REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de L'Enseignement Supérieure et de la Recherche Scientifique
Université Mouloud MAMMERI de Tizi-Ouzou
Faculté de Génie Electrique et d'Informatique
Département d'Informatique

MEMOIRE de FIN D'ETUDES

DE MASTER ACADÉMIQUE

Domaine : Mathématiques et informatique

Filière : Informatique.

Spécialité : ***Réseaux, Mobilité et Systèmes Embarqués****.*

Présenté par :

CHABA Lydia *TAMI Lydia*

Thème

Localisation indoor

Mémoire soutenu publiquement le 26/09/2017 devant le jury composé de :

Président : M^{r} M.DAOUI

Promotrice : M^{me} M.BELKADI

Examinatrice : M^{me} R.HADAOUI

Examinatrice : M^{me} R.AOUJIT

Promotion 2016/2017

Remerciements

Nous remercions le bon dieu pour le courage, la patience qui nous ont été utiles tout au long de notre parcours.

Nous tenons très sincèrement à remercier notre promotrice M^me^ BELKADI Malika de nous avoir encadrés, guidés et pour ses pertinents conseils.

Nous adressons nos remerciements les plus sincères aux membres du jury qui nous font l'honneur de juger notre travail.

Ces remerciements ne seraient pas complets si on n'avait pas pensé, à nos chers parents qui nous ont offert un environnement favorable afin de mener à terme notre travail.

Nous tenons à remercier très particulièrement BILAL et GHILES pour leur aide.

Enfin nous remercions tous ceux qui ont contribués de prés ou de loin afin que notre projet puisse voir le jour.

Dédicace

D'un cœur plein d'amour et de fierté, je dédie ce modeste travail à mes trois bougies qui brulent pour m'éclairer le chemin :

Á mon cher père qui n'a jamais cessé de combattre pour me voir réussir un jour, aux deux personnes qui me sont les plus chères au monde :
Ma mère et yemma qui m'ont élevés, éduqués et sacrifiés toutes les belles années de leurs vie pour moi, et, que dieu les protège pour nous.

Á mes chères sœurs : IMANE et MYLISSA
Á mon petit frère : SAID
Á ma chère cousine et ces petits anges : YASMINE, AMINE et RAYANE
Á mes oncles, à mes cousins et cousines
Á ma très chère copine : NINA
Á mon très cher ami : LAHCENE
Á ma camarade : LYDIA ainsi qu'à toute sa famille
Á tous mes ami(e)s
Á tous les étudiants de la promotion RMSE (2016/2017)
Á toutes les personnes qui m'ont aidé, soutenu et contribué de près ou de loin à la réalisation de ce modeste travail.

Lydia Chaba

Dédicace

D'un cœur plein d'amour et de fierté, je dédie ce modeste travail à mes deux bougies qui brulent pour m'éclairer le chemin :

Á mon cher père qui n'a jamais cessé de combattre pour me voir réussir un jour, à la personne qui est la plus chère au monde :
Ma mère qui m'a élevé, éduqué et sacrifié toute les belles années de sa vie pour moi, et, que dieu les protège pour nous.

Á ma bien aimée grand-mère
Á mon cher frère : YANIS
Á mon cher fiancé BILAL et sa famille, ainsi la regretté belle mère

Á mes chers cousins et cousines
Á mon Binou, ma Eva et Marina
Á ma camarade : LYDIA ainsi qu'à toute sa famille
Á tout mes ami(e)s
Á tous les étudiants de la promotion RMSE (2016/2017)
Á toutes les personnes qui m'ont aidé, soutenu et contribué de près ou de loin à la réalisation de ce modeste travail.

Lydia Tami

Liste des figures

Figure I.1 : Trilateration

Figure I.2 : La triangulation

Figure I.3 : Géolocalisation par angle d'arrivé

Figure I.4: Principe de géolocalisation par GPS

Figure I.5: Principe de géolocalisation par WIFI

Figure I.6: Principe de fonctionnement d'un système RFID

Figure I.7 : Système de localisation par ultrason et GPS

Figure II.1: L'Architecture de la plateforme Android

Figure II.2 : Etapes d'exécution d'un programme Android

Figure II.3 : Le cycle de vie d'une activité

Figure II.4 : Structure d'un projet Android

Figure III.1 : Plan 2D du laboratoire LARI

Figure III.2 : Représentation des points de calibrage sur le plan

Figure III.3 : K-Plus proches voisins

Figure III.4 : Cas d'utilisation « Se localiser »

Figure III.5: Diagramme entité / association

Figure IV.1 : Capture d'écran d'une fenêtre Android Studio

Figure IV.2 : AndroidManifest.xml

Figure IV.3 : Enregistrement des adresses mac dans la base de données

Figure IV.4 : Filtrage des points d'accès

Figure IV.5 : Création des tables de la base de données

Figure IV.6 : Calcul de la moyenne des puissances des signaux en temps réel

Figure IV.7 : L'objet mobile se trouve au couloire

Figure IV.8: L'objet mobile se trouve au labo 4

Liste des tableaux

Tableau I.1: Synthèse des techniques de mesures

Tableau II.1 : Tableau comparatif des versions officielles d'Android

Tableau II.2: Comparaison entre Android Studio et Eclipse

Tableau III.1 : Calcul de la moyenne pour le point de référence (5,10.4)

Tableau III.2 : Calcul de la moyenne des points de référence du Labo 3

Tableau III.3 : Calcul de la moyenne des points de référence du Labo 4

Tableau III.4 : Spécification de la tâche

Tableau III.5 : Spécification du scénario

Sommaire

Introduction générale

Chapitre I : Généralités sur la géolocalisation

Introduction 1

1. Définition 1
2. Méthodes de géolocalisation 1
   2.1. Méthodes géométriques 1
      2.1.1.La trilateration 1
      2.1.2.La triangulation 2
   2.2.Méthodes statiques 3
      2.2.1.Empreintes radio 4
   2.3.Méthodes hybrides 4
3.Techniques de mesures 4
   3.1.Identifiant de cellule (CID) 4
   3.2.Temps d'arrivé (ou ToA pour Time of Arrival) 5
   3.3.Différentiel d'arrivée (ou TDoA pour Time Difference of Arrival) 5
   3.4.Angle d'arrivée (ou AoA pour Angle of Arrival) 5
   3.5.Puissance du signal reçu (ou RSSI pour Received Signal Strength Indicator) 6
4.Les systèmes de positionnements 8
   4.1.Système de positionnement Outdoor 8
      4.1.1.Géolocalisation par GSM (Global System for Mobile Communications) 8
      4.1.2.Géolocalisation par GPS (Global Positioning System) 8
   4.2.Système de positionnement indoor 9
      4.2.1.Géolocalisation par adresse IP 10
      4.2.2.Géolocalisation par WIFI 10
      4.2.3.Géolocalisation par RFID 11
      4.2.4.Géolocalisation par Beacon 12
      4.2.5.Localisation par ultrason 13
      4.2.6.Localisation par infrarouge 13
5.Combinaison de différentes techniques 13
6.Exemples de systèmes existant 14

6.1.Drishti 14
6.2.Système de localisation à infrarouge 15
6.3.Système de localisation par RFID 15
6.4.Système Ekahau Real Time Location System (RTLS) 15
Conclusion 16

Chapitre II : Android

Introduction 17

1. Définition 17
2. Caractéristiques de système Android 17
3. Historique et versions d'Android 18
4. Architecture d'Android 20
4.1. Applications 20
4.2. Le framework (Application Framework) 20
A) CorePlateform Service 20
B) Hardwar Services 21
4.3. Les bibliothèques (Librairies) 21
4.4. Le moteur d'exécution Android (Android Runtime) 22
4.5 Le noyau Linux (Linux Kernel) 23
5. Le Développement d'applications Android 23
5.1. Définition 23
5.2. Les composants d'une application android 23
5.2.1. Activité 23
5.2.2. Les Services 25
5.2.3. Le Broadcast Receivers 25
5.2.4. Le Content providers 25
5.3. La structure d'un projet android 26
5.4. Comparaison entre Android Studio et Eclipse 27
Conclusion 28

Chapitre III: Etude, Analyse et Conception

Introduction 29

1.Présentation du laboratoire LARI 29
2.Etude 30

2.1.Phase de calibrage....30
2.1.2.Eléments nécessaires....30
2.1.3.Fonctionnement....30
2.2.Phase de positionnement....35
2.2.1.Méthode de positionnement....35
2.2.1.1. Méthode du plus proche voisin....35
3.Analyse et conception....37
3.1. Présentation d'UML....37
3.1.1. Historique....37
3.1.2. Définition....37
3.1.3.Modélisation avec l'UML....38
3.2.Analyse....38
3.2.1. Spécification des besoins....38
3.2.1.1. Identification de l'acteur....38
3.2.2. Spécification de la tâche....39
3.2.3. Spécification du scénario....39
3.2.3.1. Définition d'un scénario....39
3.2.3.2. Le scénarios de notre système....39
3.2.4. Les cas d'utilisations....40
3.2.4.1. Définition d'un cas d'utilisation....40
3.2.4.2. Spécification de cas d'utilisation....40
3.3. Conception....40
3.3.1. Conception de la base de données....40
3.3.1.1. Diagramme entités/associations....41
3.3.1.2. Schéma de la base de données....41
3.3.2. Le modèle relationnel de la base de donné……42
Conclusion....43
Chapitre IV: Réalisation
Introduction....44
1.Environnement et outils de développement....44
1.1.Partie matérielle....44
1.2.Partie logiciel....44
1.2.1.Android Studio 2.3....44

1.2.2.Le SDK android 45
1.2.2.1. Émulateur 45
1.2.3.Android Virtual Device (AVD) 45
1.2.4.Le langage de programmation Java..... 46
1.2.5.La base de données SQLite 46
2.Fonctionnement de l'application 46
3.Présentation de l'application 50
Conclusion 53
Conclusion générale
Bibliographie
Webliographie

Introduction générale

Se déplacer est une nécessité vitale pour chaque personne. De tout temps, l'homme a développé des moyens techniques pour faciliter ses déplacements et augmenter son autonomie : depuis la carte et la boussole, les systèmes d'assistance à la navigation ont connu un formidable essor depuis les années 2000 avec l'avènement du GPS (Global Positioning System). Mais les signaux de ce dernier parviennent difficilement à traverser les murs et les toits de nos bâtiments, rendant ainsi impossible son utilisation à l'intérieur. De là s'est ressenti le besoin de trouver des systèmes de substitution qui franchissent la limite du GPS et combinant l'efficacité et la précision à l'intérieur. Parmi ces systèmes on peut citer la technique RFID (Radio Frequency Identifier) qui nécessite la mise en place d'un réseau spécifique, le Beacon qui se base sur la technologie Bluetooth et le Wifi.

Notre travail consiste à la mise en place d'une application mobile de géolocalisation indoor sous android, au niveau du laboratoire LARI de l'université Mouloud Mammeri, qui va permettre à l'utilisateur de se localiser à travers un plan.

Pour bien présenter notre travail, nous l'avons structuré comme suit :

Chapitre I : « Généralités sur la géolocalisation » où nous avons fait une description détaillée des principaux systèmes de géolocalisation ainsi que leurs modes de fonctionnement.

Chapitre II : « Android » où nous avons fait une description détaillée du système android.

Chapitre III : « Etude, Analyse et Conception» décrit les phases d'analyse et de conception de notre application.

Chapitre IV : «Réalisation» qui est consacré à la réalisation et l'implémentation de notre application**.**

Chapitre 1 :

Généralités sur la géolocalisation

Chapitre I : Généralités sur la géolocalisation

Introduction

Les hommes ont toujours eu besoin de se localiser et de se situer dans l'environnement. Pour répondre à cette nécessité, plusieurs techniques ont été utilisées.

Dans ce chapitre, il sera question d'introduire le cadre théorique dans lequel s'inscrit notre travail, en définissant dans un premier temps la géolocalisation, ensuite nous présentons les différentes méthodes de cette dernière, ainsi les techniques de mesure et enfin nous terminons par les systèmes de positionnement les plus conventionnels.

1. Définition [10]

La géolocalisation (ou géoréférencement) est un procédé permettant de positionner un objet ou une personne sur un plan ou une carte à l'aide de ses coordonnées géographiques. Cette opération est réalisée à l'aide d'un terminal capable d'être localisé (grâce à un système de positionnement par satellites ou d'autres techniques) et de publier ses coordonnées géographiques (latitude, longitude). Les positions enregistrées peuvent être stockées au sein du terminal ou être transmises en temps réel vers une plateforme logicielle de géolocalisation. La transmission temps réel nécessite que le terminal soit équipé d'un moyen de télécommunication de type GSM, GPRS, UMTS, radio ou satellite lui permettant d'envoyer les positions à des intervalles réguliers. Ceci permet de visualiser la position du terminal sur une carte à travers une plateforme de géolocalisation le plus souvent accessible depuis internet.

2. Méthodes de géolocalisation

Les méthodes de géolocalisation peuvent être divisées en trois catégories: méthode géométrique, méthode statique et méthode hybride.

2.1. Méthodes géométriques [2]

Elles utilisent les théorèmes géométriques sur les relations dans les triangles en particulier, nous citons :

2.1.1. La trilateration

La trilateration consiste à utiliser uniquement les distances pour localiser l'objet cherché**.** Ce principe utilise le plus souvent le système de signal radio. Il se base sur la connaissance de la vitesse de propagation de l'onde.

