RAPPORT DE PROJET IMA4
P40 – RFID/NFC

Xinwei HU & Jean de Dieu NDUWAMUNGU

Année universitaire 2018 - 2019

Encadrants : Vincent COELEN & Thomas DANEL

Table of Contents

Remerciements

Nous tenons à remercier M. Vincent COELEN et M. Thomas DANEL, qui ont proposé ce sujet et qui nous ont encadré dans la réalisation du projet. Les échanges lors des différentes rencontres nous ont permis de nous améliorer, de faciliter la compréhension du sujet et de le réaliser correctement. Nous tenons à vous remercier d'avoir été à l'écoute et pour la confiance que nous avez accordé en mettant à notre disposition un des Robotino 3 utilisés en compétition.

Je tiens à remercier M. Xavier REDON et M. Thomas VANTROYS d'avoir suivi de près ce projet et de nous avoir également conseillé. Vos différentes interrogations nous ont poussé à approcher ce sujet de manière adéquate. Nous vous remercions également pour le matériel supplémentaire (téléphone portable sous Android) que vous avez prêté afin de réaliser nos tests indispensables à la réalisation du travail demandé.

Figure 1 Robotino 3 sur lequel nous avons travaillé

Introduction

Dans le cadre de notre formation à Polytech Lille en IMA4 nous avons réalisé ce projet RFID/NFC. Un projet très intéressant qui nous a permis de découvrir la communication RFID/NFC qui permet l'échange de données entre un lecteur et un terminal compatible.

Ce projet a été proposé par M. Vincent COELEN et M. Thomas DANEL afin de faciliter l'identification, la configuration et la prise en main des Robotino qu'ils utilisent lors des différentes compétitions où sont utilisées ces robots.

Nous allons dans un premier temps expliquer en quoi consiste la technologie RFID/NFC et comment nous l'avons appliqué dans notre projet. Nous présenterons les objectifs du projet et analyseront les concurrents existants. Une étude détaillée sur la réalisation de notre travail sera fournie et nous finirons par les problèmes rencontrés et les solutions envisagées.

1. Analyse du projet

1.1. Description

La RFID et la NFC sont deux technologies reposant toutes deux sur la même idée d'établir une communication sans contact malgré qu'elles soient différentes. Dans le cas de la RFID les données sont récupérées via des ondes radio fréquence alors que la NFC qui dérive de la RFID a été conçue pour faciliter l'échange sécurisée en permettant aux lecteurs et puces NFC de communiquer entre eux.

Le Robotino est un système robotique opérationnel de grande qualité à entraînement omnidirectionnel. Il peut se déplacer dans plusieurs directions. Il est muni d'une webcam et de plusieurs capteurs afin d'assurer son déplacement et différentes opérations qu'on adviendrait à effectuer. Il peut être utilisé directement sans PC.

Pour effectuer sa commande le Robotino dispose d'un ordinateur de bord sur lequel se trouve une carte Compact Flash qui contient plusieurs applications de démonstration ainsi qu'un système d'exploitation. Les démonstrations peuvent être directement lancées à l'aide du clavier de commande du Robotino.

Traditionnellement sa programmation est faite soit depuis un PC à l'aide du logiciel Robotino View via un réseau local sans fil ou en le branchant directement par une connexion filaire ou sans fil en ssh.

Dans ce projet nous allons développer une méthode nous permettant de le configurer automatiquement sans devoir passer par un réseau quelconque à l'aide de puces RFID/NFC et aussi en exploitant la technologie NFC disponible sur nos téléphones Android.

Il existe plusieurs générations de Robotino et nous travaillerons sur la dernière d'entre elles la troisième (Robotino 3).

1.2. Objectifs initiaux

- Aspect RoboCup

Le but de ce projet est d'avoir des puces RFID/NFC intégrés au niveau des têtes des Robotino ou sur la coque et de pouvoir avoir plusieurs applications possibles. Une des applications possibles serait d'avoir des cartes qu'on pourrait aimanter afin de choisir la configuration du robot pour différents aspects. Dans la coupe de robotique, RoboCup, les robots sont de deux couleurs différentes : cyan ou magenta. Au début de la compétition cette configuration est faite manuellement. Le travail que nous allons faire sera de le réaliser automatiquement cette tâche à l'aide de puces RFID/ NFC en leurs affectant un numéro, une couleur. Automatiser la tâche nous permettra de gagner donc du temps et d'éviter les erreurs liées à la configuration des Robotino surtout quand on dispose plusieurs de ces derniers.

