nœud capteur avec le capteur de température LM35.

- **Capture numérique :**

Nous avons utilisé la capture numérique pour pouvoir détecter une intrusion, une ouverture de porte par exemple. Pour cela, nous avons choisi le capteur ILS (Interrupteur à Lame Sous vide).

Ce capteur est composé d'un interrupteur magnétique et d'un aimant. L'un est placé sur la porte et l'autre sur le cadre de la porte, de façon à ce que les deux éléments soient proches quand la porte est fermée. Quand l'aimant est rapproché du contact, ce dernier fonctionne comme un interrupteur fermé. Si on éloigne l'aimant, L'interrupteur est ouvert. On pourra ainsi savoir si la porte est ouverte ou fermée.

Le capteur ILS doit être alimenté avec une tension comprise entre 3V et 5V, dans notre cas, nous avons utilisé une alimentation de 3.3V. La sortie 'Vout' qui est notre sortie de capture sera reliée à la pin19 de notre nœud capteur (le XBee).

Le schéma suivant représente le branchement du capteur au module XBee :

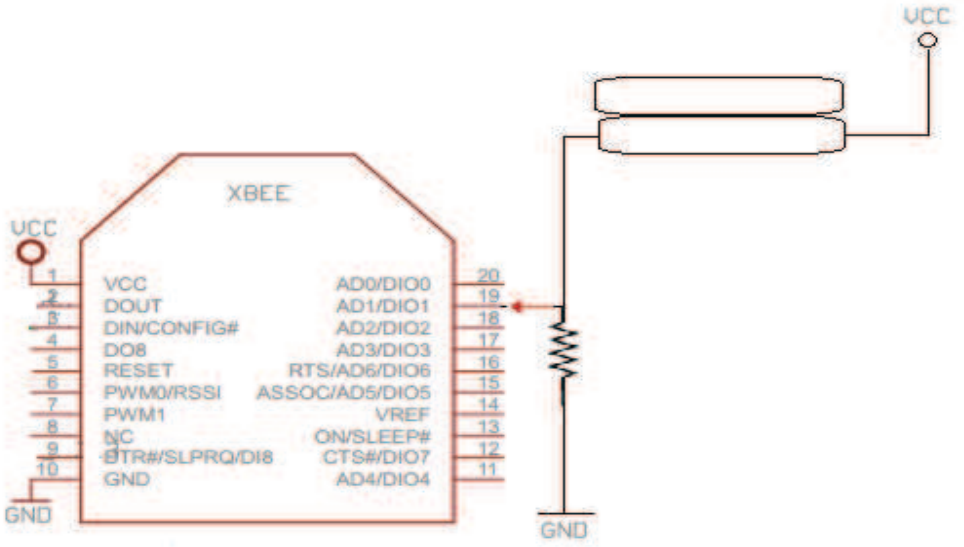

Figure III.7 : Branchement du nœud capteur avec le capteur d'intrusion ILS.

III.4.1.2 Le nœud routeur :

Cet élément est indispensable pour étendre le réseau par l'acheminement des trames d'un module à un autre. Il permet aussi aux autres modules de s'enregistrer sur le même réseau, et non exclusivement chez le coordinateur. La fonction de ce nœud est *la communication (routage)* il permet de transmettre les données recueillies par les nœuds capteurs vers le nœud coordinateur. De plus il permet d'envoyer les commandes à exécuter du coordinateur vers le nœud actionneur.

Le nœud routeur est câblé avec un branchement minimal, c'est-à-dire que le VCC doit être relié à la pin 1 du XBee, le GND à la pin 10. Il est alimenté avec une tension comprise entre 2.8V et 3.3V.

Ce nœud doit être mis à portée radio du coordinateur et du nœud capteur. Si toutefois la distance entre ces derniers est très grande, des routeurs supplémentaires peuvent être utilisés pour aider à l'acheminement des données.

La **figure III.8** montre le branchement de notre nœud routeur :

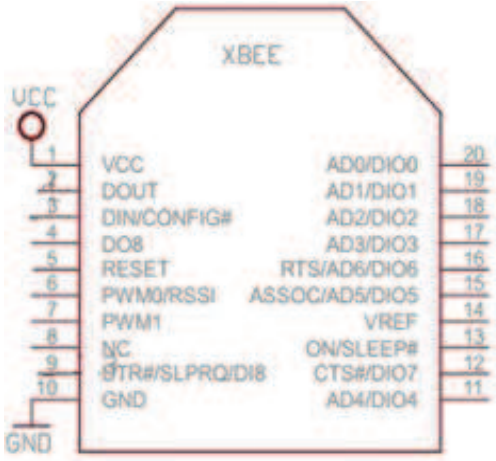

Figure III.8 : Branchement du nœud routeur.

III.4.1.3 Le nœud actionneur :

La fonction de ce nœud est l'exécution d'une tâche sur ordre du coordinateur. Si par exemple, une intrusion est détectée, l'action pourrait être le déclenchement d'une alarme ou la fermeture d'une porte. Si la température captée franchit un seuil donné, l'action pourrait être la mise en marche de la climatisation ou le déclenchement d'un arroseur. Dans notre cas, nous avons choisi comme action l'allumage d'une LED, à titre d'illustration.

Le schéma de câblage de notre actionneur est présenté dans la **Figure III.9** :

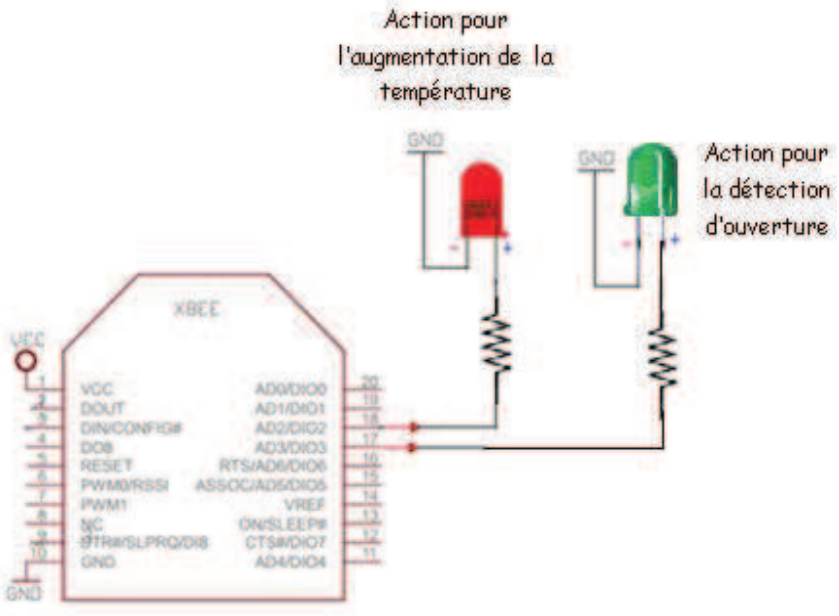

Figure III.9 : Branchement du nœud actionneur.

III.4.1.4 Le nœud coordinateur :

Ce module assure les fonctions telles que l'authentification, l'initiation de la communication, la sécurité et l'ajout des nœuds au réseau. Un coordinateur unique (sink) doit être présent dans le réseau et il est indispensable pour la mise en place de ce dernier. Il doit être actif en permanence pour répondre à tout moment aux requêtes des autres éléments du réseau.

Le coordinateur est rattaché au PC via une station d'accueil USB. Elle lui permet d'établir une connexion série avec ce dernier. Le PC est l'élément qui exécute l'application, ce qui fait de lui et du nœud coordinateur l'entité principale du réseau.

La figure suivante montre la station d'accueil sur laquelle le module XBee est branché :

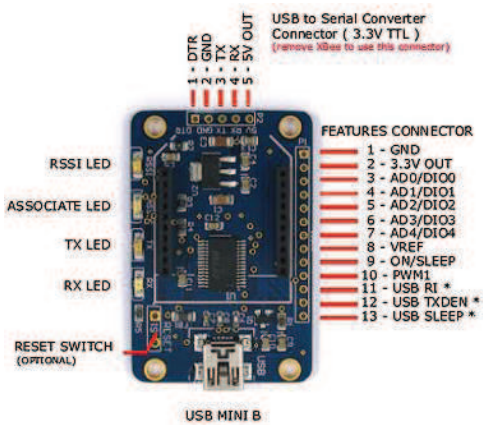

Figure III.10 : Station d'accueil USB pour XBee.

Le branchement du nœud coordinateur au PC est montré dans la **figure III.11**:

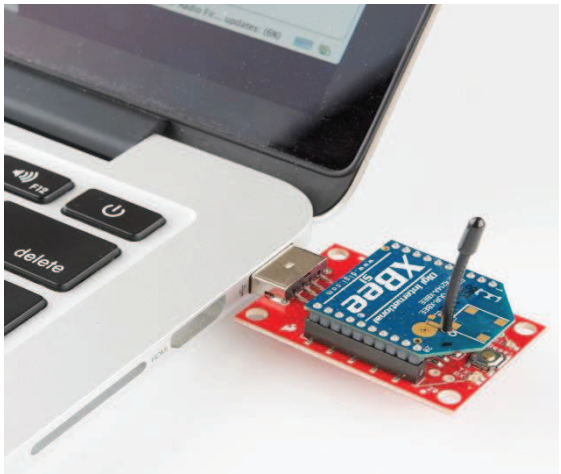

Figure III.11 : Branchement du nœud coordinateur au PC.

III.4.2 Conception logicielle :

III.4.2.1 Structure de l'application :

Notre application est constituée d'un ensemble classes qui collaborent afin de mettre en place notre réseau. La **figure III.12** Donne le diagramme de classes de notre application. Dans ce qui suit nous allons définir le rôle de chacune d'entre elles.

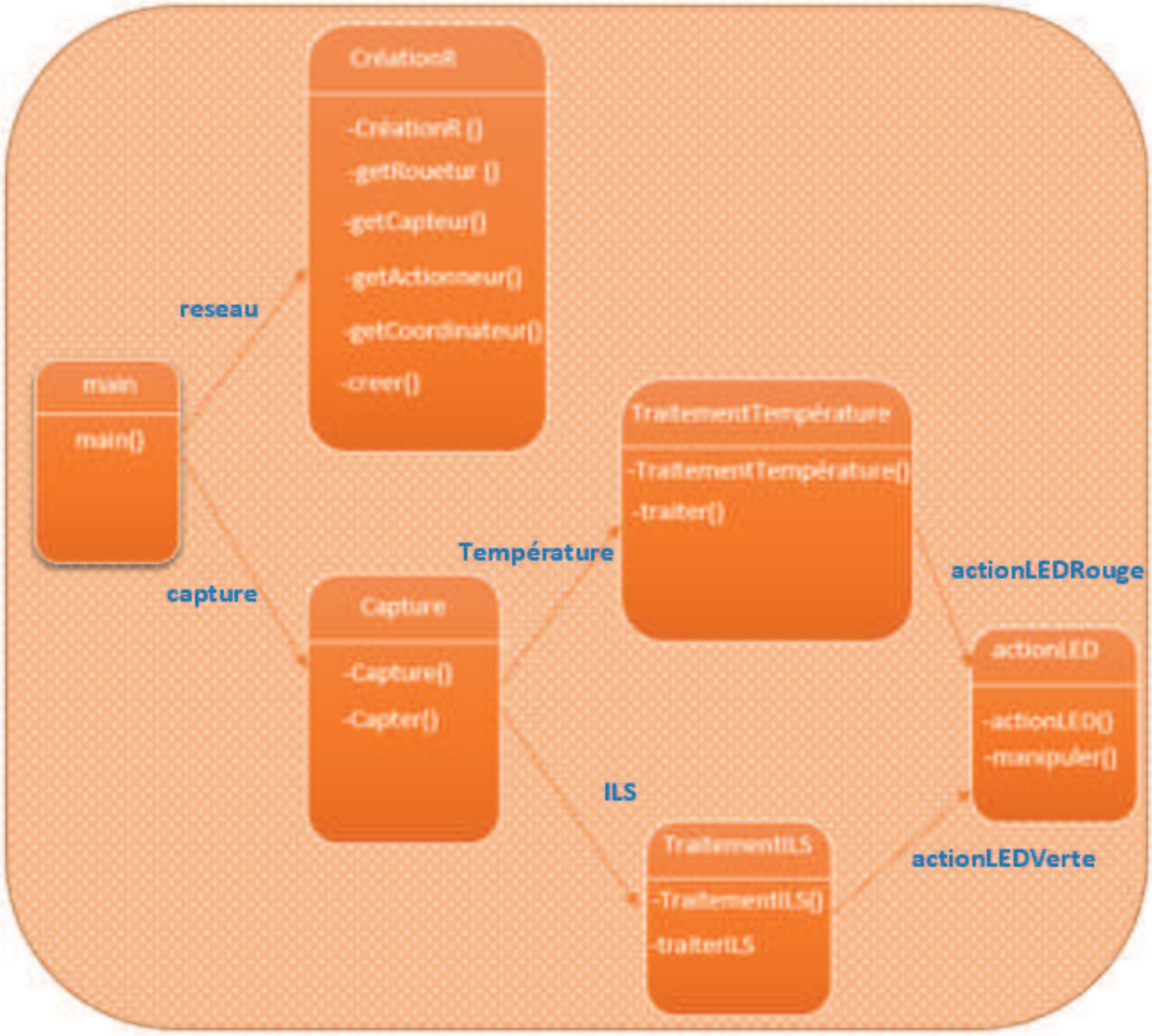

Figure III.12 : Structure de l'application.

