|  |
| --- |
| Présentiel pour openSpace |

Table des matières

[1 Analyse préliminaire 4](#_Toc516582734)

[1.1 Introduction 4](#_Toc516582735)

[1.2 Organisation 4](#_Toc516582736)

[1.3 Objectifs 4](#_Toc516582737)

[1.4 Planification initiale 5](#_Toc516582738)

[2 Analyse / Conception 5](#_Toc516582739)

[2.1 Concept 5](#_Toc516582740)

[2.1.1 Maquette schématique de l’application 6](#_Toc516582741)

[2.1.2 Cas d’utilisation 6](#_Toc516582742)

[2.2 Stratégie de test 7](#_Toc516582743)

[2.2.1 Fonctionnement 8](#_Toc516582744)

[2.2.2 Erreurs 8](#_Toc516582745)

[2.2.3 Paramètres de connexion 9](#_Toc516582746)

[2.3 Planification 9](#_Toc516582747)

[2.4 Dossier de conception 9](#_Toc516582748)

[2.4.1 Hardware 9](#_Toc516582749)

[2.4.2 Software 9](#_Toc516582750)

[2.4.3 Schéma de montage boitier 10](#_Toc516582751)

[2.4.4 Conception du code 11](#_Toc516582752)

[3 Réalisation 13](#_Toc516582753)

[3.1 Dossier de réalisation 13](#_Toc516582754)

[3.1.1 Projet Arduino 13](#_Toc516582755)

[3.1.2 Projet application 14](#_Toc516582756)

[3.2 Description des tests effectués 15](#_Toc516582757)

[3.2.1 Méthode 15](#_Toc516582758)

[3.2.2 Presser le bouton 15](#_Toc516582759)

[3.2.3 Changer le statut sur l’application 19](#_Toc516582760)

[3.2.4 Changer le statut Skype 22](#_Toc516582761)

[3.2.5 Déconnecter Skype 24](#_Toc516582762)

[3.2.6 Déconnecter le boitier 24](#_Toc516582763)

[3.2.7 Déconnecter l’application 25](#_Toc516582764)

[3.2.8 Cliquer sur le message « boitier déconnecté » 25](#_Toc516582765)

[3.2.9 Ouvrir la page de paramètres 26](#_Toc516582766)

[3.2.10 Changer les paramètres 26](#_Toc516582767)

[3.3 Erreurs restantes 27](#_Toc516582768)

[3.4 Liste des documents fournis 27](#_Toc516582769)

[4 Conclusions 27](#_Toc516582770)

[5 Annexes 28](#_Toc516582771)

[5.1 Sources – Bibliographie 28](#_Toc516582772)

[5.2 Glossaire 29](#_Toc516582773)

[5.3 Archives du projet 29](#_Toc516582774)

# Analyse préliminaire

## Introduction

Prenant peu de place et facilitant la communication entre employés, les open space se répandent. Malheureusement il y a aussi des effets négatifs comme le fait qu’une personne soit trop accaparée par les autres.et aie par conséquent moins de temps pour travailler. Ce projet vise à réduire ce souci en proposant un indicateur de disponibilité lumineux lié par Bluetooth au poste de travail de la personne ainsi qu’à son Skype.

Ayant un grand intérêt pour l’électronique et la programmation, ce projet me motive beaucoup. Autre fait motivant :

J’ai pu utiliser un Arduino communiquant avec un PC en Bluetooth lors de mon projet de pré-TPI ce qui me permettra d’aisément mettre en place cette partie du système. La liaison avec Skype est quelque chose d’encore inconnu pour moi, ce qui m’intéresse car nouveau.

## Organisation

Les personnes prenant part au projet sont les suivantes :

|  |  |  |  |
| --- | --- | --- | --- |
| **Fonction** | **Nom** | **E-mail** | **Téléphone** |
| Candidat | Kalbfuss Cyril | cyril.kalbfuss@cpnv.ch | 079 839 39 25 |
| Chef de projet | Ithurbide Julien | julien.ithurbide@cpnv.ch | 079 255 67 08 |
| Expert 1 | Malherbe Roger | r.malherbe@rmsoft.ch | 079 230 72 37 |
| Expert 2 | Folomietow Borys | borys@folomietow.ch | 076 366 45 06 |

Le temps nécessaire au développement de certaines parties étant très incertains, je vais travailler avec une méthode agile. Avec le chef de projet, nous fixerons chaque semaine un nouveau sprint.

