

---

# ASSISTANCE GLOBALE POUR AIDE AU PARKING

---

**Rapport de projet de fin d'études**  
Cinquième année du département  
Informatique, Microélectronique et Automatique

**Auteurs :**

Mathieu GERIER  
Céline LY

**Année :**

2014/2015

**Nom de la matière :**

Projet de fin d'études

**À l'attention de :**

M. Alexandre BOÉ  
M. Thomas VANTROYS

# Remerciements

---

Nous tenons tout d'abord à remercier toute l'équipe pédagogique de Polytech Lille et les responsables de la formation Informatique, Microélectronique et Automatique de nous avoir enseigné les bases pour réaliser ce projet dans de bonnes conditions.

Nous tenons aussi à remercier et à témoigner toute notre reconnaissance à Monsieur Alexandre BOÉ et Monsieur Thomas VANTROYS, nos responsables de projet, pour leur soutien permanent et leur disponibilité tout au long du projet.

Nous souhaitons également remercier Monsieur Xavier REDON et Monsieur Thierry FLAMEN pour nous avoir aidés sur certains aspects du projet lorsque nous en avons besoin.

# Sommaire

---

<b>Remerciements .....</b>	<b>1</b>
<b>Sommaire.....</b>	<b>2</b>
<b>Introduction .....</b>	<b>5</b>
<b>I. Présentation du projet .....</b>	<b>6</b>
1. Présentation générale .....	6
a. Contexte .....	6
b. Présentation du projet.....	6
2. Matériel et outils nécessaires .....	8
a. Matériel utilisé .....	8
b. Outils nécessaires.....	9
3. Étapes du projet .....	10
<b>II. Avancement du projet.....</b>	<b>12</b>
1. Partie réseau de capteurs .....	12
a. Choix des solutions liées aux capteurs .....	12
b. Choix des solutions liées au réseau.....	14
c. Système d'exploitation .....	15
d. Travail réalisé .....	16
2. Partie commune : gestion de la communication entre le module capteur et le serveur web 19	
a. Communication entre le système et le concentrateur général .....	20
b. Communication entre le système et le serveur web .....	20
3. Partie applicative .....	21
a. Choix retenus.....	22
b. Travail réalisé .....	23
<b>III. Améliorations possibles .....</b>	<b>28</b>
1. Partie réseau de capteurs .....	28
2. Partie applicative .....	29
<b>IV. Bilan du projet .....</b>	<b>30</b>

1. Récapitulatif .....	30
2. Difficultés rencontrées .....	30
3. Bilan personnel.....	31
<b>Conclusion.....</b>	<b>33</b>
<b>Annexes .....</b>	<b>34</b>



# Introduction

---

Au cours de notre cinquième année à Polytech Lille, spécialité Informatique, Microélectronique et Automatique, nous avons effectué un projet de fin d'études ayant pour objectifs principaux, l'acquisition de nouvelles connaissances et la mise en pratique de celles étudiées durant notre formation afin de les transformer en compétences. L'un des intérêts principaux de ce projet était la combinaison de deux domaines : les systèmes embarqués et le développement web.

Les réseaux de capteurs sans fil font de plus en plus partie de la vie quotidienne. En effet, on les retrouve dans les domaines du militaire, de la sécurité civile ou encore dans le transport. Ils sont de plus en plus présents afin de nous faciliter la vie et d'améliorer notre confort. Le but du projet est d'utiliser une telle technologie afin d'aider les automobilistes dans la recherche de places de parking.

Pour présenter au mieux ce projet, nous évoquerons, les objectifs fixés par le cahier des charges. Ensuite, nous exposerons le contexte du projet. Enfin, nous ferons un bilan de la situation actuelle et enfin finirons par évoquer les perspectives futures.

# I. Présentation du projet

---

## 1. Présentation générale

### *a. Contexte*

De nos jours, de par l'augmentation du nombre de véhicules circulant, il est de plus en plus stressant de se rendre à son lieu de travail ou dans des grandes surfaces. Les embouteillages, les accidents sont des facteurs de stress pour les automobilistes. De plus, lors de l'arrivée de ces derniers dans les parkings, ils perdent de précieuses minutes à chercher des places pour se garer. Ceci peut augmenter le stress, la pollution et l'énerverment. C'est dans ce contexte que nous proposons une solution aux automobilistes afin de les aider à trouver des places de parking plus aisément et dans de bonnes conditions.

### *b. Présentation du projet*

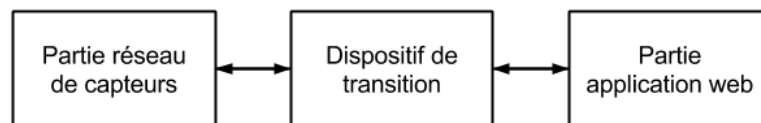
L'objectif de ce projet était de concevoir un prototype répondant à la problématique. Afin d'aider au mieux les automobilistes, nous avons pensé à utiliser une application qui affiche des informations comme par exemple, le nombre de places disponibles, le nombre de places totales ou encore les types de places qu'il y a dans un parking.

Un réseau de capteurs est nécessaire pour connaître des données telles que l'emplacement, l'état de chaque place de parking. Un capteur est un dispositif capable de mesurer une grandeur physique. Dans le cadre du projet, il sert à détecter la présence d'un obstacle (une voiture en l'occurrence). Le réseau de capteurs permet d'établir la communication entre les différents capteurs afin de remonter les informations à l'application (cf. point précédent) pour qu'elle puisse se mettre à jour et interagir avec les automobilistes.

Pour que l'application récupère les données du parking, nous utiliserons des balises permettant de récupérer les informations des places grâce aux capteurs : ce sont les concentrateurs. Il y aura un concentrateur général, où toutes les données seront acheminées à partir des autres concentrateurs, que l'on appellera concentrateurs relais, et des capteurs ; le concentrateur général permettra de communiquer les données à l'application grâce à Internet. Ces dernières seront stockées dans une base de données qui permet d'enregistrer des données de façon efficace et organisée.

Ainsi, le projet s'articule autour de trois points :

- Le réseau de capteurs permettant de récupérer les informations comme la disponibilité de certaines places, le type de place ou encore la morphologie.
- L'application permettant d'interagir avec l'automobiliste et d'afficher les informations indispensables à l'optimisation de son temps pour le parcage.
- L'association des deux parties précédentes afin de mettre à jour l'application grâce au réseau de capteurs. Il s'agit de la communication entre l'application et le réseau de capteurs.



*Figure 1 : Vue globale des parties du projet*



## 2. Matériel et outils nécessaires

### a. Matériel utilisé

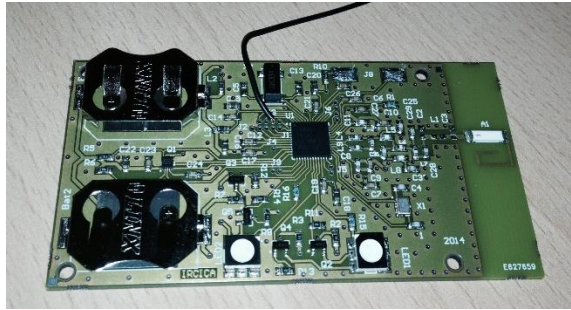


Figure 2 : Carte CC430

Afin de réaliser au mieux notre projet, nous avons choisi d'utiliser plusieurs cartes CC430 réalisées par Monsieur BOÉ et Monsieur VANTROYS. Il s'agit de cartes électroniques comprenant un module RF (*Radio Frequency*) permettant d'envoyer et de recevoir des informations en communication sans fil à une certaine fréquence. Elles disposent également d'un microcontrôleur dont l'utilité, dans notre projet, est de connaître la disponibilité d'une place de parking grâce à un capteur connecté à ce dernier et de gérer les échanges de données entre les différents modules (concentrateur ou capteur). Une carte serait présente sur chaque place de parking avec un capteur, de même une carte fait partie de chaque concentrateur afin d'échanger les informations grâce à la communication sans fil. On désignera désormais un module capteur comme l'association d'une carte et d'un capteur.

L'utilisation de ces cartes n'était pas la plus adaptée pour le projet mais il était intéressant de les utiliser afin de concevoir un prototype. En effet, elle est plus performante que ce dont nous avons réellement besoin (mémoire, puissance de calcul, etc.).