Les points autour de celui que l'on cherche à connaître émettent une onde, à partir de ce moment, un chronomètre se déclenche. Lorsque l'objet inconnu reçoit l'onde, il la renvoie vers son émetteur. Quand celui-là la capte, il stoppe le chronomètre. Le calcul devient alors simple (en théorie) :

$$v = \frac{d}{t} \leftrightarrow d = v \square t$$

Où : **v** : la vitesse
d : la distance
t : le temps

Or c'est un aller-retour que l'onde fait, il suffit donc de diviser la distance obtenue et nous obtenons la distance entre le point inconnu et l'émetteur.
Une fois les distances connues, il suffit de tracer des cercles de centres des points connus et de rayons qui sont égaux à la distance propre de chaque point à l'objet inconnu.
Le point d'intersection de ces cercles indique l'endroit où se situe le point recherché (cible x).
Comme le montre la ***figure I.1*** :

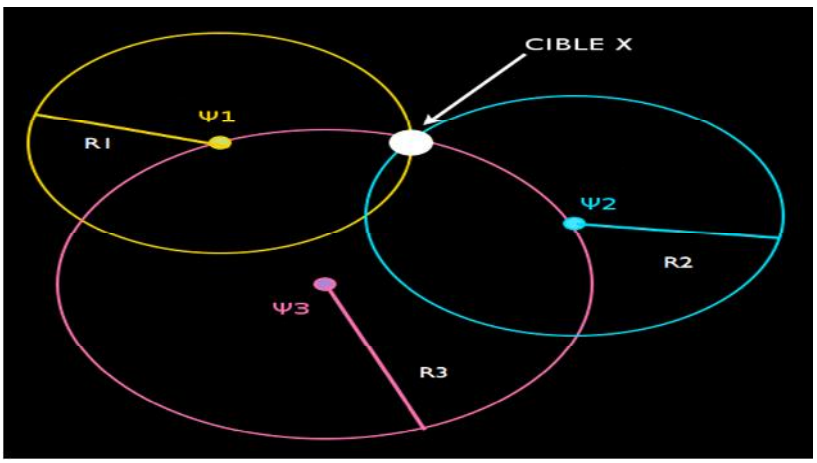


Figure I.1 : Trilateration.

2.1.2. La triangulation

Dans cette méthode, deux Points de Références de coordonnées connues forment avec l'Objet Mobile (OM) un triangle. On utilise ensuite les propriétés géométriques des triangles pour calculer la position de l'Objet Mobile (OM) : la loi des sinus, le théorème de Pythagore, …etc.
Cette méthode utilise donc moins de Points de Références (PR) que la méthode de trilateration mais nécessite la connaissance de plus d'informations liant les trois nœuds. Il est nécessaire de connaître les angles du triangle formé par les trois nœuds et la distance entre les points de référence.
Pour illustrer cette technique, basons-nous sur un exemple concret :
Soit un triangle ABC. Le point que nous cherchons est le point B. Nous connaissons AC, ainsi que les angles $\hat{a}$ et $\hat{c}$. Illustrer par la ***figure I.2*** :

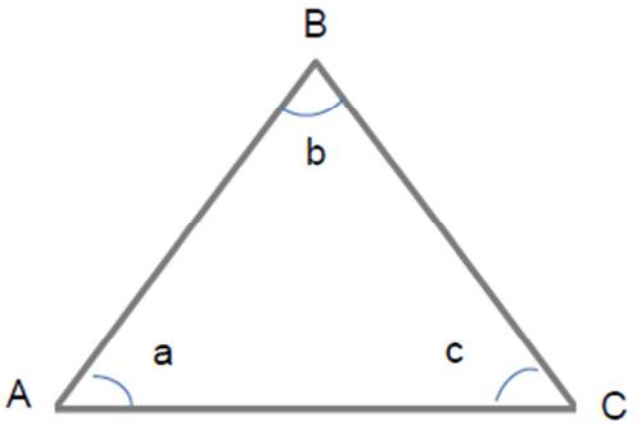


Figure I.2 : La triangulation.

L'angle $\hat{b}$ peut être calculé, sachant que la somme des angles d'un triangle est égale à 180°, il suffit de faire **180-(a+c).**
Ensuite, on utilise la loi des sinus:

$$\frac{AC}{\sin b} = \frac{AB}{\sin c} = \frac{BC}{\sin a}$$

Ainsi, grâce à deux produits en croix, nous connaissons les distances AB et BC.

Cette technique est utilisée surtout en télémétrie optique, notamment dans le domaine militaire, en l'absence de radars, à l'aide de télémètre. Elle permet d'évaluer des distances importantes, mais nécessite pour cela une distance importante entre les deux télémètres pour obtenir une précision accrue de la mesure des angles.

2.2. Méthodes statiques [3]

Ces méthodes consistent à effectuer des mesures « références », puis à comparer les mesures réalisées pour l'Objet Mobile (OM) avec ces mesures de références pour en déduire sa position.
Ces approches comprennent donc deux étapes: une étape de **calibrage** ou **d'apprentissage** et l'étape **d'estimation de la position**. Une des méthodes les plus efficaces est la méthode **d'empreinte radio** (en anglais RF fingerprinting), qui se base sur les mesures des puissances reçues (RSSI pour Received Signal Strengh Indicator) par l'OM à partir de balises radios (comme les points d'accès Wifi par exemple). Cette méthode a été proposée en 2000 par P. Bahl et V. Padmanabhan dans un premier système appelé RADAR utilisant les ondes radiofréquences émises à partir d'interfaces réseaux (WIFI).

2.2.1. Empreintes radio [4]

La méthode d'empreintes radio se fait en deux phases :

✓ **Phase de formation :** appelée aussi calibrage. Des balises radio sont installées sur le site désiré, des points échantillons (des postions sur le site) sont choisis et les RSSI mesurés sont sauvegardés dans une base de données.

✓ **Phase de positionnement (localisation) :** La position de l'Objet Mobile est déterminée en comparant les RSSI mesurés par l'OM avec les RSSI sauvegardés dans la base de données. Plus le nombre de balises radios augmente, plus la précision s'améliore. Plusieurs mesures successives de puissances doivent être réalisées pour une meilleure évaluation de la position, les ondes étant affectées par plusieurs facteurs (la réflexion, la réfraction, la diffraction, la diffusion et les interférences).

2.3. Méthodes hybrides [3]

Ces méthodes comportent deux étapes : dans la première étape, elles utilisent la méthode d'empreinte radio avec une phase rapide de formation (ceci indique qu'on a peu de PR (Points de Références) et que la base de données est très petite) pour obtenir une première évaluation de positionnement (indiquant par exemple dans quelle pièce du bâtiment se trouve OM). Dans la deuxième étape, un modèle empirique précis de la propagation du signal sera utilisé pour calculer la distance exacte entre l'émetteur et le récepteur. La triangulation pourra alors être utilisée pour calculer la position de l'OM plus précisément.

3. Techniques de mesures

Les distances et les angles dans les méthodes de positionnement déjà citées sont mesurés selon plusieurs techniques de mesure, on peut citer : Identifiant de cellule (**CID)**, temps d'arrivée (TOA), différence de temps d'arrivée (DTOA), angle d'arrivée (AOA) et puissance du signal reçu (RSS- Received Signal Strength).

3.1. Identifiant de cellule (CID) [5]

Aussi appelé **Cell ID (Identification de la cellule radio).** Cette méthode consiste à récupérer les identifiants des antennes GSM auxquelles le terminal est connecté. Par la suite, grâce à une base de données faisant le lien entre les identifiants des cellules et les positions géographiques des antennes, le terminal est capable de déterminer sa position et d'émettre une estimation.

Ces bases de données peuvent être mises à disposition par les opérateurs pour leurs abonnés, ou par des sociétés privées qui recensent les antennes GSM ou ayant des partenariats avec les opérateurs.

Des bases de données communautaires existent et sont le plus souvent alimentées par les utilisateurs eux-mêmes.

Étant donné que les bases de données Cell ID ne sont pas stockées localement dans le terminal, une connexion internet de type GPRS/EDGE ou 3G peut être nécessaire afin d'émettre une requête pour obtenir la correspondance Cell ID / longitude et latitude.

3.2. Temps d'arrivé (ou ToA pour Time of Arrival) [3]

Cette technique se base sur le temps de propagation du signal. La source envoie un signal daté qui est reçu par les récepteurs, ces derniers vont dater à leurs tours l'heure d'arrivée du signal. Un système de géolocalisation va alors se baser sur ces informations pour positionner la source, en supposant le plus souvent que la propagation se fait en ligne directe. Une synchronisation complète entre l'émetteur et le récepteur est nécessaire pour des calculs précis.

L'inconvénient majeur de cette technique est la nécessité d'une synchronisation temporelle efficace entre les points de référence et le récepteur. Un défaut de synchronisation résulte à des erreurs dans le calcul du temps pris par le signal et en conséquence conduit à des erreurs de localisation.

3.3. Différentiel d'arrivée (ou TDoA pour Time Difference of Arrival) [3]

Le principe est similaire au ToA, mais en utilisant une source quelconque qui ne date pas son émission. Le signal est alors reçu par les récepteurs et le système de géolocalisation détermine la position de la source en fonction de la différence des temps d'arrivée sur les récepteurs. Cette méthode exige de même une horloge très précise et très bien synchronisée au niveau des récepteurs. Cette solution est bien adaptée dans les environnements ouverts où le signal se propage en ligne directe. Comme l'approche TOA, elle peut connaître des limites en intérieur à cause des obstacles (le signal pouvant rebondir sur les murs avant d'atteindre un récepteur) et des effets de réflexion, réfraction ou diffusion.

3.4. Angle d'arrivée (ou AoA pour Angle of Arrival) [11]

Cette technique est basée sur l'estimation d'angles d'arrivée utilisant des antennes directionnelles pour mesurer la direction d'arrivée du signal provenant du trajet direct, émis par le mobile. Cette technique est illustrée par la ***figure I.3.*** Les directions du trajet direct forment, respectivement, un angle et avec un axe prédéfini aux stations de base. Leur intersection donne la position exacte du mobile. Cependant, étant donné que les antennes de réception disposent d'une marge d'erreurs ± θ, alors chaque station de base localise le mobile plutôt dans un faisceau égale à la direction mesurée plus ou moins la marge d'erreur. Le mobile se trouve ainsi dans la région formée par l'intersection des deux faisceaux $\mathbf{G_1G_2}$.

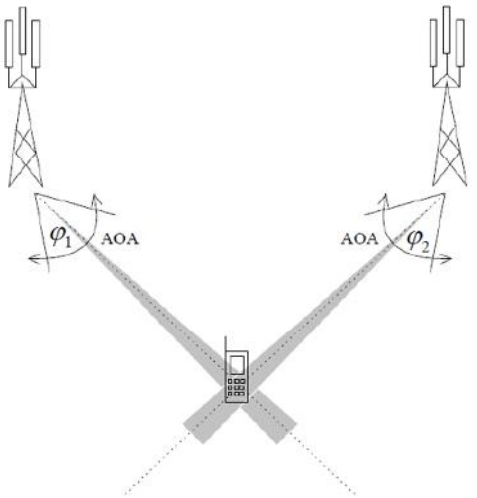


Figure I.3: Géolocalisation par angle d'arrivé.

Le principal inconvénient de la technique AoA est la nécessité de disposer de réseaux d'antennes qui augmentent la taille et les coûts des équipements. L'autre point négatif est la forte influence des trajets multiples sur la précision de l'estimation.

3.5. Puissance du signal reçu (ou RSSI pour Received Signal Strength Indicator)

Cette méthode est basée sur le calcul de la distance entre l'émetteur et le récepteur à partir de la puissance du signal reçu associée à un modèle de propagation dans l'environnement. En effet, la puissance du signal direct reçu par un récepteur est la fonction de la distance **d** séparant l'émetteur et le récepteur.

La méthode de RSSI est très utilisée pour les systèmes de localisation urbain, rural et aussi à l'intérieur. Le principal désavantage de l'utilisation du RSSI est l'imprécision de la localisation en présence de multi-trajets dans l'environnement. Par exemple, pour la géolocalisation indoor, il faut estimer les perturbations liées aux obstacles comme les murs, cloisons, vitres, équipements électromagnétiques, etc. Quand l'application est à l'extérieur les perturbations sont moindres et le calcul de l'atténuation est simplifié. Par conséquent, les algorithmes de positionnement basés sur le RSSI sont sensibles à l'estimation des paramètres de l'environnement.

Synthèse des techniques de mesures

Un résumé des principales techniques de mesures et leurs principaux avantages et inconvénients est donné dans le tableau suivant:

Technique de mesures	Avantages	Inconvénients
RSSI (Puissance du signal reçu)	- Coût d'implantation peu élevé -Disponibilité des modèles mathématiques d'atténuation -Algorithme de positionnement simple	-Nécessité d'avoir le trajet direct - Précision faible -Performance mauvaise dans un canal ayant un profil de propagation par trajets multiples.
AoA (Angle d'arrivée)	- Moins de stations de base fixes nécessaires - Algorithme de positionnement simple	-Nécessité d'avoir le trajet direct - Coût d'implantation élevé -Précision faible -Performance mauvaise dans un canal ayant un profil de propagation par trajets multiples
TOA (Temps d'arrivée)	-Paramètres généralement bien estimés -Algorithme de positionnement simple -Précision plus élevée en milieu confiné	-Synchronisation d'horloge nécessaire entre le mobile et les stations de base -Nécessité d'avoir le trajet direct -Nécessité d'une résolution temporelle élevée au récepteur
TDOA (Différence des temps d'arrivée)	-Paramètres généralement bien estimés - Algorithme de positionnement simple -Précision plus élevée en milieu	-Nécessité d'avoir le trajet direct -Synchronisation d'horloge nécessaire entre les paires de stations de base - Nécessité d'une résolution

	confiné -Pas besoin de synchronisation d'horloge entre le mobile et les stations de base	temporelle élevée au récepteur
Technique basée sur les empreintes	- Implantation facile -Prise en compte du profil de propagation par trajets multiples - Précision généralement plus élevée en milieu interne	- Ne peut être utilisée que dans un espace limité - Les performances se dégradent dans un canal non stationnaire - Les empreintes ne sont pas toujours uniques et répétitives

Tableau I.1: Synthèse des techniques de mesure.

4. Les systèmes de positionnement

4.1. Système de positionnement Outdoor

4.1.1. Géolocalisation par GSM (Global System for Mobile Communications) [6]

Cette technique permet le positionnement d'un terminal GSM en se basant sur certaines informations relatives aux antennes GSM auxquelles le terminal est connecté. La précision du positionnement par GSM peut aller de 200 mètres à plusieurs kilomètres, selon si le terminal se trouve en milieu urbain (où la densité d'antennes est supérieure), ou en milieu rural. La méthode GSM la plus utilisée est celle du **Cell ID**.