## Objectifs

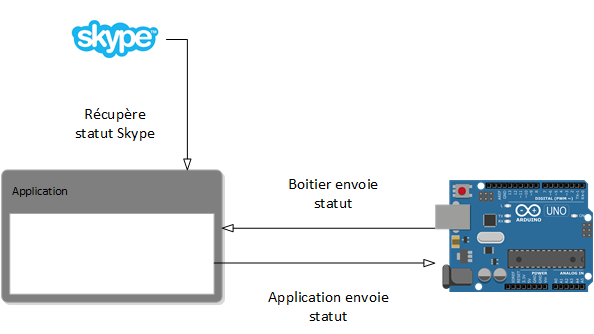
* Le boitier a une LED qui doit s’allumer en vert, orange ou rouge.
* Un bouton sur le boitier permet de changer la couleur.
* Une application permet de changer cette couleur.
* L’application peut communiquer en USB avec le boitier.
* L’application peut communique en Bluetooth avec le boitier.
* L’application a un paramètre pour changer entre USB et Bluetooth.
* L’application affiche la couleur de le LED (Si on change sur le boitier, ça change sur l’application).
* L’application peut connaitre le statut Skype et change la couleur en fonction.
* L’application peut changer le statut sur Skype.

## Planification initiale

Voir annexes.

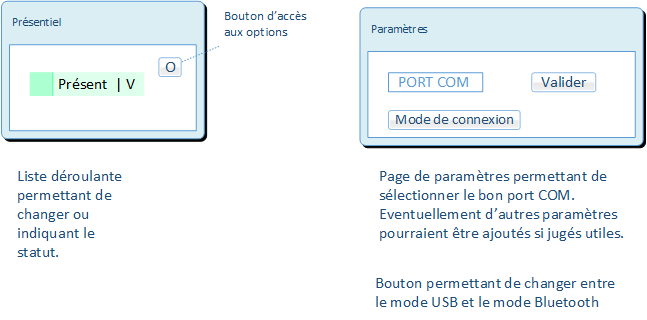
# Analyse / Conception

## Concept



Ce schéma montre les interactions entre les différents composants du projet. Lors d’un changement de statut côté boitier, l’application adapte son affichage et lors d’un changement côté application, le boitier change sa couleur et Skype modifie son statut. L’application récupère le statut Skype et se met à jour en fonction.

### Maquette schématique de l’application



L’application se composera d’une fenêtre principale avec l’indicateur de statut accompagné d’une pastille de la couleur correspondante et d’un bouton pour accéder aux option. L’indicateur sera un menu permettant de changer de statut.

Une fenêtre d’options contenant le choix du mode de connexion (USB ou Bluetooth) et un champ pour indiquer le port COM. D’autres options pourraient être nécessaire au bon fonctionnement du programme et seraient ajoutées ici.

### Cas d’utilisation

#### Boitier

|  |  |
| --- | --- |
| **Evénement** | **Action** |
| Bouton pressé | La LED passe à la couleur suivante (Bleu->Vert->Orange->Rouge->Bleu…). |
| Un message est envoyé à l’application en serial. |
| Message serial reçu (changement de statut côté application) | La LED passe à la couleur indiquée dans le message. |

#### Application

|  |  |
| --- | --- |
| **Evénement** | **Action** |
| Changement manuel du statut | L’indicateur change pour afficher le nouveau statut. |
| Message serial envoyé au boitier. |
| Mise à jour du statut Skype. |
| Changement des paramètres de connexion | Les paramètres sont enregistrés. |
| Réception d’un message du boitier | L’indicateur change pour afficher le nouveau statut. |
| Mise à jour du statut Skype. |
| Détection changement statut Skype (pas dû à l’application) | L’indicateur change pour afficher le nouveau statut. |
| Message serial envoyé au boitier. |

#### Erreurs

|  |  |
| --- | --- |
| **Evénement** | **Action** |
| Côté application : Skype pas connecté | Affiche un message l’indiquant |
| Côté application : Boitier pas connecté | Indique que le boitier n’est pas connecté. Une aide est disponible. |
| Côté boitier : Application pas connectée | Indique une couleur standby. |

## Stratégie de test

Des tests unitaires seront effectués tout au long du projet pour les différentes fonctionnalités. Des tests complets seront finalement effectués, portant sur le projet fini. Ces tests seront réalisés sur mon poste avec le prototype du boitier. Un test sur un autre ordinateur sera effectué.