- Aspect AIP

L'école Polytech Lille reçoit le long de l'année des visites sur le campus lors des journées portes ouvertes ou des événements comme la CREP. Sur les différents robots se trouvent des vidéos de démonstration. L'objectif côté AIP sera de développer une technique permettant de lancer automatiquement une démonstration à l'aide d'un tag préprogrammé qu'on viendra déposer sur le robot.

- Application Smartphone

La technologie NFC s'est étendue sur plusieurs modèles de smartphone. Un autre objectif de ce projet sera d'exploiter cette possibilité en effectuant une configuration automatique des Robotino lorsqu'on approche un smartphone sur ces derniers. Dans notre cas il s'agira dans un premier temps de connecter automatiquement le robot sur le même réseau wifi que le smartphone que l'on approche du Robotino et dans un second temps de piloter le robot ou de visualiser les images vidéo à l'aide du joystick virtuel sur une application Android que nous allons développer.

1.3. Positionnement par rapport à l'existant

Dans ce projet nous allons travailler sur des Robotino qu'on retrouve à Polytech. Notre solution sera nouvelle dans la mesure où elle va permettre un gain considérable en temps et en nombre de tâches à effectuer lors de la configuration des robots, en automatisant les tâches. Les Robotino disposent des supports NFC le but est d'exploiter cette possibilité en apportant une solution nouvelle.

1.4. Analyse des concurrents

a. NEATO Robot aspirateur connecté

Neato est un robot aspirateur connecté qui permet à son utilisateur de le contrôler via une application. Cela se fait par un point d'accès créé sur le robot lui-même et l'utilisateur après avoir téléchargé l'application iOS ou Android vient prendre le contrôle du Robot. L'application NEATO permet d'accéder aux réglages de l'aspirateur, de le diriger durant un cycle de nettoyage ou de sélectionner un mode de nettoyage particulier.

Figure 2 NEATO Concurrent 1

b. NFC DIN Rail

Il s'agit d'une démonstration développée par NXP destiné à montrer comment prendre le contrôle de réglages qui peuvent parfois être complexes sur un écran tactile. Grâce à une puce NFC situé sur le téléphone on peut mettre à jour l'ensemble du système, effectuer un diagnostic du système en positionnant le téléphone près du système ou le paramétrer de la même manière.

Figure 3 Concurrent 2 NXP

2. Réalisation du projet

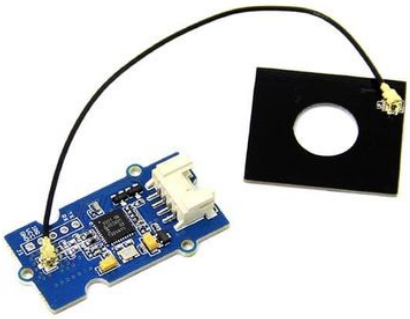
2.1. Choix du matériel

Afin de réaliser le projet nous avons à notre disposition comme matériel :

- Robotino 3
- Transpondeur RFID/NFC Grove, référence mouser : 713-113020006
- Assortiment de tag RFID/NFC, référence mouser : 485-365

- Tag RFID/NFC Sticker, référence mouser : 485-362
- Un téléphone/tablette sous Android 0S





Nous avons sur l'image ci-dessus respectivement un Robotino 3, un kit de Tag Mifare Classic, et notre antenne NFC qui est une PN532.

2.2. Configuration du Robotino

a. Scénario envisagé

L'équipe de recherche de Polytech Lille participe à la RoboCup 2019. Elle veut améliorer ses performances en instaurant une technique de configuration automatique des Robotino sans avoir à le faire manuellement pour chaque robot en lui assignant à l'aide d'un tag NFC un numéro ainsi qu'une configuration adéquate. Des Tags Cyan et Magenta pour chaque ont été fabriqués. On a juste à coller le tag sur la tête du Robotino pour le configurer en Cyan ou Magenta.

b. Réalisation

La première étape du projet consistait à établir cette configuration automatique du robot. Les Robotino 3 disposent du système d'exploitation Linux. Le Robot sur lequel on

travaillait disposait d'un fichier de configuration à la racine dans le dossier config. Dans ce dossier se trouve un fichier robotConfig.yaml dans lequel se trouvent les informations suivantes :

```yaml
teamName: PYRO
teamColor: magenta
robotNumber: 1
robotName: R1

cryptoKey: randomkey
```

Figure 4 Fichier config du Robotino

Le champ teamName correspond au nom de l'équipe.

teamColor : la couleur du robot

robotNumber : le numéro du robot

cryptoKey : est une clé qui est modifiée de manière aléatoire.