Par ailleurs, afin d'assurer une bonne communication sans fil, nous avons décidé d'utiliser la bande de fréquence 868 MHz. Grâce à cette fréquence, la communication peut être réalisée à une plus grande distance par rapport à une fréquence de 2,4 GHz et ce, pour une même taille d'antenne.

Nous avons également besoin du MSP430 LaunchPad. Il s'agit d'une carte comprenant un programmeur permettant d'envoyer un programme sur une carte CC430 à partir d'un ordinateur. Le programme permet, dans ce cas précis, de récupérer l'information du capteur et d'envoyer et de recevoir des informations venant du réseau de capteurs.

Pour notre prototype, nous avons utilisé un interrupteur pour chaque carte où il y a besoin d'un capteur. L'interrupteur permet de remplacer le capteur. En effet, ils ont la même fonction, c'est-à-dire, que si un obstacle apparaît devant le capteur, ce dernier changera d'état. Il en va de même pour l'interrupteur, où l'on va pouvoir simuler un obstacle en changeant de valeur l'interrupteur.

### *b. Outils nécessaires*

Certains outils ont été nécessaires quant à la réalisation du projet. Tout d'abord, un choix dans l'utilisation d'un serveur web a été fait. Il s'agit d'un serveur informatique utilisé pour publier des sites web sur Internet. L'objectif de l'application est de transmettre des informations à tous les utilisateurs qui ont accès à Internet. Cet accès est indispensable car l'état des places peut changer à tout moment. L'application, stockant ces informations dans une base de données, doit être capable de les récupérer à tout moment pour en informer l'utilisateur afin qu'elle puisse être à jour. Il existe des hébergeurs de sites web gratuits. Cependant, ils présentent quelques contraintes comme l'utilisation des langages PHP et MySQL que nous évoquerons ultérieurement.

Afin de réaliser une page web, nous avons besoin de certains outils comme les langages HTML et CSS. Ces derniers sont des langages de balisage permettant de structurer une page web. PHP est un langage de programmation utilisé pour le développement de pages web ; c'est celui qui est le plus utilisé dans ce domaine. Il est gratuit et ne nécessite aucune licence d'utilisation. Il existe un nombre incommensurable de bibliothèques et de tutoriels.

Pour que l'application fonctionne correctement, le site web doit stocker les informations relatives aux parkings renseignés. Pour cela nous utilisons une base de données. Il s'agit d'un outil capable de stocker et de récupérer des informations comme défini précédemment. Le réseau de capteurs renseignera la base de données permettant à l'application d'aller chercher les informations dans cette dernière et d'afficher celles souhaitées par l'utilisateur. Nous utiliserons le langage MySQL qui est un langage de programmation permettant la gestion de base de données (ajout, mise à jour ou suppression d'informations). Ce langage présente quelques avantages : il est gratuit et présente beaucoup de fonctionnalités. Il est également très répandu et il est très facile de rechercher des informations en cas de problème.

### **3. Étapes du projet**

Afin de mener à bien le projet, différents objectifs ont été posés :

- Compréhension du sujet de projet : première appréhension du sujet avec des échanges entre nos tuteurs et nous afin de réellement connaître les attentes de ce projet ;
- Étude des solutions à choisir : établissement d'un cahier des charges précis, et étude théorique des solutions qui pourraient être appliquées dans le cas où le prototype est fonctionnel ;
- Prise en main du matériel et familiarisation avec les outils choisis : familiarisation avec le matériel et ses fonctionnalités ainsi que découverte et approfondissement des connaissances et des possibilités des outils choisis ;
- Premières phases de tests pour le réseau de capteurs : réalisation de premiers tests basiques pour s'approprier le matériel (carte CC430 et MSP430 LaunchPad) et prise en main des fonctionnalités nécessaires au projet ;
- Premières phases de tests pour la partie application : conception d'une page web assez basique avec base de données et affichage des données souhaitées à l'utilisateur ;
- Réalisation d'un prototype simple : prototype répondant au cahier des charges et ne prenant pas forcément en compte toutes les erreurs et problèmes qui peuvent survenir ;

- Amélioration de ce prototype : prise en compte les erreurs et problèmes rencontrés lors du premier prototypage en situation réelle et d'apporter de nouvelles fonctionnalités.

## II. Avancement du projet

---

### 1. Partie réseau de capteurs

Cette partie du projet permet de récupérer les informations des capteurs (disponibilité des places). Dans cette partie, nous verrons, dans un premier temps, les choix technologiques que nous avons retenus, tels que les solutions liées aux capteurs. Nous évoquerons, dans un second temps, la topologie du réseau retenue. Ensuite, nous argumenterons sur le fait d'utiliser ou non un système d'exploitation. Enfin, nous présenterons les tests réalisés, ainsi que les résultats obtenus.

#### *a. Choix des solutions liées aux capteurs*

Il est à noter qu'avant de concevoir le réseau de capteurs, nous avons dû passer par une première étape d'étude du sujet. En réalisant cette étude, nous sommes parvenus à des choix technologiques qui sont les solutions que l'on aurait appliquées dans une seconde phase de prototypage, et qui auraient été plus adaptées aux besoins. En effet, pour notre projet, la première phase de prototypage consiste à faire fonctionner le réseau de capteurs avec le matériel déjà à disposition. Ce matériel est, dans notre cas, surdimensionné, mais cela nous a permis de travailler concrètement sur des solutions qui ne sont pas les plus adaptées aux besoins du projet.

Une première phase a consisté à réfléchir sur la solution liée aux capteurs que nous allions utiliser pour le projet. Nous nous sommes demandé s'il était possible de récupérer les informations de plusieurs places de parking grâce à un seul capteur. Cette solution aurait permis de ne pas avoir de contraintes de consommation d'énergie car le capteur aurait été placé sur un lampadaire et il aurait été alimenté par ce dernier. Cependant, dans un souci de rendre notre

projet générique, c'est-à-dire adaptable à n'importe quel parking, nous avons conclu que cette solution ne pouvait pas être utilisée. En effet, chaque parking est différent : certains sont intérieurs, d'autres extérieurs et chacun a un agencement "unique" (arbres, trottoirs, etc.). Nous avons donc retenu la solution d'un capteur par place, ce qui signifie que le module capteur permet de savoir si une voiture est en stationnement ou non sur une place de parking uniquement. Cette information sera alors par la suite remontée à l'application. Les inconvénients de cette solution sont que le module capteur devra être alimenté par une batterie ou une pile. Il faudra ainsi limiter la consommation énergétique le plus possible afin d'éviter d'avoir trop souvent recours à la maintenance pour remplacer ou recharger un module capteur.

Nous avons également réfléchi sur les endroits où l'on pouvait placer les modules capteurs afin d'éviter des erreurs potentielles comme des erreurs de détection d'obstacles ou de mauvaise communication sans fil. Ils auraient pu être placés en hauteur, sous ou derrière les places de parking, etc. Nous avons pensé à placer les capteurs au milieu des places de parking car il est plus facile de repérer une voiture en cas de pluie ou de brouillard. Ce choix pourra être changé par la suite en cas de tests non concluants.

En ce qui concerne le placement des concentrateurs qui devront se charger de récupérer les données des modules capteurs, nous avons établi l'hypothèse que leur position devra être connue à l'avance. Chaque concentrateur sera placé :

- De telle sorte à garantir, du mieux possible, l'homogénéité de l'émission et de la réception des données. La consommation énergétique sera plus équitable pour chaque module capteur car plus un module capteur est loin d'un concentrateur, plus il va dépenser de l'énergie afin de communiquer entre eux ;

- Sur un lampadaire afin d'être alimenté à une source d'énergie "infinie", c'est-à-dire sans problème de décharge. Il pourra ainsi envoyer et recevoir des données sans tomber en panne d'énergie.

Une étude a été consacrée à la recherche de capteurs que l'on aurait pu utiliser pour le projet. Deux solutions ont été retenues. La première est l'utilisation d'un capteur à ultrasons : son coût est faible mais présente un inconvénient non négligeable, à savoir son incapacité à détecter des obstacles en cas d'obstruction au niveau de l'antenne. La deuxième solution est d'utiliser un capteur électromagnétique : il est plus fiable que la solution précédente, mais il est plus cher aussi.

