

Programmation Orientée Objet

Bureau d'étude C/C++

Souleymane BAH

Chadi AMOUR

4AE – SE – Groupe 2

Présenté à :

Mr. Thierry Monteil

Mr. Raphaël Deau

29/05/2020

Système de contrôle d'accès connecté



C/C+

Table des matières

Introduction	1
I. Mise en contexte et positionnement du problème	2
a. Compréhension du système	2
b. Bilan des entrées et sorties du système de contrôle connecté.....	3
II. Compréhension du système et des bibliothèques développées.....	6
a. Diagramme de classes.....	6
b. Diagramme d'activités	8
c. Bibliothèques développées	9
III. Retours personnels sur le déroulement du projet	10
a. Bilan des objectifs	10
b. Compétences acquises	12

Introduction

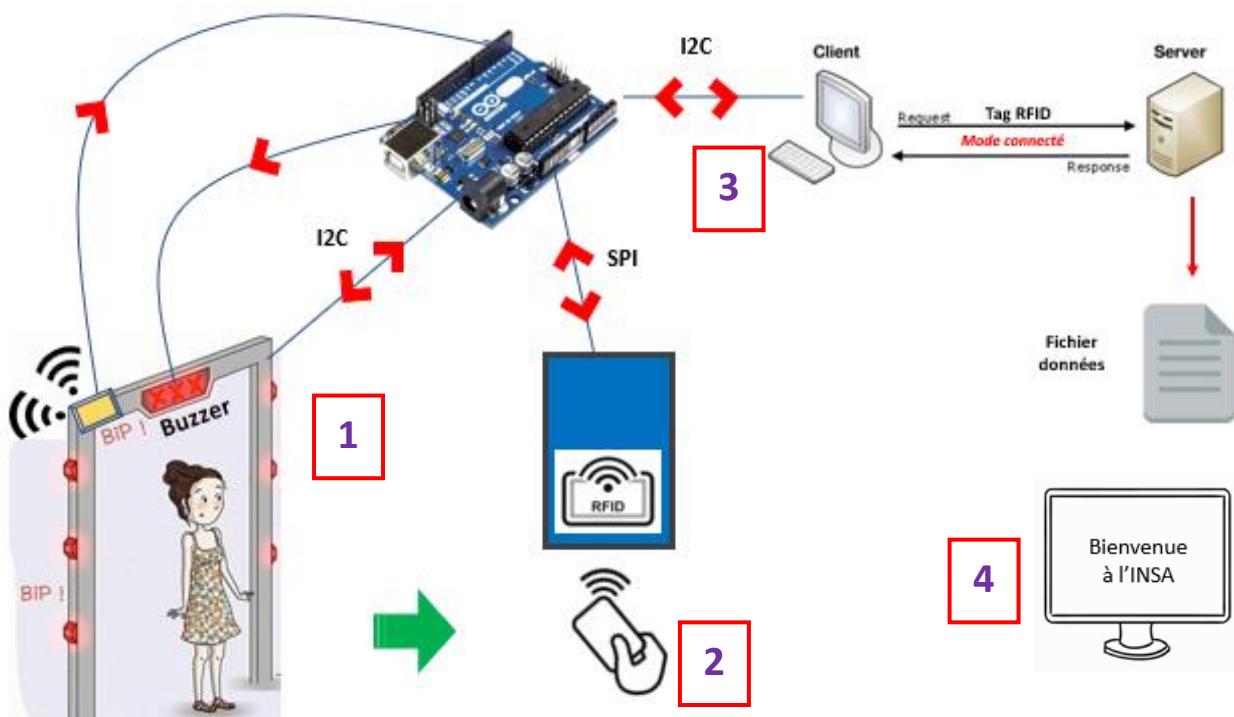
Dans la continuité de notre formation en systèmes embarqués, à l’Institut National des Sciences Appliquées de Toulouse (INSA), nous avons réalisé un bureau d’étude en Programmation Orienté Objet (POO). L’objectif est de programmer en langage C/C++ le système de notre choix, à l’aide des compétences acquises lors des Travaux Pratiques. Pour cela, nous utilisons une carte Arduino, élément coordonnant l’ensemble de notre application. Nous avons ainsi fait le choix d’implémenter un **système de contrôle d'accès connecté** dont nous expliquerons le fonctionnement dans un premier temps. Nous proposerons ensuite des diagrammes de classes et d’activités mettant en avant les différentes relations qui régissent notre application. Cette seconde partie s’accompagnera également d’un diagramme de séquences pour illustrer le comportement du système vis-à-vis des utilisateurs. Enfin nous terminerons ce compte-rendu en effectuant un retour d’analyse sur le déroulement du projet.