- **Main :**

  C'est la classe principale de notre application, elle s'occupe de :

  - Créer le réseau.
  - Lancer la capture.

- **CréationR :**

  Cette classe est utilisée par la classe main pour :

  - Etablir une connexion série avec le coordinateur.
  - Initier le réseau.
  - Ajouter les autres équipements au réseau.
  - Configurer les entrées de capture.
  - Configurer la destination des données recueillies (vers le coordinateur).

- **Capture :**

  Cette classe est utilisée par la main pour :

  - Configurer l'intervalle entre chaque capture.
  - Envoyer la donnée au coordinateur où un écouteur d'événements est implémenté, qui est en attente de capture.
  - Vérifier la provenance de la capture (capteur ILS ou capteur LM35).

- **TraitementTempérature :**
  - Vérifier si la température est supérieure à un seuil prédéfinit.
- **TraitementILS :**
  - Vérifier Si l'interrupteur est ouvert.
- **ActionLED :**
  - Allumer ou éteindre la LED verte.
  - Allumer ou éteindre la LED rouge.

III.4.2.2 Comportement de l'application :

La **figure III.13** montre les interactions entre les méthodes **:**

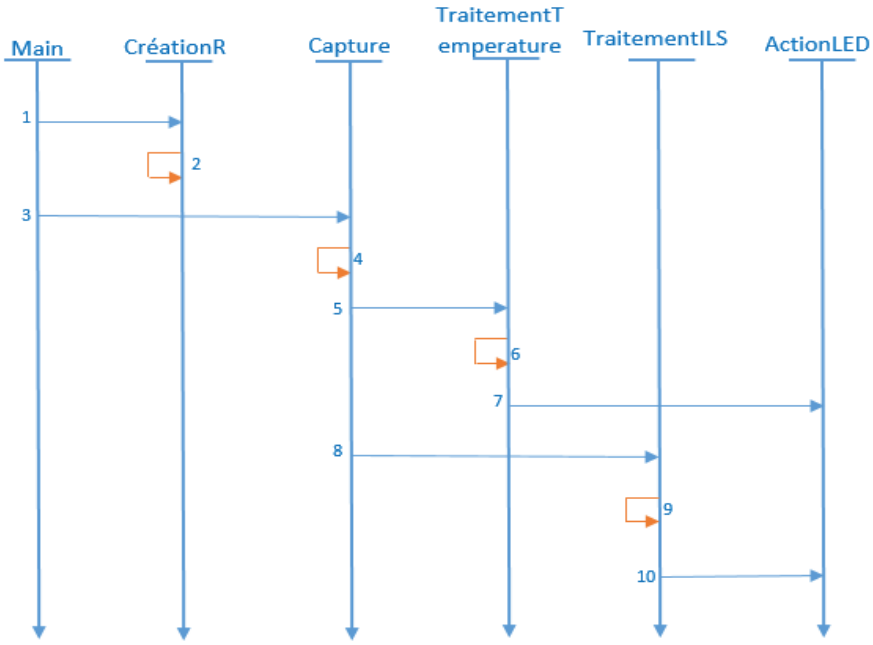

Figure III.13 : Interaction entre les classes.

1- La classe ***main*** fait appel à la classe ***CréationR.***
2- ***CréationR*** s'occupe d'initier le réseau et le mettre en marche.
3- La ***main*** fait appel à la classe ***Capture.***
4- ***Capture*** récupère les données provenant des capteurs.
5- ***Capture*** fait appel à la classe ***TraitementTemperature.***
6- ***TraitementTemperature*** vérifie si la température dépasse un seuil.
7- ***TraitementTemperature*** fait appel à la classe ***ActionLED.***
8- ***Capture*** fait appel à la classe ***TraitementILS.***
9- ***TraitementILS*** vérifie si l'interrupteur est ouvert ou fermé.
10- ***TraitementILS*** fait appel à la classe ***ActionLED.***

III.5 Domaines d'utilisation :

En raison de ses caractéristiques et de son positionnement par rapport aux autres standards de communication sans-fil personnels tels que Bluetooth ou Wi-Fi, ZigBee est plus particulièrement destiné à certains domaines d'applications, notamment dans la domotique, les domaines industriel et médical. Notre réseau pourrait être utilisé dans chacun de ces domaines.

❖ **Domotique en milieu résidentiel :**

Dans un environnement résidentiel, notre travail convient parfaitement aux applications telles que l'éclairage, les volets roulants, les systèmes d'alarme ou le contrôle du chauffage à l'intérieur du domicile par exemple, avec toutes les télécommandes associées.

Notre architecture est adaptée aux applications de ce domaine, et pourrait être mise en place en déployant les nœuds XBee comme suit :

- Contrôle du chauffage : les nœuds capteurs interceptent les mesures de température des pièces à vivre et les envoient à travers les routeurs au coordinateur qui se chargera de déclencher le chauffage en cas de baisse de température.
- Les systèmes d'alarme : Dans ce cas, les nœuds capteurs sont placés sur le cadre des portes et des fenêtres. Quand l'alarme est activée, l'ouverture d'une porte ou d'une fenêtre sera interceptée par les nœuds capteurs qui vont le signaler au coordinateur. Ce dernier ordonnera au nœud actionneur d'enclencher l'alarme et éventuellement prévenir les propriétaires des lieux en leur envoyant un SMS.

Outre la possibilité de faire interagir ces différentes fonctions à l'intérieur de la maison, le fait d'utiliser une technologie sans-fil telle que ZigBee permet de réduire les coûts d'installation, mais aussi de donner de la flexibilité permettant aux particuliers de modifier et de faire évoluer leurs installations.

❖ **Domaine médical :**

Notre travail peut s'adapter à ce domaine, par exemple pour le télé-monitoring de la fonction cardiaque d'un patient. Un module XBee peut être placé dans un boitier qui sera relié à des électrodes. Le patient pose le boîtier sur son thorax pour enregistrer et transmettre son ECG via le XBee vers un centre de traitement (où se trouve le coordinateur) qui en cas d'irrégularité de la fonction cardiaque déclenche (via le nœud actionneur) l'envoie d'un message vers le médecin traitant.

III.6 Conclusion :

Dans ce chapitre, nous avons tout d'abord décrit notre approche pour la réalisation des objectifs de notre projet. Nous avons analysé nos objectifs et nous en avons déduit nos besoins matériels et logiciels. Ensuite nous avons entamé la description de notre conception matérielle ainsi que de notre conception logicielle. Enfin, nous avons cité quelques domaines dans lesquels notre travail pourra être utile.

Dans le chapitre qui suit, nous allons décrire les outils qui nous ont permis de développer notre projet. Nous allons ainsi donner les résultats des tests effectués sur notre réseau.

Chapitre IV : Réalisation

IV.1 Introduction :

Dans la démarche de réalisation de notre projet, nous avons eu recours à l'utilisation de plusieurs outils tant au niveau de développement que de configuration. Dans la suite de ce chapitre, nous allons citer ces différents outils, puis nous allons présenter l'implémentation de notre application java qui se base sur une bibliothèque spécifique pour l'utilisation des équipements XBee. Pour finir, nous effectuons une série de tests afin d'observer le comportement de notre réseau.

IV.2 Outils de développement :

Pour développer notre application, nous avons eu recours à divers éléments notamment le langage de programmation JAVA et son environnement de développement Eclipse qui nous ont permis l'implémentation de notre application en utilisant la bibliothèque XBee. Le but de cette bibliothèque étant de fournir les outils nécessaires pour faire communiquer les modules XBee, qui doivent être configurés au préalable en utilisant le logiciel X-CTU.

a. JAVA :

Le langage java est un langage de programmation orienté objet. Il a été mis au point à partir de 1990 par la firme Sun Microsystems et officiellement présenté en 1995. Le but de JAVA à l'époque était de constituer un langage de programmation pouvant être intégré dans les appareils électroniques et électroménagers afin de pouvoir les contrôler.

Le langage JAVA reprend en grande partie la syntaxe du langage C++. Il a donné naissance à une machine propriétaire entièrement spécifiée par le logiciel, dite machine virtuelle ; JAVA a donné aussi naissance à une déclinaison pour les périphériques mobiles et embarqués. La portabilité du bytecode JAVA est assurée par la machine virtuelle et éventuellement par des bibliothèques standard incluses dans un JRE (JAVA Runtime Environment) **[13]**.

JRE :

Java Runtime Environment est un plug-in requis pour l'exécution de programmes Java sur différentes plateformes. Il contient le JVM, les bibliothèques de base et d'autres composants supplémentaires pour l'exécution d'applications et d'applets écrits dans Java.

JDK :

Java Development Kit est un bundle de logiciels utilisés pour développer des applications Java. Il désigne un ensemble de bibliothèques ainsi que les outils avec lesquels le code Java peut être compilé, transformé en bytecode destiné à la machine virtuelle Java **[14]**.

b. Eclipse :

« Eclipse » est un environnement de développement intégré (IDE :Integrated Development Environment). Il contient un espace de travail de base et un système de plug-in extensible pour la personnalisation de l'environnement. Écrit principalement en Java, Eclipse peut être utilisé pour développer des applications. Par le biais de divers plug-ins, Eclipse peut également être utilisé pour développer des applications dans d'autres langages de programmation: Ada, ABAP, C, C ++, COBOL, Fortran, etc…

Le code source initial de « Eclipse » provient de IBM VisualAge. Le kit de développement Eclipse (SDK), qui comprend les outils de développement Java, est destiné aux développeurs Java. Les utilisateurs peuvent étendre ses capacités par l'installation de plug-ins écrits pour la plateforme Eclipse, tels que les outils de développement pour d'autres langages de programmation, et peuvent aussi programmer leurs propres modules de plug-in.

c. Bibliothèque XBee :

La Bibliothèque Java XBee est une API facile à utiliser ecrite en Java qui simplifie le développement d'applications qui communiquent en utilisant les modules XBee.

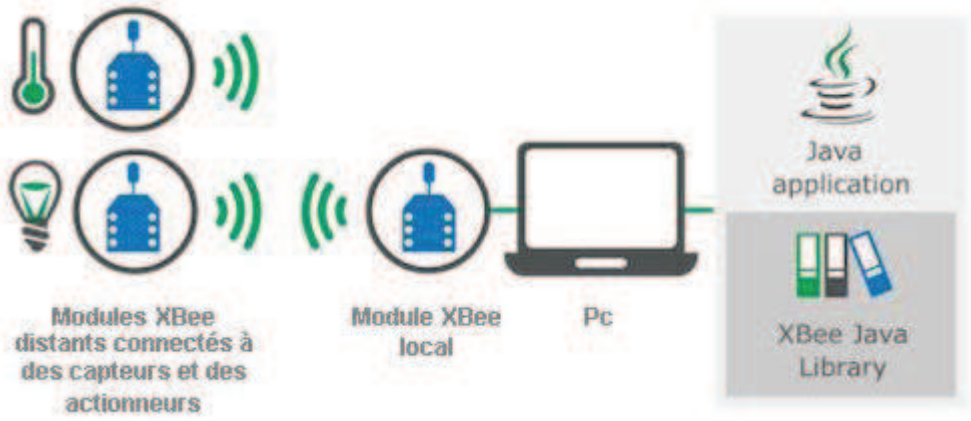

Figure IV.1 : Architecture de la bibliothèque XBee.

Cette bibliothèque comprend le code source Java, des programmes de tests de base, et de multiples exemples qui montrent comment utiliser les API disponibles. Cette bibliothèque est délivrée comme un projet Open Source. Son architecture est indiquée dans la **figure IV.1**.