4.1.2. Géolocalisation par GPS (Global Positioning System) [3]

Le système GPS est un système d'origine américain qui se base sur la méthode de trilatération et sur le temps de propagation du signal pour déterminer la position en 3D d'un objet sur le globe. Il se compose de trois parties distinctes appelées segments : le segment spatial constitué actuellement d'une constellation de 31 satellites, le segment de contrôle qui permet de surveiller et de contrôler en permanence le bon fonctionnement du système et le segment utilisateur qui regroupe l'ensemble des récepteurs GPS qui reçoivent et exploitent les informations des satellites.

Le principe de localisation est en lui-même très simple. En effet, si on imagine de vouloir localiser un point M, de la surface du globe terrestre, il suffit d'entrer en contact avec 3 satellites (4 si on veut un positionnement en 3D). Chaque satellite envoie son numéro d'identification, sa position précise par rapport à la terre, ou dans le repère lié à Greenwich,

l'heure exacte d'émission du signal ; le récepteur GPS recevant le signal, grâce à son horloge supposée synchronisée sur celle des satellites, en déduit sa distance au satellite. Le GPS travaille en 3D et le principe de calcul s'appuie sur l'intersection de sphères : le point M est sur une sphère de rayon D1 et de centre le satellite S1, l'intersection avec le globe donne un premier cercle C1 ; Le point M est aussi sur une sphère de rayon D2 et de centre le satellite S2 et sur une sphère de rayon D3 et de centre le satellite S3 ; l'intersection de ces sphères avec le globe donne deux autres cercles C2 et C3 ; le point M se trouve alors à l'intersection de ces trois cercles. Illustrer par la ***Figure I.4*** :

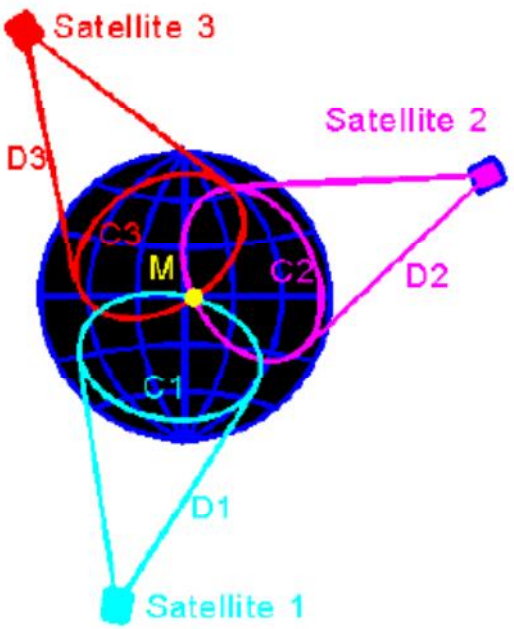


Figure I.4: Principe de géolocalisation par GPS.

La précision fournie par le GPS est estimée aux alentours de 15 mètres dans 95% des cas pour un fonctionnement de type « temps réel ». Il est cependant difficile de donner des chiffres de précision sans préciser les conditions et méthodes de mesure (en particulier la durée de ces mesures). Cette précision dépend de la visibilité des sattellites par le récepteur et de la qualité de l'antenne utilisée. Le GPS est l'un des systèmes de positionnement les plus populaires en milieu extérieur ; Cependant, il n'est pas approprié à certains environnements spécifiques, tels qu'à l'intérieur des bâtiments ou dans certains environnements urbains denses**.**

4.2. Système de positionnement indoor

La géolocalisation indoor ou la géolocalisation à l'intérieur est une technique qui permet de localiser en temps réel des biens ou des personnes dans des espaces fermés tels que les centres commerciaux, aéroports, hôpitaux, usines, complexes militaires ou industriels, etc...

Grâce aux nouveaux outils de mobilité : smartphone professionnel, tablette tactile ... et aux nouvelles technologies : WiFi, Bluetooth ou Ultra Wide Band et RFID (Radio Frequency IDentification), les utilisateurs peuvent se repérer facilement à l'intérieur d'un bâtiment. Ses informations peuvent être utilisées également par les propriétaires de sites pour suivre le

parcours des clients en temps réel et interagir avec eux en leur proposant des services et déclencher des actions de marketing.

Les systèmes de géolocalisation indoor permettent de connaitre avec une précision plus ou moins grande le positionnement d'une personne ou d'un objet dans un espace ou un lieu dans lequel l'accès aux satellites et les données GPS ne sont pas disponibles. Les réseaux sans fil sont une des solutions les plus envisagées pour la localisation à l'intérieur. **[1]**

Nous présentons dans ce qui suit les différentes technologies utilisées :

4.2.1. Géolocalisation par adresse IP [6]

Cette méthode permet de déterminer la position géographique d'un ordinateur ou de n'importe quel terminal connecté à internet en se basant sur son adresse IP. Les adresses IP sont gérées par l'IANA (Internet Assigned Numbers Authority), une organisation qui s'occupe de découper les blocs d'adresses IP disponibles et de les distribuer de façon très contrôlée aux pays qui en demandent. Toutes ces attributions étant très bien documentées, il est possible de savoir dans quel pays se trouve un terminal connecté à internet grâce à son adresse IP. On peut même obtenir un niveau de précision de l'ordre de la ville en se basant sur la distribution des adresses IP faite par les fournisseurs d'accès à internet. On peut citer comme exemple Lookup City qui est un outil en ligne effectuant cette géolocalisation.

4.2.2. Géolocalisation par WIFI [7]

Les points d'accès WiFi, émettent en permanence des signaux permettant à des ordinateurs ou à des téléphones mobiles de les reconnaître et de s'y connecter. Ces signaux contiennent notamment un numéro d'identification unique propre à chaque point d'accès (appelé "BSSID"), ce numéro est utilisé par les Smartphones pour géolocaliser une personne.

Fonctionnement de la géolocalisation à partir des points d'accès wifi :

Lorsqu'une personne lance une application de géolocalisation sur son smartphone, celui-ci peut lister les points d'accès WiFi à sa portée, et interroger une base de données qui permet d'associer ce point d'accès à une position géographique. Le smartphone va donc être capable de géolocaliser précisément le propriétaire du smartphone. Illustrer par la ***figure I.5***

Plusieurs sociétés telles que Google, Skyhook Wireless, Microsoft et Apple ont donc constitué des bases cartographiques recensant ces points d'accès WiFi pour fournir leurs services de géolocalisation.

Ces bases peuvent être créées :

- À partir des données transmises par les téléphones mobiles eux-mêmes lorsqu'ils demandent à être géolocalisés.

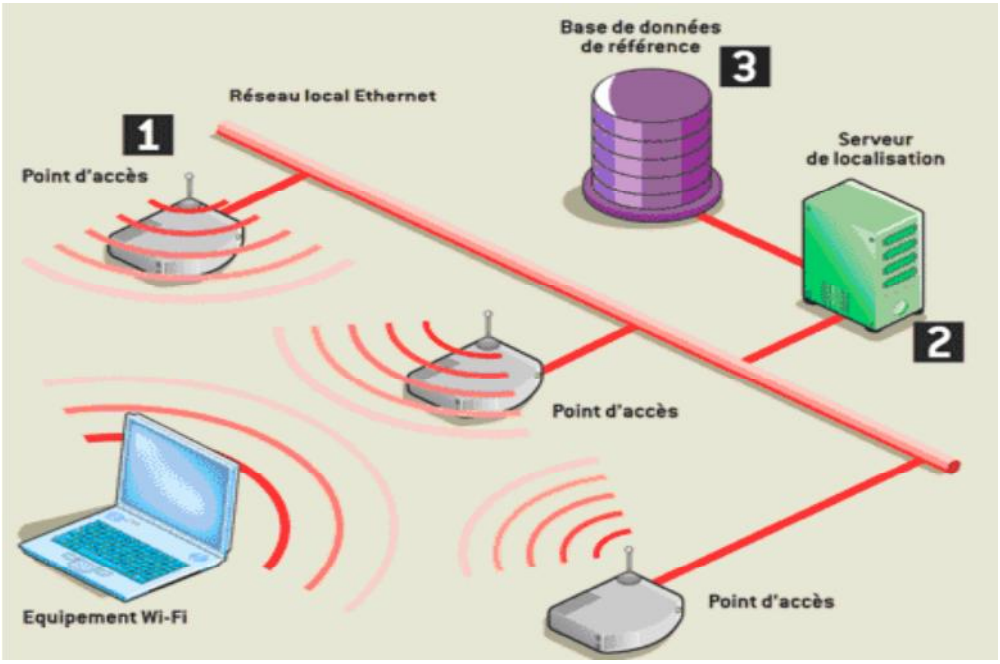


Figure I.5: Principe de géolocalisation par WIFI.

4.2.3. Géolocalisation par RFID [14]

L'identification par radiofréquence ou RFID est une technologie sans fil qui est développée pour identifier des objets, elle permet au moyen d'un lecteur radiofréquence (émet des ondes radio), de mémoriser et récupérer les données contenues sur des étiquettes (transpondeur ou tag) électroniques qui sont posées sur des objets ou y sont intégrées. Le système RFID se compose principalement de tag (l'étiquette ou transpondeur) et de lecteur.

Principe d'identification par radio fréquence : [15]

Un système RFID se compose principalement d'un lecteur, d'un tag (appelé aussi transpondeur ou étiquette) qui est composé d'une puce et d'une antenne, et d'un logiciel d'application. La ***figure I.6*** décrit le schéma général d'un système RFID.

Le lecteur active le tag (tag passif ou semi-passif) qui se trouve dans la zone de lecture, en lui envoyant des ondes électromagnétiques qui induisent le courant dans l'antenne de l'étiquette, celle-ci réagit à la réception du signal envoyé par le lecteur en lui envoyant l'information demandée.

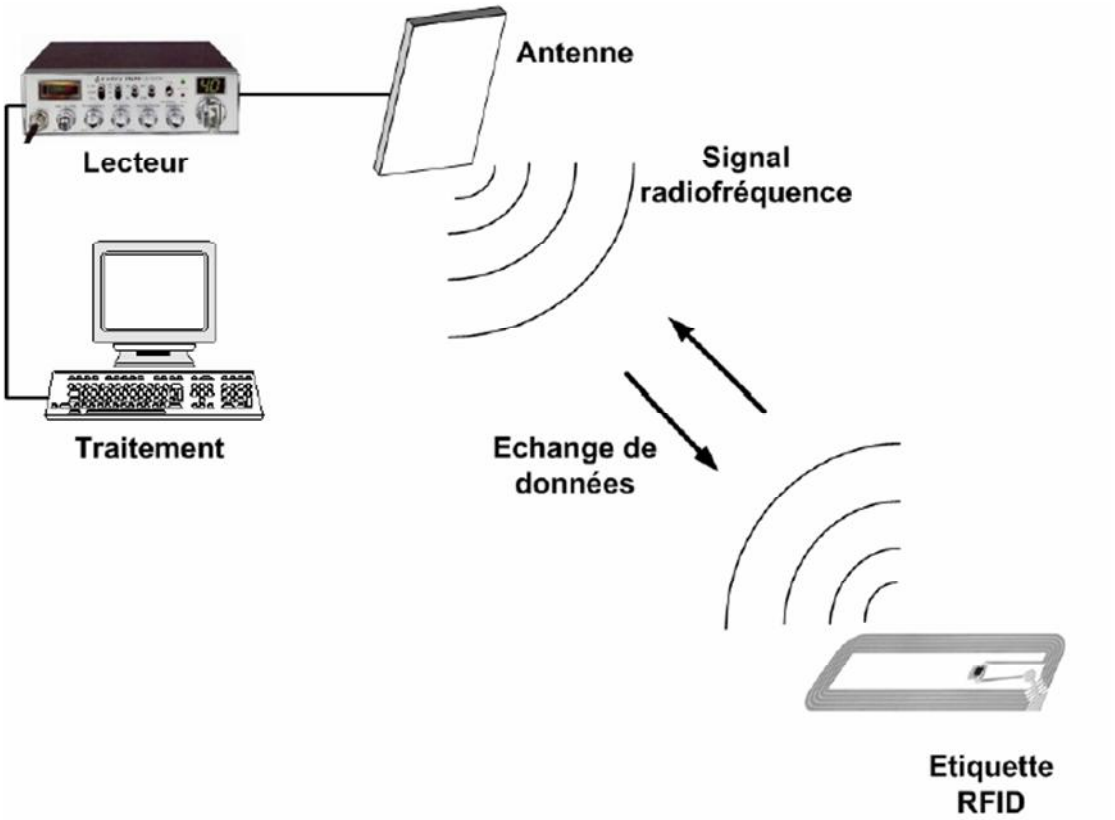


Figure I.6: **Principe de fonctionnement d'un système RFID.**

Il existe deux grandes familles d'étiquettes RFID : **[5]**

- **Les étiquettes actives :** elles sont équipées d'une batterie leur permettant d'émettre un signal. Les étiquettes actives possèdent une meilleure portée mais à un coût plus élevé et avec une durée de vie plus restreinte.

- **Les étiquettes passives :** elles utilisent l'énergie propagée à courte distance par le signal radio de l'émetteur. Ces étiquettes à moindre coût sont généralement plus petites et possèdent une durée de vie quasi-illimitée. En contrepartie, elles nécessitent une quantité d'énergie non négligeable de la part du lecteur pour pouvoir fonctionner.

La technologie RFID est beaucoup utilisée, en particulier en logistique, dans plusieurs applications dont on peut citer : le suivi des colis, l'identification des conteneurs, la gestion des stocks en particulier en e-commerce et la gestion de production.

4.2.4. Géolocalisation par Beacon

Beacon signifie simplement balise. Une balise est un boîtier de quelques centimètre qui peut être installé où l'on veut et qui émet dans un rayon jusqu'à quelques dizaines de mètres en Bluetooth Low Energy. Ces balises se signalent aux smartphones qui sont dans leurs environnement et leur permettent de se géolocaliser précisément.

La première action du beacon est de se signaler en envoyant ses données.

Cette radiodiffusion est émise à intervalles réguliers par le périphérique. Évidemment, cet intervalle aura un impact sur l'autonomie et la détection de l'appareil. L'objectif ici est de se faire connaître par plusieurs appareils en même temps, pour par exemple, détecter la proximité d'un lieu.
Il est important de noter qu'un récepteur peut se connecter à plusieurs beacon en même temps.