### b. Choix des solutions liées au réseau

Principe de la topologie choisie : *Cluster-Tree*

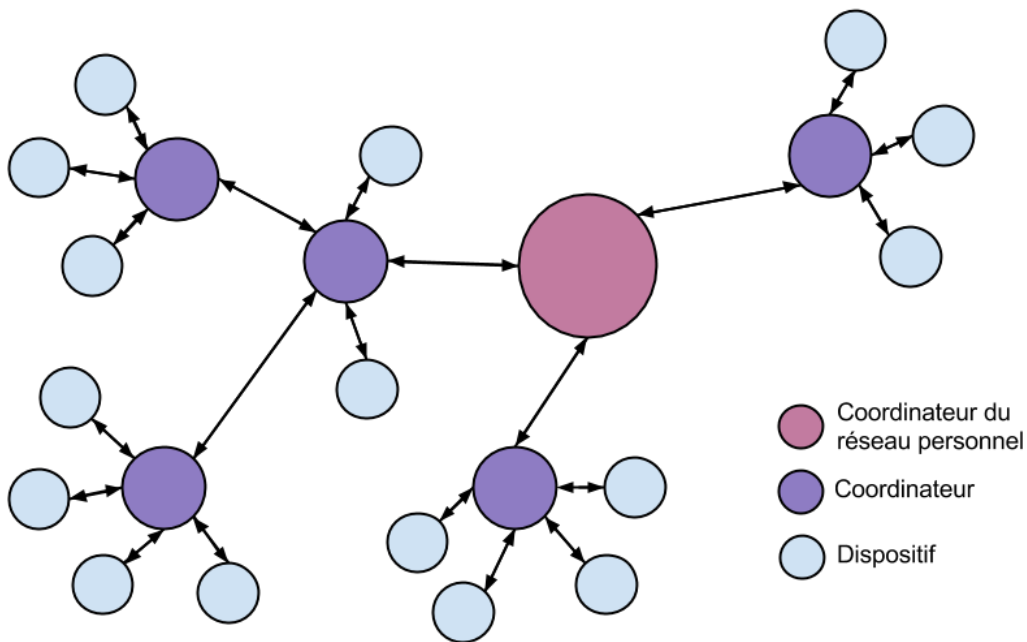


Figure 3 : Topologie Cluster-Tree adoptée pour le projet

Ce schéma montre que le réseau de capteurs est la partie qui va permettre de remonter les informations liées aux places de parking disponibles. Toute la communication réalisée est sans fil. Suivant la taille du parking, la communication des données ne peut être réalisée avec un unique concentrateur, qui regrouperait toutes les données. Ces informations doivent tout d'abord être remontées à un premier concentrateur, qui s'occupe de relayer les informations liées à une zone spécifique d'un parking. Une fois que ces informations sont récupérées par ce

premier concentrateur, si la distance vers le concentrateur général le permet, les informations sont communiquées à ce concentrateur général, sinon elles passent par un autre concentrateur relai jusqu'à atteindre le concentrateur général. Ce principe suit la topologie *Cluster-Tree*, que nous avons choisie. Cette topologie possède aussi l'avantage d'être statique : en effet, les chemins de communication sont prédéfinis. Ceci permet une implémentation plus facile, et par la même occasion une diminution de la consommation d'énergie. En effet, s'il n'y a pas de recherche dynamique à réaliser avant d'envoyer des informations, les microcontrôleurs associés aux capteurs auront moins de calculs à réaliser, réduisant leur temps de travail et donc leur consommation d'énergie. Plusieurs capteurs peuvent être liés à un seul concentrateur. Il en va de même pour la liaison entre plusieurs concentrateurs et le concentrateur général. Cette décision a été prise dans l'optique d'un premier prototype. L'idée serait cependant de pouvoir réaliser une assignation dynamique entre tous les modules afin de rendre générique le projet.

### *c. Système d'exploitation*

Nous avons pondéré la question de l'utilité d'un système d'exploitation dans notre cas. En effet, un système d'exploitation permet d'assurer la liaison entre les ressources matérielles, l'utilisateur et les applications. Son implémentation sur un microcontrôleur confère un accès plus pratique pour la modification et donc l'écriture de programmes, une certaine robustesse au niveau du traitement des données, une possibilité de gestion en temps réel, etc. Cependant, dans notre cas, nous pouvons effectuer ces opérations sans système d'exploitation même si la mise en œuvre en sera plus difficile. Par ailleurs, l'utilisation d'un système d'exploitation nécessite une certaine performance du microcontrôleur et nous limite donc dans le choix du matériel à utiliser. Ainsi, nous n'avons pas utilisé de système d'exploitation afin de choisir le matériel le plus adapté lors d'étape ultérieures dans le projet.



#### *d. Travail réalisé*

Une première partie consistait à la familiarisation du LaunchPad et des cartes CC430. Lors de cette phase, nous avons effectué des tests basiques sur le LaunchPad, tels qu'allumer une LED par appui sur bouton poussoir ou la faire clignoter à une certaine fréquence. Pour cela, nous devons programmer le microcontrôleur implanté sur le LaunchPad. Pour qu'il fonctionne d'une certaine manière, il est indispensable de paramétrer des registres. Ainsi, cette première approche a permis de découvrir les registres de base qui assignent aux ports leur type (entrée ou sortie) et leur état.

Une fois ces tests réalisés, la programmation sur carte CC430, à l'aide du LaunchPad, a commencé. Afin d'effectuer des tests, comme par exemple, connaître l'état d'un interrupteur, nous avons soudé un fil sur les cartes CC430 afin de l'y connecter. Ce dernier permet de simuler l'état des capteurs de détection d'obstacle. En effet, lorsque l'interrupteur est dans une certaine position, il simule que le capteur détecte un obstacle (la place est occupée) et lorsqu'il est dans l'autre position, le capteur détecte qu'il n'y a pas d'obstacle (la place est libre).

Cette deuxième phase de tests a donc consisté en la programmation des cartes CC430. De nouveaux registres ont été paramétrés : il a fallu paramétrer les registres d'interruption, à savoir l'autorisation d'interruption et le type d'interruption (front descendant).

Ces tests ont ainsi permis de découvrir les registres de bases d'entrée et de sortie, ainsi que les registres d'interruption.

Dans un second temps, il a fallu appréhender les registres de la communication RF et ceux de la communication série. Une première série de tests a été basée sur des morceaux de codes fournis par Monsieur VANTROYS afin de mieux comprendre le fonctionnement de la communication RF. Après la résolution de problèmes rencontrés suite à la reprise du code, une communication RF a finalement pu être réalisée. Celle-ci n'utilise cependant pas les fonctionnalités LPM (*Low-Power Mode*), car cela facilite l'implémentation de la communication

RF. La communication RF réalisée permet aux modules capteurs de communiquer avec un concentrateur ; nous avons réalisé une architecture se présentant sous la forme suivante :

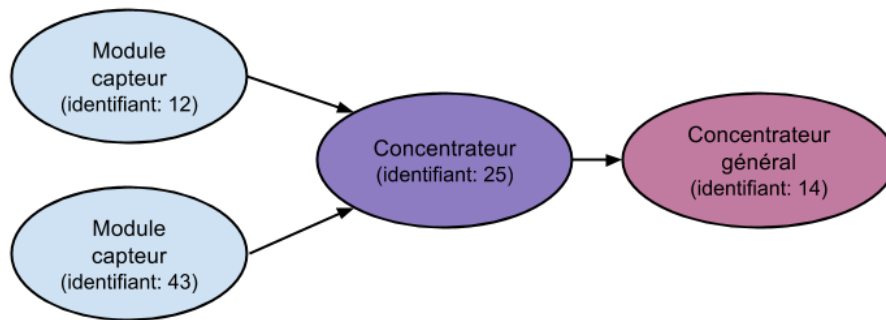


Figure 4 : Exemple de communication RF

Dans l'exemple ci-dessus, les modules capteurs 12 et 43 transmettent les informations (état du capteur) au concentrateur 25 qui va envoyer les informations au concentrateur 14 (le concentrateur général). Ce dernier permet d'envoyer les informations au niveau de la partie applicative. Nous évoquerons alors les moyens mis en œuvre pour que l'envoi de données puisse avoir lieu.

