École Polytechnique de l'Université de Tours
64, Avenue Jean Portalis
37200 TOURS, FRANCE
Tél. +33 (0)2 47 36 14 14
www.polytech.univ-tours.fr

Département Informatique
3e année
2011 - 2012

Rapport de Projet de Système d'exploitation

Tracking RFID/Bluetooth

étudiants
Jean de BAROCHEZ
jean.debarochez@etu.univ-tours.fr
Benoit BELZ
benoit.belz@etu.univ-tours.fr

Encadrants
Mathieu DELALANDRE
mathieu.delalandre@univ-tours.fr

Université François-Rabelais, Tours

DI3 2011 - 2012

Version du 9 juin 2012

Table des matières

Table des figures

Liste des tableaux

Liste des codes

Introduction

1.1 Présentation du projet

Ce projet entre dans le cursus de notre troisième année d'étude à l'école Polytech'Tours. Le but de celui-ci était de mettre en place une plateforme de suivi d'objet dans un bâtiment afin de connaître leur position. En premier lieu, nous avons étudié une solution à partir de lecteur RFID qui permettait de lire des étiquettes appliquées à des objets.

Les contraintes matérielles qu'impliquent un lecteur RFID (achat du lecteur et des antennes, puis sa livraison) nous ont poussés à orienter le projet vers une autre technologie : le Bluetooth. Celle-ci étant maintenant présente partout dans la vie de tous les jours (ordinateurs portables, téléphones mobiles, périphériques informatiques...), il est plus aisé de mettre en pratique cette solution.

Notre rapport se décline en trois parties. Tout d'abord nous étudierons les particularités des transmissions RFID et Bluetooth. Nous continuerons sur un compte rendu de notre application Bluetooth offrant une solution à notre projet. Enfin, nous donnerons les différentes ouvertures possibles données par un tel système.

Documentation

2.1 RFID

2.1.1 Utilité

Le Radio Frequency Identification est une méthode, utilisant les ondes radio, pour mémoriser et récupérer des données à distances sur des étiquettes appelées RFID-tag. Les radio-étiquettes sont des petits objets, équipés d'une puce électronique et d'une antenne pour capter les ondes radios. Ces étiquettes peuvent être placées n'importe où et sur n'importe quoi, être vivant ou objet inerte. Leur rôle premier est de pouvoir être suivie et lue par un lecteur RFID. Beaucoup utilisé dans les chaînes de production pour suivre les objets en transit, cette méthode peut aussi servir pour connaître la position d'animaux dans la nature, permettre le passage aux voitures aux péages via les boîtiers de télépéage etc.

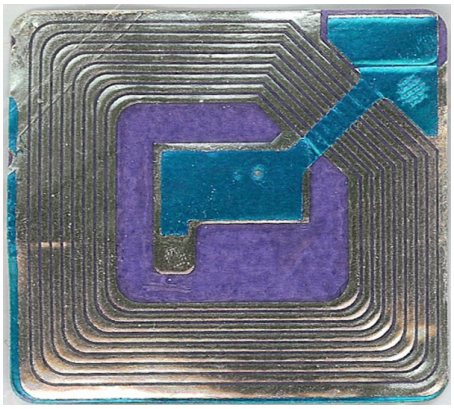

FIGURE 2.1 – Étiquette RFID

Aujourd'hui, on peut trouver ce système d'étiquetage dans de nombreuses grandes surfaces. De part sa possibilité de pister les produits, il permet facilement d'obtenir un inventaire des stocks en rayon. Il est aussi utile d'en équiper des produits à forte valeur afin de contrôler toute fraude à l'entrée du magasin. Ce système tend à remplacer le code barre, mais le coût d'une étiquette RFID est bien sûr plus important qu'une impression de code barre, ce qui freine son entrée sur le marché.

Plusieurs écoles dans le monde ont essayé d'équiper les cartes des étudiants d'étiquettes RFID. Cependant, face à la possibilité d'être pisté n'importe où, de nombreuses plaintes d'atteinte à la vie privée ont mis fin à de tels projets. N'importe qui pourrait obtenir un lecteur RFID portable et ainsi pister ces personnes en dehors de l'école.