### Fonctionnement

|  |  |
| --- | --- |
| **Test** | **Résultat attendu** |
| Presser le bouton du boitier.  *Effectuer le test en USB et en Bluetooth*.  *Effectuer le test avec Skype déconnecté.* | La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu…). |
| Le statut sur l’application change en fonction. |
| Presser le bouton du boitier.  *Effectuer le test en USB et en Bluetooth*.  *Effectuer le test avec Skype connecté.* | La LED change de couleur (Bleu->Vert->Rouge->Bleu…). |
| Le statut sur l’application change en fonction. |
| Le statut de Skype est changé en fonction. |
| Changer le statut sur l’application.  *Effectuer le test avec et sans Skype.*  *Effectuer le test en USB et en Bluetooth* | La LED prend la couleur correspondante. |
| Le statut sur l’application change en fonction. |
| Le statut de Skype est changé en fonction, si Skype est connecté. |
| Changer le statut sur Skype.  *Effectuer le test avec et sans le boitier.*  *Effectuer le test en USB et en Bluetooth* | La LED prend la couleur correspondante. |
| Le statut sur l’application change en fonction. |

### Erreurs

|  |  |
| --- | --- |
| **Test** | **Résultat attendu** |
| Déconnecter Skype. | Un message l’indique sur l’application |
| Déconnecter le boitier. | Un message indique que le boitier est déconnecté sur l’application. |
| Cliquer sur le message précédemment obtenu. | Une page d’aide s’affiche.  On peut la refermer. |
| Déconnecter l’application. | Le boitier indique la couleur de standby. |

### Paramètres de connexion

|  |  |
| --- | --- |
| **Test** | **Résultat attendu** |
| Cliquer sur le bouton de paramètres. | Une page de paramètres s’ouvre. |
| Changer le port COM et fermer la page.  *Effectuer le test en USB.* | Le nouveau port est enregistré. |
| La connexion est relancée. |
| Changer le port COM et fermer la page.  *Effectuer le test en Bluetooth.* | Le nouveau port est enregistré. |
| La connexion est relancée. |
| Cliquer sur le bouton de changement USB/Bluetooth. | Le mode de connexion est changé. |

## Planification

Voir annexes.

## Dossier de conception

### Hardware

Pour ce projet je dispose des éléments Hardware suivants :

* PC CPNV (8Go RAM, disque dur 512Go)
* Module Bluetooth USB
* Arduino UNO R3
* Module Bluetooth HC-06
* Barrette LED RGB (Adafruit NeoPixels)
* Bouton
* 1x 470Ω résistance
* 3x 1kΩ résistances
* 1x 10kΩ résistance
* 1x 1µF condensateur non polarisé