Afin d'échanger correctement les données au niveau de la communication RF, un protocole de communication basique a été utilisé. Il existe trois types de trame :

- une trame de communication entre un module capteur et un concentrateur qui permet de transmettre l'état du capteur associé au module capteur, suivant le modèle : source / destination / état du capteur ;

- une trame de communication entre concentrateurs permettant de remonter vers la base de données, l'information précédemment récupérée, suivant le modèle: source / destination / identifiant du capteur / état du capteur ;

- une trame de communication entre chaque dispositif permettant d'accuser réception de l'information, où le seul octet envoyé est l'identifiant du destinataire de l'accusé de réception. Celle-ci permet d'assurer la bonne transmission des données. Elle a donc été programmée pour chaque émission de données. Si l'accusé de réception ne parvient pas au module capteur, celui-ci renverra alors la mise à jour de son état. Par ailleurs, afin d'éviter des problèmes de collision,

un délai a été rajouté suivant l'identifiant du module : celui-ci permet donc en cas d'émission simultanée avec un second module de renouveler l'envoi de données à un intervalle différent.

Remarque : chaque donnée (i.e. source, destination, identifiant et état) est transférée sur un octet.

Par ailleurs, chaque trame commence par un octet indiquant la taille des données et se termine par deux octets : un octet de RSSI (*Received Signal Strength Indicator*), qui permet de connaître la puissance du signal reçu et un octet de LQI (*Link Quality Indicator*), qui permet d'identifier la qualité et l'exactitude du signal. Ces derniers octets ne sont cependant pas utilisés ici, et sont rajoutés automatiquement lors de l'utilisation du module RF des cartes CC430.

Ce protocole de communication permet donc de récupérer les changements d'état des capteurs et les remonter du capteur au concentrateur général.

Afin de réaliser la communication statique, chaque dispositif doit posséder des caractéristiques afin de bien communiquer. Pour ce faire, lors de la compilation, des paramètres sont assignés, comme l'explique le tableau suivant :

Dispositif	Paramètres / Commande	Explication
Module capteur	- SRC : identifiant du capteur - DEST : identifiant du concentrateur auquel le capteur est associé  make tx SRC=n1 DEST=n2 (avec n1 et n2 des entiers)	Permet d'envoyer l'information d'un module capteur source à un concentrateur destinataire
Concentrateur	- ID : identifiant du concentrateur	Permet de transférer l'information d'un concentrateur source à un

	<ul style="list-style-type: none"> <li>- NEXT : identifiant du concentrateur "suivant"</li> <li>- PARK : identifiant du parking associé au concentrateur</li> </ul> <p>make rx ID=n1 NEXT=n2 PARK=n3 <i>(avec n1, n2 et n3 des entiers)</i></p>	<p>concentrateur destinataire et de connaître l'identifiant du parking associé au concentrateur (source)</p>
Concentrateur général	<ul style="list-style-type: none"> <li>- ID : identifiant du concentrateur</li> <li>- NEXT : identifiant du concentrateur "suivant" initialisé à zéro, car le concentrateur général ne remonte pas d'informations à un autre concentrateur</li> <li>- PARK : identifiant du parking associé au concentrateur</li> </ul> <p>make rx ID=n1 NEXT=0 PARK=n2 <i>(avec n1 et n2 des entiers)</i></p>	<p>Permet de connaître l'identifiant du concentrateur général et l'identifiant du parking auquel il est associé</p>

## 2. Partie commune : gestion de la communication entre le module capteur et le serveur web

Cette partie concerne la communication entre le réseau de capteurs et l'application web. Les informations, recueillies par le réseau de capteurs, doivent être stockées dans la base de données située sur le serveur web. Il est indispensable d'avoir une connexion internet pour réaliser cette communication.

Le concentrateur général n'ayant pas de connexion à internet, nous devons passer par un système ayant une telle connexion. Pour cela, nous avons utilisé un ordinateur pour notre prototype. Par ailleurs, il est possible d'utiliser des systèmes embarqués qui sont plus adaptés à la problématique.

L'architecture est donc représentée sous cette forme :

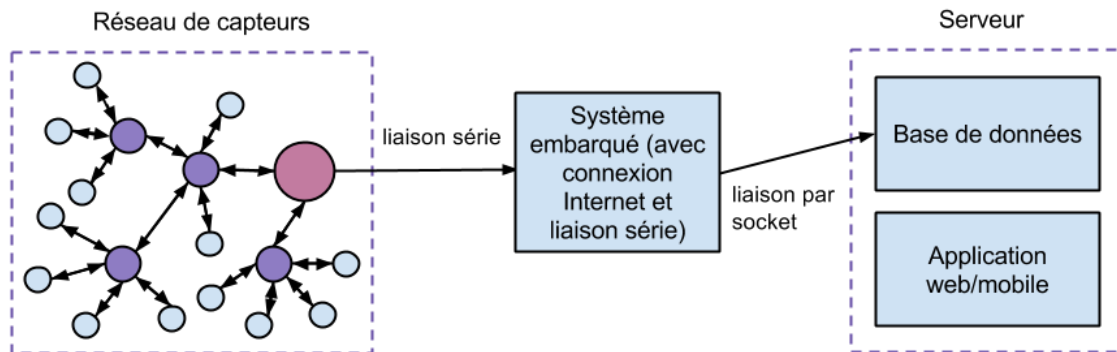


Figure 5 : Architecture principe

Le système connecté doit communiquer avec le concentrateur général ainsi qu'avec le serveur web.

#### *a. Communication entre le système et le concentrateur général*

Afin de permettre l'échange de données entre des deux dispositifs, nous avons utilisé une liaison série permettant de récupérer les informations d'un capteur telles que son état, son identifiant et l'identifiant du parking associé. Lorsque le système a récupéré ces informations, il doit ensuite les envoyer au serveur web.

#### *b. Communication entre le système et le serveur web*

Le système communique avec le serveur grâce à une *socket*. Elle permet de faire communiquer deux dispositifs entre eux. La connexion est faite grâce au système SSL (*Secure Sockets Layers*) qui est un procédé de sécurisation de transaction. Son principe consiste à établir

un canal de communication sécurisé (chiffré) entre le serveur et le système. En effet, SSL agit telle une couche supplémentaire permettant d'assurer la sécurité des données, située entre la couche application et la couche transport.

Remarque : l'utilisation de SSL était indispensable car le serveur web est un serveur sécurisé.

La communication se fait en mode connecté (TCP). De plus, il a fallu créer une requête *http* de type *post* afin d'envoyer les informations sur le serveur web. Cette méthode permet ainsi grâce à un script *php* de mettre à jour les informations stockées dans la base de données.

Principe de la communication :

La communication série est déclenchée lorsque le concentrateur général reçoit une mise à jour d'état de capteur. Plusieurs données sont alors acheminées par la liaison série :

1. un octet contenant l'identifiant du parking ;
2. un octet contenant l'identifiant du capteur ;
3. un octet contenant l'état du capteur.

Le système récupère ces informations et envoie par une requête *post* les informations à transmettre. Le script *php* permet de récupérer les données envoyées par le système et met ensuite à jour la base de données.