4.2.5. Localisation par ultrason [12]

La plupart des systèmes de localisation par ultrason sont combinés avec une autre technologie afin d'obtenir une estimation de la distance émetteur/récepteur. Le récepteur reçoit successivement l'onde ultrason et l'autre onde choisie (GPS par exemple). Il effectue une corrélation de ces deux signaux reçus pour extraire la différence des temps d'arrivée. Ceci permet d'estimer la distance le séparant de l'émetteur. En réitérant cette même mesure avec plusieurs émetteurs, on détermine précisément la position du mobile dans l'environnement.

4.2.6. Localisation par infrarouge

Le mobile à localiser est équipé d'un tag infrarouge émettant un signal infrarouge à intervalles réguliers. Les récepteurs sont installés au plafond dans chaque pièce de l'environnement. Ces récepteurs sont reliés entre eux pour former un réseau permettant de détecter le tag actif. Cette technologie n'a pas été retenue à cause de la non-robustesse du signal infrarouge. En effet, les signaux infrarouges émis ne traversent pas les murs.

5. Combinaison de différentes techniques [6]

Il existe plusieurs inconvénients à l'utilisation d'une seule technique de géolocalisation :

- **La dépendance au réseau GPS :** l'incapacité de l'utiliser en intérieur et le temps de réponse à l'allumage.

- **La dépendance au réseau GSM :** sa couverture géographique, l'accès au réseau GPRS pour exploiter l'information.

- **La dépendance à la présence de bornes d'accès WiFi :** en zone rurale par exemple.

Des dispositifs qui combinent ces trois techniques et qui sont capables de géolocaliser le terminal dans n'importe quelle situation existent. La précision de ce positionnement va varier en fonction des technologies disponibles, mais le temps de réponse à l'allumage et l'adaptabilité s'en verront améliorés. Ceci permet par exemple de géolocaliser une personne à l'extérieur en utilisant le GPS, et de garder sa trace à l'intérieur des bâtiments ou des tunnels en utilisant la technologie GSM couplée au WiFi pour plus de précision.

6. Exemples de systèmes existant

6.1. Drishti

Drishti est un système de navigation pédestre proposé, par une équipe de jeunes chercheurs Ran L, Helal S, de l'Université de Floride à Gainesville, pour guider les déficients visuels au cours de leurs déplacements au cœur du campus ou bien à l'intérieur de ses bâtiments. Ce dispositif associe le GPS différentiel (DGPS) à un Système d'Information Géographique (SIG), pour la localisation à l'extérieur et utilise un système de positionnement ultrason pour la localisation à l'intérieur : le récepteur est composé de 2 balises attachées aux épaules de l'utilisateur alors que les émetteurs se sont 4 pilotes ultrasons montées dans les quatre coins du bâtiment, pour fournir les mesures de la localisation de la personne. Pour la localisation à l'intérieur, les résultats montrent que sur 22 tests, qui ont été menés dans différentes localisations, l'erreur maximale enregistrée a été 22cm avec 12 cas inférieurs à 10cm de la position réelle. Illustrer par la ***figure I.7*** :

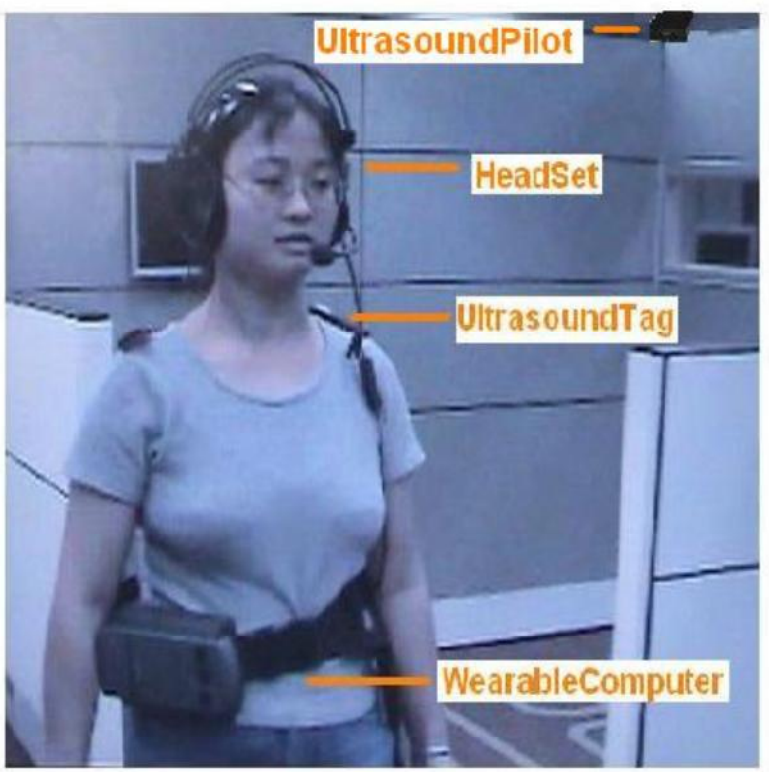


Figure I.7 **: Système de localisation par ultrason et GPS.**

6.2. Système de localisation à infrarouge

Un système de localisation à infrarouge a été proposé par Ertan S, Lee C, Willets A, Tan H, et Pentland A. Il comporte trois unités principales : un gilet qui contient une grille de 4*4 micromoteurs pour délivrer des signaux de guidage haptiques sur le dos de l'utilisateur, un ordinateur portable pour la planification d'itinéraire et un récepteur et des émetteurs à infrarouge pour localiser la personne. Pour une détection consistante des rayonnements infrarouge, les émetteurs ont été montés de telle façon qu'ils couvrent toute la trajectoire à parcourir et le récepteur IR a été tenu en hauteur pour que les signaux IR puissent être facilement détectés. Ce système a été testé par 12 étudiants. Ils l'ont utilisé pour parcourir 4 différentes trajectoires à l'intérieur du laboratoire. Les résultats ont montré que chaque trajectoire nécessite en moyenne 1.5 minutes de parcours et que le nombre moyenne d'erreurs pour chaque individu et pour chaque trajectoire varie de 0 à 3 erreurs.

6.3. Système de localisation par RFID

D'autres travaux se sont servis des RFID pour créer un réseau de communication et par conséquence assurer la localisation pédestre.
Une approche de navigation à l'intérieur des bâtiments pour les malvoyants a été présentée par Kulyukin V, Gharpure C, Nicholson J, Sute P, Graw N. D et Pavithran S. Ces derniers se sont inspirés de la navigation par chien guide pour le développement de leur système de navigation pédestre baptisé « RG ». Ce système est composé d'une plateforme robot Pioneer 2DX, d'un toolkit de navigation, d'un récepteur RFID et des radio-étiquettes pour la localisation. La plateforme robotique est rattachée au bout d'une laisse, comme un substitut à un chien-guide. Cette plateforme possède 3 roues et 16 sonars ultrasons, 8 en avant et 8 en arrière. Le toolkit de navigation comprend un ordinateur portable qui est connecté au microcontrôleur du robot via un câble USB afin de le guider. L'ordinateur portable est connecté aussi à un récepteur RFID pour assurer la localisation du robot guide. Ce prototype a été testé par 5 déficients visuels, dont 3 sont complètement aveugles et 2 qui pouvaient seulement percevoir la lumière, à l'intérieur de deux bâtiments inconnus pour eux. Tous les participants ont parvenus à atteindre leurs destinations mais ils se sont plaints de la lente vitesse de RG, 0.5 m/s alors que la vitesse de marche normale varie de 1.2 à 1.5 m/s, ainsi que de ses mouvements saccadés. En outre, ce prototype ne parvient pas à détecter les blocages de route.

6.4. Système Ekahau Real Time Location System (RTLS) [5]

La technique de positionnement proposée par la société Ekahau est basée sur la technique d'empreinte radio, basée sur les mesures des puissances des signaux reçus (RSSI) et comporte deux phases de formation et de positionnement. Ce système consiste en :

- ✓ Des clients agents : logiciel intégré dans un objet tel qu'un smartphone à positionner.
- ✓ Des étiquettes Ekahau portées par les dispositifs à localiser (optionnelles).
- ✓ Le serveur de positionnement (EPE pour Ekahau Positioning Engine) qui calcule les estimations de position.

- ✓ Le logiciel de création du modèle radio (Ekahau Manager) pour la construction de la carte radio (le calibrage du site) et la gestion du système.
- ✓ Le client agent mesure les RSSI reçus des différents points d'accès et envoie ces mesures au serveur de positionnement qui calcule la position de l'objet abritant le client agent, en comparant les mesures transmises par le client aux mesures « références » présentes dans la base de données.

Plusieurs systèmes assez proches de celui d'Ekahau existent comme Horus de l'université du Maryland, CMU-TMI, Place Lab, Magic Map, AMULET (Approximate Mobile User Location Tracking System) de l'université de Rochester, Halibut de l'université de Stanford, le système de la société Cisco (solution Cisco nified wireless).
Le temps de calibrage (qui est actuellement effectué manuellement en utilisant un terminal mobile et en se déplaçant dans le site pour enregistrer des points échantillons) ainsi que la mise à jour régulière de la base de données constituent l'inconvénient majeur de ces systèmes Par exemple, pour calibrer un site de 1200 m^2, il faut au minimum 1 heure pour le système Ekahau.

Conclusion

Les systèmes de positionnement par satellites sont largement déployés et assurent une très bonne couverture internationale mais présentent des limitations quand ils sont utilisés en indoor et la précision qu'ils offrent est limitée dans certains environnements urbains (rue étroite bordée d'immeubles élevés par exemple). En effet, la localisation basée sur les systèmes Wifi s'avère être intéressante. Elle permet d'exploiter une technologie déjà existante et adéquate. C'est pourquoi notre travail consiste à faire une localisation indoor basée sur la technologie wifi sous système android. Dans le chapitre suivant nous allons détailler ce système.

Chapitre II :
Android

Introduction

L'évolution de la technologie actuelle est telle qu'avec un appareil mobile on peut pratiquement tout faire. Ceci est rendu possible grâce à l'innovation et au progrès apportés par des systèmes d'exploitation utilisés par ces mobiles.

Nombreux sont ces divers systèmes d'exploitation, entre autres : Blackberry créé par Research In Motion (RIM) ; Windows Mobile développé par Microsoft Corporation ; iPhone OS développé par Apple pour l'iPhone, l'iPod touch et l'iPad ; Symbian conçu par Symbian ltd et Android lancé par une startup du même nom rachetée par Google.

Dans ce chapitre, nous nous intéresserons au système d'exploitation Android qui est utilisé beaucoup plus dans le domaine de Smartphones, de Tablettes ainsi que de terminaux mobiles. Il occupe la première place au niveau du marché mondial suivit par iPhone OS de Apple.

Nous aborderons, en premier lieu, la définition d'Android, son historique et ses différentes versions. Ensuite, nous présenterons quelques concepts sur les développements d'Android ; particulièrement nous illustrerons l'architecture en couche d'Android ; les composants de base d'une application Android, les approches et environnements de développement d'applications Android.

1. Définition [23]

Android est un système d'exploitation mobile pour smartphones, tablettes tactiles, PDA, smartwatches et d'autres terminaux mobiles. C'est un système open source utilisant le noyau Linux. Il a été lancé par une start-up du même nom rachetée par Google en 2005. D'autres types d'appareils possédant ce système d'exploitation existent par exemple des téléviseurs, des radio-réveils, des montres connectées, des autoradios et même des voitures.

2. Caractéristiques du système Android

Le système Android a les caractéristiques suivantes :

✓ **Open source :** Le contrat de licence pour Android respecte les principes de l'open source ; c'est à-dire-que c'est possible de le télécharger à tout moment et le modifier selon nos besoins. Android utilise des bibliothèques open source puissantes par exemple SQLite pour les bases de données et OpenGL pour la gestion d'images 2D et 3D.

✓ **Gratuit :** autant pour les utilisateurs que pour les constructeurs. Le Play Store (une plateforme de téléchargement d'applications Android), permet de publier autant d'applications que l'on souhaite gratuitement.

✓ **Disponibilité via une licence Apache version 2 :** il inclut tout les utilitaires requis par un constructeur ou par un opérateur.

3. Historique et versions d'Android [24]

L'historique des versions d'Android a débuté avec la sortie de la version 1.0 en septembre 2008. Depuis Android a connu plusieurs mises à jour. Ces mises à jour servent généralement à corriger des bugs et à ajouter de nouvelles fonctionnalités. Dans l'ensemble, chaque version est développée sous un nom de code basé sur des desserts. Ces noms de codes suivent une logique alphabétique.
Le ***tableau II.1*** ci-dessous représente une synthèse des versions d'Android :

	Nom de Version	**Caractéristiques majeures ajoutées**	**Date de sortie**
Android 1.0	**Apple pie**	✓ Téléchargement et mise à jour des applications via Android Market.	**23 / 09 / 2008**
Android 1.1	**Banana bread**	✓ Support pour sauvegarder les fichiers attachés aux MMS.	**09 / 02 / 2009**
Android 1.5	**Cupcake**	✓ Envoi de vidéos vers YouTube et Picasa. ✓ Rotation automatique.	**30 / 04 / 2009**
Android 1.6	**Donut**	✓ Interface de l'Android Market améliorée. ✓ Amélioration de la rapidité dans la recherche Google Navigation.	**15 / 09 / 2009**
Android 2.0	**Eclair**	✓ Support de plus de taille d'écran et résolutions. ✓ Réorganisation de l'interface utilisateur. ✓ Amélioration du clavier virtuel.	**26 / 10 / 2009**
Android 2.2	**Froyo**	✓ Amélioration de la vitesse d'exécution. L'envoi de fichier est supporté par le navigateur. ✓ Partage de connexion USB. ✓ Support installation d'application sur support externe.	**20 / 05 / 2010**
Android 2.3	**Gingerbread**	✓ Interface utilisateur est mise à jour. ✓ Support des grands écrans à résolutions extra-larges. ✓ Ajout d'un gestionnaire de téléchargement. ✓ Amélioration de la gestion de l'alimentation et du contrôle des applications.	**06/ 12 / 2010**

Android 3.2	**Honeycomb**	✓ Compatibilité pour les applications nonoptimisées pour les résolutions d'écran des tablettes tactiles. ✓ Possibilité d'accéder aux fichiers de la carte SD.	**15 / 07 / 2011**
Android 4.0	**Ice Cream Sandwich**	✓ Interface graphique plus ergonomique.	**18 / 10 / 2011**
Android 4.1	**Jelly Bean**	✓ Amélioration de la vitesse d'exécution. ✓ Amélioration gestion appareil photo ✓ Accessibilité : mode gestuel, prise en charge clavier externe "Braille".	**09 / 07 /2012**
Android 4.4	**KitKat**	✓ Nouvelle interface translucide. ✓ Enregistrement séquence vidéo de l'écran. ✓ Amélioration des performances.	**31 / 10 / 2013**
Android 5.0	**Lollipop**	✓ Nouvelle interface / design "Material design". ✓ Amélioration de la rapidité de la gestion de la batterie.	**17 / 10 / 2014**
Android 5.1	**Lollipop**	✓ Supporte plusieurs cartes SIM. ✓ Protection par blocage en cas de perte ou vol.	**09 / 03 / 2015**
Android 6	**Marshmallow**	✓ Support de l'authentification par empreinte digitale. ✓ Amélioration de la durée de la batterie avec un mode "deep sleep".	**05 / 10 / 2015**
Android 7.0	**Nougat**	✓ Meilleur support du multitâche. ✓ Mises à jour, système amélioré (grâce à une double partition système).	**22 / 08 / 2016**

Tableau II.1 : Versions officielles d'Android.

