El Yassem Bilal

Garibal Florian

Année 2014-2015

*e*

Sommaire

[1 What is it about ? 3](#_Toc418602094)

[2 What did we have to do ? 4](#_Toc418602095)

[3 With what ? 5](#_Toc418602096)

[4 Project steps 8](#_Toc418602097)

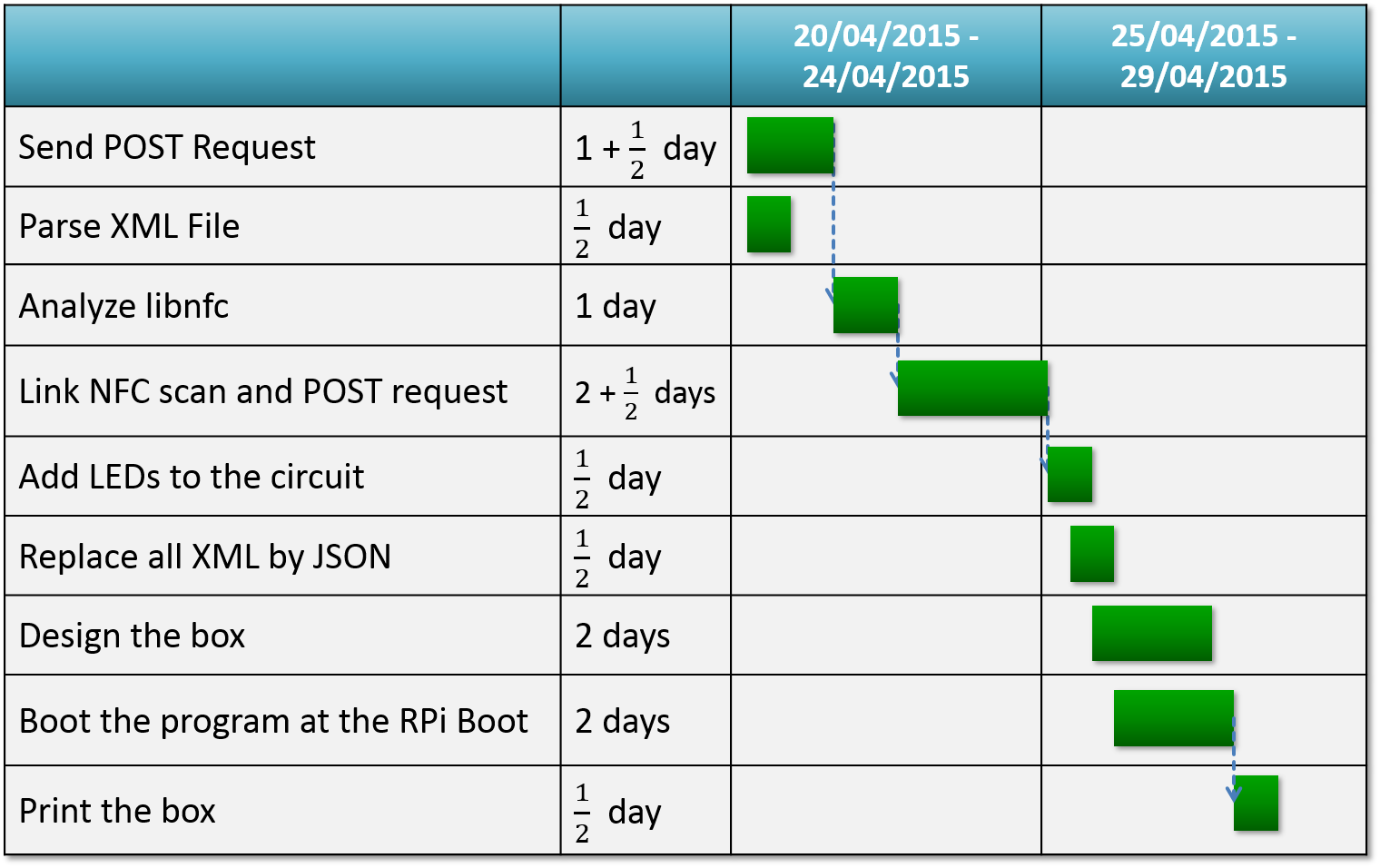
[Conclusion 9](#_Toc418602098)