Nous tenons à remercier l’ensemble du corps des enseignants qui a fait preuve d’une grande disponibilité pour nous aider à réaliser ce projet.

I. Mise en contexte et positionnement du problème

a. Compréhension du système

Nous pourrions imaginer notre système à l'entrée du bâtiment de l'INSA. Le scénario est le suivant :



1. Les étudiants passent tour à tour devant une porte-détecteur de métaux qui transmet à la carte Arduino, le type de métal détecté. Sur cette porte, nous avons intégré un buzzer pour sonner en cas de détection de métaux ainsi que des capteurs de présence visibles en jaune pour détecter l'arrivée des étudiants. Si l'étudiant détient un métal, nous le signalons par le buzzer et également sur un terminal connecté à l'Arduino, et supposons que cet étudiant pourrait se faire contrôler par un agent de sécurité.
2. Une fois la porte passée, les étudiants doivent passer un badge RFID (Radio Frequency Identification) devant le lecteur RFID. Plus précisément, dans le badge RFID est contenue une séquence d'informations appelée tag RFID, correspondant à une sorte d'identifiant unique et propre à chaque étudiant. Le lecteur RFID, recueille ce tag via des ondes électromagnétiques et transmet la séquence à la carte Arduino via un bus SPI.

- 3.** La carte Arduino fait appel alors aux services d'un PC client en lui transmettant la séquence RFID à faire analyser. Ce PC client se connecte au serveur distant (nous avons imaginé une situation réelle). Ce serveur, en s'appuyant sur une base de données, confirme au PC client si oui ou non le tag RFID fait parti de ceux enregistrés dans les fichiers du serveur. Si l'étudiant ne possède pas un tag RFID valide, nous supposons qu'il est pris en charge par un agent.
- 4.** Si l'étudiant a validé l'ensemble de ces étapes, nous lui souhaitons la bienvenue à l'INSA sur un écran connecté à l'Arduino, via un bus I2C.

b. Bilan des entrées et sorties du système de contrôle connecté

Voici un tableau récapitulatif du système global :

	Entrées	Sortie	Communicants *
Capteur présence	✓	✗	✗
Détecteur métaux	✗	✗	✓
Lecteur RFID	✗	✗	✓
PC Client	✗	✗	✓
Interrupteur (contrôlé par le PC Client)	✓	✗	✗
Buzzer	✗	✓	✗
Screen	✗	✓	✗
Terminal	✗	✗	✓

* : par systèmes communicants, nous signifions système qui envoient et reçoivent à la fois des données avec la carte Arduino via des bus.

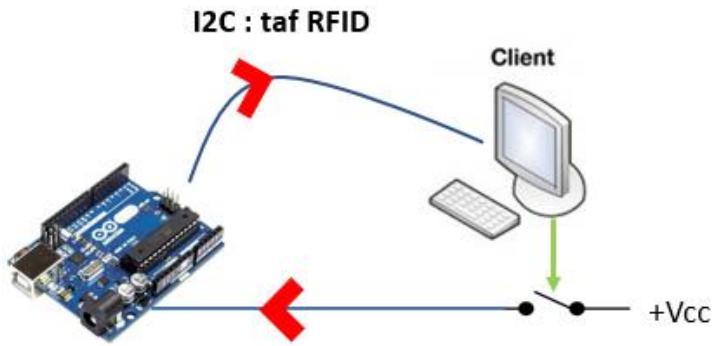
Nous allons tenter de justifier le tableau ci-dessus, en s'intéressant de plus près aux interactions présentes entre les différents composants.