4. Architecture d'Android [25]

L'architecture de la plateforme Android se décline selon une démarche bottom up (bas en haut) en quatre principaux niveaux qui sont le noyau linux, les librairies et la plateforme d'exécution, le module de développement applicatif et enfin les différentes applications.
La ***figure II.1*** illustre l'architecture en couche de la plateforme Android :

Figure II.1: L'Architecture de la plateforme Android.

4.1. Applications

Android est fourni avec un ensemble de programmes de base permettant d'accéder à des fonctionnalités comme les courriels, les SMS, le téléphone, le Web, etc. Ces applications sont développées à l'aide du langage de programmation Java. Pour l'utilisateur final, c'est la seule couche accessible et visible.

4.2. Le framework (Application Framework)

En fournissant une plateforme de développement ouverte, Android offre aux développeurs la possibilité de créer des applications riches et innovantes. Les développeurs sont libres de profiter du matériel périphérique, des informations de localisation d'accès, d'exécution des services d'arrière-plan, de définir des alarmes, d'ajouter des notifications de la barre d'état, …etc. Les services qu'offre le framwork sont les suivants :

A) CorePlateform Service

Les services cœurs de la plateforme (CorePlateform Services) fournissent des services essentiels au fonctionnement de la plateforme :

✓ **Activity Manager** : gère le cycle de vie des applications et maintient une "pile de navigation" (navigation backstack) permettant d'aller d'une application à une autre et de revenir à la précédente quand la dernière application ouverte est fermée.

✓ **Package Manager** : utilisé par l'Activity Manager pour charger les informations provenant des fichiers. "apk".

✓ **Window Manager :** juste au-dessus du "Surface Flinger", il gère les fenêtres des applications.

✓ **Resource Manager :** gère tous ce qui n'est pas du code, toutes les ressources images, fichier audio, etc.

✓ **Content Provider :** gère le partage de données entre applications, comme par exemple la base de données de contacts, qui peut être consultée par d'autres applications que l'application "Contact ". Les données peuvent être partagées à travers une base de données (SQLite), des fichiers, le réseau, etc.

✓ **View System** : fournit tous les composants graphiques : listes, grille, text box, buttons et même un navigateur web embarqué.

B) Hardwar Services

Les services matériels (Hardware Services) fournissent un accès vers les API matérielles de bas niveau :

- ✓ **Telephony Service** : permet d'accéder aux interfaces "téléphonique".
- ✓ **Location Service** : permet d'accéder au GPS.
- ✓ **Bluetooth Service** : permet d'accéder à l'interface Bluetooth.
- ✓ **WiFi Service** : permet d'accéder à l'interface Wifi.
- ✓ **USB Service** : permet d'accéder aux interfaces USB.
- ✓ **Sensor Service** : permet d'accéder aux capteurs.

4.3. Les bibliothèques (Librairies)

Android inclut un ensemble de bibliothèques C et C++ utilisées par de nombreux composants du système. Elles sont offertes aux développeurs par l'intermédiaire du « framework Android ».

La liste ci-dessous énumère quelques-unes des bibliothèques disponibles dans Android :

✓ **Bibliothèque système C** : c'est une implémentation de la bibliothèque standard C, optimisée pour les systèmes Linux embarqués.

✓ **Bibliothèques multimédias** : basées sur StageFright, elles permettent le support de nombreux formats audio et vidéo.

✓ **SurfaceFlinger** : elle permet l'accès au sous-système d'affichage.

✓ **LibWebCore** : est un moteur de rendu de pages Internet basé sur Webkit. Cette bibliothèque est principalement utilisée dans le navigateur et dans les vues web embarquées (WebView).

✓ **Skia** : moteur graphique 2D.

✓ **Bibliothèques 3D** : est une implémentation basée sur OpenGL.

✓ **FreeType** : est un rendu des polices de caractères.

✓ **SQLite** : est une bibliothèque open source écrite en C permettant d'implémenter un moteur de base de données relationnelle.

4.4. Le moteur d'exécution Android (Android Runtime)

Android inclut un ensemble de bibliothèques qui fournit la plupart des fonctionnalités disponibles dans les bibliothèques de base du langage de programmation Java.
Chaque application Android s'exécute dans son propre processus, avec sa propre instance de machine virtuelle Dalvik. Dalvik VM (Virtual Machine) est une implémentation de machine virtuelle ayant été conçue pour optimiser l'exécution multiple de machines virtuelles. Elle exécute du bytecode qui lui est dédié : le bytecode .dex.
La ***figure II.2*** montre les étapes nécessaires à la compilation et à l'exécution d'un programme Android standard :

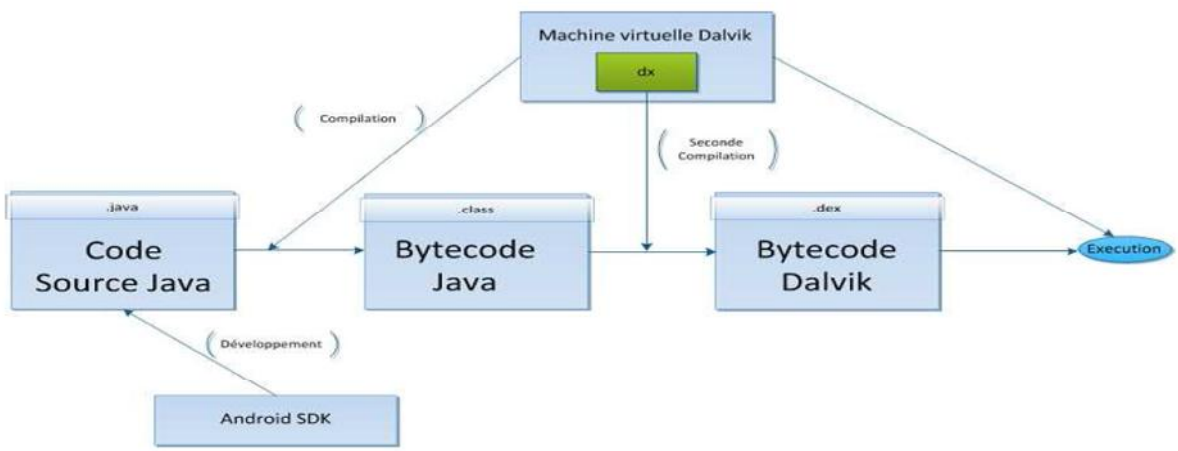


Figure II.2 : Etapes d'exécution d'un programme Android.

Un code source est une suite d'instructions que l'on trouve dans un fichier ".java" qui est traduit en une autre suite d'instructions dans un autre langage que l'on appelle le « bytecode ». Ce code est contenu dans un fichier « .class ». Le bytecode est un langage spécial qu'une machine virtuelle Java peut comprendre et interpréter. Le bytecode Java ne peut être lu que par une machine virtuelle Java et Dalvik n'était pas une machine virtuelle Java. Il faut donc procéder à une autre conversion à l'aide d'un programme qui s'appelle « dx » qui s'occupe de traduire des applications de bytecode Java en bytecode Dalvik, qui est compréhensible par la machine virtuelle Dalvik, le résultat est mis dans un fichier « .dex ».

4.5 Le noyau Linux (Linux Kernel)

Android repose sur un noyau Linux (version 2.6) qui gère les services du système, comme la sécurité, la gestion de la mémoire et des processus, la pile réseau et les pilotes. Il agit également comme une couche d'abstraction entre le matériel et la pile logicielle.

5. Le développement d'applications Android [26]

5.1. Définition

Les applications Android sont constituées d'un ensemble d'activités qui permettent l'interaction avec l'utilisateur, et d'un ensemble de services qui fonctionnent en arrière-plan et se charge d'effectuer certains traitements précis.

5.2. Les composants d'une application android

5.2.1. Activité

Une activité est la composante principale pour une application Android. Elle représente l'implémentation et les interactions de vos interfaces.

Toute activité a un cycle de vie qui correspond aux différents états lors de sa gestion par le système Android. Il est très important car il nous permet de suivre l'état de l'activité au fur et à mesure de son existence dans le système Android.
La ***figure II.3*** résume ses différents états :

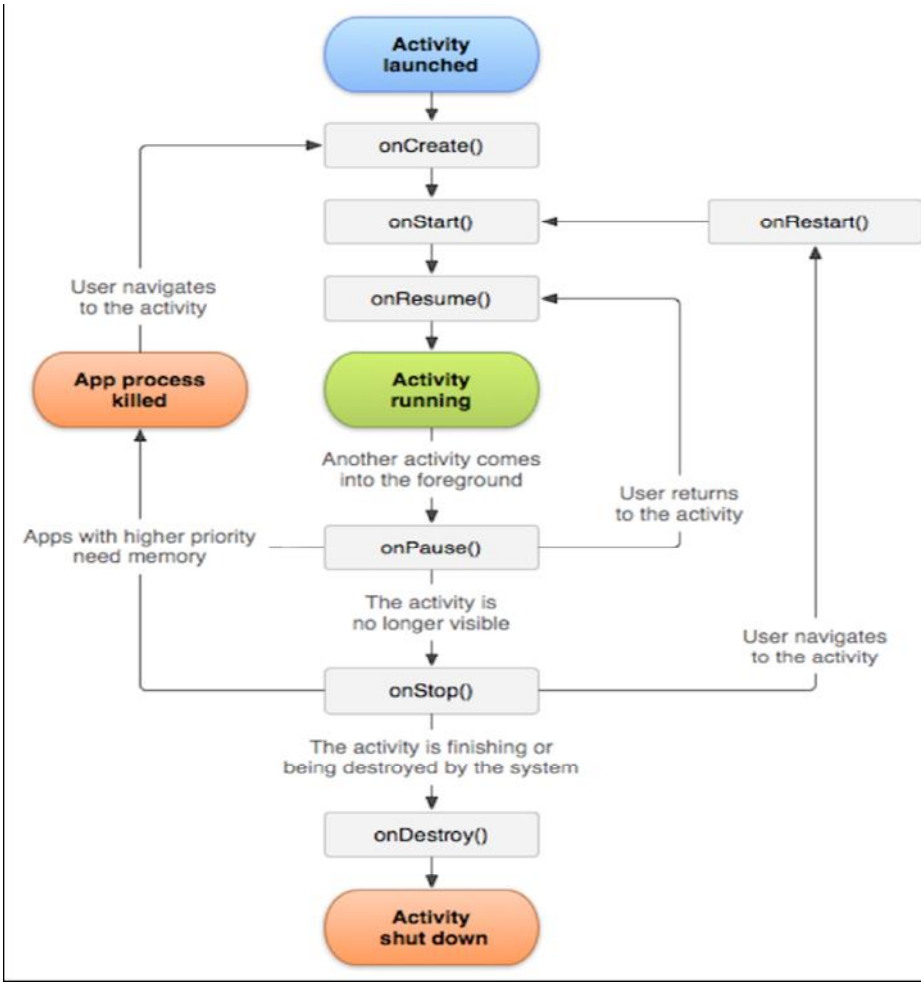


Figure II.3 : Le cycle de vie d'une activité.

Le changement d'état d'une activité est provoqué par l'exécution d'une méthode particulière. Ci-dessous nous présentons les différentes méthodes :

✓ **OnCreate ()**

Cette méthode est appelée à la création d'une activité. Elle sert à initialiser l'activité ainsi que toutes les données nécessaires à cette dernière.

Quand la méthode **OnCreate()** est appelée, on lui passe un "Bundle" en argument. Ce Bundle contient l'état de sauvegarde enregistré lors de la dernière exécution de l'activité.

✓ **OnStart()**

Cette méthode est utilisée pour signifier le début d'exécution de l'activité (début du passage au premier plan). Si l'activité ne peut pas aller en avant plan quelque soit la raison, l'activité sera transférée à **OnStop()**.

✓ **OnResume()**

Cette méthode est appelée après **OnStart** (au moment où l'application repasse en premier plan). A la fin de l'appel à la méthode **onResume()** l'application se trouve au premier plan et reçoit les interactions utilisateurs.

✓ **OnPause()**

Si une autre activité passe au premier plan, la méthode **OnPause()** est appelée sur l'activité. Afin de sauvegarder l'état de l'activité et les différents traitements effectués par l'utilisateur. A ce stade, l'activité n'a plus accès à l'écran, on doit arrêter de faire toute action en rapport avec l'interaction utilisateur (désabonner les listeners). On doit par contre continuer à exécuter des algorithmes nécessaires mais qui ne consomment pas trop de CPU.