# What is it about ?

ATTENTE EXPLICATION de Keith

# What did we have to do ?

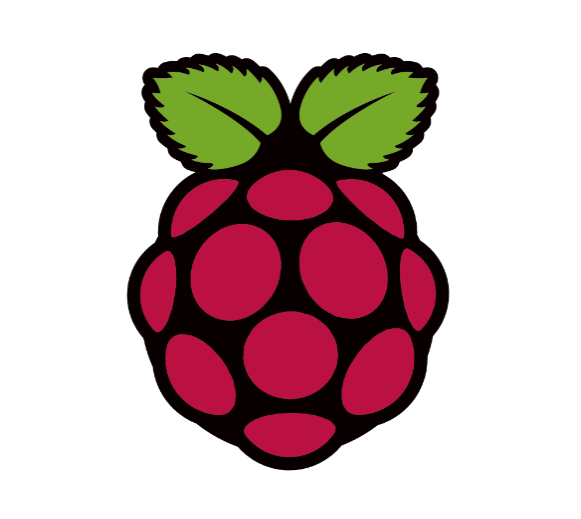
Le client souhaitait qu’un lien soit établi entre le fait de scanner une carte NFC et l’enregistrement de l’identifiant de cette carte sur un serveur spécifique. Le serveur en question étant, lui, un projet d’un précédent stagiaire qui consistait à gérer \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. Pour cela il nous a été demandé de réaliser un programme C permettant premièrement de lire un fichier de config où serait stocké l’identifiant et le mot de passe de connexion à l’API du serveur puis deuxièmement de lire l’identifiant d’une carte scannée, préparer et envoyer une requête POST au serveur puis interprêter la réponse. Concernant l’interprétation, si aucune erreur n’était présente, il nous a été demandé d’allumer une DEL verte et dans le cas contraire d’allumer une DEL rouge. En outre, étant donné que c’était un prototype, il était aussi nécessaire de concevoir une boîte capable de contenir l’ensemble du système tout en étant la plus petite possible.

 D’autre part, il nous fallait finir ce projet le plus vite possible en raison du retard important qu’il avait prit. De ce fait nous avons mis en place le Gantt prévisionnel suivant :

# With what means ?

In order to carry out the project, some hardware elements, which are explained below, were given to us.

To begin, a micro controller, the **Raspberry Pi B+ V1.2**, was provided to us. It permits to have all PC features in one little chip. Indeed, it has some RAM, input/output (HDMI, USB, etc, …) but it has also a SD slot which allows it to have an OS.

Features : ****

* Size : 85mm x 56mm x 17mm
* Processor : Broadcom SoC tournant à 700MHz
* RAM : 512MB
* 1 Ethernet port
* 4 USB ports
* 1 HDMI port
* 1 power outlet microUSB 5V/2A
* 1 GPIO connector with 40 pins
* 1 audio jack output 3.5mm
* 1 MicroSD card slot
* 4 fixation hole (drawed in beige on the diagram)

Picture and diagram of the Raspberry Pi B+ V1.2

Then, one **Adafruit PN532 Breakout Board** was provided to us.This card is the most popular card of the NFC cards and it is also this one which are in most of smartphone or device which contains NFC. Indeed, it permits to read and write data from a NFC card or even a NFC tag. Moreover this card is used by the libnfc library which allows to create program that can interact avec the card on every exploitation system (Windows, Linux, Mac).

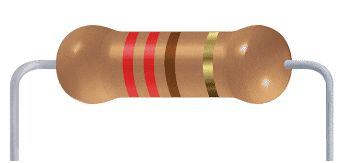
Two NFC cards were given to us in order to test our program.

Features :

* Size : 51mm x 117.7mm x 1.1mm
* This board/chip uses I2C 7-bit address 0x48

Picture of the Adafruit PN532 card and of one NFC card

On the other hand to display a feedback to the use, two LEDs, one green and one red, were connected to the circuit. In order to prevent overload, it was necessary to connect one electrical resistance before each. We were advised to use two electrical resistance of 220Ω ± 5% each.



Picture and scale of two LEDs (left side), 3D image of a 220Ω electrical resistance (right side)