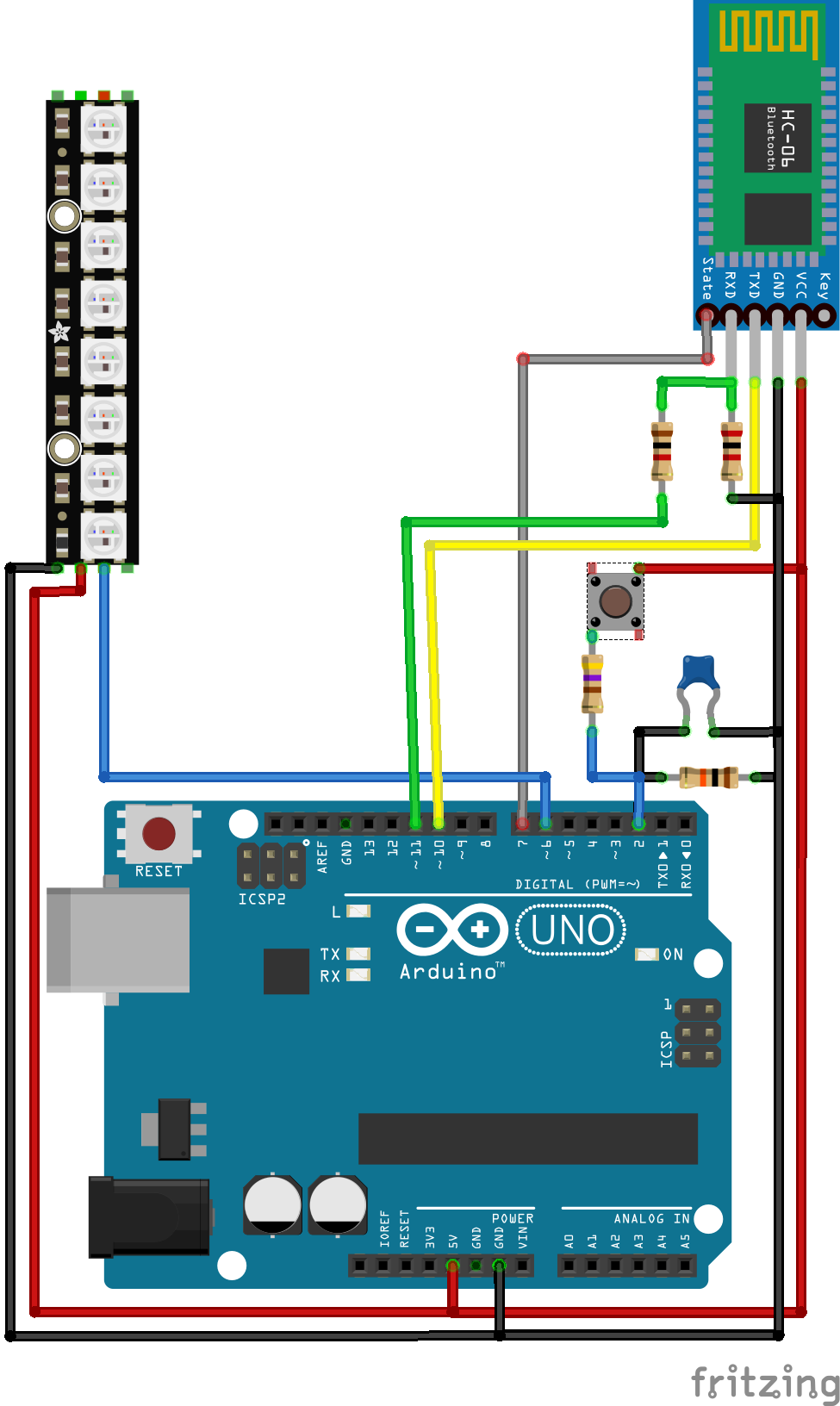
### Software

Je dispose d’une machine Windows 7 pour le développement et l’application finale tournera sur cet OS.

L’application sera programmée avec Visual Studio et le code du boitier sera fait avec l’IDE Arduino.

Skype for business sera utilisé en tant que partie du projet.

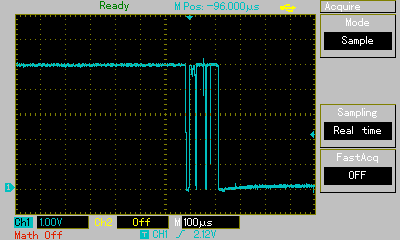
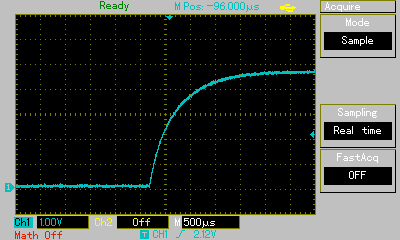
### Schéma de montage boitier



Le module Bluetooth HC-06 est connecté aux pins 10 et 11 car les pins rx/tx 0 et 1 sont utilisés par la connexion USB. La patte Rx du module ne supporte pas plus de 3.3v, c’est pour cela qu’elle est connectée avec un diviseur résistif utilisant trois résistances de 1kΩ. Il est important de noter que le Rx du module va vers le Tx de l’Arduino et le Tx du module va vers le Rx de l’Arduino.

La barrette de LEDs se connecte à l’alimentation, la terre et utilise un pin pour être contrôlée.

Le bouton est connecté à un pin et à l’alimentation ainsi qu’à une résistance 10kΩ en « pull-down » et un filtre passe bas pour filtrer les rebonds (voir images ci-dessous).



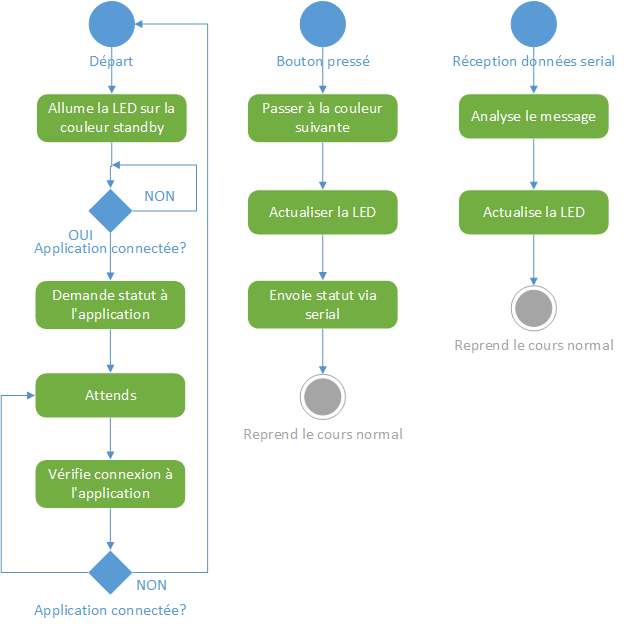
Avec filtre

Sans filtre

On voit bien ici les rebonds crées par le mécanisme du bouton (gauche) et l’effet du filtre anti-rebonds (droite). Les rebonds posent le problème suivant : étant donné qu’il y a plusieurs « montées », le programme compte comme plusieurs appuis sur le bouton.