Le projet est hébergé dans github et sa documentation est disponible en ligne. Les principales caractéristiques de la bibliothèque comprennent **[15]**:

- Prise en charge des protocoles ZigBee, 802.15.4, DigiMesh et dispositifs Point-à-Multipoint XBee.
- Prise en charge du mode de fonctionnement API.
- Gestion des objets XBee (locaux et éloignés).
- Découverte de périphériques distants XBee associés au même réseau que le périphérique local.
- La configuration des périphériques XBee locaux et distants:
- Transmission des données à tous les dispositifs XBee sur le réseau (broadcast) ou à un dispositif spécifique (unicast).
- Réception des données à partir de périphériques distants XBee.
- Réception des changements d'état du réseau liés au dispositif XBee local (connexion, déconnexion, mise en veille, …etc.)
- la gestion des lignes IO:
  - Configurer les lignes IO.
  - Mettre une valeur sur la ligne IO.
  - Lire la valeur de la ligne IO.
  - Recevoir des captures de données IO de tout dispositif XBee distant sur le réseau.

d. X-CTU :

X-CTU est une application multi-plateforme libre conçu pour permettre aux développeurs d'interagir avec des modules RF Digi grâce à une interface graphique simple à utiliser. Il comprend de nouveaux outils qui facilitent la mise en place, la configuration et le test de modules RF XBee.

X-CTU comprend tous les outils dont un développeur a besoin afin de manipuler rapidement les modules XBee. Il comprend aussi des caractéristiques uniques comme une vue graphique du réseau, qui représente graphiquement le réseau XBee avec la force du signal de chaque connexion, et le générateur de trames API XBee, qui contribue de manière intuitive à construire et interpréter ces dernières **[16]**.

IV.3 Réalisation :

La partie réalisation de notre travail comporte deux étapes : La première étape consiste à configurer les modules XBee, et la seconde étape à implémenter notre application de gestion de RCSF.

IV.3.1 Configuration des modules XBee :

Comme cité dans le chapitre précédent, nous avons besoin de trois types de modules XBee, coordinateur API, routeur API et end device API que nous allons configurer en utilisant le logiciel X-Ctu.

Dans ce qui suit, nous allons prendre pour exemple la configuration du module coordinateur.

1- Lancer l'application X-Ctu :

Figure IV.2 : Ouverture du logiciel X-CTU.

2- Ajout du module Coordinateur :

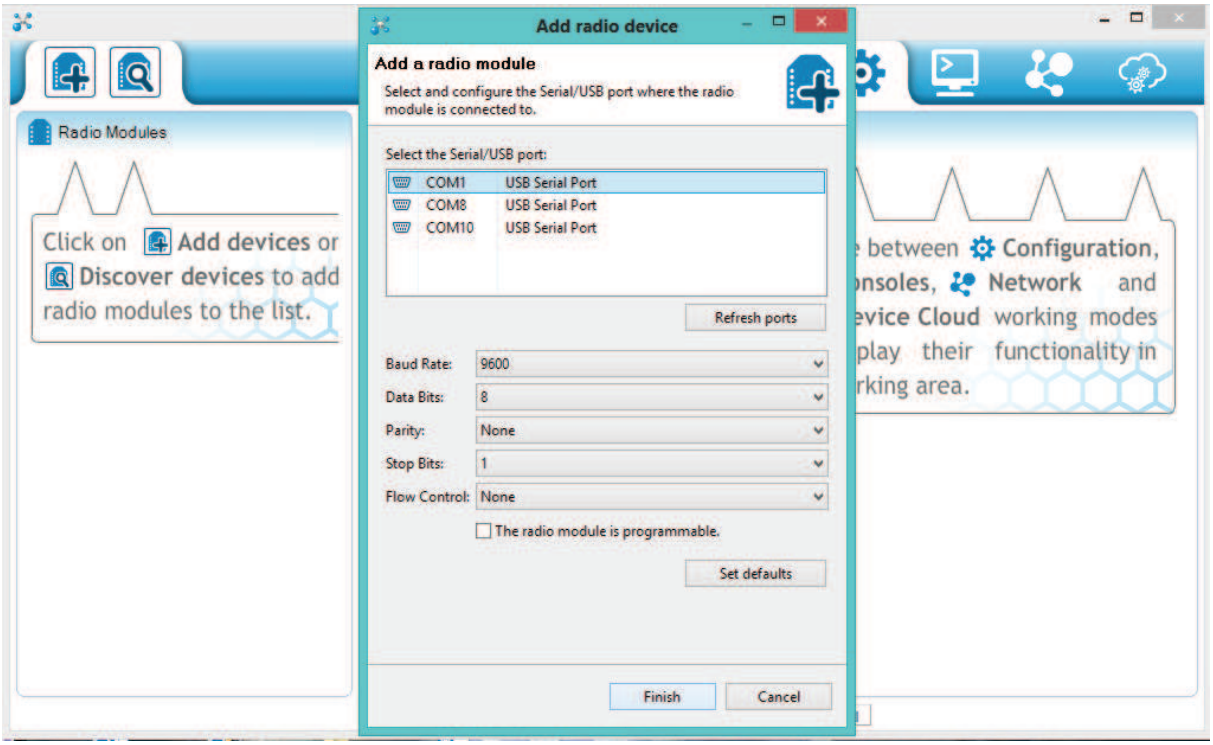

Figure IV.3 : Sélection du port de communication du coordinateur.

3- Choisir le firmware à charger sur le module XBee (Coordinateur) :

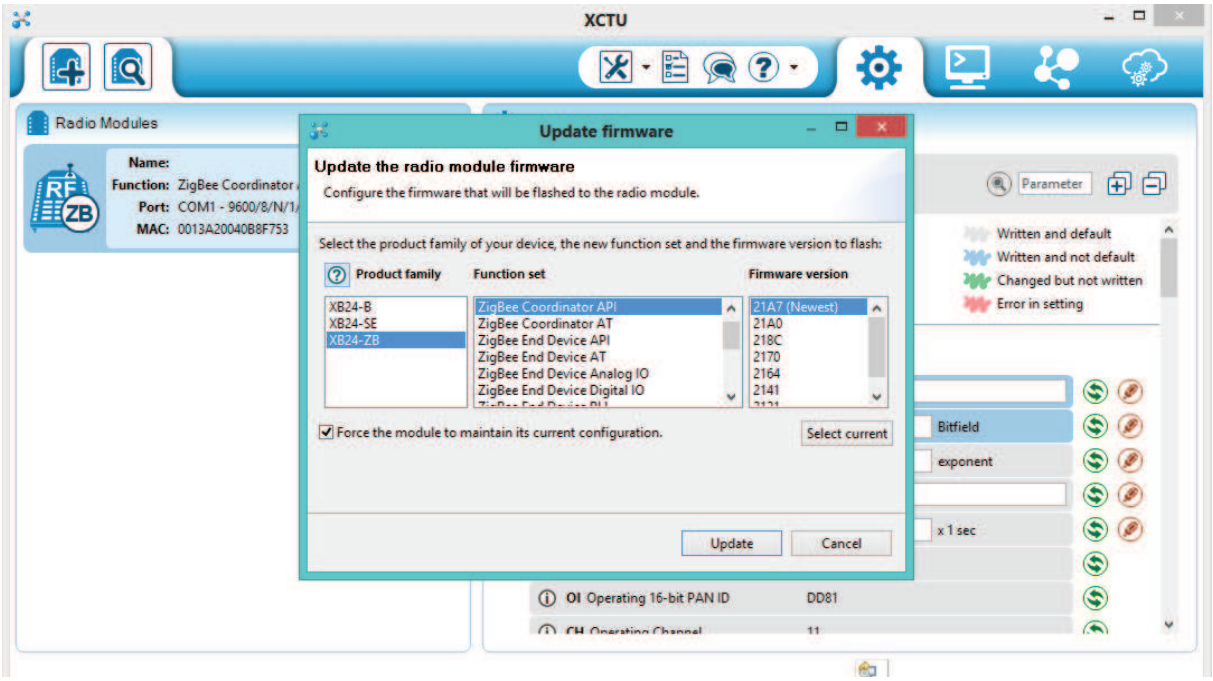

Figure IV.4 : Choix du firmware.

Après que les modules XBee aient été configurés, ils seront utilisés dans la suite de ce chapitre, pour l'implémentation de notre application.

IV.3.2 Implémentation de l'application de gestion du RCSF :

Notre application est composée de cinq classes que nous allons présenter ci-suit :

a. Classe de la création du réseau :

Cette classe s'occupe de l'établissement d'une communication série avec le coordinateur pour l'envoie des commandes nécessaires à la création, la configuration et l'ajout des équipements au réseau.

```
public class CréationR {

// Déclarer les variables et constantes utilisées.

 public void creer(){

      XBeeDevice localDevice = new XBeeDevice(PORT, BAUD_RATE); // Définition des
      paramètres de la communication, port de communication et débit.

      coordinateur=localDevice; // Le coordinateur est le XBee local rattaché au
      PC.
      ……
      try {
      localDevice.open();// Ouvrir la connexion série avec le coordinateur.

      ……

      // Déclarer les adresses des dispositifs.
      // Ajout des équipements au réseau.
      // Déclarer les entrées de capture.

      ……

      } catch (XBeeException e) {
                  e.printStackTrace();
                  localDevice.close();// Fermer la connexion série.
                  System.exit(1);
            }


      }}
```

b. Capture de données :

Cette classe est responsable de la capture de données analogiques (température) et numériques (ouverture de porte). Elle commence par définir un intervalle entre chaque capture, puis à l'arrivée d'une nouvelle capture de donnée, elle identifie sa provenance (ILS ou LM35) et fait appel à la fonction associée en vue de son traitement (TraitementILS ou TraitementTempérature).

```
public Class Capture{

public Capture () {}

public void capter(RemoteXBeeDevice remoteDevice, XBeeDevice localDevice,
RemoteXBeeDevice ActionneurDevice){

 try{
        remoteDevice.setIOSamplingRate(IO_SAMPLING_RATE); // Définir l'intervalle
        entre chaque capture.

        ..................................................

        localDevice.addIOSampleListener(new IIOSampleReceiveListener()
         { // Implémenter l'écouteur d'événements au niveau du coordinateur.
        @Override

            public void ioSampleReceived(RemoteXBeeDevice Device, IOSample
            ioSample) // Définir la valeur captée et sa provenance.
            {
            ..................................................

            IOValue value = remoteDevice.getDIOValue(DIGITAL_LINE); // recueillir
            la donnée de l'interrupteur ILS.

            // Faire appel à la classe TraitementILS.
            ..................................................

            valeurTempérature = Device.getADCValue(ANALOG_LINE); // recueillir la
            température.

            // Faire appel à la classe TraitementTempérature.

            } catch (TimeoutException e) {
            e.printStackTrace();
            }) ;
      };
```

c. Traitement de la capture de température :

Cette classe permet de tester la température recueillie, et quand cette dernière dépasse le seuil préalablement définit on fait appel à la classe ActionLED.

```
public Class TraitementTempérature{

public () {}



public void traiter(double Température, RemoteXBeeDevice ActionneurDevice) throws
TimeoutException, XBeeException {

              ……..
             if(Température >= SEUIL_TEMPERATURE){
             }
             // Faire appel à la classe ActionLED.
             ……..
              }
      }
```

d. Traitement de la détection d'ouverture (sur ILS) :

Cette classe permet de tester la donnée captée, et quand l'interrupteur est ouvert la classe ActionLED est appelée.

```
    public class TraitementILS {
public TraitementILS (IOValue value, RemoteXBeeDevice ActionneurDevice) throws
TimeoutException, XBeeException {

      }

    public void traiterils(IOValue value, RemoteXBeeDevice ActionneurDevice)
            throws TimeoutException, XBeeException {
             ……..
             if(value == IOLINE_value_HIGH){ // Si l'interrupteur est ouvert
             // Faire appel à la classe ActionLED.
             ……..
                          }
             ……..
             }      }
```

e. Le programme de manipulation de la LED :

Cette classe s'occupe d'allumer ou d'éteindre la LED rouge quand cette dernière est appelée par la classe TraitementTempérature, ou de manipuler la LED verte quand elle est appelée par la classe TraitementILS.

```
public class ActionLED {

	public ActionLED () {

	.....
	}
	public void manipuler(IOLine Ligne_LED,RemoteXBeeDevice
ActionneurDevice,IOValue value) throws TimeoutException, XBeeException{

	.....
	//Allumer/éteindre la LED verte si l'interrupteur est ouvert/fermé.
	//Allumer la LED rouge si la température dépasse le seuil fixé, sinon
	l'éteindre
	}
}
```

f. La classe main :

C'est la classe principale qui fait appel à la classe CréationR pour l'initialisation du réseau. Ensuite elle lance la capture de données grâce à la classe Capture.

```
public class main {

	static RemoteXBeeDevice Routeur;
	static RemoteXBeeDevice Capteur;
	static RemoteXBeeDevice Actionneur;
	static XBeeDevice Coordinateur;


	public static void main(String[] args) {
		// Créer le réseau.
		CréationR Reseau= new CréationR();
		Reseau.creer();

		// Récupérer les éléments du réseau.

		Coordinateur=Reseau.getCoordinateur();
		Capteur=Reseau.getCapteur();
		Routeur=Reseau.getRouteur();
		Actionneur=Reseau.getActionneur();
		// Lancer la capture.
		Capture capture=new Capture(Capteur,Coordinateur,Actionneur);
			capture.capter(Capteur, Coordinateur, Actionneur);
			}
}
```

IV.4 Câblage et tests du réseau :

Nous présentons dans cette dernière partie les schémas de câblage de nos modules et les différents tests effectués sur ces derniers.

