

4A – IT option SQR

Semestre S7

Communication sans fil

COMPTE RENDU DU TP DE COMMUNICATION SANS FIL

Réalisé par :

EL AMRI Mohamed.

CHEIKHROUHOU Yacine.

Encadré par :

Pr. Vincent Thivent

Année Universitaire : 2022/2023.

1. Introduction :

L'objectif de ces TP est d'implémenter une interface graphique permettant d'utiliser une carte MiFare Classic et son lecteur sans fil avec des clés de sécurité préconfigurées.

Pour cela, on a utilisé l'étude des références techniques faite en TD, une librairie C/C++, et l'éditeur d'interface graphique Qt Creator.

Les blocs de la carte utilisés servent à contenir un nom, un prénom et un compteur.

2. Fonctionnement de la carte :

La mémoire de la carte RFID est composée de secteurs, qui sont chacun composés de blocs.

On utilise trois blocs de données : le nom, le prénom et un compteur.

Chaque secteur utilise 2 clés (A et B). La clé A sert pour la lecture, et la clé B pour l'écriture.

Pour lire un bloc, il faut utiliser les clés du secteur dans lequel il se trouve. Les clés en question sont déjà stockées dans le lecteur RFID.

En TD, nous avons vu le détail des formats sous lesquels sont stockés les valeurs entières et les chaînes de caractères.

Cependant, la librairie fournie contient déjà des fonctions qui décodent ces formats.

3. Méthode de travail :

3.1 Outils :

Nous avons utilisé Git pour le versionnage et Qt Creator pour l'interface graphique.

Les outils de développement étaient déjà installés et préconfigurés, ce qui nous a aidé à démarrer le projet très rapidement.

Nous avons également accès à une carte MiFare Classic préconfigurée et un lecteur connecté par un câble USB. Le fait d'avoir le matériel nécessaire en accès libre nous a permis de rapidement tester les effets du code, et de voir les bugs dès leur apparition.

4. Stratégie :

Pendant le développement, nous nous sommes inspirés des « sprints » de la méthodologie agile. À chaque séance, on se relayait pour faire chacun un « sprint » d'une heure environ. Pendant ce temps, l'autre membre suivait le code et se préparait à utiliser au mieux l'accès au travail.

Nous avons utilisé cette approche pour utiliser au mieux les séances limitées, en maintenant une efficacité de travail constante sans perdre de vue l'objectif global ni perdre de temps lors de la résolution des conflits entre les commits git.

Avec cette approche, nous avons réussi à fournir un travail intense et régulier, et à finir le projet avant la fin des séances prévues.

5. Apprentissages/Acquis :

5.1 Obstacles rencontrés :

Dans ce projet, l'obstacle principal a été de se familiariser avec la bibliothèque. Nous avons eu pour cela des séances de TD, mais la documentation de la bibliothèque n'était pas forcément évidente pour un débutant dans le domaine des cartes RFID.

Cependant, une fois que nous avons compris l'usage des fonctions dont nous avons besoin, notamment avec les exemples de code fournis par le professeur et le travail réalisé en TD, le développement de l'application est devenu beaucoup plus facile.

Un autre obstacle que nous avons rencontré était la conversion entre chaînes de caractères Qt et tableaux statiques de caractères. Pour cela, nous avons principalement utilisé l'interface objet de la classe QString.

Le reste de l'application a été développé dans un style événementiel, en associant à chaque bouton le comportement attendu.