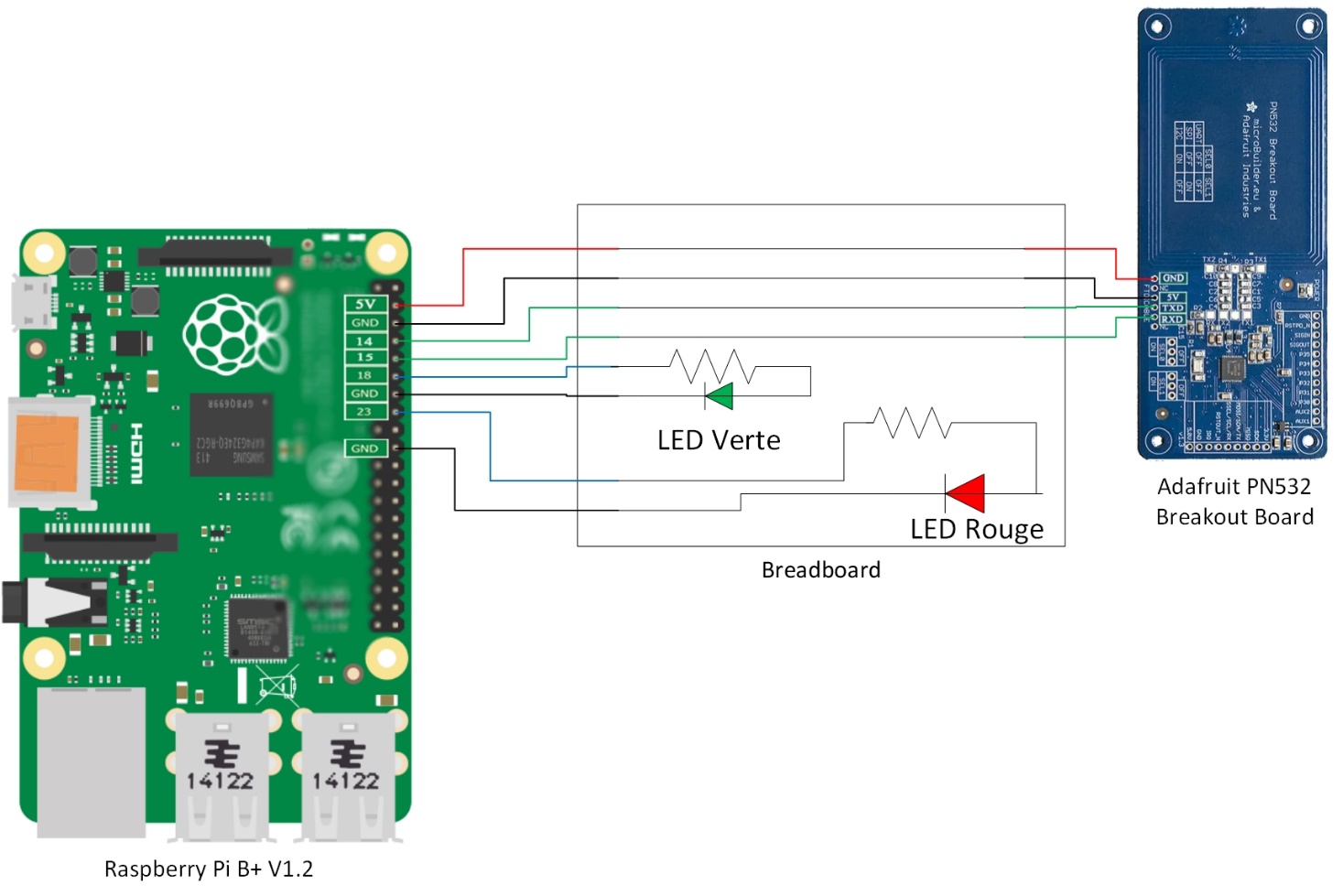
In order to link these two cards and LEDs, a breadboard and some electrical wires were given us. All of this was integrated in the following circuit : 

Diagram of the electrical circuit which joins the two cards and the two LEDs

It was asked us to work with C language and the libnf library which provide methods that allows to interact with the NFC reader.

For this project, we decided, with Bilal El Yassem, to use curl library, which make network exchanges easier, and json libraries, which permits to parse JSON files easily.

# Project steps

A REDIGER

• Ecrire un programme qui envoie une requête POST(libcurl)

• Ecrire un programme qui parse un fichier XML (ezxml)

• Comprendre la librairie libNFC (analyser tous les fichiers, recherche sur internet, ...)

Faire des tests pour arriver à récupérer l'ID d'une carte dans une variable

• Lier le fait de lire une carte et d'envoyer une requête POST à un serveur (RequestBin pour le moment car serveur down)

• Traiter la réponse du serveur en essayant de modifier le serveur pour qu'il nous la transmette en XML et non JSON

• Se rendre compte que c'est pas possible donc écrire un programme qui parse du JSON basique

• Modifier tous les parsages du programme de XML vers JSON

• Démonter le circuit des DEL et remonter pour que ça fonctionne

• Intégrer les DEL dans l'interaction avec la carte (allumage rouge puis vert si tout s'est bien passé)

• Faire démarrer le programme au boot du Raspberry

• Assembler les différents élément de la carte grâce à une project board, des fils soudés et deux "\_\_\_\_\_\_\_\_\_\_"

• Designer la boîte grâce à FreeCAD

• Imprimer la boîte

Conclusion