- **Lecteur RFID** : Nous avons mis en place une communication SPI avec la carte Arduino sur la base de registres MISO (Master Input Slave Output), MOSI (Master Output Slave Input) et d'une entrée SCK. Nous avons simplifié cette communication où l'Arduino représente le maître (Master) et le lecteur RFID l'esclave (Slave) :



Nous ne considérons pas la ligne « SS » qui représente en temps normal l'adresse de chaque système esclave connecté à l'Arduino. Comme il n'y a qu'un seul et unique lecteur RFID, nous n'avons pas besoin de manipuler cette ligne. Au cours du programme, l'Arduino va venir régulièrement demander au lecteur RFID, le tag RFID qu'il a détecté (pour ensuite transmettre cette séquence vers le PC Client). Plus précisément, la carte Arduino, vient placer la ligne SCK à 1 qui la relie au lecteur, nous pouvons traduire cela par : « Parle lecteur RFID, je t'écoute, quelle est la séquence RFID que tu as détectée ? ». C'est donc uniquement l'Arduino qui a le contrôle de la ligne SCK, le lecteur RFID vient seulement lire sa valeur et répondre en écrivant sur le bus MISO.

- **PC Client** : Ce dernier dispose d'un bus I2C relié à celui de l'Arduino. Notamment, l'Arduino écrit dans ce bus, le tag RFID à faire analyser par le Serveur, comme si nous envoyions un échantillon à faire analyser au laboratoire. Plus précisément, il suffira que l'Arduino envoie le message de la forme : « Server séquence_RFID », et là le PC Client comprendra qu'il faut invoquer les services du serveur. Pour des raisons de synchronisation, nous pouvons tomber sur une situation où le PC Client écrit dans le bus, suivie d'une écriture sur ce même bus par l'Arduino. Pour cette raison, nous avons décidé que la carte sera l'émetteur et le PC Client le destinataire de ce bus I2C. Cependant, un problème vient s'interposer si nous raisonnons de cette manière : comment le PC Client peut avertir la carte de la réponse qu'il a reçu du serveur ? Nous avons donc intégré un interrupteur connecté entre le PC Client et l'Arduino. Lorsque le PC Client obtiens, la validation par le Serveur distant pour la séquence RFID, il doit pouvoir le signaler à l'Arduino. Voici une illustration de la relation : Arduino – PC Client



- **Détecteur de métaux :** Ce dernier est lui aussi doté d'une ligne I2C car il nous faut un moyen de transmettre à l'Arduino le nom du métal détecté. Nous pourrions très bien imaginer un microcontrôleur intégré à la porte détectrice de métaux.

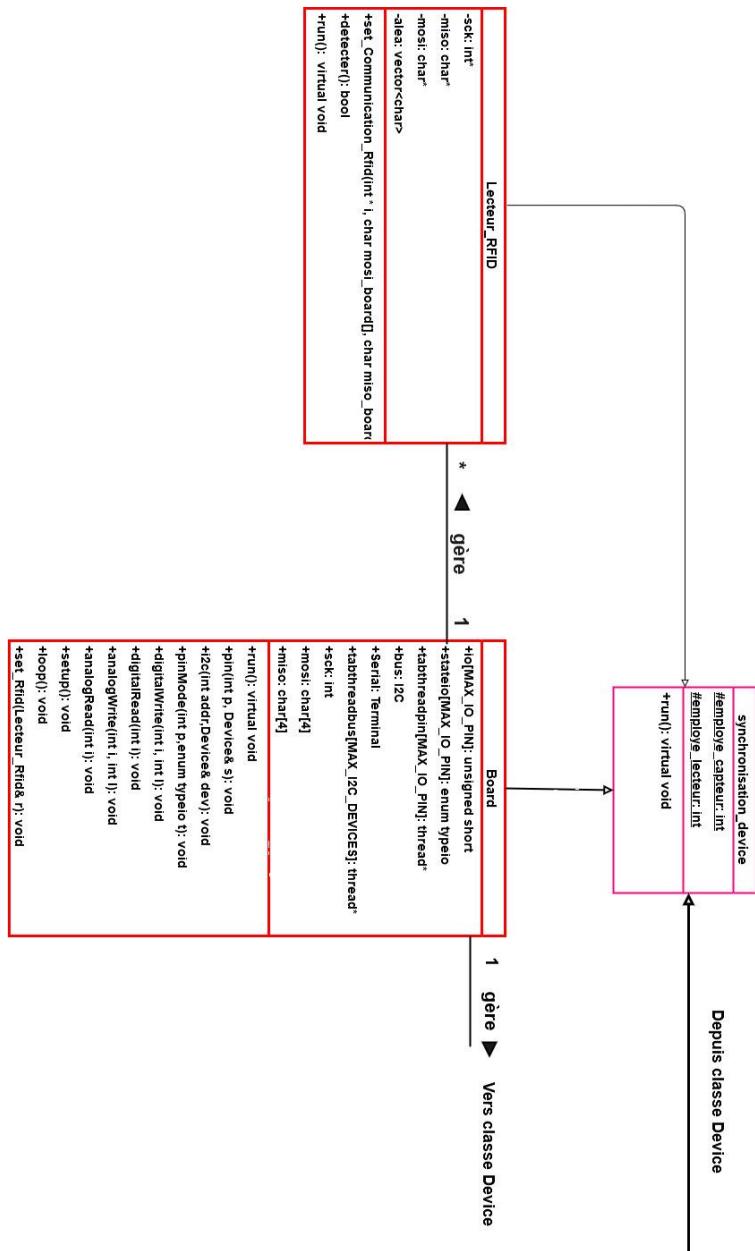
En vue de mieux de comprendre l'ensemble du système, nous allons dans un deuxième temps, nous intéresser aux diagrammes de séquences et d'activités. Ce qui nous permettra d'approcher le fonctionnement de notre application avec un point de vue différent.