L'objectif étant de pouvoir modifier sans avoir à écrire manuellement dans ce fichier nous avons utilisé deux librairies appelées libnfc et libfreefare qui nous permettent de donner accès aux périphériques NFC les différentes applications utilisateurs. Il s'agit deux librairies libres, indépendantes de toute plateforme et disposant d'une API permettant de réaliser plusieurs fonctionnalités. Elles sont écrites en langage C. libfreefare s'appuie sur libnfc et est utilisé pour lire et écrire les cartes Mifare dont nous disposons.

Nous avons dans un premier temps réalisé l'étude du code source de cette librairie et nous avons très vite pris en main sa puissance en écrivant le contenu d'un Tag dans un fichier. Ce qui nous était demandé n'étant pas cela nous donc écrit un programme basé sur le mifare-classic-read-ndef qui nous permet de lire les contenus des cartes. (Voir Annexe 1).

Il est donc impératif d'installer les librairies libnfc et libfreefare.

Une fois les bibliothèques installées nous pouvons connecter l'antenne au Robotino et modifier le fichier dans /etc/nfc/libnfc.conf. Il s'agit du fichier de configuration de la librairie. Il faut préciser à cet endroit le type d'antenne utilisé afin que notre matériel puisse être détecté en tant que périphérique NFC par le Robotino.

```
device.name="My Reader Name"
device.connstring="connstring"
connstring = "pn532_uart:/dev/ttyXXXX"
```

Figure 5 Fichier Config du PN532

Pour écrire sur les Tags nous avons utilisé un logiciel nous permettant d'écrire les données NDEF et l'on entrait le nom, la couleur et d'autres informations dans le tag NFC. Le programme sur le robot est principalement divisé en deux : lire le contenu du tag NFC et modifier le fichier Yaml dans le Robotino. Voici le diagramme d'activité du programme :

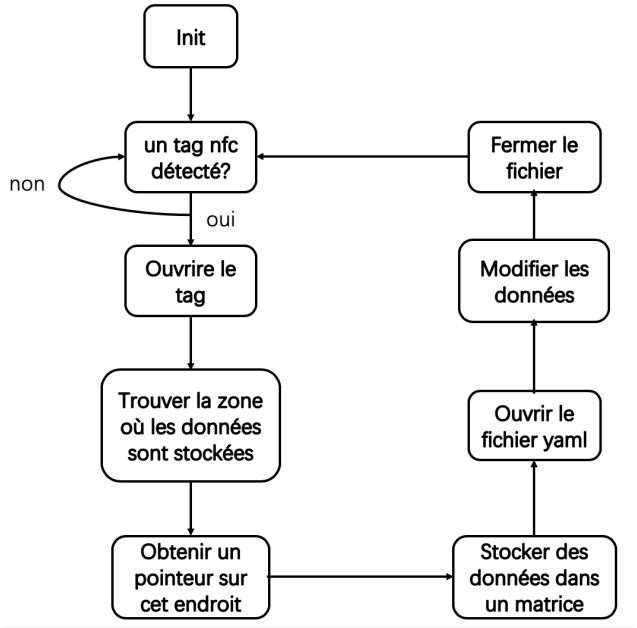


Figure 6 Schematic de fonctionnement du programme

Une fois l'initialisation terminée, le programme recherche d'abord l'interface USB correspondante et le tag et il attend jusqu'à l'apparition d'un Tag. Afin de lire le contenu de Tag NFC, on doit obtenir le MAD (Accès au secteur) et le TLV bloc qui contient les informations. Comme il existe des TLV blocs vide, il est important d'obtenir la longueur d'un bloc pour analyser tous les blocs un par un jusqu'à avoir les données. Quand le programme a trouvé les données, il place un pointeur dans le début de ces informations. Puis, il sauvegarde les

informations dans une matrice (4 lignes). Après avoir chargé le fichier yaml, il modifie les données avec cette matrice et émet les nouvelles données. Lorsque le processus ci-dessus est réussi, le programme entre un nouvel boucle et se répète.