IV.4.1 Les composants :

Notre montage nécessite les composants suivants :

- Quatre(4) modules XBee.
- Trois stations d'accueils pour XBee.
- Un capteur LM35.
- Un capteur ILS.
- Un bread board.
- Deux LEDs.
- Deux Resistances de 330 Ω.
- Des fils.
- Des pincettes.

La représentation graphique de notre réseau sous X-Ctu est montrée dans la **figure IV.5**.

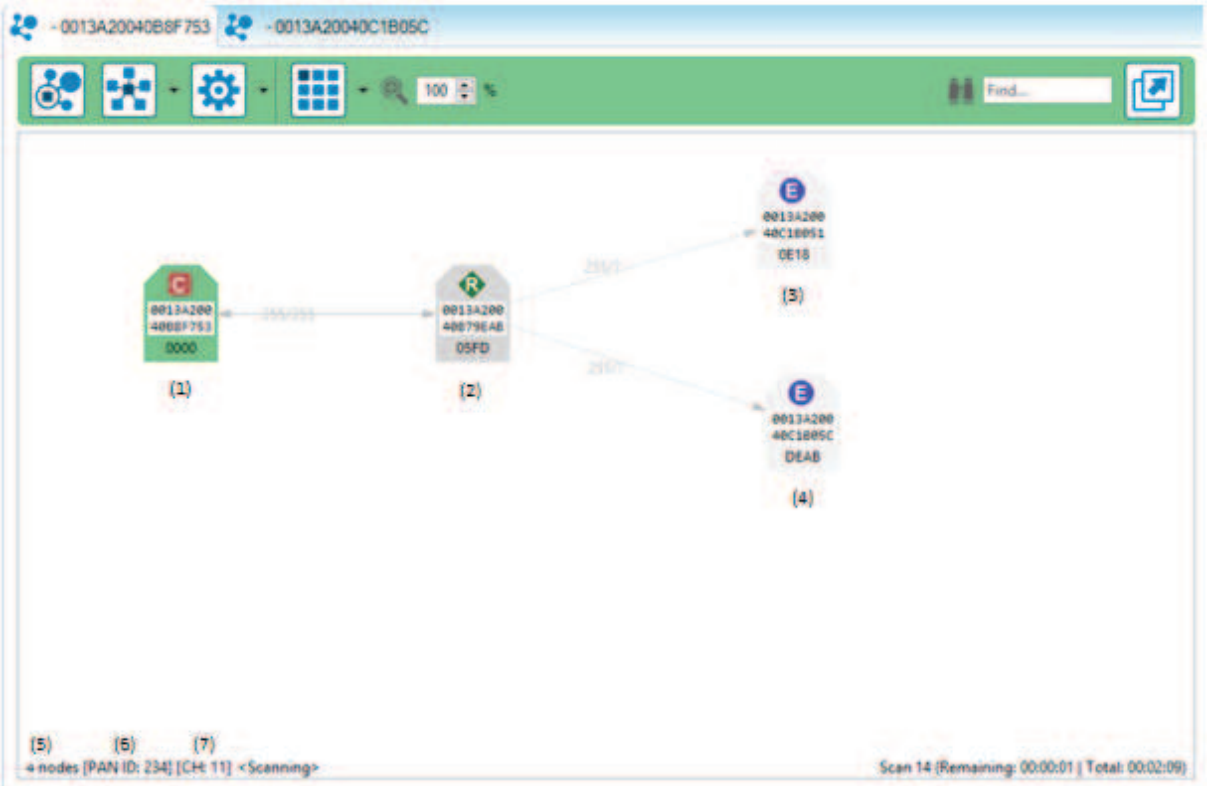

Figure IV.5 : Représentation graphique de notre réseau.

Comme le montre la **figure IV.5**, notre réseau est constitué de 4 nœuds (5) communiquant sur le canal numéro 11 (7) en utilisant le PAN ID 234 (6). Le nœud routeur (2) ayant l'adresse réseau '05FD' est rattaché au coordinateur (1) dont l'adresse réseau est '0000'. Les deux end devices (3) et (4) sont associés au routeur et prennent comme adresse '0E18' et 'DEAB' respectivement.

IV.4.2 Test de capture de données :

a- Teste de capture d'ouverture de porte :

La figure suivante montre le branchement des nœuds capteur et actionneur.

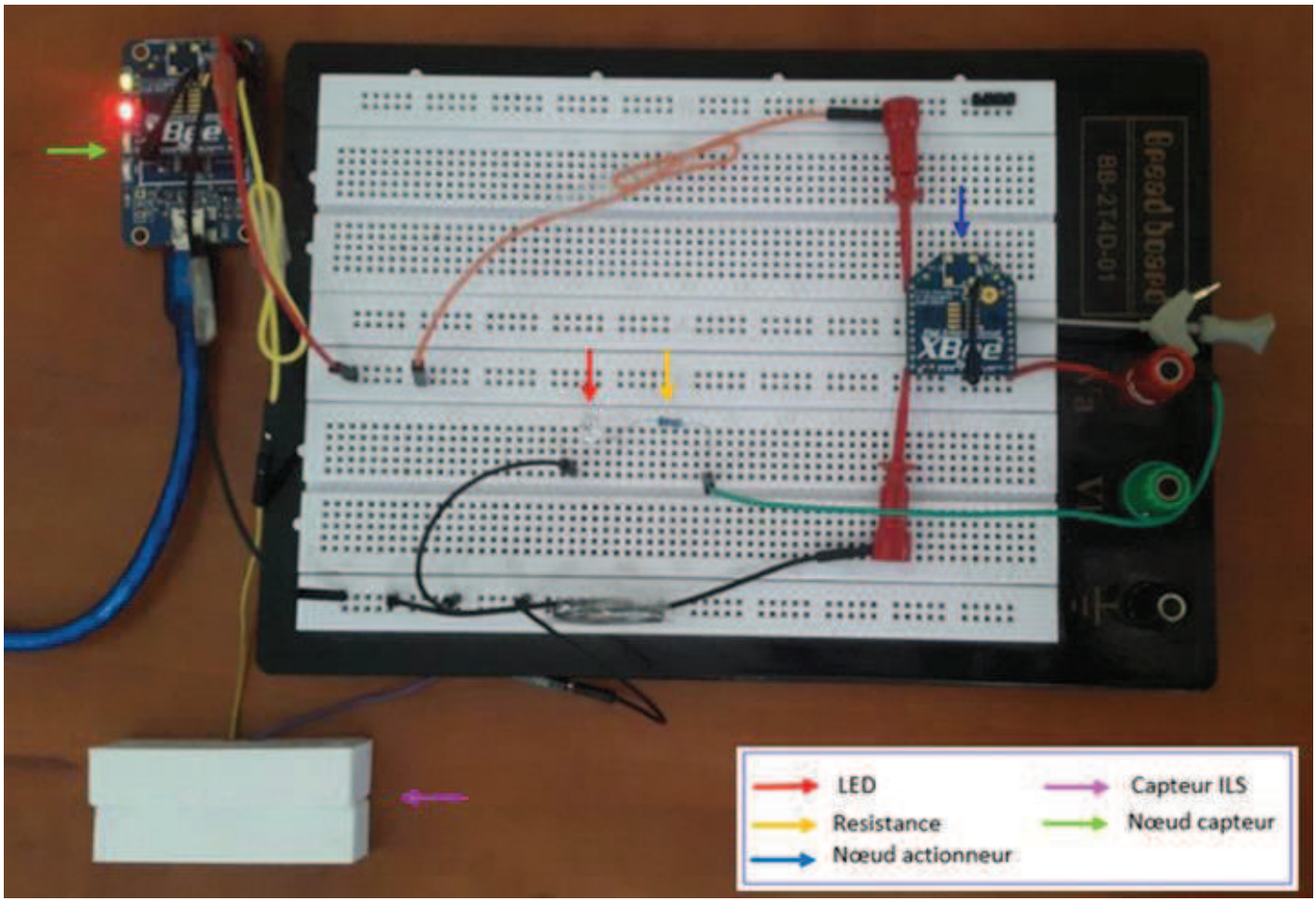

Figure IV.6 : Câblage du Capteur ILS.

La **figure IV.7** montre le résultat d'exécution du programme de notre application avec l'interrupteur ILS fermé. Nous remarquons que la donnée recueillie par le capteur dont l'adresse MAC est '0013A20040C18051' à travers la pin 'DIO1/AD1' est à l'état 'low' ce qui indique que le capteur ILS est fermé et que la LED de la **figure IV.6** est éteinte.

```
mainapp.java   capture.java   actionLED.java   traitementils.java   creationR.java   traitementtempérature.java

2
3  import com.digi.xbee.api.RemoteXBeeDevice;
11
12
13 public class capture {
14
15
16     private static final IOLine ANALOG_LINE = IOLine.DIO0_AD0;//temperature
17     private static final IOLine DIGITAL_LINE = IOLine.DIO1_AD1;//ILs
18
19     private static final int IO_SAMPLING_RATE = 1000; // 1 second.
20     traitementtempérature Température=new traitementtempérature();
21      traitementils ILS= new traitementils();
22     public capture() {}
23
24
25 public void capter(RemoteXBeeDevice remoteDevice, XBeeDevice localDevice, RemoteXBeeDevice ActionneurDevice){
26
27
28
29     try {
30         System.out.println(" +---------------------------------------------+");
31         System.out.println(" | ************Capture de données*************|");
32         System.out.println(" +---------------------------------------------+\n");
```

Problems Javadoc Declaration Console

```
<terminated> mainapp [Java Application] C:\Program Files\Java\jre1.8.0_31\bin\javaw.exe (14 juin 2015 19:47:36)
 +---------------------------------------------+
 | ************Capture de données*************|
 +---------------------------------------------+

New sample received from 0013A20040C1B051 - {[DIO1/AD1: Low], [DIO0/AD0: 226]}
New sample received from 0013A20040C1B051 - {[DIO1/AD1: Low], [DIO0/AD0: 226]}
```

Figure IV.7 : Résultat d'exécution du programme avec interrupteur fermé.

La figure suivante montre le résultat de l'ouverture du capteur ILS.