II. Compréhension du système et des bibliothèques développées

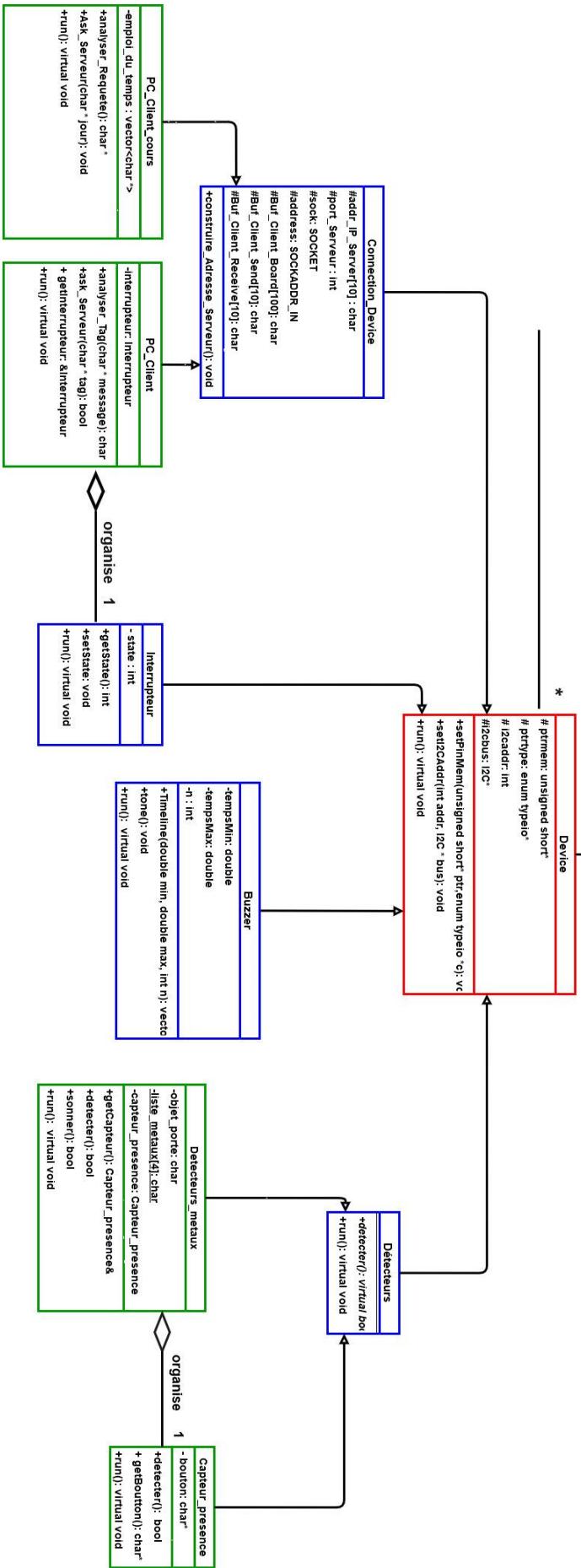
a. Diagramme de classes

Intéressons-nous ci-dessous au diagramme de classe général caractérisant le système (*nous conseillons de voir les fichiers jpeg qui sont beaucoup plus nettes*).