2.3. Lecture de la configuration Réseau

a. Scénario envisagé

Xinwei souhaite obtenir les informations sur la configuration du Robotino notamment l'adresse IP, le nom du Wi-Fi sur lequel il est connecté, le mot de passe du Wi-Fi. Pour cela il approche de l'antenne son smartphone disposant du système d'exploitation Android et apparaît sur son smartphone les informations souhaitées.

b. Réalisation

Nous nous sommes appuyés sur la librairie libnfc qui permet d'émuler c'est-à-dire de donner un comportement similaire à un Tag NFC une antenne. Il nous a fallu comprendre comment un Tag est écrit, comprendre le format NDEF qui est le format(protocole) sous lequel sont transmis les informations d'un téléphone Android à une antenne ou à un autre téléphone Android.
Pour comprendre le NDEF il faudrait expliquer sa structure. NDEF veut dire étymologiquement NFC Data Exchange Format. Il s'agit d'un format binaire organisé en plusieurs éléments, chacun contenant des enregistrements (records) comme nous pouvons le voir sur la figure ci-dessous. Chaque enregistrement est constitué d'un entête contenant les informations concernant le type de données (URL, texte, …), la taille des données et enfin le contenu lui-même.

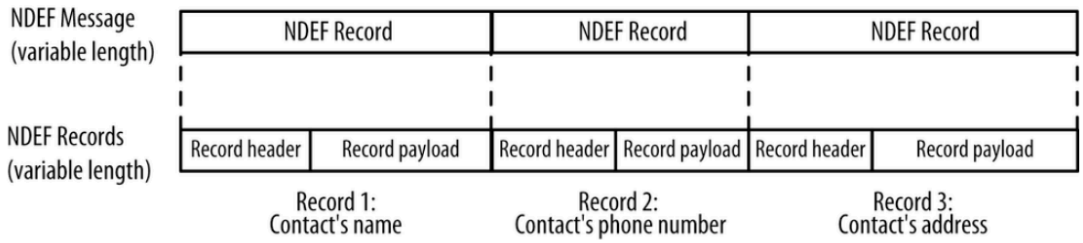


Figure 7 Structure NDEF

L’entête NDEF Records contient la description de comment accéder aux différentes données. Les enregistrements peuvent être plusieurs mais aussi de différents types. Nous pouvons voir sur l’image ci-dessous la structure complète. Le NDEF Record correspond à un TNF (Type Name Format). Il indique la manière dont on va lire le contenu.

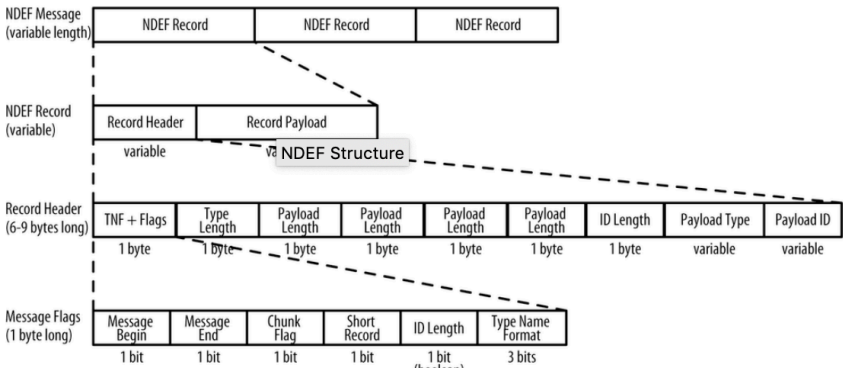


Figure 8 Strucuture complète NDEF

Il existe 7 valeurs possibles pour le TNF (Type Name Format) :

- 0 : enregistrement vide
- 1 : Type prédéfini par le NFC Forum
- 2 : Internet Media Type
- 3 : Lien URL
- 4 : Externe – défini par l’utilisateur
- 5 : Inconnu
- 6 : Inchangé
- 7 : Non alloué, réservé pour une utilisation ultérieure

Les plus fréquents 01, 02 ainsi que 04 qui est utilisé par le système Android pour communiquer par NFC.