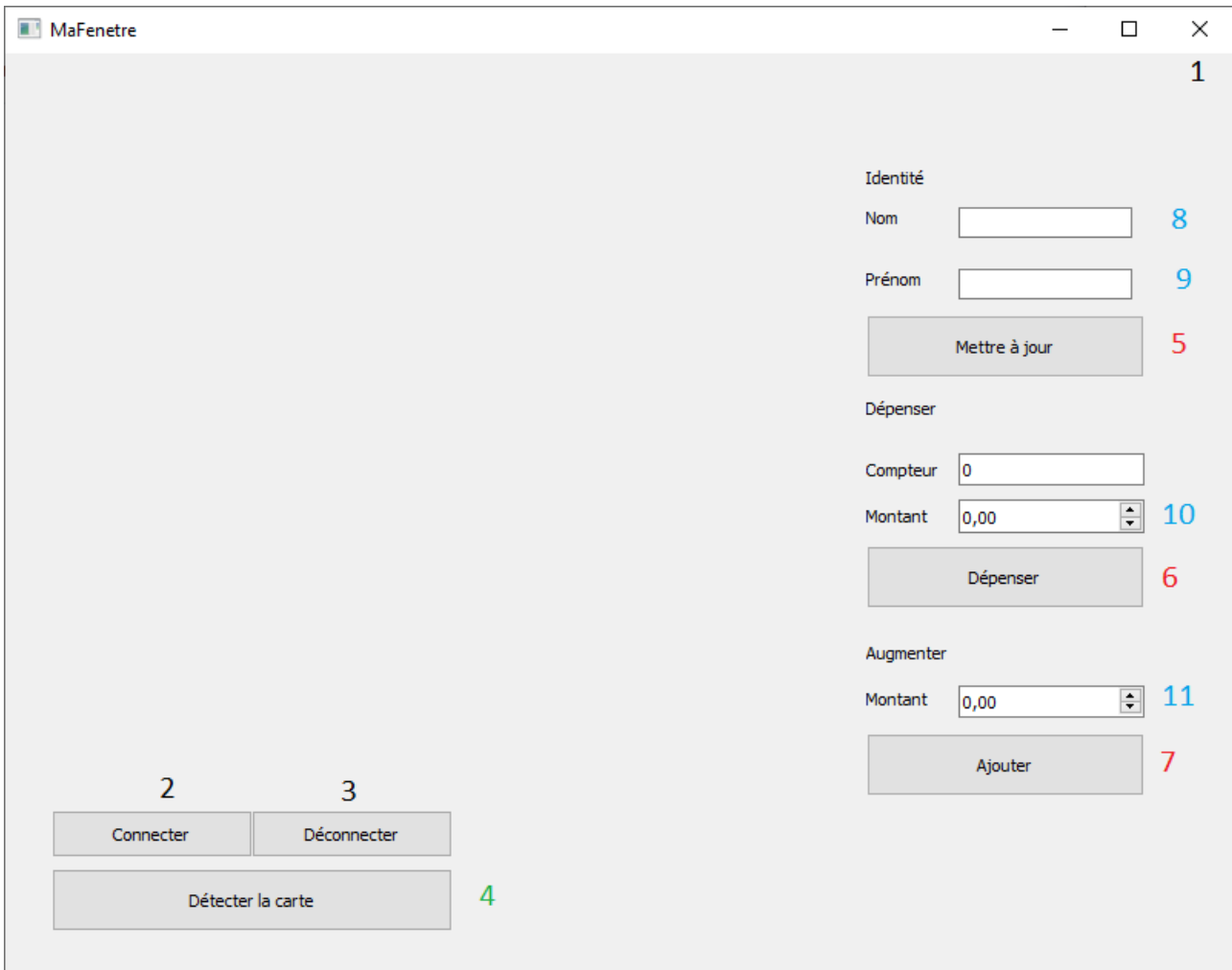
6. Résultats atteints :

Notre application est complètement fonctionnelle et inclut des gardes pour que n'importe qui puisse s'en servir sans problème.

En particulier, nous avons lié la fonction de déconnexion à l'évènement de fermeture du programme, et inclus une sécurité pour se déconnecter automatiquement avant de se reconnecter (lorsque la fonction d'ouverture de session est appelée plusieurs fois, la connexion devient inutilisable)

Cependant, il aurait été possible d'ajouter des options d'ergonomie, comme par exemple connecter le lecteur automatiquement lorsque l'utilisateur souhaite lire une carte, ou surveiller en permanence la présence d'une carte à lire.

7. Structure de l'application :



The screenshot shows a window titled 'MaFenetre' with the following elements and callouts:

- 1**: Close button (X) in the top right corner.
- 2**: 'Connecter' button.
- 3**: 'Déconnecter' button.
- 4**: 'Détecter la carte' button.
- 5**: 'Mettre à jour' button.
- 6**: 'Dépenser' button.
- 7**: 'Ajouter' button.
- 8**: 'Nom' text input field.
- 9**: 'Prénom' text input field.
- 10**: 'Montant' spinner box for the 'Dépenser' section.
- 11**: 'Montant' spinner box for the 'Augmenter' section.

Figure 1 : interface graphique.

7.1 Légende :

Contrôle global :

- 1 = Quitter l'application (après déconnexion)
- 2 = Connecte (ou reconnecte) l'application au lecteur
- 3 = Déconnecte le lecteur
- 4 = Lecture de la carte :

4 = Lit les données de la carte si elle est présente et les affiche sur l'interface

Écriture sur la carte

5 = Écrire le nom et le prénom sur la carte

6 = Décrémenter du montant la valeur sur la carte et rafraîchir le compteur

7 = Incrémenter du montant la valeur sur la carte et rafraîchir le compteur

Champs de données :

8 = Nom (lu ou à écrire)