2.1.2 Fonctionnement

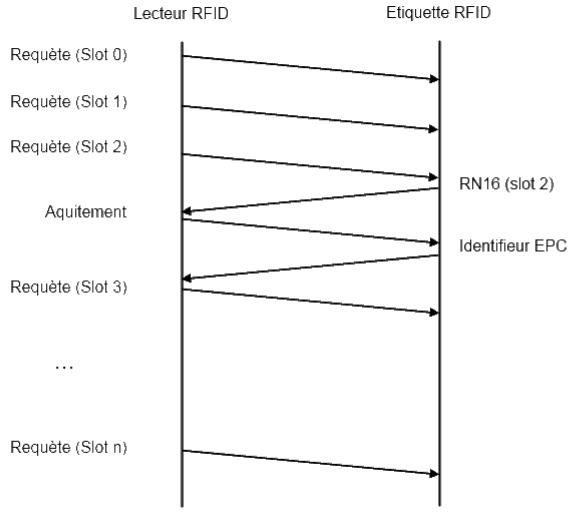

FIGURE 2.2 – Échange de message pour identifier une étiquette

Toutes les 400ms, le lecteur émet un signal sur la bande de fréquence 902-928 Mhz ISM. Ces fréquences sont dites Ultra Hautes Fréquences. Les ondes envoyées par le lecteur réagissent un peu comme un radar. Si celles-ci rencontrent une étiquette, alors il y a rétrodiffusion des ondes. C'est à dire que les étiquettes emmagasinent l'énergie reçue des ondes afin de les renvoyer à l'émetteur. Ainsi, l'étiquette se signale au lecteur. Dans ces ondes émises est transporté un nombre aléatoirement généré sur 16-bit (RN16 message). S'il n'y a pas eu de collision entre le tag et le reader, alors ce dernier acquitte le message en renvoyant un ACK Message (Acknowledgement Message). L'étiquette renvoi enfin son code produit électronique (EPC), codé sur 96 bits, et c'est ainsi que se termine la plupart des discussions. Cependant, dans certains cas, le lecteur peut effectuer d'autres opérations sur l'étiquette, comme par exemple écrire des données dessus.

2.1.3 Application d'une solution RFID

Au début de notre projet, nous avons eu pour but de comprendre comment mettre en place un réseau RFID afin de pouvoir connaître à tout instant la position d'objet dans un bâtiment (imprimantes, unités centrales etc). Nous avons donc recherché quel type de matériel serait requis pour répondre à nos attentes. Voilà quelles étaient les contraintes données :

- Un lecteur fixe
- Rayon de portée d'environ 10m
- Manipulable de l'extérieur

Nous avons donc cherché tout d'abord quelle norme de produit correspondait à ces critères. Notre choix s'est ainsi porté sur la norme 18000-6c. Transmettant sur les Ultra Haute Fréquences (UHF), les appareils peuvent ainsi détecter sur une dizaine de mètre de rayon les étiquettes passant dans leur champ.

Avec la norme, nous avons pu commencer à chercher l'appareil correspondant. Les entreprises construisant ces machines ne vendent en général qu'aux professionnels, si bien que les prix ne sont pas publics sur Internet et nécessitent de faire des devis. Nous avons donc contacté plusieurs entreprises françaises et étrangères afin d'en connaître plus sur leurs produits. Bien que très peu d'entre elles ai répondu, nous avons pu trouver un lecteur satisfaisant. Ce lecteur de la compagnie CaenRFID nécessite d'être accompagné d'antenne et peut être connecté jusqu'à quatre par lecteur. Cela dépassait nos attentes, car au lieu de

nécessiter plusieurs lecteur et de les disposer à différents endroits, nous avons un lecteur centrale possédant ses antennes pouvant contrôler une large zone.

Cependant, la livraison du produit s'est faite attendre et nos contraintes temporelles pour notre projet a poussé celui-ci vers une autre direction : le Bluetooth.