Le lien peut être de type URN (ex : monApp), URI (ex : com.polytech.robotino) qui est un lien inversé, URL (ex : http://polytech.com/monApp)

Le Payload ID (ID de l'enregistrement) est arbitraire. Il faut que celui qui émet et celui qui reçoit soient d'accord sur cet identifiant. Le message n'étant pas transmis dans le cas contraire.

Le Record Layout

Les 5 premiers bits sont des drapeaux indiquant comment lire et où se trouve l'information.

Ils sont organisés de la manière suivante :

- MB (Message Begin) Vrai si c'est le premier enregistrement sur le Tag
- ME (Message End) Vrai si c'est le dernier message sur le Tag
- CF (Chunk Flag) Vrai si le message est partitionné sur plusieurs secteurs
- SR (Short Record) Vrai si c'est le format de message court est utilisé
- IL (ID Length) Vrai si le champ de longueur de l'ID du Tag est présent

Record Header

Il contient l'information requise pour la lecture des données.

En résumé un message NDEF est formé de la manière suivante : En entête nous avons le TNF expliqué ci-haut, vient ensuite le NDEF record header qui contient la longueur du type de données sur un byte. Vient ensuite le contenu (le message à envoyer) suivi du SR cité ci-haut. Si le SR est initialisé à 1 la taille du message est de 1 byte dans le cas contraire le message est sur 4 bytes. Si le champ ID Length est initialisé à 1 il sera suivi d'un byte correspondant au à la taille de l'ID. Le Record Type va déterminer combien de bytes seront lus. La taille maximale du message étant de de 2^{31}-1 bytes.

Voici comment nous avons configuré notre antenne afin d'envoyer un message personnalisé et pouvant être lu par les différents lecteurs NFC :

0x00, 0x0F, Notre fichier d'entête sera sur 15 bytes
0x20, Mapping version 2.0 *
0x00, 0xFB, MLe Maximum R-ADPU taille

0x00, 0xF9, MLc Maximum C-ADPU taille
0x04, T field of the NDEF File-Control TLV
0x06, L field of the NDEF File-Control TLV

0xE1, 0x04, File identifier
0xFF, 0xFE, Taille maximale du message NDEF

0x00, Droit de lecture NDEF
0x00, Droit d'écriture NDEF

Cette configuration nous permet d'envoyer les données souhaitées. Les données que nous envoyons sont séparées par un délimiteur (%) et on vient concaténer l'adresse IP, l'information Wi-Fi (nom Wi-Fi et mot de passe) sur une chaîne de caractère. L'adresse IP est récupérée grâce à la fonction inet_ntoa(*((struct in_addr*)host_entry->h_addr_list[0]));
La commande iwconfig nous permet de récupérer le nom du SSID sur lequel nous sommes connecté et le mot de passe correspondant à ce SSID est récupéré dans le fichier /etc/NetworkManager/system-connection/ Il s'agit d'un fichier protégé qui requiert des autorisations d'administrateur pour le lire.

c. Notre application Android

L'application que nous avons développée a été écrite en langage Java. Le logiciel que nous avons utilisé est Android Studio. Il s'agit de l'environnement de développement officiel de Google. Nous l'avons choisi pour sa simplicité et son efficacité et la possibilité de relier directement nos comptes Git au logiciel. Nous avons la possibilité d'écrire notre programme en Kotlin ou en Java mais notre choix s'est tourné vers ce dernier.

Figure 9 Logo logiciel Android Studio

Les téléphones Android ont simultanément trois modes de fonctionnement concernant la technologie NFC :

- Le mode d'écriture/lecture qui permet d'écrire ainsi que de lire les Tags qui sont approchés au téléphone
- Le mode P2P (Peer to Peer) permettant d'échanger avec les autres périphériques NFC (par exemple un autre téléphone)
- Le mode d'émulation qui donne la possibilité au téléphone de se comporter lui-même comme un Tag

Le mode qui nous intéressera sera de lecture et d'écriture des Tags. Le schéma suivant explique comment le téléphone sous Android arrive à lire les Tags qui lui sont reliés.