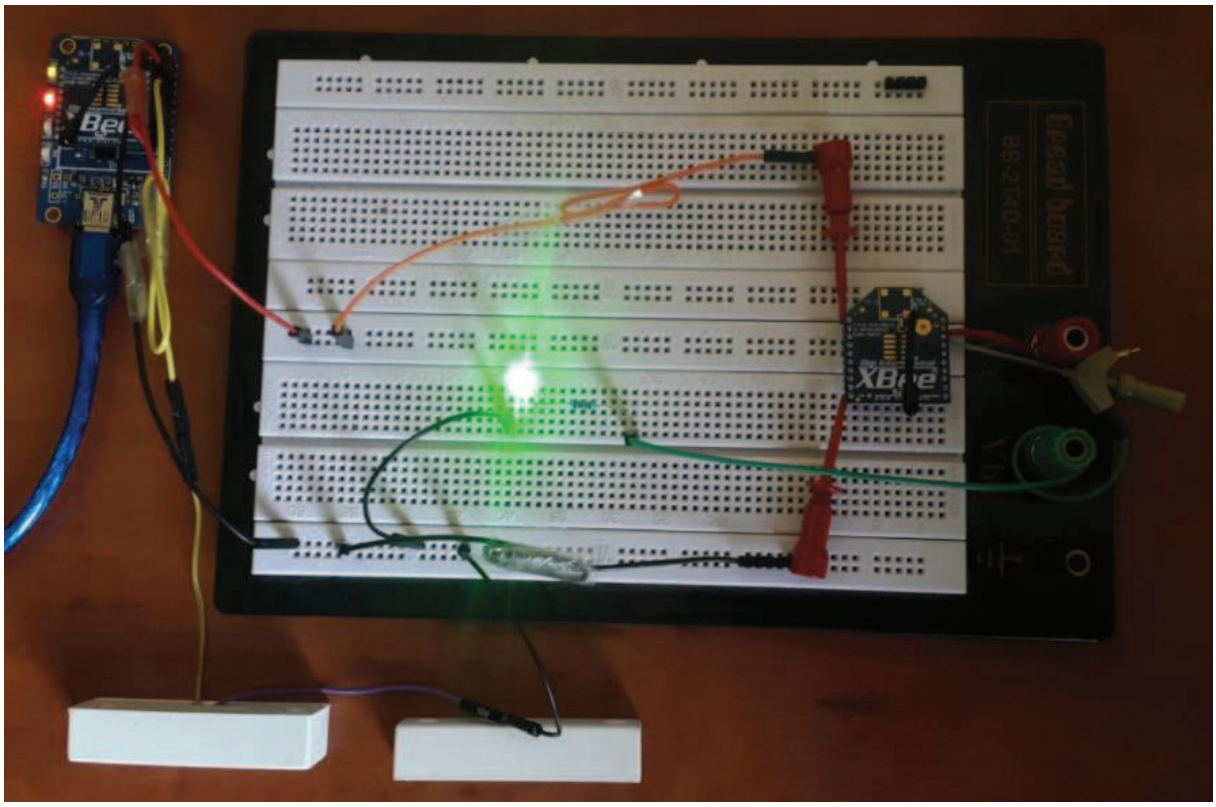

Figure IV.8 : Ouverture de l'interrupteur ILS.

La **figure IV.8** montre que l'ouverture de l'interrupteur ILS est interceptée par le nœud capteur, ce qui déclenche l'allumage de la LED verte par le nœud actionneur. On peut retrouver ce scénario dans l'exécution de notre application qui est montrée par la **figure IV.9**.

La donnée captée sur la pin 'DIO1/AD1' par le nœud capteur est à l'état 'high' ce qui indique l'ouverture du capteur ILS, ce qui est traduit par l'allumage de la LED par le nœud actionneur et l'affichage d'un message d'alerte.

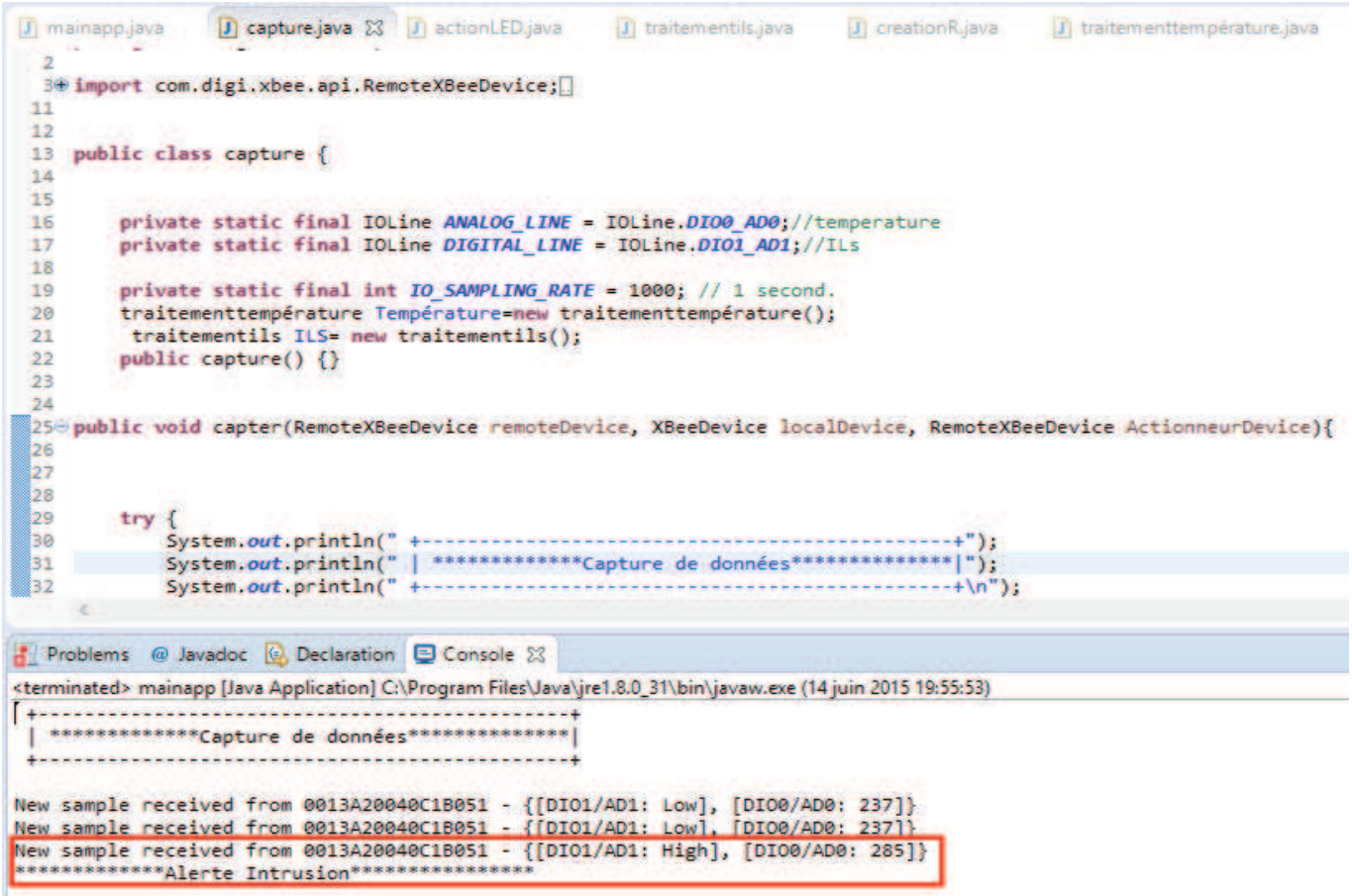

Figure IV.9 : Résultat d'exécution du programme avec interrupteur ouvert.

b- Teste de capture de température :

Notre câblage pour la capture de la température est représenté par la **figure IV.10** :

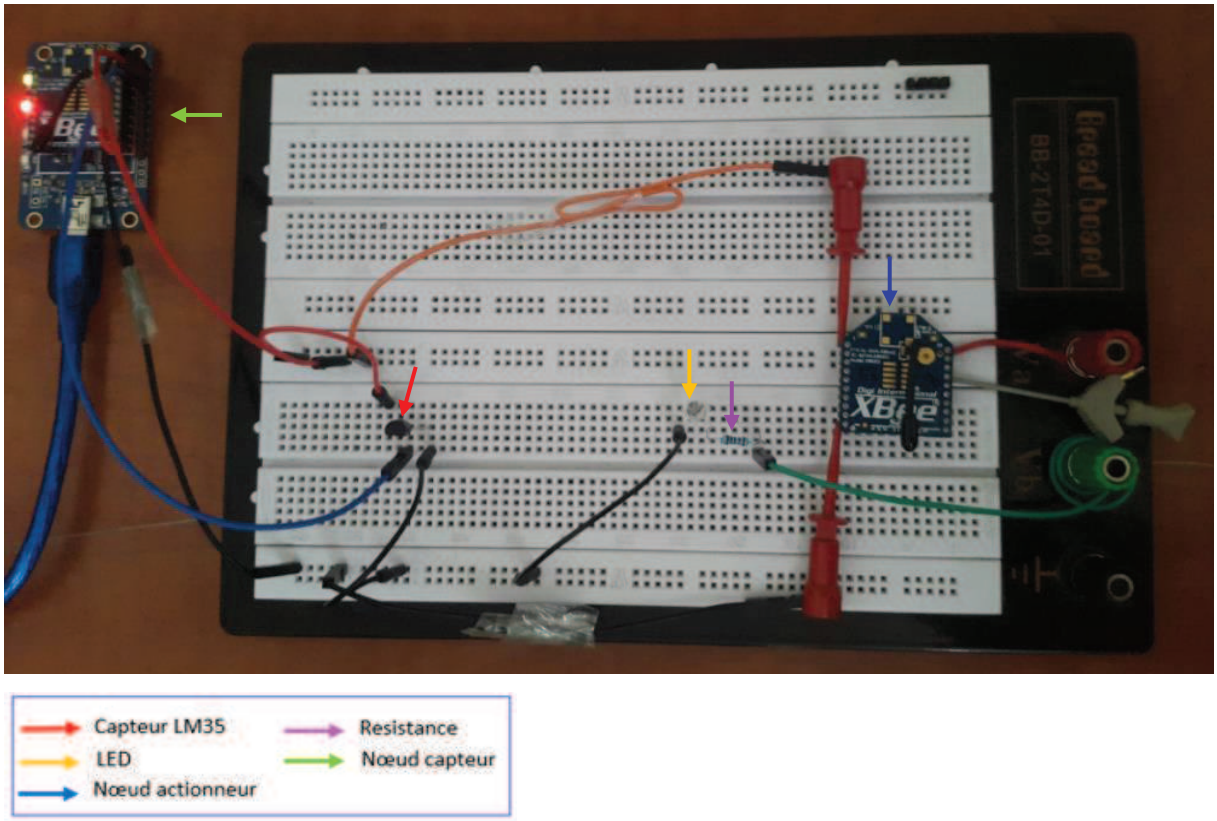

Figure IV.10 : Câblage du capteur LM35.

Lors de l'exécution de notre application montrée par la **figure IV.11**, la température captée est affichée à l'écran. Quand cette dernière ne dépasse pas le seuil fixé, dans notre cas, par exemple, 21.5°C, l'exécution continue sans aucun changement.

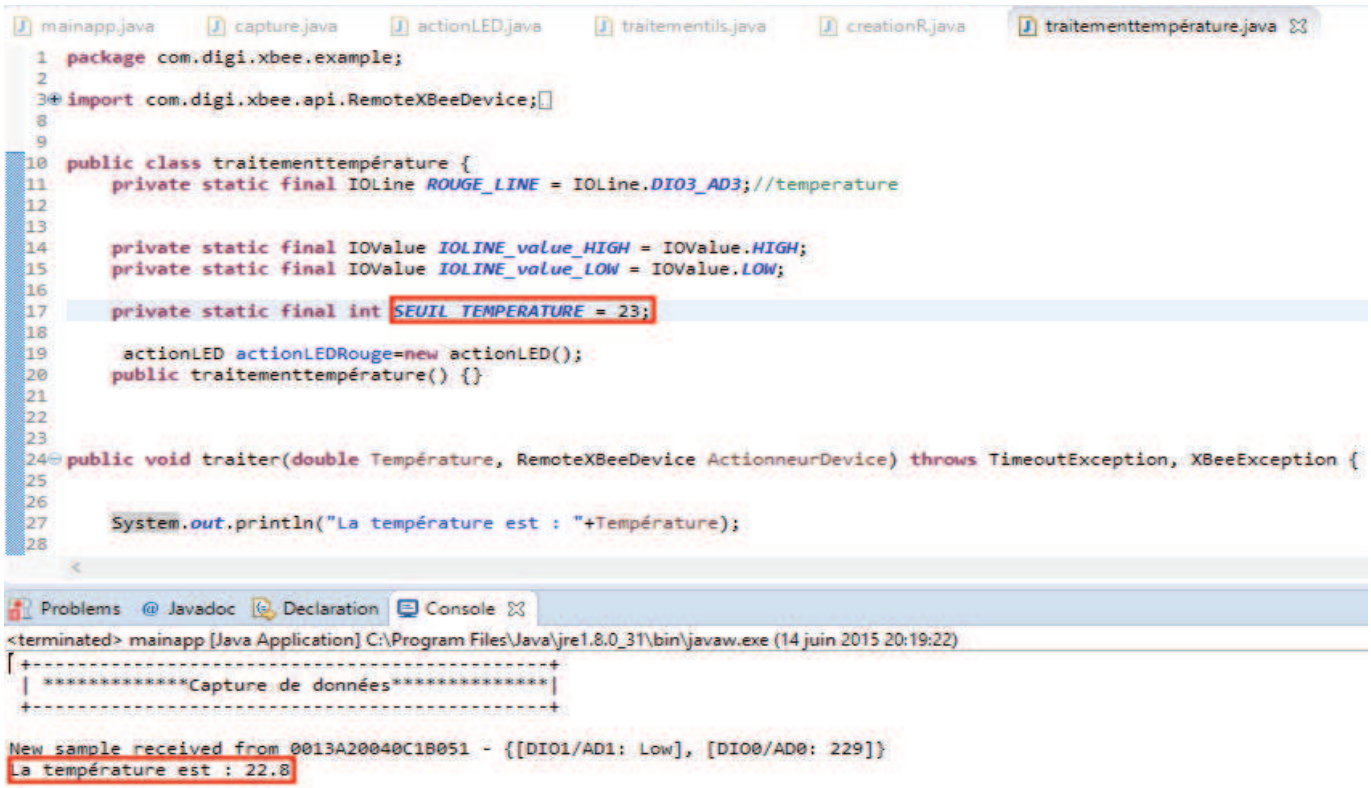

Figure IV.11 : Résultat d'exécution du programme traitement de température.