FIGURE 2.3 – Lecteur RFID Ion R4300P

2.2 BlueTooth

2.2.1 Historique

L'idée de Bluetooth est issu de la société L. M. Ericsson en 1994. Elle cherchait à développer un procédé de connexion sans fil de téléphones mobiles à d'autres périphériques. En 1998, IBM, Intel, Nokia et Toshiba rejoignent Ericsson dans leur recherche et créent un consortium nommé Bluetooth SIG (Special Intereset Group) afin de trouver un système radio pour connecter des appareils électroniques sur de courtes distances, avec un coût suffisamment bas pour que la mise sur le marché soit rapide et efficace. En 1999, le Bluetooth SIG publient la première version du protocole. Depuis, de nombreux appareils électroniques utilisent le Bluetooth, du téléphone mobile à l'ordinateur portable en passant par les casques audio, les imprimantes, etc.

La première version permettait aux appareils de se trouver et de se connecter entre eux. Cette action est nommée "pairing" puisque les deux produits s'associent afin de procéder à des échanges sécurisés de données.

Les versions 2.0, 3.0 et 4.0 sortiront respectivement en 2004, en 2009 et en fin d'année 2009. Chacune permettant d'améliorer la vitesse d'échange des données. La version 3.0 apporta aussi une importante nouveauté puisqu'à partir de ce moment là, le Bluetooth peut se combiner avec les standards 802.11 (Wi-Fi) afin d'obtenir un débit de transfert élevé. Ceci est rendu permit par l'ajout d'un système "Alternate MAC/PHY" permettant, comme son nom l'indique, d'utiliser les couches MAC et Physique du modèle OSI pour transporter les données Bluetooth. Le protocole Bluetooth est toujours utilisé pour rechercher des dispositifs à portée, pour se connecter, configurer le profile Bluetooth à utiliser et sécuriser la liaison. Cependant, quand la quantité de données à envoyer devient trop importante, le protocole 802.11 prend le relais pour transporter les données. Un des défauts majeurs du Bluetooth était son faible débit et si un transfert important de donnée était nécessaire, la durée de copie était parfois très longue. Maintenant, avec cette "aide" du Wifi, le Bluetooth est capable d'atteindre un débit de 54 Mbps. Bien sûr, permettre l'alternance entre le protocole Wi-Fi et le protocole Bluetooth nécessite que l'appareil possède les puces Wi-Fi et Bluetooth.

2.2.2 Pile des protocoles Bluetooth

Le standard Bluetooth rassemble différents protocoles dans ses différentes couches de sa pile. Si bien que sa structure ne suit pas le modèle OSI ou tout autre modèle. On peut y faire des rapprochements, mais il reste très différent.

La première couche, la couche radio, pourrait être associé à la couche physique du modèle OSI. Elle représente l'une des spécificités du Bluetooth. De par son coût faible, il permet au protocole d'être facilement implanté sur le marché. La deuxième couche, le contrôleur de liaison ou bande de base, pourrait être associé au protocole MAC, mais inclue aussi des éléments de la couche physique. Elle gère donc les adresses matérielles des périphériques "BD_ADDR" codées sur 48 bits. Elle permet aussi aux maîtres de gérer leurs ouvertures de socket et l'envoi de trames.

Ce contrôleur de liaison s'accompagne de deux autres protocoles. Le premier est le gestionnaire de liaison. Il s'occupe de la mise en place des liens logiques entre les appareils équipés de connectivité Bluetooth, de les appairer et du chiffrement des données, ainsi que de gérer la qualité de service. L'autre protocole est l'interface hôte-contrôleur. Il permet de faciliter la distinction entre ce qui est implémenté sur la puce Bluetooth et ce qui est installé sur l'appareil qui héberge la puce et d'en faire la liaison.