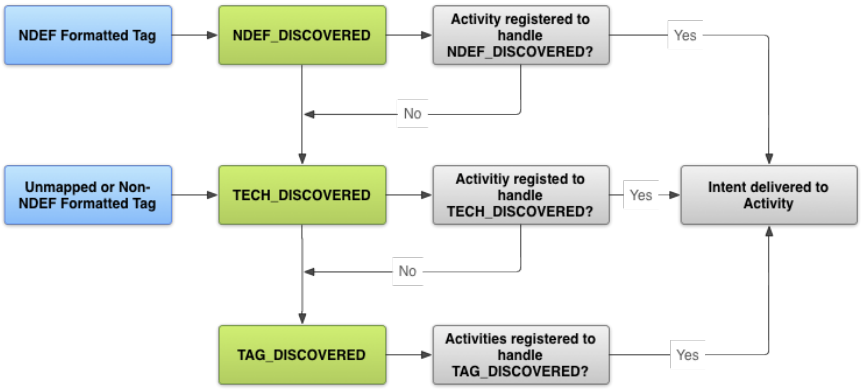


Figure 10 Schematic de fonctionnement NFC Android

Lorsqu'un Tag NFC est approché au téléphone, le téléphone donne priorité à la recherche d'une application prête à prendre son contrôle et la lance si les données sont au bon format c'est à dire les données NDEF. Pour cela nous avons besoin de le déclarer dans notre application. Si les données ne sont pas sous format standard NDEF c'est à nous de programmer un protocole nous permettant de lire ces données. Cependant dans notre projet nous utiliserons que les données NDEF. C'est grâce à ce mécanisme que nous pourrons donner la priorité à notre application afin qu'elle s'ouvre dès qu'un Tag sera approché.

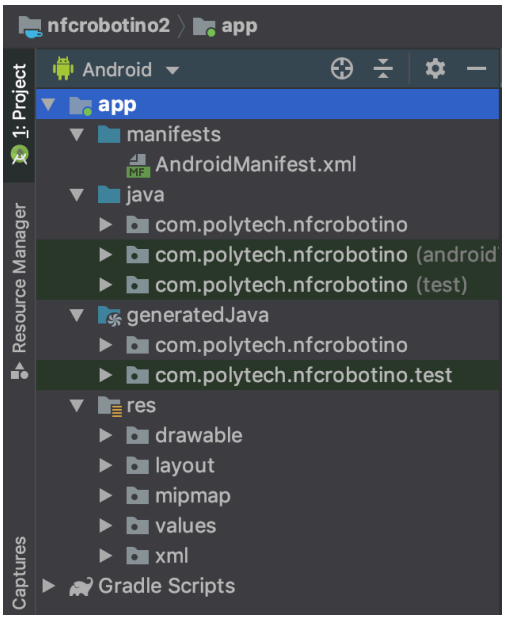


Figure 11 Strucutre des fichiers sur Android

La figure ci-dessus nous montre l'organisation des fichiers de notre programme Android. Dans manifests se trouve le fichier AndroidManifest.xml qui est la racine du projet de tout application Android. Il décrit à l'application l'essentiel de l'information requise à son bon fonctionnement au système Android. C'est également dans ce fichier qu'on déclare le nom de l'application, les différents autorisations (accès à la caméra, à internet, etc.)

Dans le dossier java se trouve le code source du projet

Dans le dossier generated.java se trouvent les fichiers java qui sont généré automatiquement à la compilation.

Dans res se trouve les fichiers contenant les visuels de l'application, les images et les ID ou constantes que l'on souhaite déclarer. Le code entier de l'application est sur notre wiki en annexe.

Les pages de l'application sont organisées en activité. Nous avons donc différentes activités nous permettant de réaliser notre application. L'activité nous permettant de lire les Tags émulées par notre antenne NFC et l'activité nous permettant de commander le Robotino.

2.4. Commande du Robotino

Afin de prendre la main sur le Robotino nous avons pensé à utiliser un client -- serveur TCP entre le Robotino et le téléphone. TCP veut dire Transmission Control Protocol. Le protocole établit une connexion virtuelle entre deux périphériques (client et serveur) via une série de requêtes et réponses envoyées à travers le réseau physique. Le client est le téléphone et se connecte au Robotino. En premier temps nous avons écrit un programme nous permettant de contrôler le Robotino sans l'intervention du téléphone. C'est un programme que l'on tournait sur le robot auquel on fournissait les paramètres x et y et le programme se déplaçait jusqu'à atteindre l'objectif et s'arrêtait. L'idée était de pouvoir contrôler le robot manuellement dans un premier temps et par la suite recevoir des données via le protocole TCP/IP. Nous avons développé un menu sur notre application envoyant la chaîne de caractère avancer lorsque l'on clique sur le bout haut, reculer sur le bouton bas, gauche et droite. Le serveur arrivait à bien recevoir les données malgré qu'il s'arrêtât après le premier envoi.