Pa contre, lorsque la température dépasse le seuil, un message d'alerte est affiché, et l'allumage de la LED est déclenché. Ces deux résultats sont montrés dans les deux **figures IV.12** et **IV.13**.

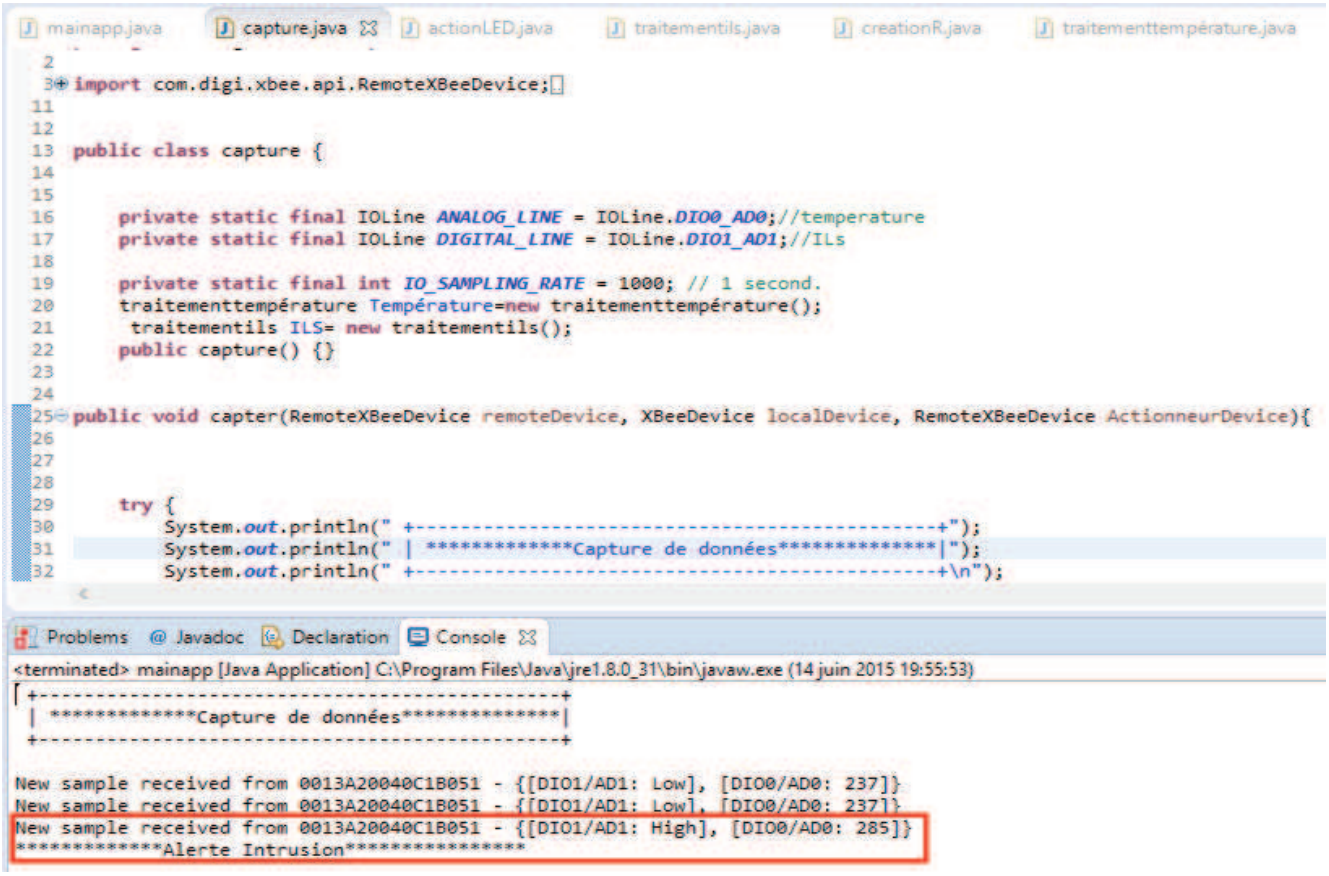

Figure IV.12 : Résultat de l'exécution après atteinte du seuil.

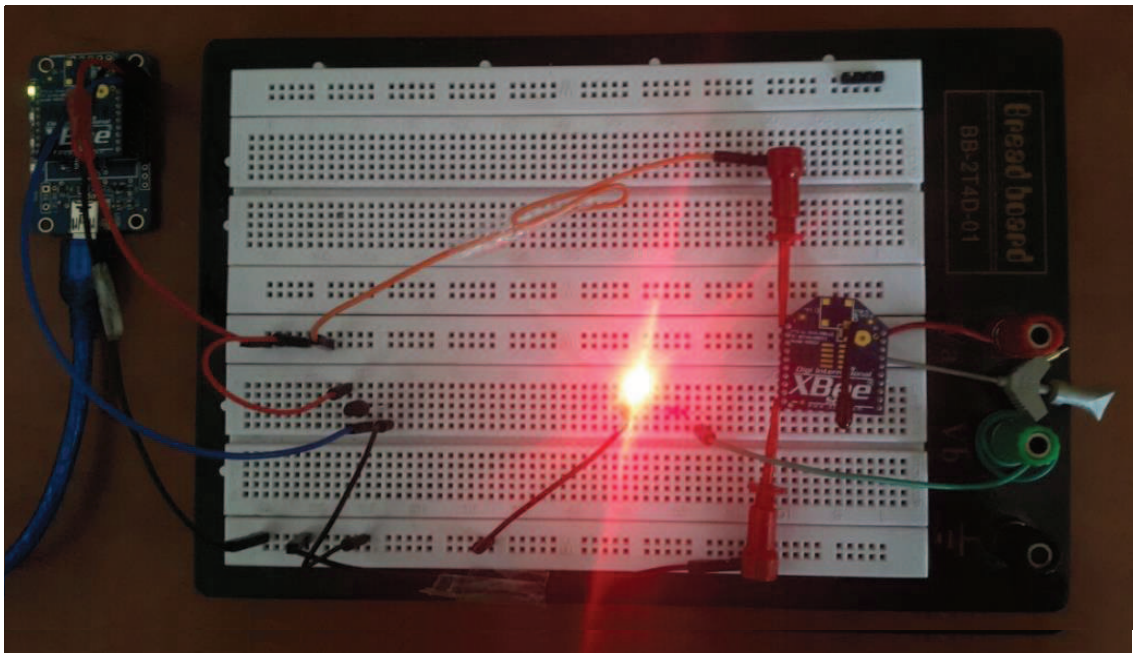

Figure IV.13 : Action associée au dépassement du seuil de température prédéfini.

IV.4.3 Tests de fonctionnement des routeurs dans le réseau :

Nous avons effectué une deuxième série de tests en utilisant deux routeurs et un nœud qui prendra en charge la capture et l'exécution des actions. Ces tests ont été effectués au sein de notre laboratoire LARI.

En premier lieu, nous avons testé une architecture à deux routeurs en série. Ensuite, nous avons utilisé une architecture avec deux routeurs qui fonctionnent en parallèle.

a- Architecture à deux routeurs en série :

La figure suivante montre les endroits où nous avons placé chacun de nos modules afin de pouvoir avoir une architecture en série.

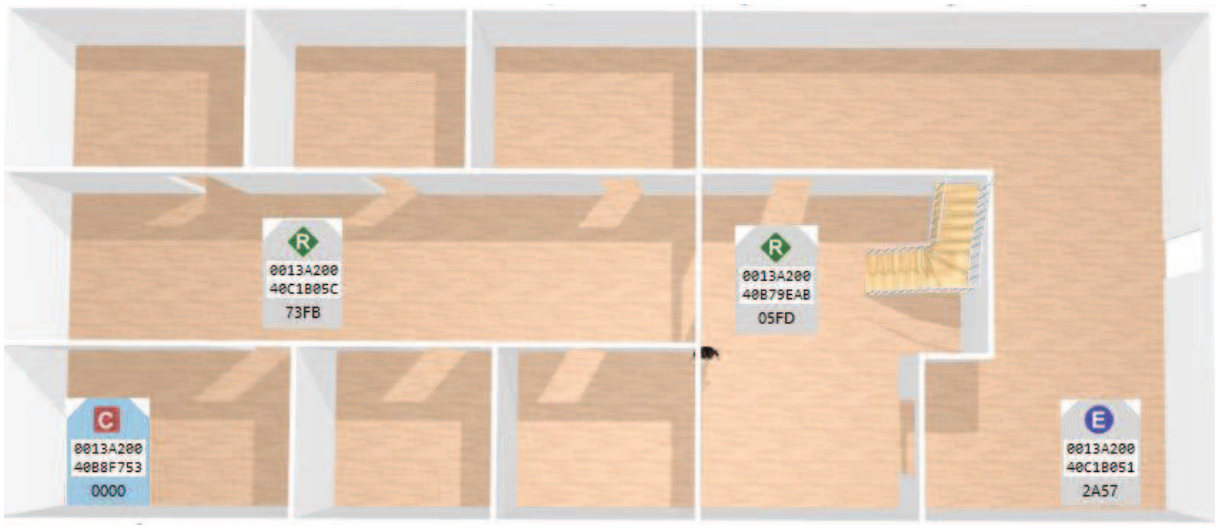

Figure IV.14 : Emplacement des modules dans le Laboratoire LARI

Nous avons ainsi utilisé le visionneur graphique du logiciel X-CTU afin de pouvoir visualiser graphiquement les connexions entre nos différents modules. Une vue de notre réseau est montrée dans la **figure IV.16**.

La **figure IV.15** suivante représente la table de routage de notre réseau. Elle est composée de cinq champs :

- **Role :** Ce champ indique la fonction du module XBee.
- **MAC :** Il donne l'adresse MAC du module sur 64 bits.
- **Network Address :** correspond à l'adresse réseau des équipements (sur 16 bits).
- **Last scan :** il représente le numéro du scan réseau.
- **Connections :** ce champ donne la table de routage de chaque dispositif dans le réseau en précisant la qualité de la liaison radio (LQI) et son statut (Status).

Role	MAC	Network Address	Last scan	Connections
Coordinator	0013A20040B8F753	0000	15	Show connections
Router	0013A20040B79EAB	05FD	15	Show connections
End device	0013A20040C1B051	1982	15	Show connections
Router	0013A20040C1B05C	73FB	15	Show connections

Figure IV.15 : Table de routage du réseau.