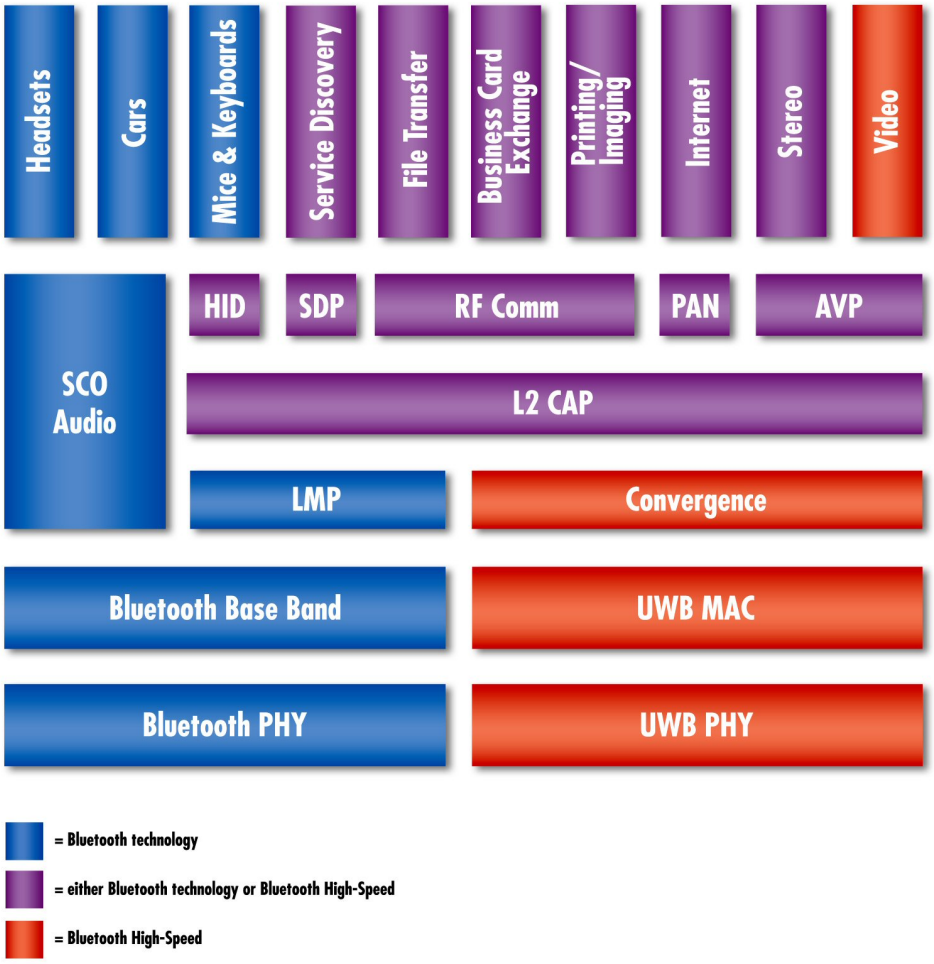

FIGURE 2.4 – Pile des protocoles Bluetooth

Au dessus de cet HCI (Host-Controller Interface) se trouve la couche application, composée de plusieurs protocoles. Comme on expliquait précédemment, la pile Bluetooth ne respecte pas de modèle standard. Ainsi, la couche application s'occupe aussi bien du traitement des données que de la gestion applicative du Bluetooth. Le protocole de contrôle et d'adaptation de liens logiques L2CAP (Logical Link Control Adaptation Protocole) supporte le haut niveau de multiplexage, la segmentation et le réassemblage des paquets envoyées, ainsi que la qualité de service si nécessaire. De nombreux protocoles de la couche application utilisent le L2CAP. Par exemple, le service de détection d'appareil à proximité (Service Discovery Protocole) ou RFcomm (Radio Frequency communication) qui émule le port série des claviers, souris, etc des ordinateurs afin de les connecter comme s'ils étaient connecté par des moyens traditionnels. Ce "choix" est rendu possible par un système de profile, autrement dit d'application du Bluetooth, permettant de sélectionner les protocoles à utiliser en fonction des besoins de l'application.

2.2.3 Les profils Bluetooth

Les profils correspondent à l'usage dont notre périphérique a besoin. A l'inverse du 802.11 qui procure la connexion et laisse aux applications le choix de l'utilisation de la bande passance, les profils décrivent

l'architecture protocolaire de l'appareil à utiliser. Un clavier n'a pas les mêmes besoins qu'un échange de fichier entre deux ordinateurs. Cette solution de profil permet ainsi de restreindre au strict minimum les ressources nécessaires à l'appareil Bluetooth. Ainsi, on obtient des systèmes embarqués non-surchargés par des fonctionnalités qui ne leur seraient jamais utiles.