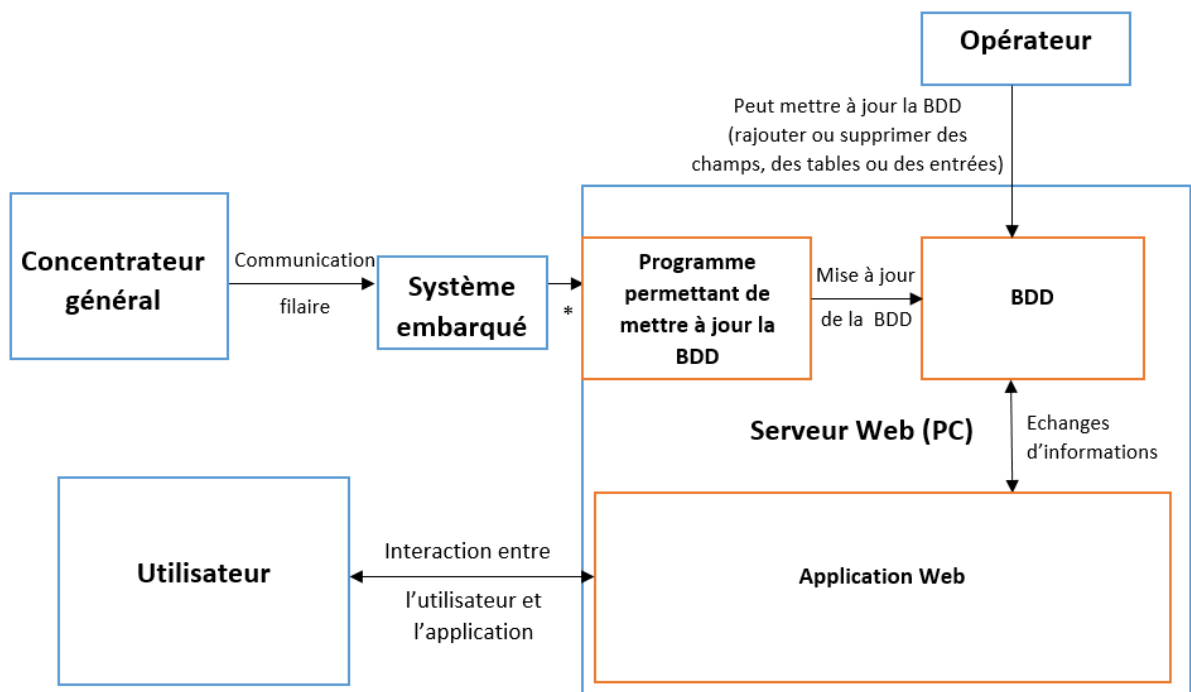
### **3. Partie applicative**

Cette partie du projet permet de récupérer les données envoyées par le réseau de capteurs et de mettre à jour la base de données en conséquence. Elle permet ensuite d'afficher les informations nécessaires à l'utilisateur pour qu'il puisse savoir s'il peut ou non se garer dans un parking choisi ainsi que de connaître les caractéristiques de ce dernier. Enfin, il est possible qu'un opérateur puisse mettre à jour la base de données, rajouter de nouvelles informations ou enlever. Il peut de plus, créer grâce à une interface, un parking virtuel représentant le parking réel qu'il souhaite ajouter. Ce dernier sera visualisable par un utilisateur de l'application afin d'avoir un aperçu des places disponibles où il peut se garer.

### a. Choix retenus

Une première phase d'étude a consisté à chercher le type d'application que nous allions utiliser (application Android, application web, etc.). Afin que tout le monde puisse avoir accès à l'application, nous avons choisi de faire une application web. En effet, dans le cas d'une application Android, certains utilisateurs de téléphones comme Apple ou Microsoft n'auraient pas pu y avoir accès. Le fait d'utiliser une application web permet à toute personne ayant une connexion Internet, qui est indispensable pour le projet dans un souci de mise à jour des informations, d'utiliser l'application.

Voici le schéma illustrant l'architecture du système :



\* : communication socket TCP avec système ssl et requête post

Figure 6 : Architecture du système

Une décision a été prise dans le choix de conception de la base de données (cf. Annexe II). En effet, il est important que les données soient stockés de telle manière à avoir y avoir accès facilement.

L'objectif de cette partie est d'afficher les informations utiles pour un automobiliste comme le nombre de places disponibles d'un parking ou les types de places (créneau, bataille, épi) pour un parking choisi par l'utilisateur.

Enfin, une dernière phase a consisté à réfléchir sur la manière dont un opérateur pouvait créer un plan de parking. Le but de ce dernier est qu'il puisse être affiché à un utilisateur en renseignant si une place est libre ou non ; en effet, chaque capteur est associé avec une place de parking. L'interface sera présentée ultérieurement.

#### *b. Travail réalisé*

Une première phase a consisté à se familiariser avec les outils nécessaires pour le projet. Cette étape a permis un gain de temps lors de la phase de programmation. Pour cela, des tutoriels ont été réalisés sur un site web créé sur un serveur local. À la suite des phases de tests, le site web a été mis sur un serveur pour pouvoir y accéder par Internet.

Une étape importante du projet a consisté à réaliser la gestion des droits utilisateurs. Il existe quatre types de droits :

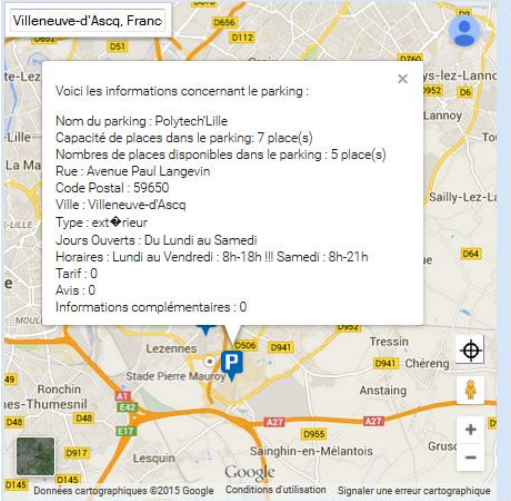
- l'administrateur, qui peut modifier les droits des utilisateurs ;
- le configureur, qui peut créer des parkings (plan, caractéristiques, etc.) ;
- le membre, qui peut modifier ses informations ;
- le "super utilisateur", qui a tous les droits précédemment cités.



En fonction de leurs droits, les utilisateurs peuvent accéder à des pages particulières afin d’y effectuer des actions comme la création d’un plan de parking accessible pour un configurateur. Cette gestion est gérée grâce à la base de données (cf. Annexe II) qui sauvegarde les informations. Ensuite, l’objectif, étant d’afficher les informations d’un parking sur le site web pour qu’un utilisateur puisse les visualiser, des méthodes ont été créées. En effet, dans un premier temps, l’utilisateur pouvait choisir un parking parmi une liste, en fonction de la localité (département et ville). Dans un second temps, afin de rendre plus agréable l’application, une carte *Google Maps* a été rajoutée. Ainsi, si l’on veut trouver parking, il suffit de rentrer l’adresse proche de celle du parking et la carte affiche les parkings à proximité. Afin de visualiser les informations, il suffit de choisir le parking souhaité.

En voici une représentation :

### Affichage Parking avec Carte



Voici les informations concernant le parking :

- Nom du parking : Polytech'Lille
- Capacité de places dans le parking: 7 place(s)
- Nombres de places disponibles dans le parking : 5 place(s)
- Rue : Avenue Paul Langevin
- Code Postal : 59650
- Ville : Villeneuve-d'Ascq
- Type : ext<sup>érieur</sup>
- Jours Ouverts : Du Lundi au Samedi
- Horaires : Lundi au Vendredi : 8h-18h !!! Samedi : 8h-21h
- Tarif : 0
- Avis : 0
- Informations complémentaires : 0

### Affichage Parking sans Carte

Veillez choisir un département et une ville pour cibler les parkings aux alentours

Nord (59)

Villeneuve-d'Ascq

Polytech'Lille, Avenue Paul Langevin, 59650

Voici les informations concernant le parking :

- Nom du parking : Polytech'Lille
- Capacité de places dans le parking: 6 place(s)
- Nombres de places disponibles dans le parking : 5 place(s)
- Rue : Avenue Paul Langevin
- Code Postal : 59650
- Ville : Villeneuve-d'Ascq
- Type : ext<sup>érieur</sup>
- Jours Ouverts : Du Lundi au Samedi
- Horaires : Lundi au Vendredi : 8h-18h !!! Samedi : 8h-21h
- Tarif : 0
- Avis : 0
- Informations complémentaires : 0

Figure 7 : Affichage de parking

**Remarque :** certaines fonctionnalités ont été rajoutées pour l’affichage avec la carte *Google Maps*. La gestion d’un itinéraire a été rajoutée ainsi que la gestion de la géolocalisation.

Dans un souci de rendre l'application assez fluide, une étape a consisté à mettre à jour les informations de la base de données sans que l'utilisateur s'en aperçoive. Ainsi, les informations se mettent à jour sans rechargement de la page et l'utilisateur peut voir assez rapidement le changement des informations d'un parking sans action de sa part.

Afin de faciliter au mieux l'accès à une place de parking, une partie a consisté à créer une interface graphique représentant le plan de parking. Ainsi, un utilisateur pourra visualiser les différentes places disponibles afin de s'y rendre pour se garer. L'interface permet de visualiser de manière détaillée un parking. Ce dernier est représenté par une grille. Sur cette interface, les places de parking, ainsi que les routes et obstacles (bâtiments, arbres, etc.) sont identifiés par des couleurs différentes (routes en gris, obstacles en orange). Une place de parking peut s'afficher de trois couleurs différentes : si elle est libre, elle s'affiche soit en vert, soit en bleu si c'est une place handicapée libre ; si elle est occupée, elle s'affiche en rouge dans tous les cas. Chaque place de parking est définie par son statut (libre ou occupée), sa taille (petite, moyenne ou grande) et son type (épi, bataille ou créneau), du point de vue de l'utilisateur. Ainsi l'utilisateur peut choisir sa place en fonction de ses préférences.