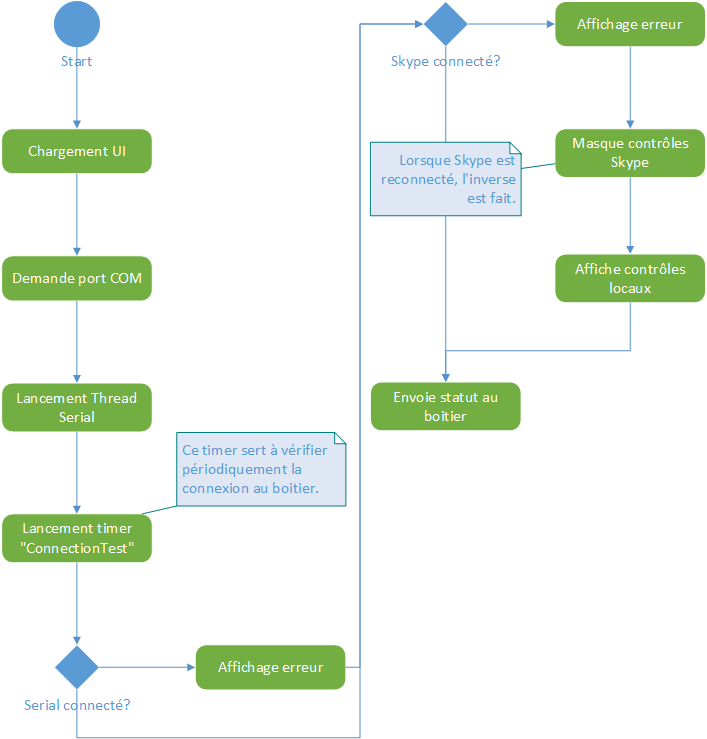
### Conception du code

#### Boitier



Lorsque le boitier s’enclenche, il s’allumera avec une couleur d’attente pouvant indiquer qu’il n’y a personne. Si le boitier ne détecte pas de connexion avec l’application, il reste ou revient sur cette couleur. Lorsque quelqu’un pèse sur le bouton ou que le boitier reçoit un message serial, l’attente s’interrompt et le boitier change de couleur.

#### Application



Le logiciel C# sera un projet Visual studio WPF car c’est dans ce type de projet que les contrôles de l’API de Skype for business sont présents. Ceux-ci permettent une interaction quasi-automatique entre l’application et Skype.

La partie principale du programme lance un thread qui gèrera le serial et un qui gèrera Skype. Puis il initialise le statut en fonction des choses connectées.

Les threads serial et Skype enverront des évènements lors de la réception d’un message ou d’un changement de statut.

Un timer est également lancé afin de vérifier périodiquement l’état des connexions.

# Réalisation

## Dossier de réalisation

Lors de ce projet, les divers documents et dossiers (projet de code, etc…) sont enregistrés sur le lecteur réseau dont je dispose et sont régulièrement synchronisés sur GitHub.

Le dossier est organisé de la manière suivante :

Doc Contient les documents utilisés pour la documentation.

Archives Contient des sauvegardes de la documentation.

Conception Contient les documents de conception (pseudo codes, schéma de montage, …)

Images Contient les images utilisées pour la documentation.

Planning Contient les fichiers pour le planning initial et suivants.

Documentation.docx Fichier Word de la documentation.

JournalTravail.xlsx Fichier Excel du journal de travail.

Code Contient les dossiers des projets de code.

Boitier Contient le projet Arduino IDE du boitier.

Presentiel Contient le projet Visual Studio de l’application

Légende : Dossier Fichier

### Projet Arduino

La programmation du boitier a été effectuée avec l’IDE Arduino (version 1.6.5). Le projet ne comporte qu’un fichier de code mais ce dernier utilise deux librairies :

* SoftwareSerial.h
* Adafruit\_NeoPixel.h

La première permet de gérer la communication serial et est donc utilisée pour la communication USB et Bluetooth.

La seconde permet de gérer la barre LED et est proposée par le fabriquant.

Concernant le boitier physique, le montage montré dans la partie conception a été suivi et un boitier imprimé en 3D a été créé sur mesure pour accueillir ce prototype.