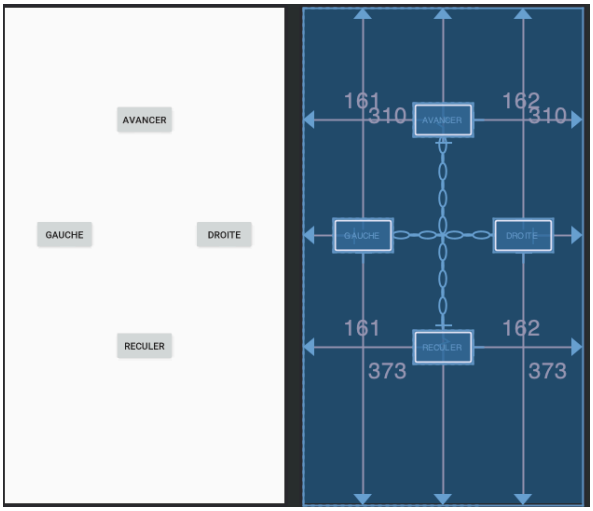


Figure 12 xml commande Robotino

Dans notre serveur nous testons la chaîne de caractère et suivant que c'est avancer, reculer, gauche ou droite nous initialisons avec des valeurs prédéfinis x et y. L'utilisation des threads était judicieuse afin de lire ce que l'utilisateur envoie au Robotino et en même temps le bouger. L'objectif étant de distinguer les appuis longs et les appui court.

3. Résultats obtenus

a. Difficultés rencontrées

Lorsqu'on récupérait l'adresse IP nous récupérions l'adresse IP local 127.0.1.1 alors que le test sur les ordinateurs dans la salle E304 nous affichaient la bonne adresse. De plus le contrôle du Robot en appuyant sur les boutons se lançait et on voyait la chaîne de caractère reçu par le robot mais celui-ci bougeait une fois et le téléphone coupait la connexion. Nous pensons que cela réside dans notre approche pour nous connecter au serveur.

b. Ce qui n'a pas été fait

Nous n'avons pas pu malheureusement pu tester le retour camera du Robotino. Dans nos objectifs initiaux cela nous était demandé. L'idée qu'on avait été de pouvoir se servir du programme d'exemple nous permettant de capturer des photos et de le mettre dans une boucle de manière et prendre plusieurs images en un temps déterminé et lire ces images de manière continue. Nous pourrions envoyer ces images via le protocole TCP au téléphone et celui aurait reconstruit les images et les aurait affichés également de manière continue de façon à constituer une vidéo.

Conclusion

Nous avons choisi ce projet parce que la technologie nous intéresse et réaliser ce projet était pour nous une occasion de pouvoir découvrir ce qu'est la NFC. En 5ème année nous aurons l'occasion de revenir dessus avoir eu un projet dessus sera bénéfique pour nous, de plus Xinwei de retour en Chine finira son Master 2 dans cette spécialité.

Malgré que tout ce qui avait été demandé dans le cahier de charge n'a pas pu être réalisé, ce projet était très formateur car il nous a permis d'acquérir de connaissances qui nous seront utiles dans nos futurs projets. La technologie NFC étant de plus en plus utilisée avoir ce premier contact et pouvoir mettre en application les notions de programmation Java et Réseau que nous voyons était un plus pour nous. Ce projet nous a appris à faire de la recherche d'information à nous organiser en partageant le travail et en nous fixant des objectifs hebdomadaires.

Sources

Les codes sources du projet sont sur le lien suivant :
https://framagit.org/PyroTeam/nfcrobotino

Sites Web que nous avons utilisé :
https://developer.android.com/guide/topics/connectivity/nfc
https://github.com/nfc-tools
https://www.oreilly.com/library/view/beginning-nfc/9781449324094/ch04.html
https://www.nxp.com/docs/en/application-note/AN1305.pdf
https://rex.plil.fr/Enseignement/Reseau/Reseau.IMA4sc/reseau.html
https://wiki.openrobotino.org/index.php?title=API2