Représentation de l'interface graphique permettant de visualiser le plan de parking :

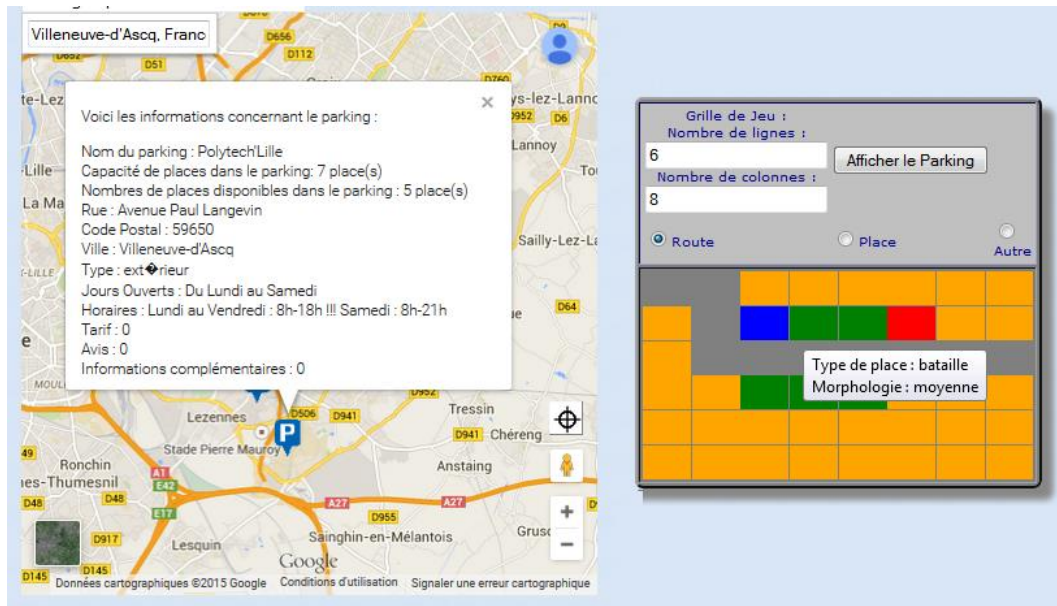


Figure 8 : Application avec interface utilisateur

Afin de créer un plan de parking, le configurateur peut créer un parking selon le même affichage. Il peut définir la taille du parking (en déterminant le nombre de lignes et de colonnes de la grille représentant le parking), les éléments (route, place ou obstacle) du parking, ainsi que les caractéristiques des places (type, morphologie, etc.). Ensuite, afin de faire le lien avec la partie réseau de capteurs, lorsque le configurateur renseigne une place, il définit l'identifiant de la place ainsi que l'identifiant de capteur associé. Ainsi, cela permet du côté visualisation de représenter correctement l'état d'une place.

Les données de l'interface graphique sont elles aussi sauvegardées dans la base de données. En effet, chaque case de la grille possède une caractéristique (route, obstacle ou place de parking) ; et dans le cas de la place de parking, celle-ci est directement associée à un identifiant de capteur.

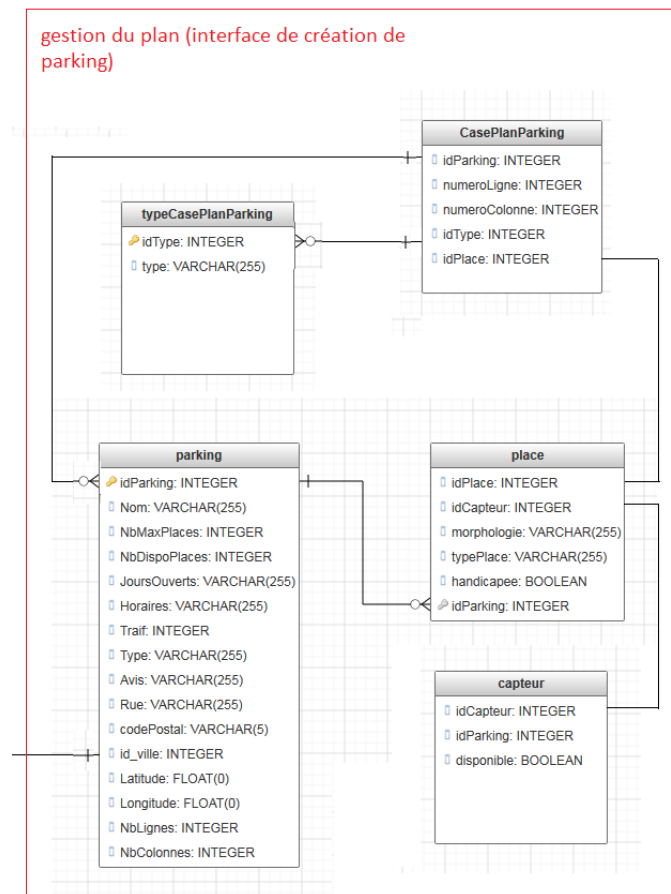


Figure 9 : Base de données de l'interface graphique

Cette interface permet alors de représenter un plan, avec la disponibilité des places et leurs caractéristiques.

idParking	Nom	NbMaxPlaces	NbDispoPlaces
1	Polytech'Lille	6	5

Modifier

Annuler

Grille de Jeu :

Nombre de lignes :

Nombre de colonnes :

Route  Place  Autre

ID du capteur correspondant :

ID de la place :

Morphologie : petite

Type de place : creneau

Type de place :  place handicapée

Figure 10 : Mode configurateur

Une dernière partie a consisté à faire la gestion d'inscription, de connexion et de mot de passe oublié. Il est possible de s'inscrire sur le site web, de se désinscrire ou encore de se connecter. De plus, lorsqu'un utilisateur oublie son mot de passe, l'application lui envoie un mail avec de nouveaux identifiants. En effet, le mot de passe est changé.

Par ailleurs, une page permettant à un administrateur d'ajouter, de modifier ou de supprimer un ou plusieurs utilisateurs a été créée. Ainsi, elle permet à un administrateur de ne pas modifier les informations directement sur la base de données.

### III. Améliorations possibles

---

#### 1. Partie réseau de capteurs

En ce qui concerne la communication RF, plusieurs améliorations sont possibles, notamment au niveau de la consommation d'énergie. En effet, il existe une fonctionnalité de LPM (*Low-Power Mode*) intégrée aux CC430, qui aurait pu être utilisée. De plus, un mode WOR (*Wake On Radio*) existe aussi afin de réduire la consommation d'énergie. Celui-ci permet de ne "réveiller" le CPU qu'en cas d'interruption RF, i.e. lorsqu'un message doit être envoyé ou est reçu, et de le "rendormir" une fois la tâche réalisée. Par ailleurs, la gestion des collisions est gérée en renvoyant les paquets, ce qui augmente alors la consommation d'énergie. La fonctionnalité de CCA (*Clear Channel Assesment*) aurait alors pu être implémentée afin de gérer ces problèmes de collisions : en effet, elle permet de détecter si un canal est libre ou non, et ainsi permet de savoir si une émission est possible sans problème de collisions. Dans le cas où le canal serait occupé, l'algorithme CSMA/CA (*Carrier Sense Multiple Access/Collision Avoidance*) permet de varier les temps d'émission et d'éviter au plus possible les collisions.

Par ailleurs, le réseau de capteurs est actuellement réalisé de manière statique : tous les trajets sont pré-établis et tous les identifiants sont eux aussi pré-établis, ce qui rendraient les choses peu pratiques en réalité (lors de la mise en service, par exemple). Il aurait alors été meilleur de pouvoir réaliser une assignation dynamique des identifiants des modules capteurs, suivant par exemple la distance avec le concentrateur le plus proche.

Une autre amélioration possible est la sécurisation de la communication RF ; en effet, il est possible de crypter les données à envoyer par communication RF afin de mieux les protéger.