L’Arduino doit être alimenté via le câble USB pour fonctionner. Ce dernier ne doit toutefois pas forcément être branché sur un ordinateur : Une batterie externe fonctionne parfaitement.

### Projet application

L’application a été développée en C# sous forme d’un projet WPF avec le logiciel Microsoft Visual Studio Entreprise 2017 (Version 15.5.7). Les différents fichiers et dossiers du projet sont organisés sous le forme suivante :

presentiel.sln

presentiel

properties Contient les settings utilisés.

referencies Contient les librairies importées.

Resources Contient les fichiers importés en tant que ressource.

utils Un dossier contenant les classes « utiles » du projet.

serialCom.cs La classe du gestionnaire de communication serial.

MainWindow.xaml Fichier de l’UI. Décrit les différents éléments graphiques.

MainWindow.xaml.cs Fichier du code lié à l’UI.

Légende : Dossier Fichier

Pour ce projet, j’ai importé la librairie suivante :

* Microsoft Lync Client SDK

Celle-ci est fournie par Microsoft mais il faut éditer le fichier d’installation fourni car ce dernier ne reconnait pas Skype for Business en tant que Lync.

En plus de cette librairie, j’utilise une partie des références de base de System et System.Windows. notamment Threading et IO.Ports pour la communication.

## Description des tests effectués

### Méthode

Les tests suivants ont été réalisés sur le PC de développement avec le prototype de boitier ainsi que l’application lancée depuis Visual Studio. Pour les tests en connexion Bluetooth, le boitier est branché à une batterie externe au lieu de PC pour s’assurer que ce soit bien ce mode de présentation qui soit utilisé. Des images illustrent les différents états résultants du test effectué.

### Presser le bouton

**Eléments connectés :** Boitier, Application, Connexion USB

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).
* Le statut de l’application prend la même couleur et le statut correspondant.

**Résultat obtenu :**

* Le boitier se comporte correctement.
* L’application se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Eléments connectés :** Boitier, Application, Skype, Connexion USB

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert ->Rouge->Bleu->…).
* Le statut de l’application prend la couleur et le statut correspondants (Vert->Rouge->Rouge->Vert->…) Bleu n’étant pas disponible pour Skype.
* Le statut Skype correspond à l’application.

**Résultat obtenu :**

* Le boitier se comporte correctement.
* L’application se comporte correctement.
* Skype se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Eléments connectés :** Boitier, Application, Connexion Bluetooth

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).
* Le statut de l’application prend la même couleur et le statut correspondant.

**Résultat obtenu :**

* Le boitier se comporte correctement.
* L’application se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Eléments connectés :** Boitier, Application, Skype, Connexion USB

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert ->Rouge->Bleu->…).
* Le statut de l’application prend la couleur et le statut correspondants (Vert->Rouge->Rouge->Vert->…) Bleu n’étant pas disponible pour Skype.
* Le statut Skype correspond à l’application.

**Résultat obtenu :**

* Le boitier se comporte correctement.
* L’application se comporte correctement.
* Skype se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

### Changer le statut sur l’application

**Eléments connectés :** Boitier, Application, Connexion USB

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).

**Résultat obtenu :**

* Le boitier se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Eléments connectés :** Boitier, Application, Skype, Connexion USB

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).
* Le statut Skype correspond à l’application.

**Résultat obtenu :**

* Le boitier se comporte correctement.
* Skype se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Eléments connectés :** Boitier, Application, Connexion Bluetooth

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).

**Résultat obtenu :**

* Le boitier se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Eléments connectés :** Boitier, Application, Skype, Connexion Bluetooth

**Résultat attendu :**

* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).
* Le statut Skype correspond à l’application.

**Résultat obtenu :**

* Le boitier se comporte correctement.
* Skype se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

### Changer le statut Skype

**Eléments connectés :** Boitier, Application, Skype, Connexion USB

**Résultat attendu :**

* Le statut de l’application correspond au statut Skype.
* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).

**Résultat obtenu :**

* L’application se comporte correctement
* Le boitier se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Eléments connectés :** Boitier, Application, Skype, Connexion Bluetooth

**Résultat attendu :**

* Le statut de l’application correspond au statut Skype.
* La LED change de couleur (Bleu->Vert->Orange->Rouge->Bleu->…).

**Résultat obtenu :**

* L’application se comporte correctement
* Le boitier se comporte correctement.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

### Déconnecter Skype

**Eléments connectés :** Boitier, Application, Connexion USB

**Résultat attendu :**

* L’application indique que Skype est déconnecté.