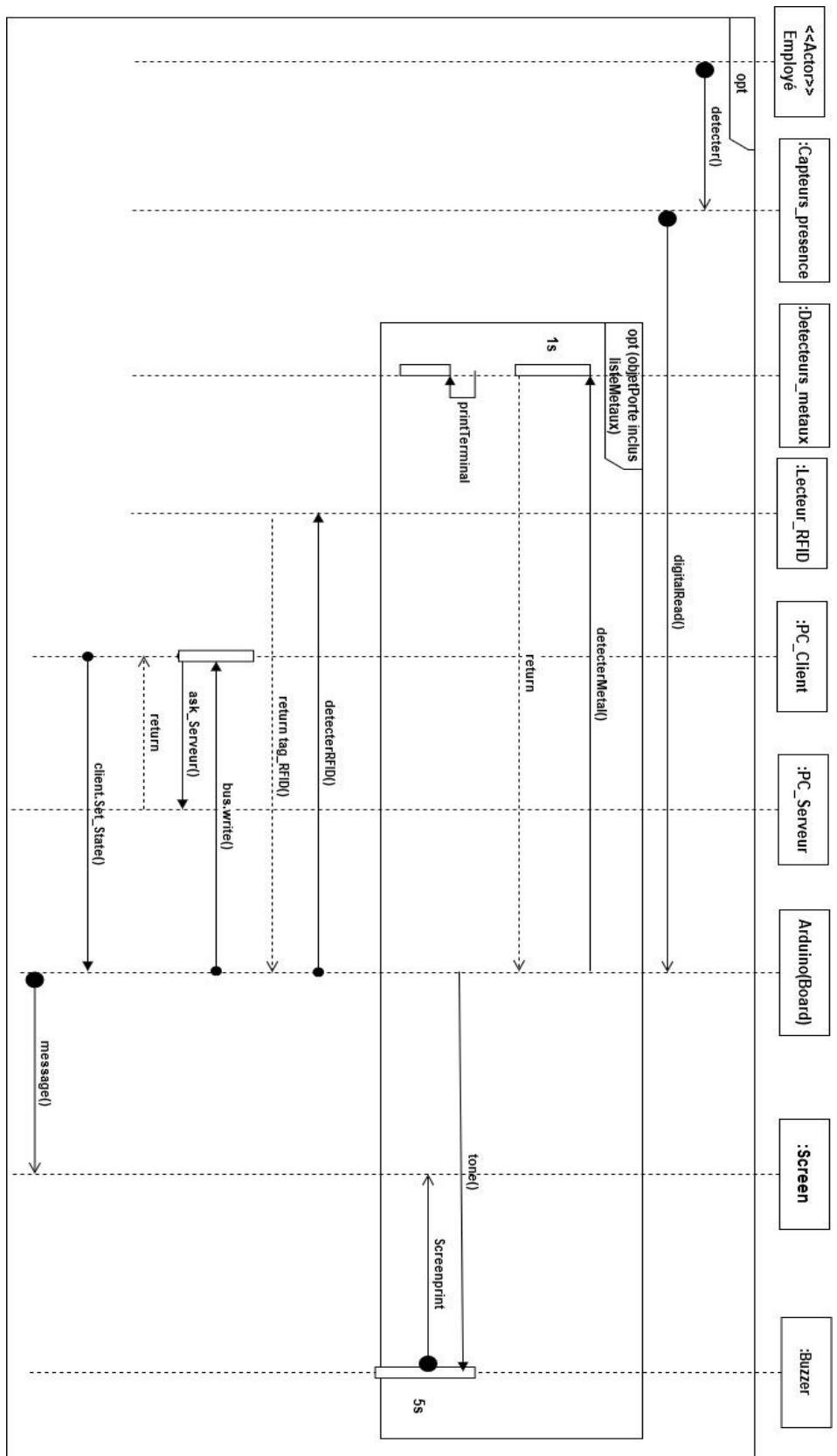
Remarque : par manque de temps, nous n'avons pas pu développer la classe « *synchronisation_device* ». Mais en nous plaçant dans la situation d'apprenti-ingénieurs, nous tenions à proposer une classe solution pour pouvoir synchroniser les différents threads de chaque classe fille.