## 2. Partie applicative

Certaines améliorations pour la partie applicative pourraient se réaliser, à savoir :

- Rajouter des fonctions permettant de modifier plus facilement les parkings pour le configurateur ;
- Réaliser une meilleure structuration du code afin de le rendre moins redondant ;
- Travailler le design afin d'avoir un rendu d'affichage plus propre ;
- Enregistrer les préférences des utilisateurs sur les places qu'ils affectionnent pour leur faciliter l'accès aux places les plus adaptées à leur besoin ;
- Réaliser un GPS dans le parking pour guider l'utilisateur à une place avec gestion de la voix.

# IV. Bilan du projet

## 1. Récapitulatif

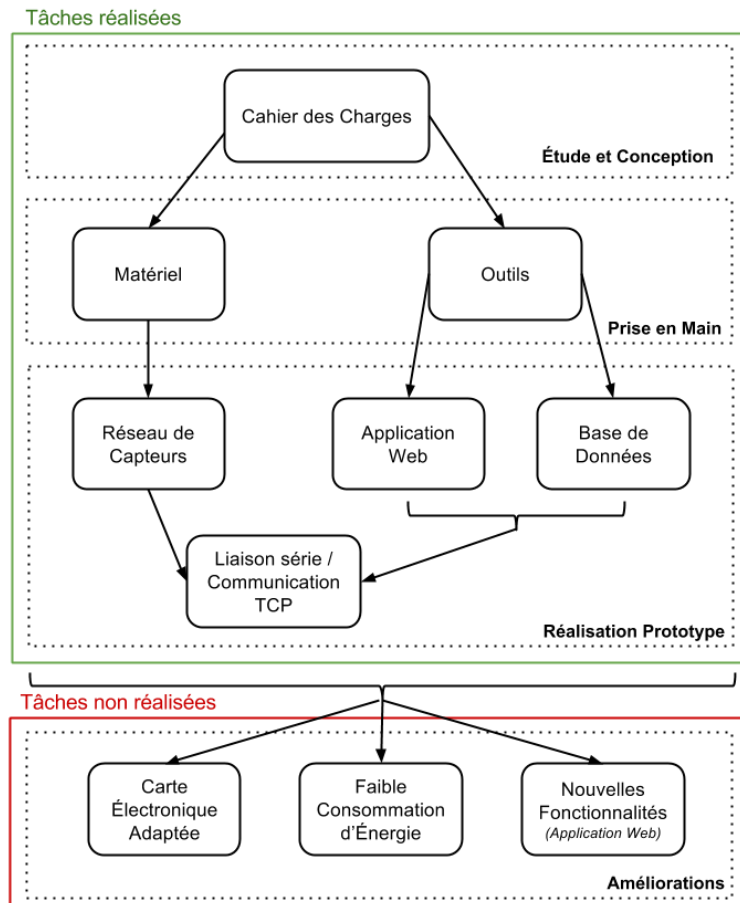


Figure 11 : Schéma récapitulatif

## 2. Difficultés rencontrées

Tout au long du projet, nous avons dû faire face à plusieurs problématiques :

- La compréhension du sujet a été une des tâches les plus complexes du projet. En effet, il est très important de comprendre en tout point le sujet donné afin de répondre au mieux à la problématique du projet. Nous avons pris rendez-vous avec nos tuteurs de projet afin de connaître précisément son contexte et de savoir les tâches indispensables à réaliser.

- La maîtrise de nouveaux procédés : l'utilisation de nouveau matériel comme les cartes CC430, de nouveaux outils tels les langages de programmation. La recherche d'informations via des livres, Internet, des articles rédigés à cet effet ainsi que des échanges avec des membres de l'équipe pédagogique de Polytech Lille ont été nécessaires afin de pallier les problèmes rencontrés. Il était indispensable d'assimiler ces procédés afin d'avoir un bon déroulement de projet.

- La recherche et les choix des solutions à adopter. Cette partie a été très problématique. En effet, nous n'avons pas beaucoup d'expérience dans les domaines dans lesquels nous avons travaillé et il a été difficile, de trouver les meilleures solutions et d'en choisir une. Afin de résoudre ceci, nous avons discuté avec des personnes plus expérimentées et nous avons pu échanger nos idées, les argumenter afin de retenir les meilleures.

- Le respect du planning s'est révélé pour cette moitié de projet difficile à suivre du fait de la survenue de problèmes ponctuels qu'il a fallu résoudre au fur et à mesure. Il a été nécessaire de trouver des solutions rapidement afin de remplir, dans les temps impartis, les objectifs fixés. Il est important de ne pas sous-estimer la tâche à effectuer afin de respecter les délais définis (gestion de projet).

### **3. Bilan personnel**

Nous avons trouvé ce projet très enrichissant tout d'abord d'un point de vue technique, il nous a permis de mettre à profit nos compétences acquises durant le cursus ingénieur. Nous avons également pu partager nos compétences et connaissances.

Ensuite d'un point de vue gestion de projet nous avons dû apprendre à se répartir les tâches afin de rendre notre travail plus productif et efficace. Il est également nécessaire d'anticiper les risques possibles du projet. Il est très difficile de suivre les objectifs fixés sans avoir de problèmes et c'est pourquoi, c'est à nous d'anticiper pour qu'il se déroule sans anicroche. Nous serons confrontés aux mêmes choix dans notre vie professionnelle, nous aurons des délais à respecter, nous devons alors prendre les dispositions nécessaires pour cela.



Nous aurions voulu ajouter plus de fonctionnalités sur l'interface de notre application ainsi que réaliser la communication basse consommation. Ces étapes n'ont pas été réalisées car le temps ne nous le permettait pas. Nous pouvons retenir qu'il faut toujours faire un prototype avec les fonctionnalités minimales afin de respecter le cahier des charges puis d'améliorer le projet par la suite. En effet, la partie Interface Homme-Machine par exemple prend un certain temps, si l'on privilégie ceci avant les fonctionnalités principales, on peut vite se retrouver en difficulté.

# Conclusion

---

Nous avons effectué ce projet dans le cadre de notre cinquième année dans la spécialité Informatique Microélectronique et Automatique à Polytech Lille.

Le projet consistait à créer un système permettant à des automobilistes de se garer plus facilement dans les parkings en les aidant à trouver des places de parking. La rédaction d'un cahier des charges, la conception d'un réseau de capteurs, la réalisation de l'interaction avec la base de données ainsi que la création d'une application web ont été les différentes étapes qui nous ont été nécessaires afin de créer un tel dispositif.

Nous avons alors pu, lors de ces vingt-et-une semaines de projet, mettre en pratique nos connaissances acquises durant la formation et en développer de nouvelles. Cette expérience nous a aussi permis d'anticiper les différents problèmes qui se sont dressés devant nous, de travailler en équipe ainsi que de gagner en maturité et de prendre confiance.

## Annexes

---

# Annexes

---

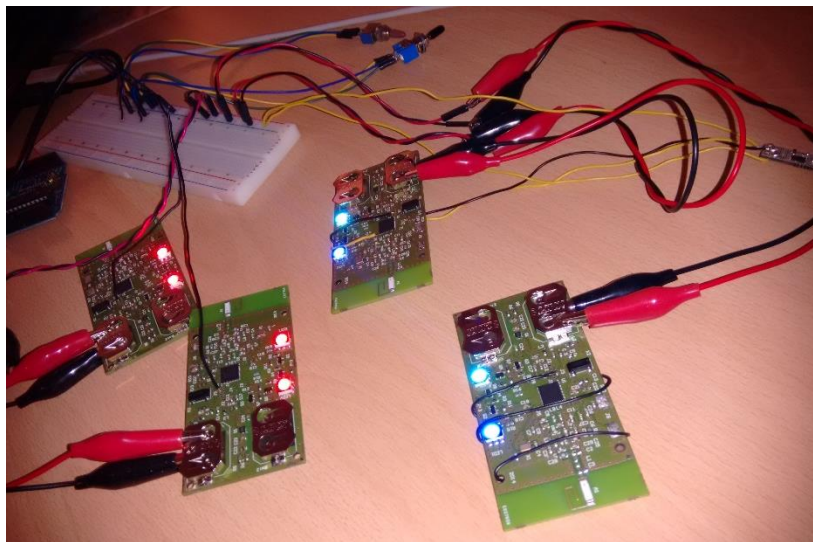
Annexe I : Branchement des cartes CC430.....	36
Annexe II : Base de données.....	38
Annexe III : Utilisation d'Ajax.....	39