9 = Prénom

10 = Montant à retirer du compteur

11 = Montant à ajouter au compteur

8. Diagramme de cas d'utilisation :

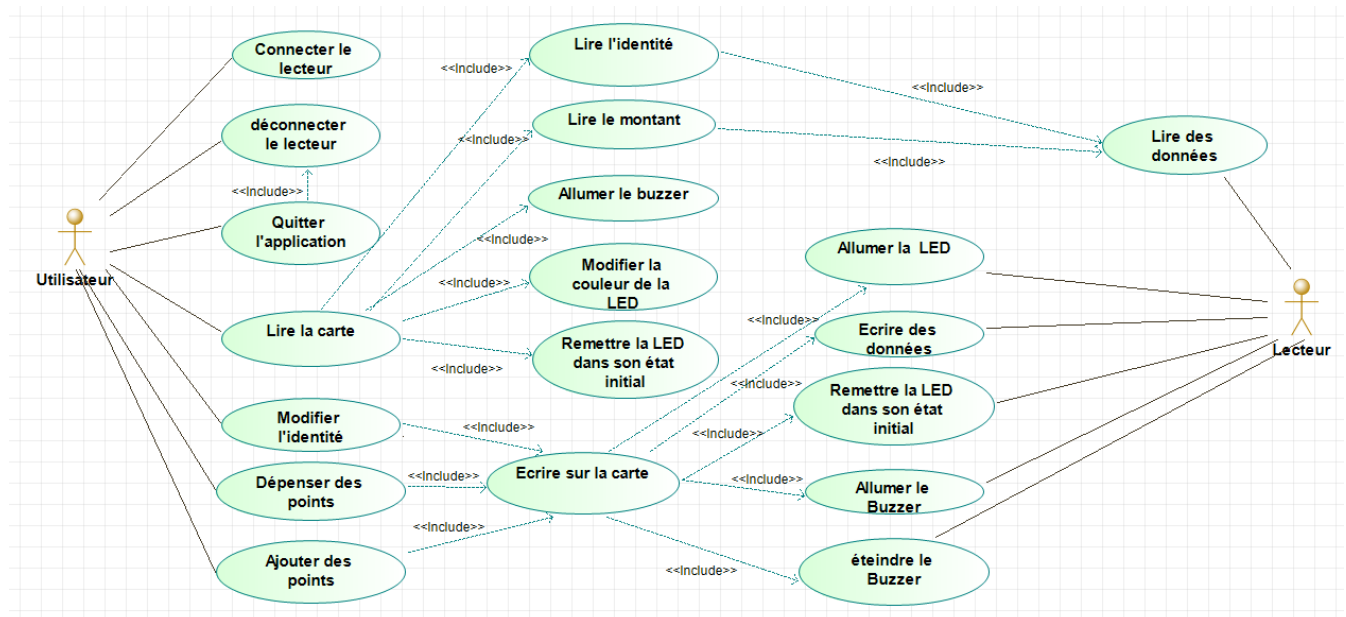


Figure 2 : Diagramme de cas d'utilisation de l'interface graphique

9. Architecture du code :

Connexion :

- OpenCom
- LEDBuzzer
- Version
- RF_Power_Control

Déconnexion/Quitter :

- Close COM
- RF_Power_Control

Lire la carte:

- Mf_Classic_Read_Value
- Mf_Classic_Read_Block
- LED Buzzer
- ISO14443_3_3_A_Pollcard

Dépenser :

- Mf_Classic_Decrement_Value
- Lire la carte

Ajouter:

- Mf_Classic_Increment_Value
- Lire la carte

Mettre à jour l'identité ;

- Mf_Classic_Write_Block.

Conclusion

Dans ce projet, nous avons utilisé une librairie de haut niveau qui permet de communiquer avec un lecteur de carte via un protocole d'interactions bas niveau.

Nous avons pu voir à quel point l'existence d'outils de développement adaptés peuvent faciliter la résolution de problèmes complexes, en se détachant des spécificités du matériel pour penser en terme de fonctionnalités et se concentrer sur le problème immédiat.

Nous avons également découvert que les langages C/C++ permettent d'adapter ses outils, notamment en modifiant la librairie pour corriger des éventuels bugs, comme par exemple les constantes associées aux codes couleurs de la LED. De plus, ils permettent aussi de développer ses propres outils, puisque la librairie que nous avons utilisé est elle-même écrite en C.

L'approche de la décomposition d'un problème en des couches haut niveau et bas niveau peut, dans un problème qui à première vue semble impossible à résoudre, permettre de résoudre facilement les différents aspects de ce problème.

Nous nous également familiarisés avec le domaine de la communication RFID et les protocoles qui sont utilisés dans ce domaine. Ce projet nous a permis non seulement de comprendre l'architecture spécifique au modèle MIFARE Classic, mais aussi les types de protocoles et de fonctionnement qu'on peut trouver dans les autres modèles de carte, tels que l'usage de cryptographie, le besoin de passer par des formats de données spécifiques et les sessions de communication qui permettent de contrôler une carte RFID.