Vers classe synchronisation_device



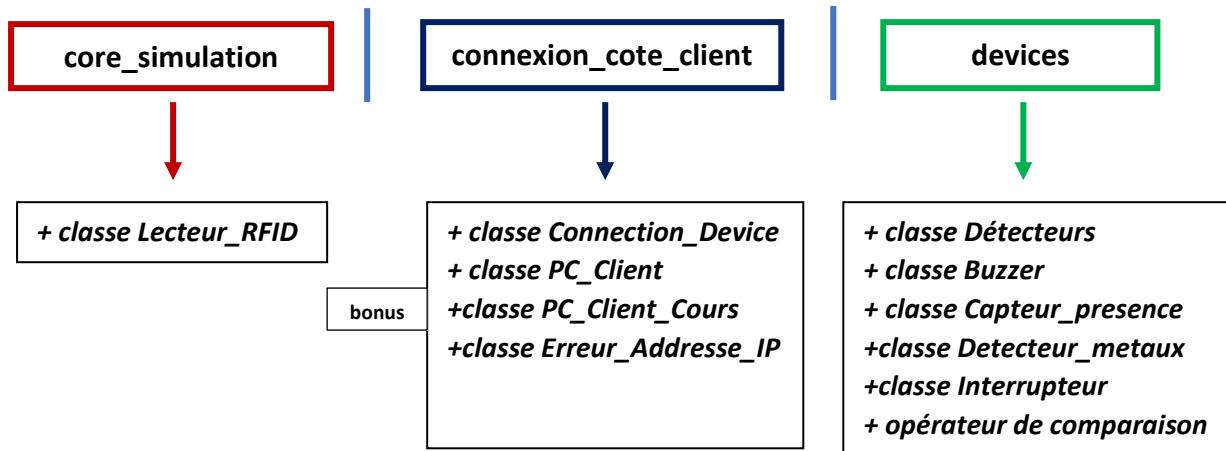
b. Diagramme d'activités



c. Bibliothèques développées

Nous présentons seulement ce que nous avons apporté par rapport au simulateur déjà mis en place.

Nous avons développé deux bibliothèques importantes et modifié une autre déjà présente :



Remarque :

Nous avons implémenté la classe `PC_Client_Cours`, pour tout utilisateur qui voudrait l'utiliser. Nous n'avons cependant pas eu le temps d'utiliser une instance de cette classe dans notre programme. En effet, par manque de temps, nous avons préféré nous focaliser sur la partie PC Client, qui était connectée à l'Arduino.

L'opérateur de comparaison a été redéfini pour comparer deux instances de `Capteur_presence`.

Nous utilisons cet opérateur dans la méthode `run()` du détecteur de métaux

Nous avons créé une classe d'exception correspondant à une erreur dans l'adresse IP du serveur.

Une instance de cette classe exception est lancé dans la méthode « `construire_Adresse_Serveur` »

III. Retours personnels sur le déroulement du projet

a. Bilan des objectifs

Pour réaliser notre projet , nous devions respecter quatre principaux objectifs :

- Comprendre le fonctionnement du simulateur : **OBJECTIF ATTEINT**

Cette tâche était primordiale. Ceci nous a pris le temps d'une séance de BE. Mais il était fondamental de saisir comment relier et connecter les différents appareils à la carte Arduino. Nous avons surtout saisi que la connexion physique entre celle-ci et un appareil était réalisée par le mécanisme d'adressage (pointeurs). Nous nous sommes d'ailleurs basés sur cette technique pour réaliser nos liaisons SPI et I2C.

- Effectuer une bonne conception du projet pour ensuite développer les bibliothèques : **OBJECTIF PARTIELLEMENT ATTEINT**

Même si nous avons fait l'effort de réaliser un diagramme de classes, clair, pour démarrer ce projet, notre travail a néanmoins consisté en des va et vient entre le diagramme de classes et son développement de code. Ce qui montre que nous avons peut-être manqué de rigueur dans la réalisation du diagramme de classes. Ce Bureau d'Étude a constitué une réelle leçon, pour nous, dans la manière de réaliser des projets informatiques. Nous avons pris conscience de l'importance d'effectuer une bonne conception avant de passer au développement des programmes.

- Réaliser une bibliothèque pour le lecteur RFID : **OBJECTIF ATTEINT**

Nous avons rencontré certaines difficultés dans la réalisation du lecteur RFID. Il nous a fallu en premier lieu comprendre son fonctionnement. Nous avons ensuite dû réfléchir à la façon dont nous allions développer cette classe, comment générer les séquences RFID, ce qui n'était pas évident. Cependant notre volonté de bien comprendre le fonctionnement d'un système RFID nous a permis d'atteindre l'objectif.