✓ **OnStop()**

Est appelée quand l'activité n'est plus du tout visible quelque soit la raison. Dans cette méthode on doit arrêter tous les traitements et les services exécutés par l'application.

✓ **OnDestroy()**

Est appelée quand l'application est totalement fermée (Processus terminé). Toutes les données non sauvegardées sont perdues.

✓ **OnRestart ()**

Exécutée lorsque l'activité est arrêtée via un **OnStop()** (repasse en premier plan).

5.2.2. Les Services

Un service, à la différence d'une activité, ne possède pas de vue mais permet l'exécution d'un algorithme sur un temps indéfini. Il ne s'arrêtera que lorsque la tâche est finie ou que son exécution est arrêtée.

Il peut être lancé au :

- Démarrage du téléphone.
- Moment d'arrivée d'un événement (arrivée d'un appel, SMS, mail, etc).
- Lancement de l'application.
- Moment de réalisation d'une action particulière ou autres actions sur l'application.

5.2.3. Le Broadcast Receivers

Un Broadcast Receiver permet d'écouter ce qui se passe sur le système ou sur l'application et de déclencher une action prédéfinie. C'est souvent par ce mécanisme que les services sont lancés.

5.2.4. Le Content providers

Les content providers servent à accéder à des données depuis une application. Notamment pour accéder aux:

- Contacts stockés sur le téléphone.
- Agendas.
- Photos.
- Autres données.

5.3. La structure d'un projet android [27]

Le système de construction d'un programme Android est organisé sous la forme d'une arborescence de répertoires spécifiques à un projet, exactement comme n'importe quel projet Java. Les détails sont spécifiques à Android. Illustré par la ***figure II.4*** :

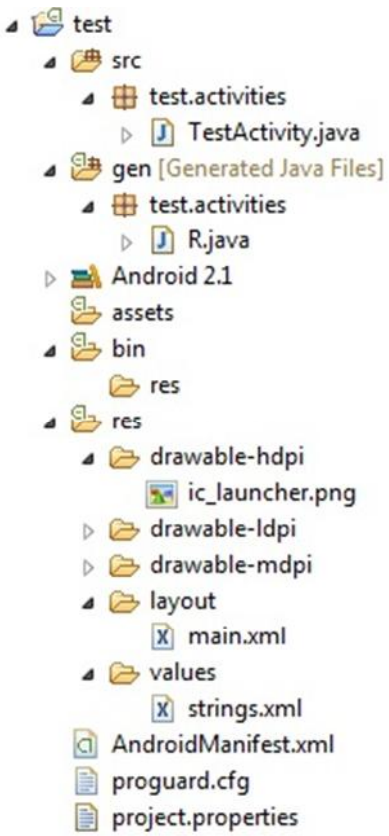


Figure II.4 : Structure d'un projet Android.

Tout comme technologies actuelles, les sources d'une application Android possèdent une structure bien définie qui doit être respectée. Ces arborescences permettent non seulement de rendre les projets plus lisibles et organisés, mais aussi de simplifier le développement. Lors de la création d'un nouveau projet, voici l'arborescence qui est automatiquement générée :

- **src** : répertoire contenant l'ensemble des sources du projet. il contient les classes de type activité qui gèrent entre autre le cycle de vie, mais aussi les classes permettant de piloter les différentes fonctions de l'application;
- **gen** : répertoire contenant l'ensemble des fichiers générés par le plugin correspondant à l'environnement de développement. Aucune modification ne doit être faite dans ces fichiers;
- **androidManifest.xml** : fichier XML décrivant l'application et ses composants, tels que les activités, les services, etc. Lors de la création d'une activité, une erreur courante pour un premier projet Android est d'oublier de la déclarer dans le fichier Manifest. C'est une étape indispensable pour le fonctionnement de l'application. Le Manifest est, en quelque sorte, la

carte d'identité de l'application, et permet d'autoriser l'exécution des activités et autres actions de l'application.

- **res** : répertoire contenant toutes les ressources telles que l'images, les vues de l'interface graphique, etc, nécessaires à l'application. Ce répertoire est structuré par défaut de la maniere suivante:
- **res/drawbale** : contient les ressources de type image;
- **res/layout** : contient les descriptions des interfaces graphiques au format XML(les vues);
- **res/xml** : contient les fichiers XML supplémentaires (non présents par défaut);
- **res/menu** : contient la description des menus, composants très courants d'une vue;
- **res/values** : contient diverses ressources, telles que les textes, qui sont empaquetées sans aucun traitement.

Au moment de la compilation du projet, l'application finale est générée au format APK, dans le répertoire bin de l'arborescence. C'est ce fichier qu'il faut ensuite déployer sur les équipements afin de pouvoir faire tourner l'application.

5.4. Comparaison entre Android Studio et Eclipse

Même si ces deux plateformes de développement permettent de développer des applications complètes on peut noter des différences ; le tableau suivant montre un comparatif entre ces deux solutions. **[28]**

Critères de comparaison	ADT (Eclipse)	Android Studio
Facilité d'installation	Moyen	Simple
Langue	Plusieurs	Anglais
Performance	Peut être lourd	Rapide
Système de construction	Ant	Gradle
Génération de variante et de multiple APK (Android Package)	Non	Oui
Le code d'exécution et de refactorisation Android (completion code et refactoring)	Base	Avancé
Editeur d'interface graphique	Oui	Oui
Signature d'APK et gestion de Keystore	Oui	Oui
Support NDK : Le Kit de développement natif	Oui	A venir

Tableau II.2: Comparaison entre Android Studio et Eclipse.

Conclusion

Android est un système d'exploitation très accessible. Il a une documentation très riche et disponible gratuitement sur différents sites internet dont le site officiel pour le développeur Android. De plus, différentes plateformes de développement d'applications : l'Eclipse + le SDK et l'Android Studio, offrent des outils nécessaires pour le développement d'applications Android.

En conséquence, les applications Android et les différentes versions d'Android avec nouvelles fonctionnalités ne cessent de voir le jour, facilitant ainsi le maniement quotidien de certains appareils et apportant de plus en plus de nouveauté quant à l'usage de ses dispositifs.

Le prochain chapitre consiste à introduire la modélisation de notre application en s'appuyant sur le langage UML.

Chapitre III :
Etude, Analyse et Conception

Introduction

Les chapitres précédents se sont focalisés sur la présentation de l'état de l'art lié au domaine de la géolocalisation. Une attention particulière a été consacrée à la géolocalisation en intérieur. L'application à développer est une application mobile sous un système Android qui sert à géolocaliser en intérieur sur un plan, en utilisant des données stokées dans une base de données embarquée.

Dans ce chapitre, nous présentons nos contributions dans le domaine de la géolocalisation en intérieur dans les réseaux Wi-Fi. Nous nous intéressons à la façon de trouver l'emplacement du terminal mobile à partir des informations de puissances des signaux.

Nous étudions dans ce qui suit l'environnement d'expérimentation ainsi que l'analyse et la conception de notre application.

1. Présentation du laboratoire LARI

L'objectif de notre travail est de se localiser à l'intérieur du laboratoire de recherche LARI de **29.65** mètres de longueur et **16.25** mètres de largeur, composé de 6 salles de surfaces égales, dont 3 salles à droites et 3 salles à gauches séparées par un long couloir.

La ***figure III.1*** ci-dessous schématise le plan de ce laboratoire :

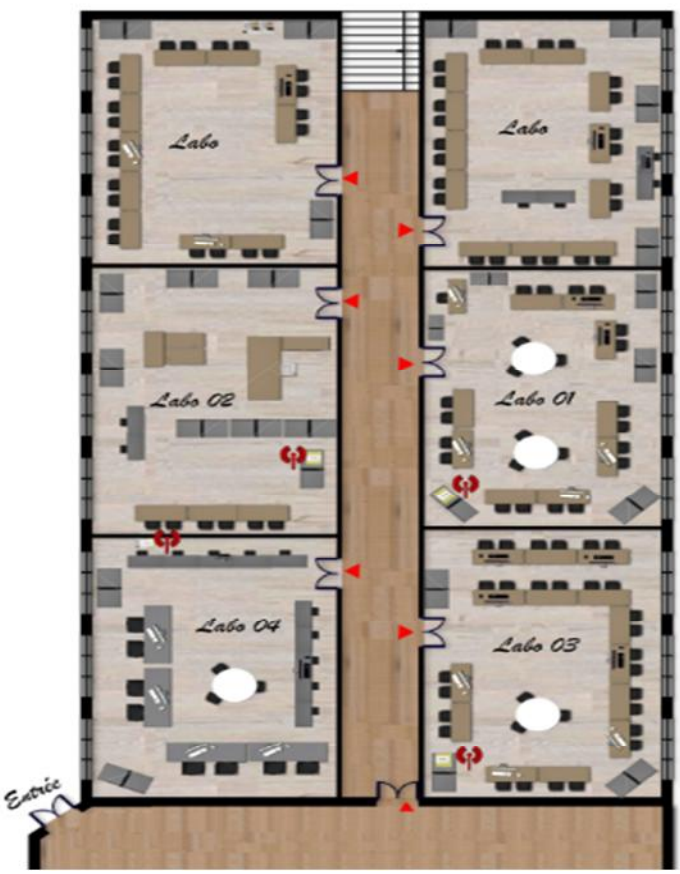


Figure III.1 : Plan 2D du laboratoire LARI.

2. Etude

Pour la réalisation de notre travail nous avons opté pour la technologie de localisation indoor Wifi, d'une part pour son accessibilité, d'autre part pour ses portées et sa mobilité .En effet, la portée en intérieur d'un périphérique Wifi est de l'ordre de plusieurs dizaines de mètres, pouvant atteindre la centaine de mètres en extérieur. Le débit atteint, quant à lui, 54 Mbps voire 300 Mbps pour la norme 802.11n. Ceci permet d'envisager des applications incluant des médias riches.

Comme vu précédemment (dans le chapitre II) deux familles de systèmes de géolocalisation existent ayant des avantages et des inconvénients. D'une part, la grande quantité de données mesurées ou calculées dans le cadre des systèmes basés sur le référencement de points de mesures permet à ces systèmes d'avoir une précision intéressante, mais les rend en contrepartie plus lents à déployer et à adapter à des changements d'environnement. D'autre part, les systèmes de géolocalisation basés sur la trilatération grâce aux distances obtenues par une relation entre la puissance du signal et la distance sont rapidement adaptables mais souffrent d'une précision inférieure à celle des systèmes basés sur le référencement de mesures.

C'est pourquoi nous nous sommes orientées vers la méthode statique (le fingerprint) qui se repose sur deux phases :

2.1. Phase de calibrage

2.1.1. Eléments nécessaires

- ✓ Un plan du site (image réalisée sur **Autocad,** ***Figure III.1***).
- ✓ Des points d'accès répartis sur le site (4 points d'accès).
- ✓ Un terminal équipé d'une carte WiFi avec le logiciel de calibrage (Wifi Analyzer).

2.1.2. Fonctionnement

La phase de calibrage consiste à se déplacer sur l'ensemble du site avec un terminal équipé de l'interface WiFi et du logiciel de calibrage. Au cours du déplacement, des mesures sont effectuées régulièrement.

Le terminal se trouvant à chaque fois dans la zone de couverture d'un ou de plusieurs points d'accès (APs), le logiciel de calibrage mesure les puissances des différents signaux (RSSI) reçus des APs et les sauvegarde dans la base de données.

On appelle échantillon l'ensemble formé des adresses MAC des APs et des puissances des signaux reçus de ces APs, ainsi que les coordonnées cartésiennes du point du site où la mesure a été effectuée. Les mêmes étapes sont répétées pour plusieurs points du site. La figure III.2 illustre ces points:

Figure III.2 : Représentation des points de calibrage sur le plan.

Démarche suivie

Pour chaque point de référence nous avons enregistré 30 mesures de signaux, puis nous avons calculé leurs moyennes, vu les variations des puissances des signaux reçus pour un même point de référence.

Un exemple de mesures reçues par les quatre points d'accès ayant comme **SSID** : **LARI 1**, **LARI 2, LARI 3 et LARI** placés respectivement dans les laboratoire**s Labo 1, Labo 2, Labo 3 et Labo 4**. Le tableau ci-dessous montre les mesures qui sont enregistrées pour le point de référence (5, 10.4) se trouvant dans le **Labo 4**.

LARI 1	LARI 2	LARI 3	LARI
-80	-75	-67	-56
-73	-62	-70	-42
-69	-63	-67	-50
-70	-67	-68	-45
-69	-70	-66	-46
-71	-61	-67	-41
-70	-69	-67	-43
-70	-58	-64	-44
-71	-59	-67	-43
-69	-62	-68	-45
-68	-61	-67	-44
-70	-62	-73	-43
-70	-58	-65	-42
-71	-62	-66	-44
-69	-60	-67	-45
-68	-84	-66	-50
-69	-60	-68	-45
-70	-62	-68	-43
-69	-59	-69	-51
-72	-59	-70	-42
-72	-60	-67	-46
-69	-73	-68	-50
-75	-78	-71	-56
-77	-84	-73	-54
-76	-77	-72	-56

	-80	-83	-77	-54
	-81	-73	-72	-61
	-80	-86	-72	-57
	-81	-64	-75	-54
	-73	-69	-79	-53
Moyenne (db)	**-72.4**	**-59.2**	**-64.8666667**	**-45.3666667**

Tableau III.1 : Calcul de la moyenne pour le point de référence (5, 10.4).

Afin d'estimer en moyenne les puissances reçues par les APs dans chaque laboratoire nous avons calculé la moyenne des puissances des points de références présents dans chacun d'eux. Les tableaux ci-dessous présentent des exemples de calculs effectués dans le **Labo 3** et le **Labo 4**:

Référence	LARI 1	LARI 2	LARI 3	LARI
(1, 1.10)	-65.5	-76.43	-59.33	-69.4
(0.35, 5.85)	-58.6	-78.4	-52.37	-55.87
(2, 3.10)	-70.73	-88.57	-56.3	-71.1
(3.75, 1.6)	-56.93	-71	-53.6	-72.77
(5, 1.10)	-65.03	-66.93	-59.13	-66.5
(5, 3.85)	-57.53	-75.87	-53.8	-67.9
(6.5, 5.6)	-60.03	-68.6	-51.7	-59.57
(8.5, 3.35)	-56.1	-69.93	-53.2	-64.87
(9, 1.10)	-66.27	-75.37	-50.5	-71.1
(10, 5.85)	-57.47	-67.23	-52.6	-78.72
Moyenne labo 3 (db)	**-61.419**	**-73.833**	**-54.253**	**-67.78**

Tableau III.2 : Calcul de la moyenne des points de référence du Labo 3.