**Résultat obtenu :**

* L’application se comporte correctement

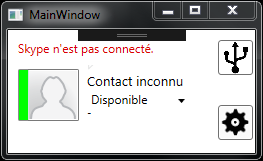
**Eléments connectés :** Boitier, Application, Connexion Bluetooth

**Résultat attendu :**

* L’application indique que Skype est déconnecté.

**Résultat obtenu :**

* L’application se comporte correctement



### Déconnecter le boitier

**Eléments connectés :** Application, Connexion USB

**Résultat attendu :**

* L’application indique que le boitier est déconnecté.

**Résultat obtenu :**

* L’application se comporte correctement

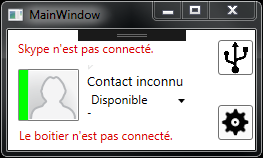
**Eléments connectés :** Application, Connexion Bluetooth

**Résultat attendu :**

* L’application indique que le boitier est déconnecté.

**Résultat obtenu :**

* L’application se comporte correctement



### Déconnecter l’application

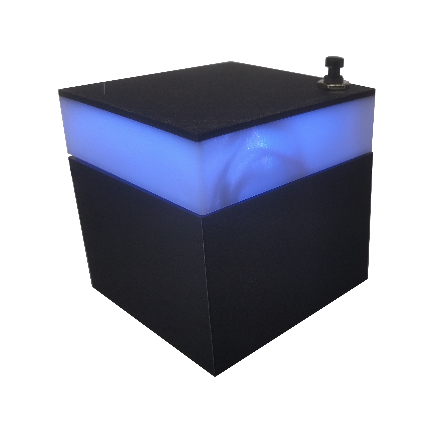
**Eléments connectés :** Boitier

**Résultat attendu :**

* La couleur passe au bleu (couleur stand-by).

**Résultat obtenu :**

* Le boitier se comporte correctement : Change sa couleur après quelques secondes de délai. (Image à droite)



### Cliquer sur le message « boitier déconnecté »

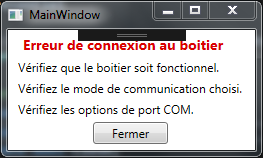
**Eléments connectés :** Application

**Résultat attendu :**

* Une page avec des informations s’affiche.
* Cette page se ferme lorsqu’on clique sur son bouton.

**Résultat obtenu :**

* L’application se comporte correctement.



### Ouvrir la page de paramètres

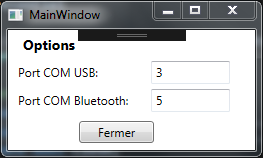
**Eléments connectés :** Application

**Résultat attendu :**

* La page des paramètres s’ouvre.
* Cette page se ferme lorsqu’on clique sur son bouton.

**Résultat obtenu :**

* L’application se comporte correctement.



### Changer les paramètres

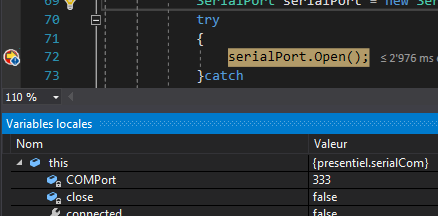
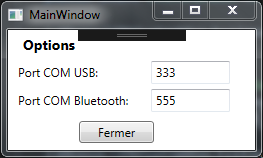
**Eléments connectés :** Boitier, Application, Connexion Bluetooth/USB

**Résultat attendu :**

* Le changement est enregistré.
* Le changement s’applique à la communication serial.

**Résultat obtenu :**

* L’application se comporte correctement.

****

On voit ici que le paramètre est bien appliqué au relancement de la communication qui intervient à la fermeture de la page de paramètres. En revenant sur cette page, les paramètres restent inchangés (333 et 555).

## Erreurs restantes

Lorsque l’on survole l’un des boutons de l’interface principale, l’animation de focus se déclenche. Cela a pour effet de changer le fond par un dégradé et donc de masquer le logo. Cette erreur n’a qu’une conséquence visuelle et aucune incidence sur le fonctionnement technique du produit.

Pour résoudre ce problème, il faudrait créer un style personnalisé pour ces boutons soit reprenant l’ensemble du style existant sans le focus et écrasant le style par défaut soit en créant un style contenant une nouvelle animation n’écrasant pas le style par défaut. Dans les deux cas cela devrait être assez long à mettre en place. Compte tenu du faible impact de cette erreur, elle n’a pas été corrigée.