Chaque appareil comporte au moins un profil en fonction de leur utilisation. Faire une liste des différents profils ne serait pas nécessaire. Cependant dans notre recherche de solution, le choix de l'un d'entre eux aurait pu être envisagé.

Il s'agit du Device ID Profile. Il permet à un appareil d'être identifié au-delà des limitations des classes d'appareil Bluetooth. Ces classes permettent de distinguer les différents types d'appareil des uns et des autres (Ordinateur, portable ou de bureau, téléphone, mobile ou fixe etc). Là où le Device ID Profile aurait été utile pour notre système est qu'il apporte d'autres informations à propos de l'appareil : l'identification du constructeur, l'identification du produit et sa version de produit. Ce profil sert principalement pour les ordinateurs pour reconnaître un périphérique se connectant et ainsi télécharger les drivers appropriés. Si notre bâtiment à surveiller possédait des centaines de produit à suivre, il aurait été peut-être plus intéressant d'utiliser ce système par rapport à notre solution finale. Nous expliquerons pourquoi dans la suite.

2.2.4 Fonctionnement

Le Bluetooth utilise la bande de fréquence 2,4GHz ISM (Industrial, Scientific and Medical radio band), identique à la norme 802.11. Cette bande est ensuite divisée en 79 canaux de 1MHz chacun qui consiste à découper la bande de fréquence en "morceaux" de 1Mhz. Afin d'éviter toute interférence avec d'autres réseaux utilisant ces fréquences, l'étalement de spectre par saut de fréquence (Frequency Hopping Spread Spectrum) est appliqué. C'est une méthode de de transmission de signaux utilisant différents canaux selon une séquence pseudo aléatoire connue que de l'émetteur et du récepteur. Ainsi, les interférences avec d'autres ondes réseaux sont plus difficiles à obtenir.

Les composants se connectent en tant qu'esclave à un maître. Cela forme un réseau appelé piconet. Un piconet peut comporter jusqu'à 7 appareils en mode actif. Une cohabitation de plusieurs piconet est possible par l'intermédiaire d'un des esclaves qui permettrait le relais d'information entre les différents groupes. Il est aussi possible d'associer des appareils en mode "parked" à un piconet. Ils n'ont alors pas d'adresse, mais ne peuvent pas communiquer avec le maitre. En plus des sept appareils actifs dans un piconet, 255 périphériques "parked" peuvent être associer à ce réseau. Par ce mode, l'appareil peut économiser son énergie. Il ne sera réveillé que depuis un activation ou un signal du maître. Toute connexion se fait entre maître et esclave. Les communications entre esclaves ne sont pas possibles.

2.2.5 Connexions

La connexion de 2 périphériques Bluetooth suit le processus suivant :
- Découverte des composants à portée
- Synchronisation avec un composant.
- Découverte des services de ce composant.
- Création d'un canal avec le composant et envoie de la séquence FHS
- Pairage