Référence	LARI1	LARI2	LARI3	LARI
(1, 10.04)	-73.8	-66.6	-56.9333333	-40.3
(2, 15.54)	-72.2	-73.2666667	-69.3666667	-59.7666667
(3.5, 16.29)	-68.9	-61.9	-64.9666667	-49.9333333
(4, 12.29)	-66.4333333	-62.9333333	-70	-53.3
(5.25, 12.54)	-67.8666667	-59.0333333	-69.0333333	-49.5
(5, 10.4)	-72.4	-59.2	-64.8666667	-45.3666667
(7, 10.04)	-65.1333333	-53.9	-73.0666667	-55.6666667
(7, 13.29)	-66.6	-64.1333333	-69.9333333	-59.3333333
(8.5, 15.29)	-74.1333333	-63.9666667	-67.3	-51.1
Moyenne labo 4 (db)	-69.7185185	-62.7703704	-67.2740741	-51.5851852

Tableau III.3 : Calcul de la moyenne des points de référence du Labo 4.

Après avoir analysé les résultats obtenus nous avons remarqué que la moyenne des puissances des signaux de chaque point d'accès pour chaque laboratoire est logique.

✓ **Pour le labo 3 :** la valeur maximale des quatre valeurs (LARI 1, LARI 2, LARI 3, LARI) est égale à celle capté par **LARI 3** ce qui est logique du moment que le **LARI 3 est placé dans le Labo 3.**

✓ **Pour le labo 4 :** la valeur maximale des quatre valeurs (LARI1, LARI2, LARI3, LARI) est égale à celle capté par **LARI** ce qui est logique du moment que le **LARI est placé dans le Labo 4.**

✓ **Pour le labo 1 :** la valeur maximale des quatre valeurs (LARI1, LARI2, LARI3, LARI) est égale à celle capté par **LARI1** ce qui est logique du moment que le **LARI1 est placé dans le Labo 1.**

✓ **Pour le labo 2 :** la valeur maximale des quatre valeurs (LARI1, LARI2, LARI3, LARI) est égale à celle capté par **LARI2** ce qui est logique du moment que le **LARI2 est placé dans le Labo 2.**

2.2. Phase de positionnement

Pour se localiser nous avons utilisé la méthode statistique**,** après la phase de calibrage, nous avons constitué la base de données de référencement. La localisation est ensuite effectuée par comparaison des mesures en temps réel avec les informations contenues dans la base de données.
Notre base de données contient des informations sur les coordonnées cartésiennes (x,y) et les moyennes des mesures de puissance pour chaque point d'accès, on obtient alors une base de données composée d'une suite d'enregistrements relatifs chacun à un point de référence.
Un enregistrement est de la forme **(x , y, LARI 1, LARI 2, LARI 3, LARI, Zone)** où :
x et y : sont les coordonnées cartésiennes du point de référence.

LARI 1, LARI 2, LARI 3, LARI: sont l'ensemble des moyennes de puissance du signal reçu par les quatre points d'accès.

Zone : les noms des laboratoires ou bien des zones du couloire.

2.2.1. Méthode de positionnement

Vu les variations des RSSI et la susceptibilité des ondes radio aux erreurs et à différents facteurs qui peuvent affecter leur qualité, la simple comparaison des mesures reçues du client mobile avec les mesures prises dans un temps précédent et qui sont sauvegardées dans la base données ne sera pas précise. Il est donc nécessaire d'utiliser certains algorithmes ou méthodes pour l'estimation de la position afin de bien exploiter les mesures relevées durant la phase de calibrage. En effet, ces mesures n'ont été effectuées qu'aux certains points et il faut interpoler au mieux en tous les points de la zone de localisation.

Plusieurs algorithmes sont utilisés à ce propos : des algorithmes simples comme celui des k plus proches voisins (en anglais k-Nearest Neighbor ou k-NN), ou des méthodes statistiques plus complexes comme les modèles de Markov cachés (Hidden Markov model ou HMM) et les approches bayésiennes.

2.2.1.1. Méthode du plus proche voisin

L'algorithme des k-plus proches voisins est une méthode de localisation par proximité. Comme on peut le voir à la ***Figure III.3***, l'idée derrière cette méthode est de délimiter une zone dans laquelle le nœud cible se situe en déterminant quels sont ses plus proches voisins (les nœuds de référence ayant reçus les signaux plus proches de celles du nœud cible)**.** Ensuite, la position est estimée en effectuant une moyenne pondérée des positions des K plus proches voisins.

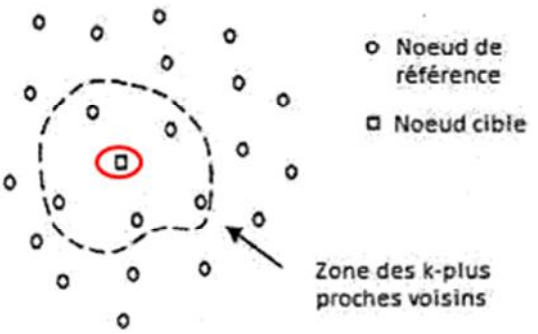


Figure III.3 : K-Plus proches voisins.

L'algorithme du plus proche voisin calcule la distance euclidienne entre les puissances des signaux mesurées et celles sauvegardées dans la base de données. Supposons que les puissances transmises par l'objet mobile soient notées **RSSI_TR** (puissance du signal reçu de l'APi). L'algorithme cherchera dans la base de données, les échantillons de la même série d'APs (soit AP1, AP2, AP3 … APn), et calculera, pour chaque échantillon trouvé, la distance euclidienne entre les puissances reçues par l'objet mobile et les puissances sauvegardées dans la base de données **RSSI_BD**. La distance est donnée par la formule suivante :

$$d(RSSI\_TR, RSSI\_BD) = \sqrt{\sum_{j=1}^{M} (RSSI\_TR[j] - RSSI\_BD[j])^2}$$

Plus cette distance est faible, plus le degré de similarité entre ces deux entités est grand.
M étant le nombre de points d'accès (4 dans notre cas).

L'algorithme :

M = nombres de points d'accès (4 dans notre cas)
d= 0
pour j=1 **à** M
d = d + (**RSSI_TR[j] - RSSI_BD[j])^2**
fin
d=sqrt(d)

3. Analyse et conception

Dans le but d'une meilleure organisation et une bonne maîtrise du travail, tout processus de développement d'applications ou d'un système informatique doit suivre une méthode ou une démarche bien définie. Nous avons adopté la conception avec UML.
Dans ce qui suit nous allons expliquer le processus de développement et de conception de notre application.

3.1. Présentation d'UML

3.1.1. Historique [16]

L'UML est né en octobre 1994 chez Rational Software Corporation à l'initiative de G. Booch et de J. Rumbaugh.UML 1.1 a été standardisé par l'OMG (Object Management Group) le 17 novembre 1997 suite à la demande émanant de la collaboration de plusieurs entreprises (Hewlett-Packard, IBM, i-Logix, ICON Computing, IntelliCorp, MCI Systemhouse, Microsoft, ObjecTime, Oracle, PlatinumTechnology, Ptech, Rational Software Corporation, Reich Technologies) **La version actuelle est UML 2.5**.
Voici le récapitulatif de ces évolutions :

- **En 1995:** Méthode unifiée 0.8 (intégrant les méthodes Booch'93 et OMT).
- **En 1995:** UML 0.9 (intégrant la méthode OOSE).
- **En 1996:** UML 1.0 (proposée à l'OMG).
- **En 1997:** UML 1.1 (standardisée par l'OMG).
- **En 1998:** UML 1.2.
- **En 1999:** UML 1.3.
- **En 2000:** UML 1.4.
- **En 2003:** UML 1.5.
- **En 2005:** UML 2.0.
- **En 2007:** UML 2.1.
- **En 2009:** UML 2.2.
- **En 2010:** UML 2.3.
- **En 2011:** UML 2.4.
- **En 2015:** UML 2.5.

3.1.2. Définition [17]

UML (Unified Modeling Language), que l'on peut traduire par "langage de modélisation unifié) est une notation permettant de modéliser un problème de façon standard. Ce langage est né de la fusion de plusieurs méthodes existantes auparavant, et est devenu désormais la référence en terme de modélisation objet, cette notion consiste à modéliser informatiquement un ensemble d'éléments d'une partie du monde réel (que l'on appelle domaine) en un ensemble d'entités informatiques. UML n'est pas une méthode dans la mesure

où elle ne présente aucune démarche. A ce titre UML est un formalisme de modélisation objet.

3.1.3. Modélisation avec l'UML [16]

UML permet de représenter des modèles, mais il ne définit pas leurs processus d'élaboration. Les acteurs d'UML conseillent tout de même une démarche pour favoriser la réussite d'un projet, cette démarche doit être :

- **Une démarche itérative et incrémentale** : pour comprendre et représenter un système complexe, pour analyser par étapes, pour favoriser le prototypage et pour réduire et maitriser l'inconnu.

- **Une démarche guidée par les besoins des utilisateurs** : tout est basé sur le besoin des utilisateurs du système, le but du développement lui-même est de répondre à leur besoin. Chaque étape sera affinée et validée en fonction des besoins des utilisateurs.

- **Une démarche centrée sur l'architecture logicielle** : c'est la clé de succès d'un développement, les choix stratégique définiront la qualité du logiciel.

3.2. Analyse

La phase d'analyse débute par la mise en évidence des différents acteurs qui interviennent dans le système, leurs scénarios ainsi que les besoins fonctionnels du système (ce que le système doit faire en réponse à une requête d'un utilisateur), qui aboutira à un ensemble de diagrammes représentant le modèle d'analyse. Ensuite, modéliser les objectifs à atteindre, dans la phase de conception en s'appuyant sur la phase d'analyse.

3.2.1. Spécification des besoins

3.2.1.1. Identification de l'acteur

Un acteur est un utilisateur type qui a toujours le même comportement vis-à-vis d'un cas d'utilisation. Ainsi les utilisateurs d'un système appartiennent à une ou plusieurs classes d'acteurs selon les rôles qu'ils tiennent par rapport au système. **[18]**

On distingue 2 types d'acteurs :

- ✓ **Acteur principal :** celui pour qui le cas d'utilisation produit un résultat observable.
- ✓ **Acteur secondaire :** est souvent sollicité pour des informations complémentaires ; il peut uniquement consulter ou informer le système lors de l'exécution du cas d'utilisation.

Un seul acteur qui intervient dans notre application qui est **l'utilisateur**.

3.2.2. Spécification de la tâche

L'acteur que nous avons défini précédemment, effectue une seule tâche qui est « se localiser ». Illustrer par le ***tableau III.4*** :

Acteur	Tâches
Utilisateur	**T1** : Lancer l'application **T2** : Se localiser

Tableau III.4 : Spécification de la tâche.

3.2.3. Spécification du scénario

3.2.3.1. Définition d'un scénario

Un scénario représente une succession particulière d'enchainements s'exécutant du début à la fin du cas d'utilisation, un enchainement étant l'unité de description de séquence d'actions **[19]**.
En générale, il existe 2 types de scénarios **: [18]**

- ✓ **Scénario nominal :** qui se termine de façon normale et correcte.
- ✓ **Scénario alternatif :** qui se termine en échec avec détection d'erreurs.

3.2.3.2. Le scénarios de notre système

Afin d'effectuer la tâche associée à notre acteur, celui-ci doit effectuer un certain nombre d'actions. **Le tableau *III.5*** montre le scénario suivit par l'acteur :

Acteur	Tâches	Scénario
Utilisateur	**T1** : lancer l'application	**S1** : cliquer sur l'icone de l'application
	T2 : Se localiser	**S2 :** La localisation est faite automatiquement dès le lancement de notre application

Tableau III.5 : Spécification du scénario.

3.2.4. Les cas d'utilisation

3.2.4.1. Définition d'un cas d'utilisation

Un cas d'utilisation (use case) correspond à un certain nombre d'actions que le système devra exécuter en réponse à un besoin d'un acteur. Un cas d'utilisation doit produire un résultat observable pour un ou plusieurs acteurs ou parties prenantes du système **[19]**.

3.2.4.2. Spécification de cas d'utilisation

Cas d'utilisation : Se localiser.
Scénario : S1, S2.
Rôle : Utilisateur.
Description :
1- L'utilisateur lance l'application après avoir cliquer sur l'icone de l'application.
2- La localisation est faite automatiquement dès le lancement de notre application.

Figure III.4 : Cas d'utilisation « Se localiser ».

3.3. Conception

3.3.1. Conception de la base de données

Pour l'implémentation de la base de données, on aura besoin d'élaborer un modèle logique de données. Nous présentons dans ce qui suit le modèle relationnel et la structure des tables de la base de données.