- **Développer une classe pour se connecter à un serveur DISTANT : OBJECTIF ATTEINT**

Atteindre cet objectif a constitué un réel défi puisque réaliser une connexion client-serveur nécessite certaines notions de réseaux que nous avions acquises l'an passé mais qu'il nous a fallu reprendre. Deux difficultés se sont présentées à nous pour cette bibliothèque. Paramétrier l'IDE pour qu'il puisse reconnaître les différentes fonctions utilisées pour la mise en place des sockets (ce fût l'étape la plus difficile). La deuxième difficulté a concerné une erreur de compilation dans l'utilisation des primitives Recv et Send (comprises dans les librairies pour les sockets). Nous avons en effet obtenu des comportements non souhaités. Par exemple, la méthode Recv est une méthode bloquante qui attend de recevoir un message sur la totalité de la longueur souhaitée (c'est le développeur, donc nous, qui imposons la taille) avant de pouvoir laisser le programme passer à la suite. Or le programme restait bloqué sur cette même ligne de code (celle où nous appelons Recv). Pour surmonter cette difficulté, nous avons mis en place, avec nos professeurs référents un code plus « propre », basé sur l'ajout de blocs « if » qui permettent de vérifier l'ensemble du bon déroulement dans la création et l'utilisation des sockets. Surtout nous avons compris la réelle signification de l'avant-dernier argument de la méthode Recv. Ce dernier n'indique non pas la taille du message maximum accordée mais bien la taille que nous, développeurs, souhaitons. Ce qui est totalement différent. Ce qui montre qu'il faut bien analyser une bibliothèque de fonctions avant de les utiliser.

- **Synchroniser l'ensemble des threads : OBJECTIF PLUTÔT ATTEINT**

Nous ne disposions pas de vrai capteurs ou actionneurs en temps réel pour matérialiser les événements extérieurs. Événements qu'il a fallu néanmoins simuler par les threads de chaque appareil. Cependant, faire en sorte que ces événements proviennent de manière cohérente dans le temps a constitué une tâche assez difficile. Nous y sommes tout de même parvenus grâce à l'utilisation de la fonction sleep et la mise en place d'un compteur. Nous avons appelé ce compteur « relaxeur » car il permet de « relaxer » l'affichage sur le terminal (diminuer le flux d'affichage).

b. Compétences acquises

Au niveau technique, les compétences qui étaient évidemment en jeu, furent celles liées à l'apprentissage du langage C++. Nous avons compris de nouvelles notions comme les mécanismes d'encapsulation et d'héritage, qui nous ont permis de proposer aux utilisateurs des bibliothèques cohérentes, offrant chacune des fonctionnalités qui leur sont propres.

Nous retenons également les notions de polymorphisme et de méthodes virtuelles. Par exemple, même si nous avons implémenté, de manière séparée, les threads de chaque appareil héritant de la classe « Device », il est important de comprendre que ces mécanismes (méthodes virtuelles et polymorphisme) nous ont permis d'appeler chacun de ces threads depuis la classe Board, ce, en ayant pour seule information que ces appareils héritaient de la classe « Device ». Ce qui a d'ailleurs grandement simplifié le code de la classe « Board ».

Concernant le projet, nous avons su faire preuve d'efficacité en proposant un système assez complet. Ce dernier offre une solution de contrôle d'accès connecté, répondant à de nombreuses situations quotidiennes. C'est par ailleurs le fait de choisir le système de notre choix qui nous a d'autant plus motivé. Nous sommes en général habitués à réaliser des systèmes déjà pensés par nos professeurs. Ici, nous devions nous-même penser le système. Ce rôle de concepteur-développeur fut très ainsi très motivant.

C'est également la gestion des délais qui était importante. Dans un intervalle de temps assez restreint, nous avons su proposer un système fonctionnel et innovant. Nous sommes d'ailleurs grandement satisfaits de constater son fonctionnement. Ce projet représentait un défi que nous avons su relever, ce qui nous a fait gagner en confiance pour la réalisation de futurs travaux de groupe.

Nous devons cependant progresser dans la manière d'aborder un projet. Par cela, nous signifions que nous devons consacrer plus de temps sur le plan conception, comme mentionné en III. a, pour mieux entrevoir l'organisation de nos programmes et revenir moins souvent sur l'agencement des classes.



INSA Toulouse

135, avenue de Rangueil
31077 Toulouse Cedex 4 - France
www.insa-toulouse.fr



MINISTÈRE
DE L'ÉDUCATION NATIONALE,
DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE