YVERNAULT Bertrand
THEBAULT Louis

Promotion 2016

IMA4

Rapport de projet Semestre 8

Aimant intelligent

Département Informatique-Microélectronique-Automatique
Polytech' Lille
Université Lille 1

SOMMAIRE

Table des figures

I. Introduction

Le “Bluetooth” est une méthode populaire de communication entre des appareils. De nos jours, beaucoup de smartphones ont la capacité de communiquer en utilisant le Bluetooth.

Le “Bluetooth Low Energy” (BLE) est destiné le plus souvent pour des objets connectés. Il possède une faible consommation d’énergie comparée au Bluetooth classique. Le BLE étend l’utilisation de la technologie sans fil Bluetooth à différents domaines comme le sport, la santé, les périphériques d’ordinateurs (souris et claviers), dans les balises, dans le vestimentaires et dans d’autres appareils utilisés pour le divertissement.

Notre projet consiste à concevoir un objet connecté aimanté, utilisant la technologie “Bluetooth Low Energy”, permettant de délivrer des rappels et qui sera utilisable avec une application Android.

On va donc vous présenter tout d’abord un peu plus précisément le projet que nous avons réalisé avec le cahier des charges que nous avons mis en place au début de notre projet. Nous vous présenterons ensuite le développement réalisé avec le Rfduino et l’application Android. Et pour finir on vous montrera la conception de notre prototype pour le boîtier.

1. Présentation du projet

Le but de ce projet est de développer un "aimant" qui utilisera une balise BLE (Bluetooth Low Energy) qui nous permettra de partager des données simplement et localement depuis un téléphone portable : cela reprend l'idée du post-it sur le réfrigérateur. Ce post-it qui nous rappelle un rendez-vous chez le dentiste, une liste de course ou juste un petit mot pour sa famille.

On a séparé ce projet en deux parties :

- un système embarqué permettant de recevoir et de transmettre des informations
- une application Android permettant de communiquer avec le système

On concevra à l'aide de l'imprimante 3D le boîtier accueillant la balise.

2. Cahier des charges

Voici quels ont été nos objectifs tout au long du projet :

Étape 1 : Étude bibliographique sur le BLE et sur le développement d'application Android (java)

Étape 2 : Faire fonctionner la balise BLE avec la transmission et la réception d'un message

Etape 3 : Développer l'application Android (Connexion, transmission, réception, affichage,...)

Étape 4 : Finition de l'application et réflexion sur la forme du boîtier

Étape 5 : Réalisation du boîtier

3. Planning prévisionnel

Nous avons ensuite mis en place un planning prévisionnel :

Semaine 1 : Recherche bibliographique sur le fonctionnement de la BLE et sur les applications Android (développement en java)

Semaine 2 à 6 : Faire fonctionner la balise BLE en transmission et réception

Semaine 2 à 6 : Développement de l'application sous Android

Semaine 6 à 9 : mise en marche des deux systèmes ensemble (BLE + application Android)

Semaine 7 à 9 : Fabrication du boitier

Semaine 10 à 11 : Réalisation de la vidéo

II. Développement de l'application Android

Nous avons développé une application Android en utilisant le logiciel Android Studio. Nous n'avons jamais développé d'application Android ou codé en java auparavant. Nous avons donc commencé par suivre des tutorats / cours en ligne pour apprendre la base de la programmation Android/java.

Nous avons ensuite pu mettre en place une liste de fonctionnalités à réaliser pour faire marcher notre application.

- Mise en marche du Bluetooth, si celui-ci n'est pas déjà en marche
- Détection des appareils utilisant la technologie BLE.
- Connexion et récupération des informations de la balise BLE
- Communication entre la balise BLE et l'application android (envoie et réception d'un message)
- Suppression de messages choisis.

1. Détection des appareils utilisant la technologie BLE

Un des premiers points important à développer était la détection des appareils utilisant la technologie BLE. Pour cela nous avons crée un bouton "scan" qui va nous permettre de scanner l'appareil le plus proche de notre tablette. Puis nous affichons les données de notre balise.

Les appareils doivent d'abord se découvrir les uns les autres. On utilise un service du système qui s'intitule BluetoothManager pour permettre cela. Pour lancer un scan, on appelle la fonction "startLeScan()" avec une référence à "BluetoothAdapter.LeScanCallBack" où les résultats du scan seront livrés par le rappel "onLeScan()"

Figure 1 : Détection d'appareils

2. Connexion à la balise

Une fois notre balise BLE découvert, nous avons besoin d'établir une connexion entre la balise et l'application. La première étape dans l'interaction avec un dispositif BLE est de se connecter dans un premier temps au serveur GATT sur l'appareil.

Pour se connecter à un serveur GATT, nous avons une action "ACTION_CONNECTED" qui permet cela. Il faut savoir qu'un GATT définit la façon dont les deux dispositifs "Bluetooth Low Energy" transfèrent des données en utilisant des concepts appelés "services" et "caractéristiques".

Nous avons créé un bouton "Connecter" avec un texte qui nous indique si nous sommes "Déconnecté" ou "Connecter".

Figure 2 : Connexion

On observe le passage de déconnecter à connecter entre les deux figures.

3. Communication entre l'application et la balise BLE

Un des points difficiles dans le développement de l'application a été de faire une communication entre l'application et la balise BLE. Donc permettre l'envoie et la réception d'un message.

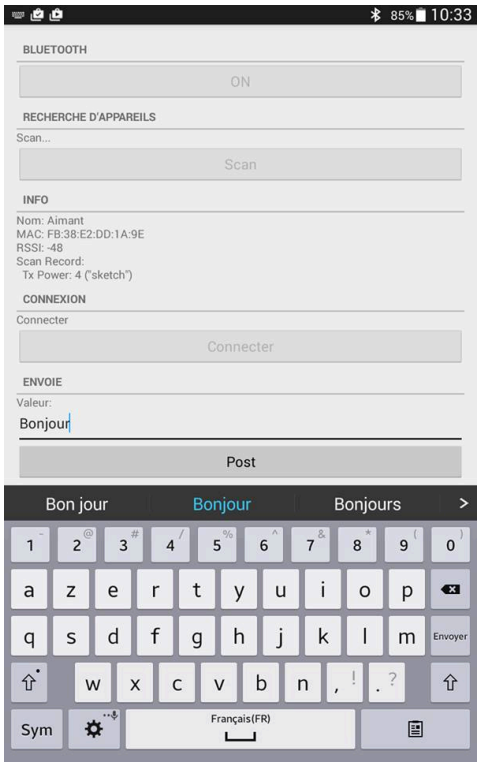

Figure 3 : Envoie d'un message

Nous pouvons ensuite voir la réception du mot. La balise BLE nous envoie exactement la même chose.

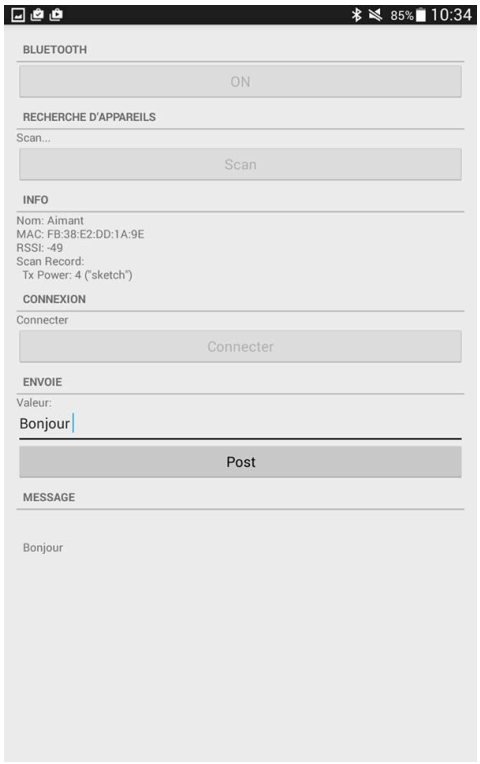

Figure 4 : affichage du message

III. Développement du code Rfduino

Puis il a fallu développer un code pour la balise RFduino en utilisant l'IDE Arduino. Mais qu'est-ce qu'un RFduino ?

Un RFduino est un Arduino de très petite taille, disposant d'une connexion Bluetooth.

Pour nous aider, une librairie du RFduino nous est disponible sur internet. Nous l'avons donc utilisée.

1. Activation du Bluetooth

Pour l'activation du Bluetooth, il y a une fonction présente dans la librairie : *RFduino.begin().* Cette fonction lance la pile de la BLE et fait apparaitre la balise aux autres appareils.

De plus, nous pouvons choisir le nom que nous donnons à la balise. Par exemple, nous voulons appeler notre balise "Aimant", nous utilisons : *RFduino.deviceName = "Aimant"*. Le nom par défaut de la balise est "RFduino".

Il existe d'autres fonctions qui permettent d'envoyer des informations lors de la découverte de la balise mais nous ne les avons pas utilisées.

2. Connexion et déconnexion

Il n'y pas de fonctions pour permettre de connecter ou de déconnecter la balise. Cependant on peut vérifier qu'un appareil est : soit connecté, soit déconnecté. Pour cela, nous avons affiché sur le port série un message pour savoir si un appareil s'est connecté ou déconnecté.

```
void RFduinoBLE_onConnect() {
  //Ecris "Nous sommes connectés" sur le port série
  Serial.print("Nous sommes connectés\n");
 }
void RFduinoBLE_onDisconnect() {
  //Ecris "Nous sommes déconnectés" sur le port
  Serial.print("Nous sommes déconnectés\n");
 }
```

3. Réception d'un message

Pour recevoir un message, nous avons utilisé une fonction contenue dans la librairie RFduino : *RFduinoBLE_onReceive()*.

Cette fonction permet de retourner ce que la balise reçoit. C'est dans cette fonction que nous allons faire la gestion des messages reçus.

4. Envoie d'un message

Il existe plusieurs fonctions pour pouvoir envoyer des mots de différents types. Celles ci sont présentes dans la librairie. Nous avons utilisé une seule fonction qui permet d'envoyer des types char. Pour pouvoir envoyer, nous avons besoin du mot à envoyer et de sa taille.

```
RFduinoBLE.send(data, len);
```

"Data" représente ce que nous envoyons et "len" correspond à la taille.

5. Sauvegarde des messages

Cette partie fût celle qui nous a posé beaucoup de problèmes et nous n'avons pas réussi à les résoudre.
Nous avons pensé à créer un type de variable avec plusieurs champs : Message.

```
struct Message
{
  char *mot;
  bool use;
  int num;
};
```

La variable "*mot" va recevoir la donnée reçu par la balise.
La variable "use" est pour savoir si ce mot est utilisé.
Le variable "num" est pour savoir c'est le combientième mot reçu. Cette variable aura été utilisée pour la suppression des messages.

Le but était de déclarer un grand nombre de messages et d'utiliser une fonction qui permettait de classer les données reçu dans un message. Par exemple, pour deux messages, nous avions la fonction suivante :

```
void SauvegardeMessage(char *recept){

 if (m1.use != true)
   {m1.mot=recept;
    m1.use=true;
    m1.num=1;
   }
  else if(m2.use== false)
   {m2.mot=recept;
    m2.use=true;
    m2.num=2;
   }
```

Cependant, nous avions des problèmes. Lorsque nous recevions un seul message, il n'y avait pas de problème. Lorsque nous recevions un autre message, il était bien mis dans un autre message, mais le premier message était effacé par celui-ci.

IV. Conception du boîtier

Le but final de notre projet est d'avoir un objet connecté que nous pouvons utiliser sans l'aide d'un ordinateur. Pour cela nous a dû créer un boîtier pouvant accueillir la balise BLE. La réalisation de ce boîtier a été faite en deux parties : la première s'agissait de la modélisation de notre boîtier puis de la réalisation de celui-ci.

1. Modélisation du boîtier

Dans un premier temps, nous avons réfléchis à la forme que nous voulions donner au boîtier. Nous nous sommes dit qu'il devait être assez esthétique c'est-à-dire un boîtier qui ne prends pas trop de place et qui ne soit pas gênant. Nous avons d'abord pensé à une sphère cependant après réflexion, nous avons pensé qu'il y aurait un problème pour accrocher le boîtier sur mur par exemple. Nous en avons donc choisi de faire une demi-sphère.

Puis est venu la question de comment allons-nous faire pour fermer notre boîtier. Nous avons pensé à plusieurs solutions mais celle qui nous a parue la meilleur d'un point de vu réalisation est de faire dépasser des petits cubes des deux pièces pour pouvoir les assembler soit avec des élastiques soit avec des vis.

Une fois la réflexion sur notre boîtier terminé, nous nous sommes lancés dans la modélisation. Pour cela, nous avons utilisé le logiciel de modélisation "SolidWorks". Celui-ci est facile d'utilisation et très pratique. Nous avons pu donc modéliser nos deux pièces ci-dessous :

Le dessus :

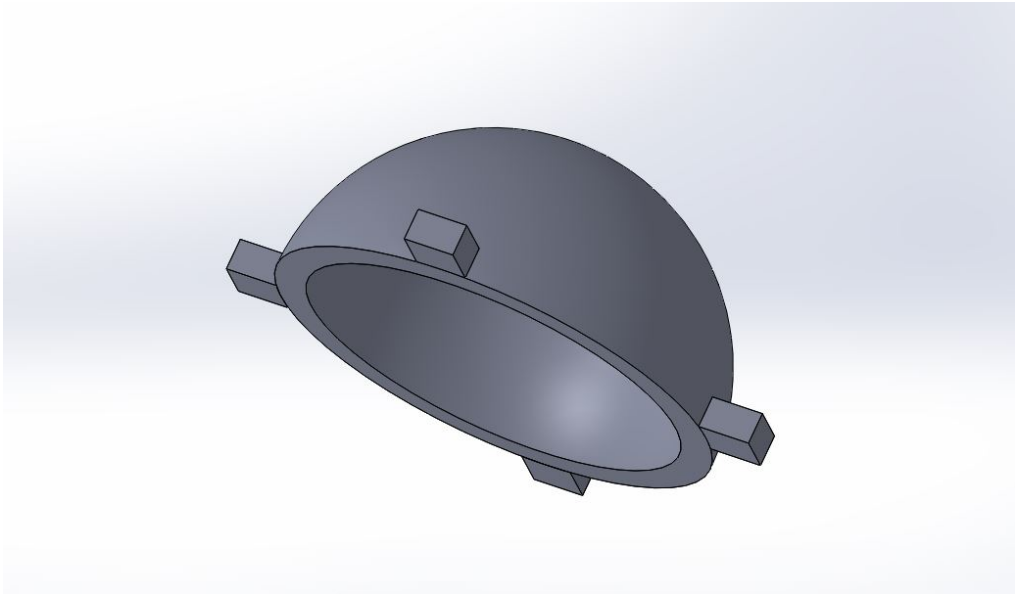

Figure 5 : Dessus de la pièce

Le couvercle :

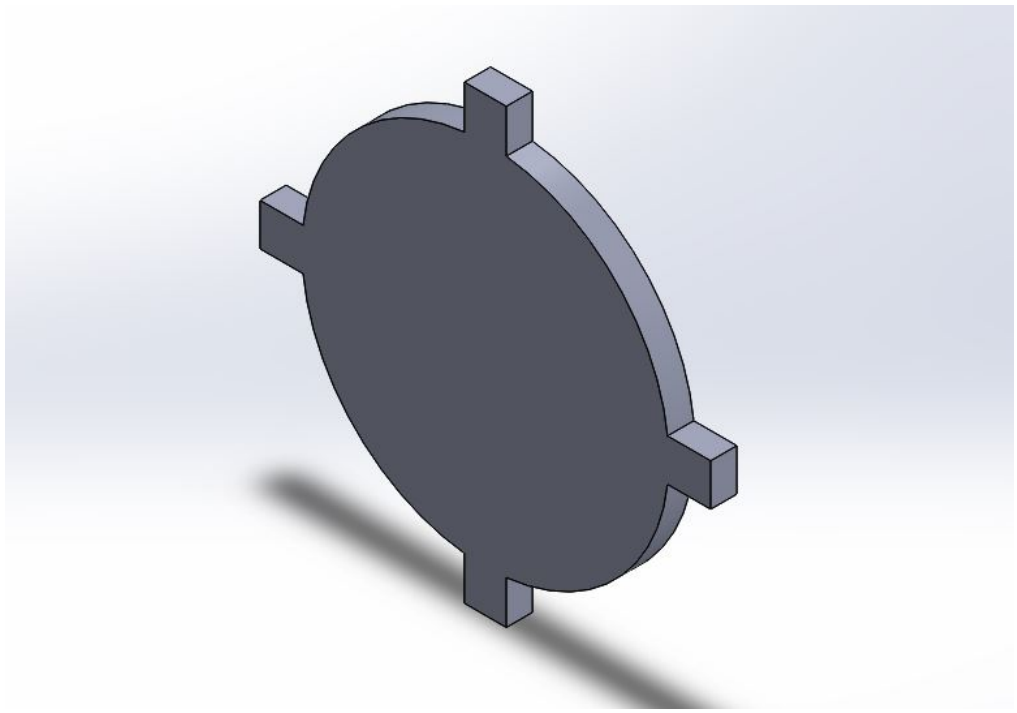

Figure 6 : Le couvercle de la pièce

2. Réalisation du boîtier

Après avoir fait la modélisation de nos pièces sous Solidworks, nous avons pu nous lancer dans la réalisation de notre boîtier. Pour cela nous avons utilisé une imprimante 3D pour le fabriquer. On a dû utiliser le logiciel Cura pour pouvoir convertir nos fichiers SolidWorks en un type de fichier utilisable par l'imprimante 3D. Après avoir effectué cela, on transmet nos fichiers à l'imprimante 3D grâce à une carte SD. Puis nous avons dû régler l'imprimante 3D pour que le plateau accueillant la pièce soit bien placé. Enfin nous avons lancé l'impression des pièces séparément.

Le dessus a mis environ 3h à être imprimé.

Figure 7 : Dessus de la pièce imprimée

Le couvercle a mis environ 1h à être imprimé.

Figure 8 : Couvercle de la pièce imprimée

Une fois que nos pièces ont été imprimées, nous avons rajouté des aimants pour permettre de mettre notre boîtier sur un frigo par exemple.

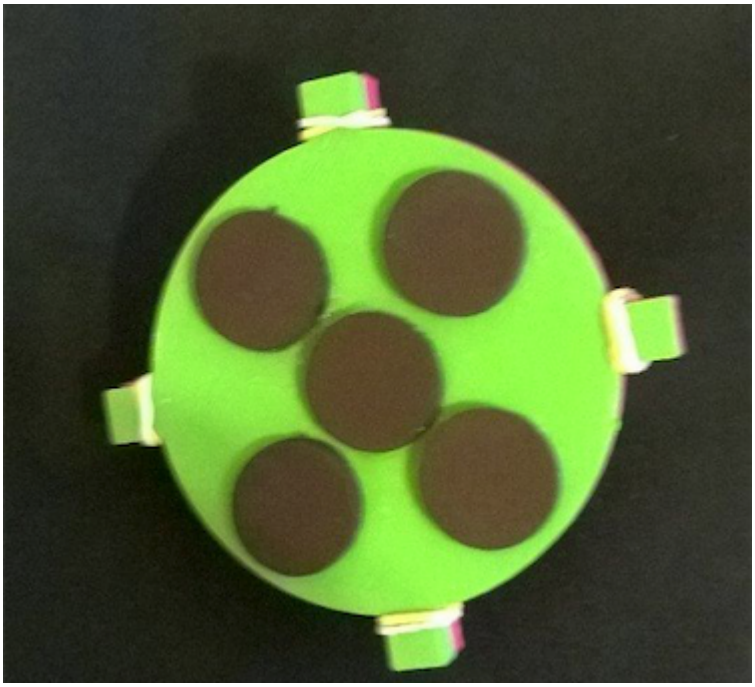

Figure 9 : Pièce final

V. Conclusion

Ce projet a été, pour nous, une excellente expérience personnelle et pédagogique. Nous avons pu mener un projet de A à Z, avec la mise en place d'un cahier des charges et d'un planning prévisionnel. Nous avons su nous montrer autonomes face aux difficultés rencontrées. Nous avons dû apprendre à gérer notre projet au niveau de l'organisation du travail et la répartition du temps. Nous avons su aussi être débrouillards dans la recherche d'informations qui s'est avérée essentielle.

Nous avons rencontré quelques difficultés, en partie pour le développement de l'application Android qui était pour nous une quelque chose de nouveau. Nous avons donc mis du temps à assimiler le langage java et nous pouvons le dire, nous avons encore un peu de mal. Nous avons eu aussi du mal à s'adapter au RFduino. C'était un nouveau microcontrôleur que nous avons rencontré. Nous avons appris à utiliser de SolidWorks, tout comme l'imprimante 3D qui est une technologie assez impressionnante. En somme, ce projet nous as permis donc d'apprendre à utiliser de nouvelle connaissance qui nous seront utiles dans l'avenir.

Nous n'avons malheureusement pas eu le temps de porter notre projet à bout. Il nous manque encore quelques points à traiter :

- La suppression individuelle d'un message
- Faire marcher la sauvegarde des messages par la BLE
- Le PCB de la pile qui nous permettre de rendre la balise indépendante