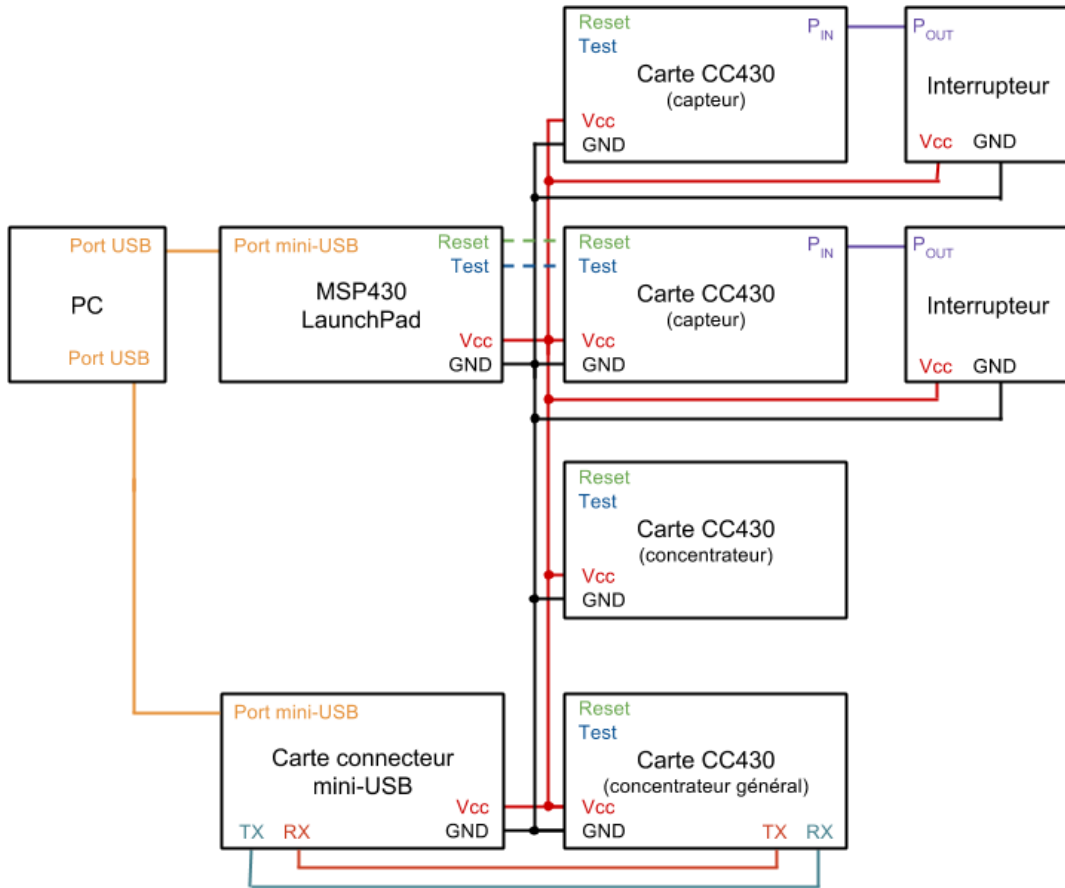
## Annexe I : Branchement des cartes CC430

---

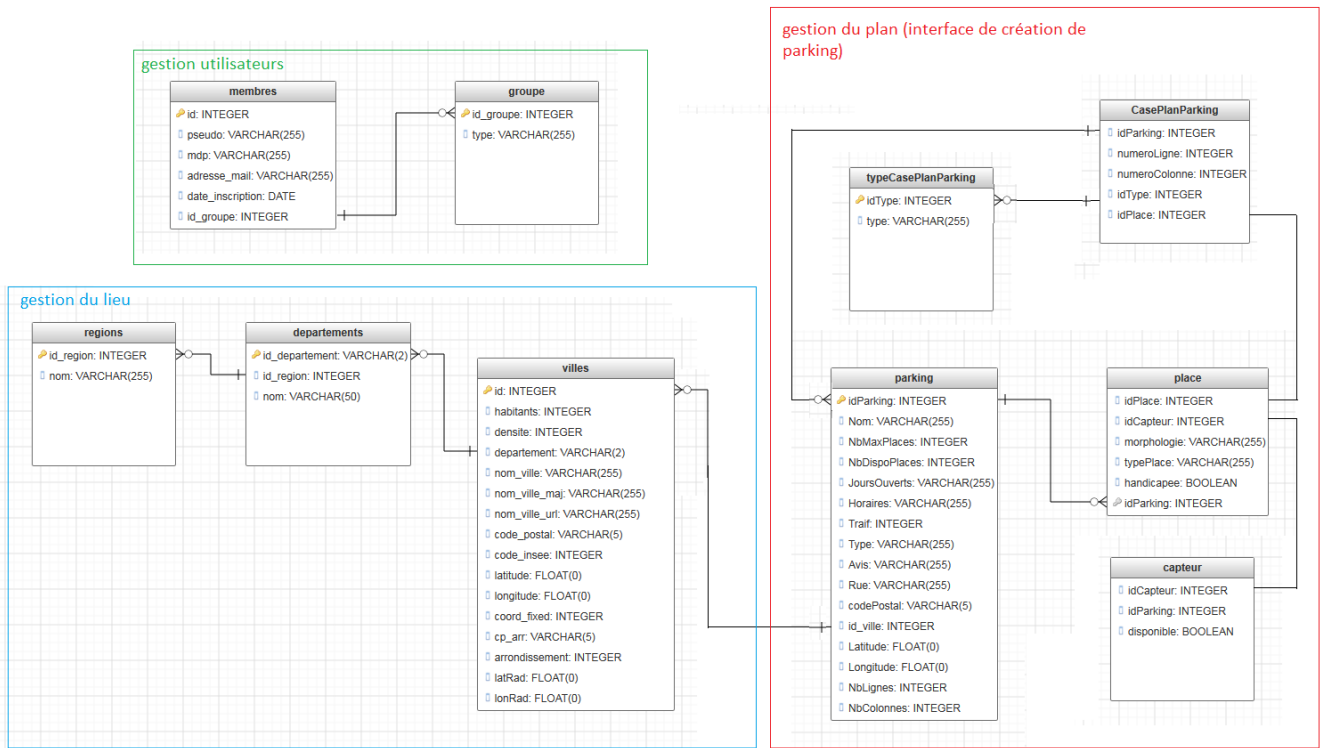
Les branchements réalisés entre le LaunchPad et les cartes CC430 est comme indiqué sur la photo et le schéma. À ces branchements s'ajoutent des interrupteurs (qui simulent l'état des capteurs) et un connecteur série permettant de faire le lien entre une carte CC430 et le PC. Ces branchements permettent le fonctionnement du réseau de capteurs, jusqu'à la liaison série vers le PC. Le branchement des ports *Test* et *Reset* ne sont nécessaires que lorsqu'un programme doit être déployé sur les cartes CC430.

Remarque : le schéma a été simplifié au niveau de l'alimentation des cartes CC430 ; en effet, de par la puissance demandée par les cartes CC430, un Arduino a été utilisé en plus afin de les alimenter.





## Annexe II : Base de données



Le schéma ci-dessus représente la base de données gérant le site web. Elle est séparée en trois parties distinctes :

- partie lieu : stocke les éléments permettant de retrouver parking en fonction de la localité renseignée;
- partie interface : stocke les éléments en relation avec l'interface graphique de de plan de parking;
- partie utilisateur : Elle permet de gérer les droits des utilisateurs.

## Annexe III : Utilisation d'Ajax

---

```
/**
 * Cette fonction permet de mettre à jour les coordonnées du parking
 */
function miseAJourPlanParking(nbreLigne, nbreColonne) {
    var parameters = {idParking: idParking, nbreLigne: nbreLigne, nbreColonne: nbreColonne};

    // appel à l'ajax pour mettre à jour la BDD
    $.ajax({type: 'post', dataType: 'json', url: 'miseAJourPlanParking.php', data: parameters});
}
```

Le code est un code *JavaScript* (côté client) permet de mettre à jour la base de données. En utilisant *Ajax*, le code *JavaScript* peut communiquer à la partie serveur les paramètres nécessaires pour la mise à jour de la base de données. Ces informations sont ensuite traitées dans un programme *php* (ici 'miseAJourPlanParking.php) qui se charge de mettre à jour la base de données.