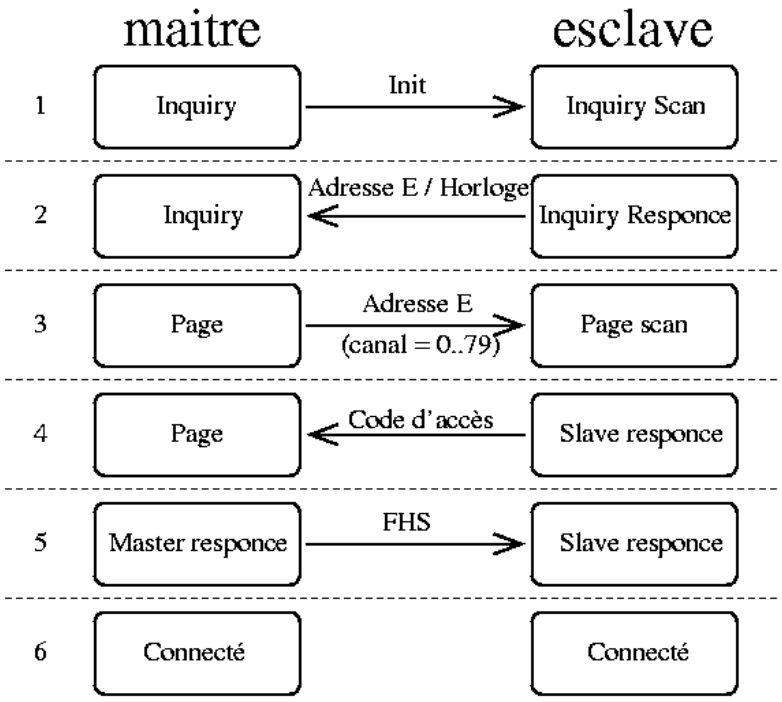

FIGURE 2.5 – Échange de message lors d'une connexion entre deux appareils

Mise en application d'une bibliothèque Bluetooth

3.1 Présentation de la librairie BlueZ

3.1.1 La librairie

Cette librairie a été d'abord développée par Max Krasnyansky pour l'entreprise Qualcomm. Ils ont décidé de la rendre publique en 2001. Un mois plus tard, elle était choisie par Linus Torvald pour être incluse dans le Kernel Linux.
C'est une librairie extrêmement vaste qui offre de nombreuses fonctionnalités (Source : Site Officiel) :

- Complete modular implementation
- Symmetric multi processing safe
- Multithreaded data processing
- Support for multiple Bluetooth devices
- Real hardware abstraction
- Standard socket interface to all layers
- Device and service level security support

Cette librairie n'est disponible que sur l'environnement Linux, c'est pourquoi nous avons réalisé nos tests sur cet OS.

3.1.2 Installation

Afin de pouvoir compiler notre code, il faut installer les packages supplémentaires fournis ici : http ://www.bluez.org/d
Le principe d'installation est assez simple, commencer par faire un "./configure" dans le dossier du package à installer. Le script peut donner une liste des packages manquant, qu'il faudra installer avant de poursuivre. Une fois toutes les dépendances installées, un "make" puis un "make install" finiront l'installation du package.
Une fois les 3 packages installés, on peut compiler le code.

3.1.3 Fonctions Essentielles

Nous n'allons pas détailler le code de chaque fonctions, le code commenté est fourni en annexe. Nous allons cependant les présenter grossièrement.

DétectionBluetooth : *hciinquiry*

Cette fonction permet, après configuration, de détecter les périphériques Bluetooth en mode "visible" à portée. Les périphériques sont stockés dans une structure de type "inquiry_info".
Les paramètres de la fonction sont les suivants :

int dev_id : L'id de l'antenne Bluetooth que l'on va utiliser. On récupère cette valeur avec la fonction hci_get_route(NULL). Il faut aussi ouvrir un socket sur cette antenne, avec la fonction hci_open_dev(dev_id)
int len : Coefficient définissant le temps pendant lequel la saisie va avoir lieu. (1,28 * len secondes)
int max_rsp : Nombre maximum de de devices que l'on va pouvoir stocker (taille de la structure).
NULL : Nous n'avons pas trouvé d'informations à ce sujet.
***inquiry_info ii :** Adresse de la structure de réception des devices. Cette structure doit être allouée avant.
int flags : Options spécifiques. (Gestion du cache etc)

Connexionàunserveur : *connect*

Cette fonction sert à se connecter à un serveur Bluetooth.
Ses paramètres sont les suivants :

int s : Socket Bluetooth utilisé pour se connecter. Le socket est créé par la fonction suivante : socket(AF_BLUETOOTH, SOCK_STREAM, BTPROTO_RFCOMM) qui renvoie un int.
***sockaddr_rc addr :** Cette structure contient l'adresse du serveur à contacter.
int sizeof(addr) : Taille de la structure.