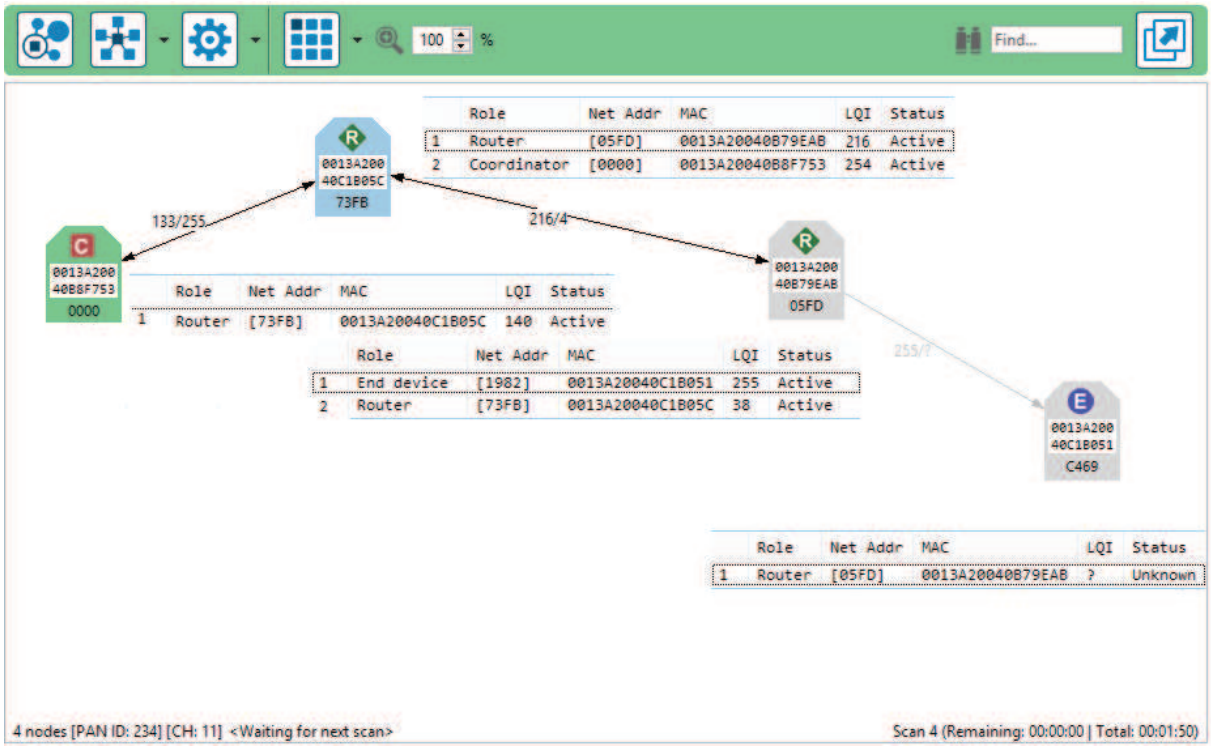

Figure IV.16 : Architecture de notre réseau avec deux routeurs en série.

Cette figure montre que le réseau forme une sorte de chaine. Plus exactement, le end device est rattaché à un routeur, ce routeur quant à lui est rattaché à un autre routeur et ce dernier est relié au coordinateur.

Les figures IV.17 et **IV.18** montrent que lorsqu'on débranche l'un des deux routeurs, celui-ci n'est plus accessible dans le réseau (sur le visionneur graphique il prend la couleur rouge).

Role	MAC	Network Address	Last scan	Connections
Coordinator	0013A20040B8F753	0000	15	*Show connections*
Router	0013A20040B79EAB	05FD	15	*Show connections*
End device	*0013A20040C1B051*	*A49A*	*13*	*Show connections*
Router	0013A20040C1B05C	73FB	15	*Show connections*

Figure IV.17 : Table de routage du réseau après débranchement du routeur.

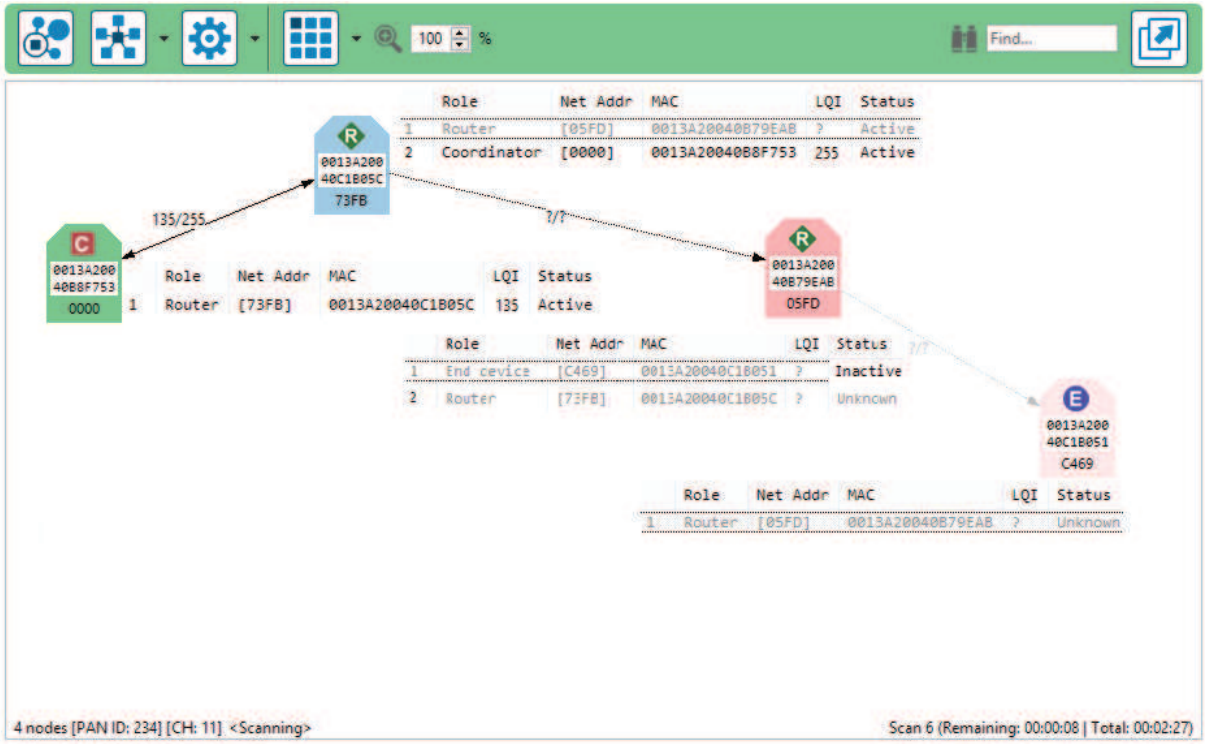

Figure IV.18 : Vue graphique du réseau après débranchement du nœud routeur.

Nous remarquons ainsi que le end device n'ayant pas la possibilité de contacter son nœud père, il essaye de se rattacher à l'autre routeur en vin, car il ne trouve aucun équipement à sa portée. Par conséquent, plus aucune communication n'est possible avec le end device qui est inaccessible dans le réseau.

b- Architecture à deux routeurs en parallèle :

Pour les besoins de ce test, nous avons placé deux routeurs au même endroit. C'est à dire que les deux routeurs doivent être à portée radio du end device.

Le but de ce test est d'observer le comportement du réseau lors du débranchement de l'un des deux routeurs. La **figure IV.19** montre la table de routage de notre réseau initial. La figure IV.20 quant à elle montre une vue de ce dernier.

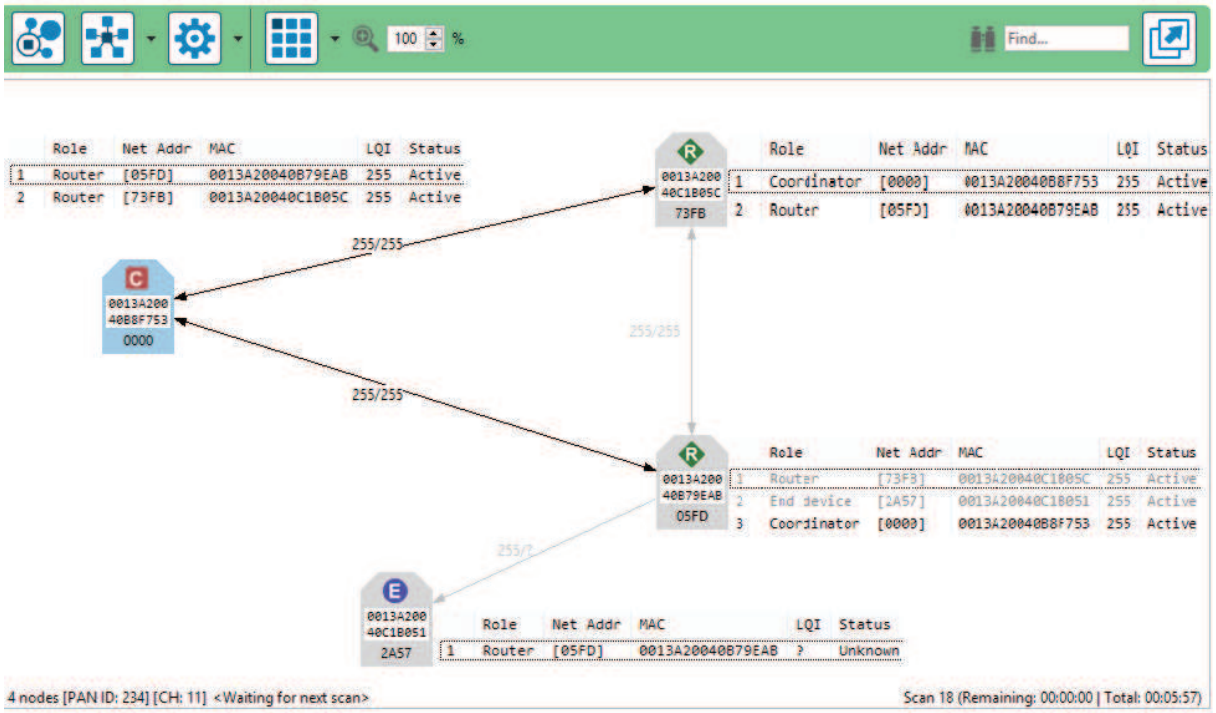

Figure IV.19 : Architecture du réseau avec deux routeurs en parallèle.

Lorsqu'on débranche le routeur auquel est relié notre end device, ce dernier rattache à l'autre routeur présent sur le réseau. Les figures suivantes montrent le comportement du réseau après le débranchement du routeur.

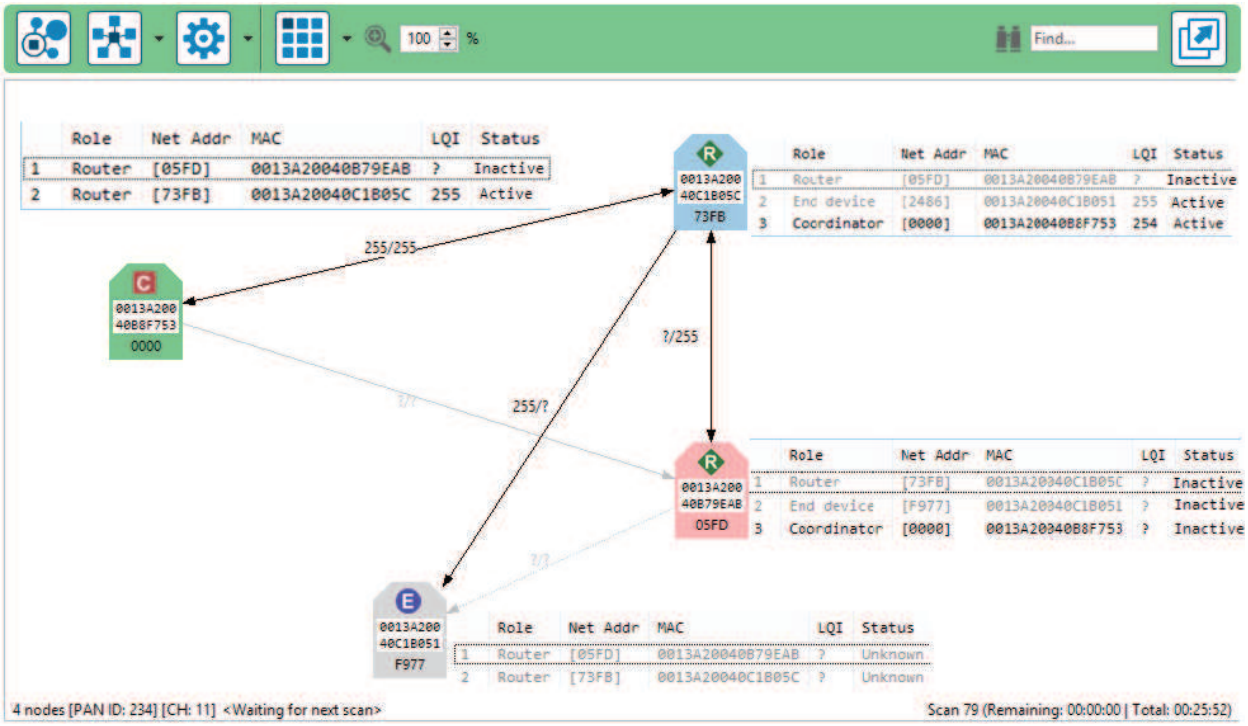

Figure IV.20 : Architecture du réseau après débranchement d'un routeur.

Nous voyons dans les figures ci-dessus que le routeur débranché n'est plus accessible dans le réseau et aucun lien de communication ne peut être établi avec ce dernier (il prend la couleur rouge sur le visionneur graphique). Après avoir perdu la connexion avec le premier routeur, le end device se rattache au second routeur et ce dernier prend le relai pour toutes les communications du end device à travers le réseau.