## Liste des documents fournis

Le produit comprend les parties suivantes :

* Boitier
* Application C#

Les documents suivants sont livrés avec le produit :

* Le présent rapport
* Un résumé
* Un manuel

Sont aussi livré sur un CD :

* Le code du boitier
* Le code de l’application
* Les documents de travail

# Conclusions

Lors de ce projet j’ai eu l’occasion d’améliorer mes connaissances en programmation embarquée, en C# ainsi qu’en électronique, notamment en ayant l’occasion d’utiliser un oscilloscope pour voir les rebonds, et de rafraichir d’autres notions vues en début de formation.

J’ai réussi à atteindre tous les objectifs fixés et l’ensemble se comporte de manière correcte et fluide.

Il m’a cependant été compliqué d’installer et de faire fonctionner le SDK Lync client du fait des restrictions dues au CPNV concernant l’installation et l’utilisation de Skype for business. Une autre difficulté aura été de prendre un main l’environnement WPF de Visual Studio tout de même bien différent de Windows Forms que j’avais l’habitude d’utiliser.

Dans son état actuel le projet est un prototype. Plusieurs choses pourraient être faites pour le rendre bien mieux.

Le plus évident serait de faire un circuit imprimé personnalisé et bien moins volumineux avec un nouveau boitier.

Une autre idée serait de permettre une configuration complète. Pouvoir choisir quelle couleur mettre pour quel statut, etc…

Finalement on pourrait rendre l’application compatible avec d’autres services de communications utilisés en entreprise.

# Annexes

Les documents suivants se trouvent en annexe :

* Résumé du rapport
* Manuel d’installation et d’utilisation
* Journal de travail
* Planifications initiale et finale

## Sources – Bibliographie

Documentation Lync Client SDK :

<https://docs.microsoft.com/en-us/lync/desktop/lync-2013-sdk-documentation>

Installation de Lync Client SDK avec Skype for business installé :

<https://www.eshlomo.us/skype-for-business-2016-sdk-and-lync-2013-sdk/>

Récupérer le statut de présence Skype :

<https://blog.thoughtstuff.co.uk/2015/02/pausing-lync-when-on-a-call-using-the-lync-client-sdk/>

Mettre à jour le statut de présence sur Skype :

<https://blog.thoughtstuff.co.uk/2016/06/skypedevq-updating-skype-for-business-presence-client-sdk/>

Multithreading pour communication Serial (voir solution 2) :

<https://www.codeproject.com/Questions/228124/serial-port-using-threading>

Création d’un timer :

<https://stackoverflow.com/questions/11559999/how-do-i-create-a-timer-in-wpf>

Lier les paramètres de l’application à un champ texte :

<http://labloguera.net/javier/2011/09/binding-wpf-properties-to-applicationsettings-in-c/>

Gestion d’un bouton sur Arduino :

<https://www.arduino.cc/en/Tutorial/Button>

Guide pour la barre de LED :

<https://learn.adafruit.com/adafruit-neopixel-uberguide/basic-connections>

J’ai aussi obtenu quelques informations de la part de M. Ithurbide et de M. Favre.

## Glossaire

|  |  |
| --- | --- |
| Arudino | Marque de microcontrôleurs produisant le microcontrôleur UNO R3 |
| API | Application Programming Interface : Série de protocoles et outils permettant la communnication entre des éléments de software |
| Bluetooth | Standard de transfert de données sans fil. |
| C# | Language de programmation orienté objet conçu et développé par Microsoft |
| Condensateur | Composant électronique qui stocke de l'énergie potentielle. |
| GitHub | Plateforme de développement permettant de mettre en ligne des projets logiciels. GitHub est leader dans le domaine. |
| IDE | "Integrated Developpement Environnement" Application fournissant un ensemble d'outils permettant le développement de logiciels. Un IDE contient normalement un éditeur de code source, un compilateur et un débuggeur. |
| LED | « Light Emitting Diode » Diode (Diode : Composant électronique à deux pattes qui conduit le courant dans une direction) qui émet de la lumière. |
| Patte | Point de connection d'un composant électrique. |
| Pin | Point d’entrée pour le contrôleur. |
| Port Com | Type de port série respectant le standard RS-232. |
| Résistance | Composant électronque qui rend plus difficile le passage du courant. |
| RGB | Red Green Blue (Rouge Vert Bleu) |
| SDK | Software developpement kit : Packet contenant des outils permettant de développer des applications. |
| Skype | Logiciel de communication en ligne proposé par Microsoft. |
| UI | User Inteface (Interface utilisateur) |
| Visual Studio | IDE de microsoft |
| WPF | Librairies permettant le développement d'UI pour la plateforme Windows. |

## Archives du projet

L’entier des documents de travail sont à disposition sur le CD joint. Ce dernier contient la documentation et ses annexes, les documents de conception ainsi que le code du boitier et de l’application.