La structure addr doit être initialisée comme ceci :

```
addr.rc_family = AF_BLUETOOTH;
addr.rc_channel = (uint8_t) 1;
str2ba( adrDest, & addr.rc_bdaddr );
```

On spécifie bien l'adresse MAC du serveur que l'on souhaite contacter.

Réceptiondesdonnées : *accept*

Cette fonction sert à récupérer une connexion entrante d'un client, et à créer une structure permettant ensuite de lire les informations envoyées par ce client.

int s : Un socket qu'il faudra avoir alloué puis associé à une antenne Bluetooth grâce à la fonction "bind".
***sockaddr rem_addr :** Structure qui contiendra l'adresse de la connexion entrante après acception de celle-ci.
socklen_t opt : Qui contient sizeof(rem_addr)

AssociationdeSockets : *bind*

Cette fonction sert à lier un socket à une antenne Bluetooth, on l'utilise dans la partie Serveur de l'application. Elle a les paramètres suivants :

int s : Le socket que l'on souhaite lier.
***sockaddr_rc loc_addr :** Adresse de l'antenne
int taille : sizeof(loc_addr)

loc_addr s'initialise de cette manière :
loc_addr.rc_family = AF_BLUETOOTH ; loc_addr.rc_bdaddr = *BDADDR_ANY ; loc_addr.rc_channel = (uint8_t) 1 ;

3.2 Notre application

Le but de notre application était de pouvoir localiser un matériel dans un bâtiment grâce à la technologie Bluetooth. Pour cela, nous avons procédé par étapes. Nous avons commencé par établir une liaison entre 2 machines via Bluetooth. Puis nous avons ajouté la notion de plusieurs machines connectées à un même serveur. Et enfin plusieurs machines se connectant à plusieurs serveurs.

3.2.1 1 Client 1 Serveur

Cette application a surtout pour but de transmettre un message entre 2 PC via Bluetooth. Pour cela nous avons utilisé une architecture Client/Serveur.

Notre programme est composé de 3 modules :

Détection

Ce module permet de détecter les périphériques Bluetooth à portée, affiche leur adresse MAC ainsi que leur "Bluetooth Name".

Serveur

Ce module lance un serveur se mettant en attente d'un paquet Bluetooth, l'affichant puis se fermant.

Client

Module demande a l'utilisateur de saisir l'adresse MAC d'un périphérique et essaye d'envoyer le message "Hello !" à ce périphérique.

3.2.2 N Clients 1 Serveur

Cette application a pour but de faire une liste des périphériques à portée d'un serveur.
Nous avons conservé la même architecture Client/Serveur, mais en modifiant les modules Serveur et Client.

Client

Ce module demande à l'utilisateur de saisir une adresse MAC, puis il va envoyer un paquet à ce serveur toutes les secondes. Ce module fonctionne avec une "while(1)", il faut donc l'arrêter avec un "Ctrl + C".

Serveur

Ce module lance un serveur qui va permettre de lister les périphériques à portée.
Pour cela, nous allons créer un tableau de char* qui contiendra les adresses MAC des périphériques à portée.

Le serveur utilise cet algorithme :

Algorithm 1 Serveur

while $ProgrammeEnCours$ **do**

for $Ide0)DELAI\_RAFRAICHISSEMENT$ **do**
Attendre Message Client

if $AdresseMacClientexistedansTableauPeriph$ **then**
RIEN

else
Ajouter AdresseMAC dans TableauPeriph

end if
Afficher TableauPeriph

end for
Effacer le Contenu de TableauPeriph

end while

Mise en pratique langage C

Pour coder cet algorithme en C, nous avons rencontré quelques soucis.
Premièrement, il y avait un soucis lorsqu'un seul périphérique envoyait des messages puis passait hors de portée. Le tableau comportait son adresse mais comme la fonction "accept()" bloque l'exécution du programme, empêchant donc le tableau de s'effacer.
Il aurait fallu pour cela rajouter une notion de temporisation à la boucle de rafraichissement. Nous avons d'abord pensé à gérer cela à l'aide d'un fork(), un des fork aurait géré le traitement des messages et l'autre l'affichage du tableau. Cependant, nous n'avons pas trouvé comment mettre ceci en pratique.
Nous avons finalement choisi de modifier la valeur du "timeout" de la fonction connect() et de permettre au programme de vider le tableau des périphériques si celle-ci est atteinte.
Nos recherches nous ont montré qu'il était très difficile de modifier ces timeout :

Unfortunately, BlueZ does not provide an easy way to change the packet timeout for a connection. A handle to the underlying connection is first needed to make this change, but the only way to obtain a handle to the underlying connection is to query the microcontroller on the local Bluetooth adapter.