IV.5 Conclusion :

Au cours de ce de dernier chapitre, nous avons décrit l'étape de réalisation et de test de notre RCSF. En premier lieu, nous avons présenté les outils utilisés pour développer et configurer notre réseau dans le but d'effectuer des captures de données environnementales. Ensuite, nous sommes passés à la dernière étape qui consiste à effectuer des tests sur notre réseau.

La première série de tests a permis de montrer que grâce à la bibliothèque XBee, nous avons eu la possibilité de créer un réseau de capteur sans fil qui recueille des grandeurs analogiques et numériques. Quant à la seconde série effectuée à l'aide du logiciel X-Ctu, elle nous a permis de tester le fonctionnement de notre réseau sous de multiples topologies.

Conclusion générale et perspectives :

Ce projet nous a permis de manipuler une jeune technologie très prometteuse qui est le ZigBee. Cette technologie, facile à mettre en place grâce à des modules comme le XBee que nous avons utilisé se montre très adaptée à des petits systèmes embarqués ne nécessitant pas un haut débit de transfert, tels que les appareils de contrôle et d'acquisition. Ce qui rend cette norme encore plus adaptée à ce type de systèmes est sa faible consommation d'énergie et la petite taille de ces modules. Elle est aussi bien adaptée à une utilisation en milieu industriel grâce à sa portée et sa tolérance aux environnements bruités.

Pour la suite de ce travail, nous avons comme perspective d'intégrer notre réseau dans des solutions embarquées finies qui pourraient être utilisée par exemple pour la détection d'intrusion ou la gestion de la climatisation et/ou du chauffage. Une autre perspective pourrait être l'implémentation d'un protocole de routage pour une meilleure gestion de consommation d'énergie, en répartissant les rôle des end devices et des routeurs entre les nœuds du réseau. Cela en s'inspirant des protocoles existants tel que *LEACH* (Low Energy Adaptative Clustering Hierarchy).

[1] : Y.Younes, « Minimisation de l'énergie dans un réseau de capteurs », mémoire de magister, 2012, UMMTO.

[2] : D.Nouali, « Agrégation de données à la volée pour un réseau de capteurs sans fil(RCSF) », mémoire de master, 2012, UMMTO.

[3] M.Pamba Capo-Chichi, « Conception d'une architecture hiérarchique de réseaux de capteurs pour le stockage et la compression de données », thèse de Doctorat, LIFC, Université de Franche-Comté, 11 mars 2010.

[4] : S.Benamara, « L'amélioration des techniques de mise en veille dans le protocole LEACH », mémoire de master, 2011, UMMTO.

[5] : Z.Bouzidi A.Benameur, « Mise en place d'un réseau de capteur sans fil pour l'irrigationintélligente », mémoire de master 2012, Université Abou Bekr Belkaid (Tlemcem).

[6] : Norme IEEE 802.15.4 sur Wikipedia : http://fr.wikipedia.org/wiki/802.15.4.

[7] : P.Kadionik, « Étude et réalisation d'un capteur sans fil ZigBee », projet avancé, 2008 , l'ENSEIRB-IMS.

[8] : T.Val E.Campo A.Van Den Bossche, « Technologie ZigBee/802.15.4 », cours Editions T.I., Université de Toulouse.

[9] : M.Essamlali S.Hamdaoui, « Exposé sur ZigBee », prezi.com, 2013.

[10] : Manuel du module Xbee : http://ftp1.digi.com/support/documentation/90000976_W.pdf

[11] : K. Navya, Dr. M. B. R. Murthy, « A Zigbee Based Patient Health Monitoring System Journal of Engineering », Research and Applications www.ijera.com Vol. 3, Issue 5, Sep-Oct 2013, pp.483-486, Inde.

[12] : F.Leccese Z.Leonowicz, « Intelligent wireless street lighting system », Environment and Electrical Engineering (EEEIC), 2012 11th International Conference on 18-25 May 2012, pp.958 – 961, Venice.

[13] : Langage java sur Wikipedia : http://fr.wikipedia.org/wiki/Java

[14] : http://www.oracle.com

[15] : http://ftp1.digi.com.

[16] : www.digi.com/xctu.

Annexes

Présentation de la Bibliothèque XBee :

La Bibliothèque XBee est une API simple d'utilisation développée en Java qui permet d'interagir avec les modules radio XBee de Digi. Elle peut être utilisée pour créer tout type d'application Java pour la manipulation de ces modules.

L'API fournit toutes les méthodes nécessaires afin d'effectuer des tâches les plus courantes liées aux dispositifs XBee. Cette API peut être également utilisée pour créer des applications puissantes.

Dans cette annexe, nous présentons quelques méthodes de l'API XBee :

Méthode	Description
XBeeDevice("Port", "baud rate")	Instancier un objet XBee local.
RemoteXBeeDevice(XBeeLocal,@MAC)	Instancier un objet XBee distant.
readDeviceInfo()	Lire les informations du module.
Open()	Ouvrir la connexion avec le module
Close()	Fermer la connexion avec le module
Manipulation des paramètres régulièrement utilisés :	
get64BitAddress()	Obtenir l'adresse MAC du module.
getNodeID()	Obtenir l'identificateur du module.
getFirmwareVersion()	Obtenir la version du Firmware.
getXBeeProtocol()	Obtenir le protocole du module.
getOperatingMode()	Obtenir le mode d'opération.
Manipulation des autres paramètres :	
getParameter(String)	Spécifier la commande AT(en String) pour obtenir sa valeur. La

Méthode	Description
	valeur est retournée dans un tableau de Bytes.
setParameter(String, byte[])	Spécifier la commande AT(en String) et la valeur à lui assigner en un tableau de bytes.
executeCommand(String)	Spécifier la commande AT à exécuter sur le module.
Appliquer les changements de configuration :	
enableApplyConfigurationChanges(boolean)	Spécifiez si oui ou non les changements sur les réglages et paramètres seront appliquées lorsqu'il est réglé.
isApplyConfigurationChangesEnabled()	Retourne si oui ou non le dispositif de XBee est configuré pour appliquer les modifications de paramètres quand ils sont fixés.
applyChanges()	Applique les changements sur les paramètres qui étaient déjà définis, mais sont en attente d'être appliqué.
writeChanges()	écrire (enregistrer) les modifications de paramètres dans la mémoire de l'appareil XBee afin qu'ils persistent à travers les réinitialisations ultérieures
reset()	Réinitialiser le module XBee.

Méthode	Description
Découverte de réseau :	
startDiscoveryProcess()	Démarre le processus de découverte, les périphériques distant trouvés sont enregistrés dans l'objet XBeeNetwork.
discoverDevice(String)	Cherche le module ayant l'identificateur égal au String donné en paramètre.
getDevices()	Retourne la liste des modules distants.
getDevices(String)	Retourne la liste des modules ayant l'identificateur égal au string donné en paramètre.
getDevice(String)	Retourne le module ayant l'identificateur égal au string donné en paramètre.
getDevice(XBee16BitAddress)	Retourne le module distant ayant l'adresse de réseau donnée en paramètre.
getDevice(XBee64BitAddress)	Retourne le module distant ayant l'adresse MAC donnée en paramètre.
addRemoteDevice(RemoteXBeeDevice)	Ajouter au réseau un module manuellement.
removeRemoteDevice(RemoteXBeeDevice)	Supprimer un module distant de la liste des modules du réseau.
clearDeviceList()	Supprimer la liste des modules distants du réseau.
Communication avec les modules XBee :	
sendData(XBee64BitAddress, XBee16Bit	Envoie synchrone de données à un

Méthode	Description
Address, byte[])	module spécifié
sendDataAsync(XBee64BitAddress,XBee16BitAddress, byte[])	Envoie asynchrone de données à un module spécifié
sendBroacastData(byte[])	Envoie à tous les modules du réseau
readData(int)	Réception de données pendant le temps spécifié par int donné en paramètre (si aucun entier n'est donné la valeur par défaut est utilisée).
readDataFrom(RemoteXBeeDevice, int)	Réception de données venant d'un module spécifique.
Manipulation des lignes IO :	
setIOConfiguration(IOLine, IOMode)	Tous les objets de XBee comprennent cette méthode où on peut spécifier la ligne IO à configurer et la fonction désirée.
getIOConfiguration(IOLine)	Obtenir la configuration d'une ligne IO.
setDIOValue(IOLine, IOValue)	Dans le cas où votre ligne IO est configurée comme sortie numérique, vous serez en mesure de définir son état (haut/ bas) d'une manière facile.

Les commandes AT :

Dans ce qui suit nous allons présenter les commandes AT pour la manipulation des modules XBee :

Commande AT	description	Format du paramètre	Valeur par défaut
WR	Sauvegarde les paramètres dans la mémoire non volatile. Il faut impérativement attendre la réponse "OK" du module avant de lui envoyer une nouvelle commande.	--	--
RE	Restaure les paramètres par défaut du module. Cette commande ne réinitialise pas le champ ID	--	--
NR	Réinitialise les paramètres de la couche réseau sur un ou plusieurs modules au sein du réseau. Toutes les informations de routages sont perdues. NR = 0 : réinitialise le module qui a reçu la commande NR = 1 : réinitialise tous les modules contenus dans le PAN	0-1	--
DH	Modifie ou lit les 32 bits MSB de l'adressage destinataire.	0 – 0xFFFFFFFF	0
DL	Modifie ou lit les 32 bits LSB de l'adressage destinataire.	0 – 0xFFFFFFFF	0xFFFF
MY	Lit les 16 bit de	0 – 0xFFFE	0xFFFE

	l'adressage réseau du module.		
MP	Lit les 16 bit de l'adressage parent du module.	0 – 0xFFFE	0xFFFE
SH	Lit les 32 bits MSB du n° de série du module	0 – 0xFFFFFFFF	Défini en usine
SL	Lit les 32 bits LSB du n° de série du module.	0 – 0xFFFFFFFF	Défini en usine
CH	Modifie ou lit le canal utilisé dans la bande 2,4 GHz.	0 - 0x1A 0 (0x0B) 1 (0x0C) 2 (0x0D) 3 (0x0E) 4 (0x0F) 5 (0x10) 6 (0x11) 7 (0x12) 8 (0x13) 9 (0x14) 10 (0x15) 11 (0x16) 12 (0x17) 13 (0x18) 14 (0x19) 15 (0x1A)	0
ID	Modifie ou lit l'adresse du Pan ID. Le changement de PAN est effectué lorsque le module est réinitialisé sur une commande FR, NR ou réalimenter.	0 - 0xFFFF	0x0234
NT	Défini ou lit le temps qu'acceptera le module pour découvrir d'autre noeud du réseau lorsque ND est activé	0 - 0xFC [x 100 msec]	0x3C
ND	Cherche et donne les modules trouvés. Pour chacun on obtient: MY + SH + SL + DB + NI. La commande se termine au bout de	optional 20-Byte NI or MY value	--

	2,5 secondes et le module renvoie un "CR"(retour chariot). On peut faire suivre la commande d'un paramètre constitué des 20 caractères du NI d'un module. Dans ce cas on obtient en réponse uniquement les paramètres de ce module.		
PL	Modifie ou lit la puissance de sortie du module.	0 - 4 0 = -10 / 10 dBm 1 = -6 / 12 dBm 2 = -4 / 14 dBm 3 = -2 / 16 dBm 4 = 0 / 18 dBm	4
BD	Vitesse de transmission	0 - 7 0 = 1200 bps 1 = 2400 bps 2 = 4800 bps 3 = 9600 bps 4 = 19200 bps 5 = 38400 bps 6 = 57600 bps 7 = 115200 bps	3
D7	Configuration contrôle de flux	0 – 1 0 = CTS non permit 1 = RTS permit	0
D6	Configuration contrôle de flux	0 – 1 0 = RTS non permit 1 = RTS permit	1
D5	Configure la LED d'association. La LED clignote 1*/sec quand le module est alimenté. La LED clignote 2*/sec quand le module rejoint un réseau	0 – 1 0 = Fonction non enclenchée 1 = Fonction enclenchée	1
VR	Lit la version firmware du module	0 – 0xFFFF	Défini en usine
CT	Modifie ou lit le Time Out qui fait repasser le module en mode IDLE »repos » si aucune commande	2 - 0x028F [x 100 ms]	0x64

	AT ne parvient.		
CN	Dit explicitement au module de quitter le mode commande AT.	--	--
GT	Modifie ou lit le temps de garde entre l'envoie de deux caractères. Cette période est utilisé pour éviter d'entrer par inadvertance dans le mode commande AT.	1 - 0x0CE4 [x 1 ms] (max of 3.3 sec décimale)	0x3E8
CC	Modifie ou lit le caractère ASCII utilisé pour passer en mode commande.	0 - 0xFF	0x2B ('+ ')