3.2.3 N Clients N Serveurs

On a enfin modifié ce système pour mettre en place une architecture N Client N serveurs. On aurait plusieurs serveurs qui permettraient de localiser les machines. Il faudrait pour cela modifier le module Client.

Client

Ce module devra contenir une liste des serveurs avec qui communiquer. Il enverra alternativement des messages à chacuns de ces serveurs. Il faut donc connaitre à l'avance, l'adresse des serveurs que l'on souhaite interroger.

Évolutions possibles du sujet

4.0.4 N Clients N Serveurs + Table Centrale

Cette évolution rajoute en plus un serveur qui rassemble les tables de tous les serveurs, ce système permettrait par exemple de localiser les périphériques dans un bâtiment en ayant 1 serveur par pièce.

Il faudrait pour cela modifier le module Serveur.

Serveur

Ce module devra envoyer la table des périphériques à portée au serveur central. Ce serveur affichera alors toutes les listes des périphériques des serveurs du système. Ce système amène une nouvelle contrainte dans la gestion des serveurs, ils devront envoyer les tables des périphériques au serveur central à un intervalle constant. Cet envoi devra se faire via une technologie ayant une portée supérieure au bluetooth, par exemple Ethernet ou WiFi. En effet, on doit pouvoir placer les serveurs dans différentes pièces d'un bâtiment. Il faudra donc modifier l'algorithme du module.

Conclusion

Le RFID permet de scanner de nombreux objets en un laps de temps très court. Très performant dans le domaine de l'industrie, il permet d'accomplir des tâches simples, mais laborieuse si elles étaient faites manuellement.

Dans notre étude, on aurait pu comparer les avantages et inconvénient du RFID par rapport au Bluetooth. On peut tout de même noter que le RFID tend à être utiliser sur des animaux, du matériel ou de l'équipement. Les cas d'identification d'être humain par RFID créé des polémiques sur la violation de vie privée. Les puces sous-cutanées injectées aux animaux sont de la taille d'un grain de riz. Un tel dispositif chez l'humain aurait certes de nombreux avantages pour identifier les individus. Cependant, il pourrait être rapidement détourné à d'autres fins que ceux-ci.

L'énorme avantage que possède le Bluetooth par rapport au RFID, c'est qu'il peut être désactivé. Ainsi, si on ne souhaite pas être détecté lors d'un scan d'entourage, éteindre son téléphone ou tout simplement coupé l'application Bluetooth résout ce problème.

Avec la mise en place d'un système de serveur de détection Bluetooth à l'entrée de bâtiment, dès qu'un individu rentrerait, il activerait son terminal Bluetooth pour indiquer sa présence et ainsi permettre de savoir combien de personne doivent être évacué en cas d'incident.

Tracking RFID/Bluetooth

Département Informatique
3e année
2011 - 2012

Rapport de Projet de Système d'exploitation

Résumé : Rapport de Projet de Système d'exploitation sur la localisation d'objet dans un bâtiment à l'aide de lecteur RFID ou capteur Bluetooth

Mots clefs : RFID, Bluetooth, repérage, bâtiment

Abstract: Draft report operating system about tracking object in building with RFID reader or Bluetooth captor

Keywords: RFID, Bluetooth, tracking, building

étudiants
Jean de BAROCHEZ
jean.debarochez@etu.univ-tours.fr
Benoit BELZ
benoit.belz@etu.univ-tours.fr

Encadrants
Mathieu DELALANDRE
mathieu.delalandre@univ-tours.fr

Université François-Rabelais, Tours

DI3 2011 - 2012