3.3.1.1. Diagramme entités/associations

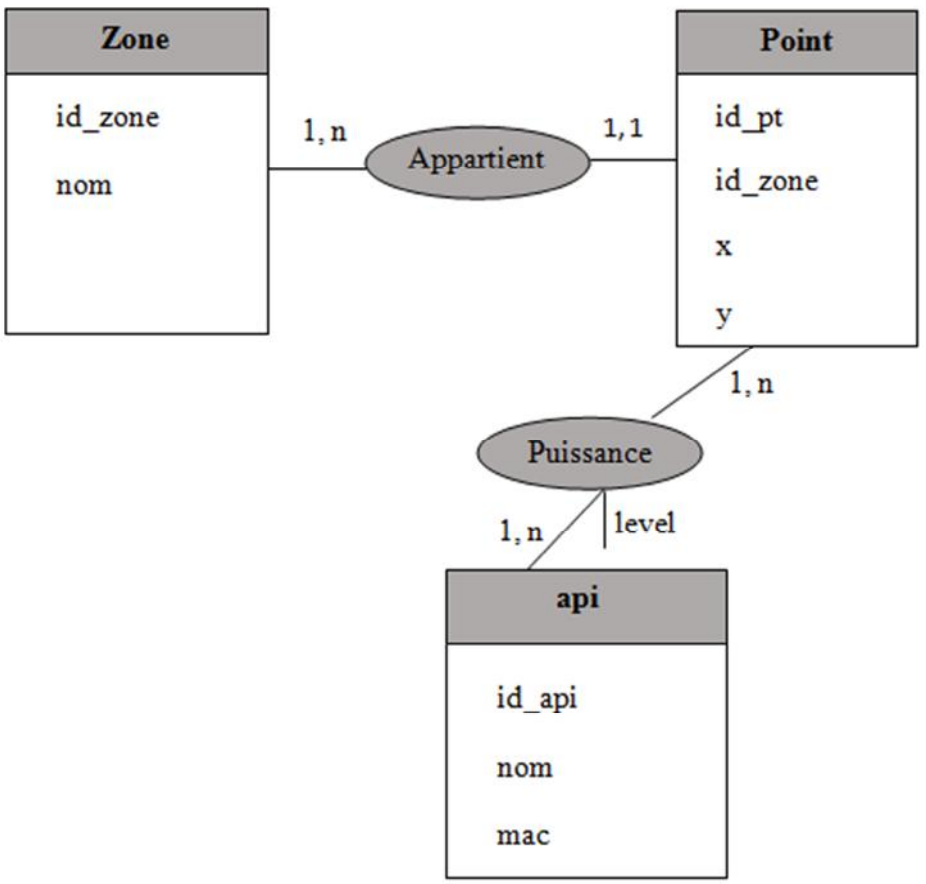


Figure III.5: Modèle entités/associations

3.3.1.2. Schéma de la base de données

Notre base de données contient quatre tables qui sont représentée comme suit :

- **Table « Point »**

Nom du champ	Type	Description	Clé
id_pt	integer	Identificateur du point	Primaire
id_zone	integer	Identificateur de la zone à laquelle le point appartient	
x	number	Les coordonnées cartésiennes	
y	number		

❖ **Table « Zone »**

Nom du champ	Type	Description	Clé
id_zone	integer	Identificateur de la zone	Primaire
nom	Text	Le nom de la zone	

❖ **Table « api »**

Nom du champ	Type	Description	Clé
id_api	integer	Identificateur de l'api	Primaire
nom	Text	Le nom de l'api	
mac	Text	L'adresse mac de l'api	

❖ **Table « puissance »**

Nom du champ	Type	Description	Clé
id_api	integer	Identificateur de l'api	Primaire
id_pt	integer	Identificateur du point	Primaire
level	number	La puissance du signal	

3.3.2. Le modèle relationnel de la base de données

point (id_pt, id_zone, x, y).
zone (id_zone, nom).
api (id_api, nom, mac).
puissance (id_api, id_pt, level).

Conclusion

A l'issue de ce chapitre, nous avons décrit en détails la méthode de géolocalisation utilisé ainsi que son fonctionnement, la démarche suivie et l'algorithme permettant d'estimer la position de l'objet mobile.

Puis nous avons introduit la fonction principale de notre application. Pour atteindre, notre objectif nous avons proposé une solution fondée sur une analyse et une conception modélisées à l'aide du langage de modélisation unifié UML. Pour cela, nous avons défini l'acteur principal de notre application, la tâche qu'il assure. Nous avons élaboré le diagramme de cas d'utilisation, élaboré le diagramme entités/associations .En fin, nous avons représenté les tables de la base de données de notre application.

Le chapitre suivant sera consacré à la réalisation de notre application, en présentant les outils de développement.

Chapitre IV :

Réalisation

Introduction

Après avoir présenté dans le chapitre précédent les différentes étapes d'analyse et de conception. A présent, nous présentons les étapes d'implémentation de l'application.

Ainsi, nous structurons ce chapitre comme suit: dans la première section, nous présentons l'environnement de développement, le langage et les outils utilisés pour la mise en œuvre de notre application, la seconde section est consacrée à la présentation de l'application réalisée à travers quelques interfaces.

1. Environnement et outils de développement

1.1. Partie matérielle

Elle est constituée d'un ensemble de ressources destinées au développement de notre application, citons :

- Un ordinateurs portables Lenovo ayant Windows8.1 comme système d'exploitation.
- Quatre points d'accès : 4 modems Djawab.
- Smartphone doté d'un système Android 5.1.1 : OPPO F1+ et SAMSUNG GRAND PRIME.

1.2. Partie logiciel

Pour pouvoir réaliser une application dans de bonnes conditions il faut en tout premier lieu bien choisir son environnement de travail. Nous avons opté pour la réalisation d'une application mobile sous système Android. Nous avons utilisé android studio et le langage JAVA.

1.2.1. Android Studio 2.3

L'Android Studio est un environnement de développement intégré, conçu spécifiquement pour développer des applications Android. Il est basé sur IntelliJ IDEA (environnement de développement intégré pour le développement de logiciel).

Figure IV.1 : Capture d'écran d'une fenêtre Android Studio.

1.2.2. Le SDK android

Le kit de développement (SDK) d'Android est un ensemble complet d'outils de développement qui permet de développer des applications spécifiques de la téléphonie mobile à mettre en œuvre sur la plate-forme [7].

Le SDK fournit un ensemble d'utilitaires qui sont :

1.2.2.1. Émulateur

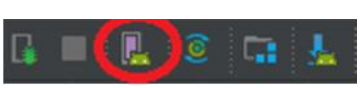
Le SDK comprend un émulateur qui permet de simuler les différentes versions d'Android, permettant ainsi aux développeurs de tester leurs applications ou de tester les fonctionnalités d'Android. Le SDK contient plusieurs images en fonction des différentes versions d'Android.

1.2.3. Android Virtual Device (AVD)

Android Virtual Device, est un émulateur de terminal sous Android; c'est-à-dire que c'est un logiciel qui fait croire à votre ordinateur qu'il est un appareil sous Android. C'est la raison pour laquelle nous n'avons pas besoin d'un périphérique sous Android pour développer et tester la plupart de nos applications avant de les déployer sur les véritables terminaux.

Pour créer un AVD :

- ✓ Nous Lançons Android Studio et nous cliquons sur cette icône :



- ✓ Nous cliquons sur **Create Virtual Device** et nous choisissons une catégorie d'appareil et un appareil dans la liste.

- ✓ Nous cliquons sur **Next** et nous choisissons la version d'Android que nous désirons installer sur l'appareil virtuel.

- ✓ Nous cliquons sur **Next** et nous choisissons un nom pour notre appareil virtuel.

- ✓ Enfin nous cliquons sur **Finish**.

1.2.4. Le langage de programmation Java

Java est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton de Sun Microsystems, et aussi un environnement d'exécution.
Java peut être séparée en deux parties. D'une part, le programme écrit en langage Java et d'autre part, une machine virtuelle (JVM) qui va se charger de l'exécution de ce programme.
C'est cette plateforme qui garantit la portabilité de Java. Il suffit qu'un système ait une machine virtuelle Java pour que tout programme écrit en ce langage puisse fonctionner.

1.2.5. La base de données SQLite

SQLite est un système de base de données qui a la particularité de fonctionner sans serveur, on dit "base de données embarquée".
L'intérêt c'est que c'est très léger et rapide à mettre en place, on peut s'en servir aussi bien pour stocker des données dans une vraie base de données sur une application pour smartphone (iPhone ou Android), pour une application Windows ou sur un serveur web.

Dans notre cas nous avons créé une base de données nommée base.db contenant 4 tables.

Autres outils

Auto CAD et SketchUp : on l'a utilisé pour dessiner le plan du laboratoire LARI.

Photoshop : pour la récupération de la longueur et de la largeur du plan du laboratoire LARI.

2. Fonctionnement de l'application

L'application que nous avons développée comporte une seule activité principale nommée **MainActivity.java.** Illustrée dans l'**AndroidManifest.xml** dans la ***figure IV.2:***

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.lydia.localisation_wifii">

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="localisation_wifii"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="localisation_wifii"
            android:theme="@style/AppTheme.NoActionBar"
            android:configChanges="screenSize|orientation|keyboardHidden">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Figure IV.2 : AndroidManifest.xml

android.permission.ACCESS_WIFI_STATE
android.permission.CHANGE_WIFI_STATE
android.permission.ACCESS_FINE_LOCATION

Ce sont des permissions qu'on doit rajouter, la première sert à chercher l'état du wifi, la seconde pour changer son état et la dernière pour le positionnement.

Afin de se localiser en intérieur précisément dans le laboratoire **LARI** du département informatique nous avons opté pour une méthode statique basée sur les puissances du signal (RSSI) en implémentant l'algorithme du plus proche voisin dit (KPP ou KNN).

En premier lieu nous avons installé et configuré **4 points d'accès** dans le labo de recherche **LARI (Lari 1**, **Lari 2**, **Lari 3** et **Lari**) se trouvant respectivement dans les pièces (**Labo 01**, **Labo 02**, **Labo 03** et **Labo 04**).

Dans la programmation nous avons enregistré les adresses mac des points d'accès (**Lari 1**, **Lari 2**, **Lari 3** et **Lari**) (***Figure IV.3***) puis nous avons filtré tout les autres (***Figure IV.4***) :

```
db.execSQL("INSERT INTO api VALUES (NULL,'LARI1','b0:c5:54:f9:8d:a4')");
db.execSQL("INSERT INTO api VALUES (NULL,'LARI2','b0:c5:54:f9:91:5a')");
db.execSQL("INSERT INTO api VALUES (NULL,'LARI3','b0:c5:54:f9:8d:9e')");
db.execSQL("INSERT INTO api VALUES (NULL,'LARI','b0:c5:54:f9:99:5e')");
```

Figure IV.3 : Enregistrement des adresses mac dans la base de données.

```
public static boolean contains(ArrayList<API> APIs,API api){
    boolean b = false;
    for(API apil : APIs){

        if(apil.mac.equalsIgnoreCase(api.mac)){
            b = true;
            break;
        }
    }
    return b;
}
```

Figure IV.4 : Filtrage des points d'accès.

Nous avons choisi **10 points** aléatoires dans chaque pièce comme points de référence et à l'aide d'une application déjà existante qui est "**wifi analyser**" (ayant comme rôle principal la capture de la puissance du signal d'un point d'accès), Nous avons pris 30 mesures de puissance du signal pour chaque point choisi par rapport aux 4 points d'accès (**Lari 1**, **Lari 2**, **Lari 3** et **Lari**) puis nous avons calculé la moyenne de ces mesures.
Ensuite, nous avons sauvegardé dans une base de données embarquée, ayant 4 tables: **point**, **zone**, **api** et **puissance** tout les points de référence choisis. Illustrés par la ***figure IV.5***:

```
db.execSQL("CREATE TABLE IF NOT EXISTS zone (id integer PRIMARY KEY autoincrement,nom text)");

db.execSQL("CREATE TABLE IF NOT EXISTS point (id integer PRIMARY KEY autoincrement,idzone integer,x number,y number)");

db.execSQL("CREATE TABLE IF NOT EXISTS api   (id integer PRIMARY KEY autoincrement,nom text,mac text)");

db.execSQL("CREATE TABLE IF NOT EXISTS point_api (id integer PRIMARY KEY autoincrement,idapi integer,idp integer,puissance number)");
```

Figure IV.5 : ***Création des tables de la base de données.***

Une fois l'application lancée, on capte en temps réel des puissances de signaux des 4 points d'accès, à chaque fois qu'elles changent et ça pendant "**3secondes**" celles-ci seront sommées pour calculer leur moyenne. La ***figure IV.6*** illustre cela:

```
int n = 30;//30
for(int j=0;j<n;j++){
    try {

        List<ScanResult> wifiScanResult = wifiManager.getScanResults();
        //Log.i("lydia", "NBR : " + wifiScanResult.size());
        for (int i = 0; i < wifiScanResult.size(); i++) {
            API api = new API(0, wifiScanResult.get(i).SSID, wifiScanResult.get(i).BSSID);

            if (contains(APIs,api)) {
                sv.get(api.mac).addLevel(wifiScanResult.get(i).level);
            }
        }
        wifiManager.startScan();
        try {
            Thread.sleep(100);//100 * 30 = 3s
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }catch (Exception ex){
        ex.printStackTrace();
    }

}
```

```
public void addLevel(double level1){
    level += level1;
    nbr++;
}

public double getLevel30() { return level/nbr; }
```

Figure IV.6 : Calcul de la moyenne des puissances des signaux en temps réel.

Exemple: si la puissance du signal a changé **5** fois pendant les "**3secondes**" les 5 résultats seront sommés puis divisé par 5 pour avoir une moyenne pour chacun des points d'accès.

Puis c'est à ce moment là que la formule KPP sera appliquée comme suit:
Vu que nous avons 4 points d'accès donc on aura 4 moyennes de puissances en temps réel et une distance euclidienne sera calculée:

$$d=\sqrt{(RSSI_{TR[1]} - RSSI_{BD[1]})^2 + (RSSI_{TR[2]} - RSSI_{BD[2]})^2 + (RSSI_{TR[3]} - RSSI_{BD[3]})^2 + (RSSI_{TR[4]} - RSSI_{BD[4]})^2}$$

Cette distance est calculée pour tous les enregistrements de la base de données.

Si **k=1** le point de référence qui sera choisi sera le point de la plus petite distance.

Si **k=2** le point qui sera choisi sur le plan c'est la différence entre les 2 plus petites valeurs trouvée par exemple.

Si **k=3** c'est le centroide des 3 plus **proches voisins.**

soit (x1 ; y1) , (x2 ; y2) et (x3 ; y3) les coordonnées cartésiennes de ces trois plus proches voisins trouvés après le tri décroissant des distances. La position est calculée comme suit:

$$\mathbf{x}=\frac{x1+x2+x3}{3}$$

$$\mathbf{y}=\frac{y1+y2+y3}{3}$$

Dans notre cas **k=1** le calcul de la distance se fait par rapport à tous les enregistrements de la basse de données, puis la position estimée est le point de la plus petite distance.

3. Présentation de l'application

L'interface de notre application est illustrée par les ***figures***:

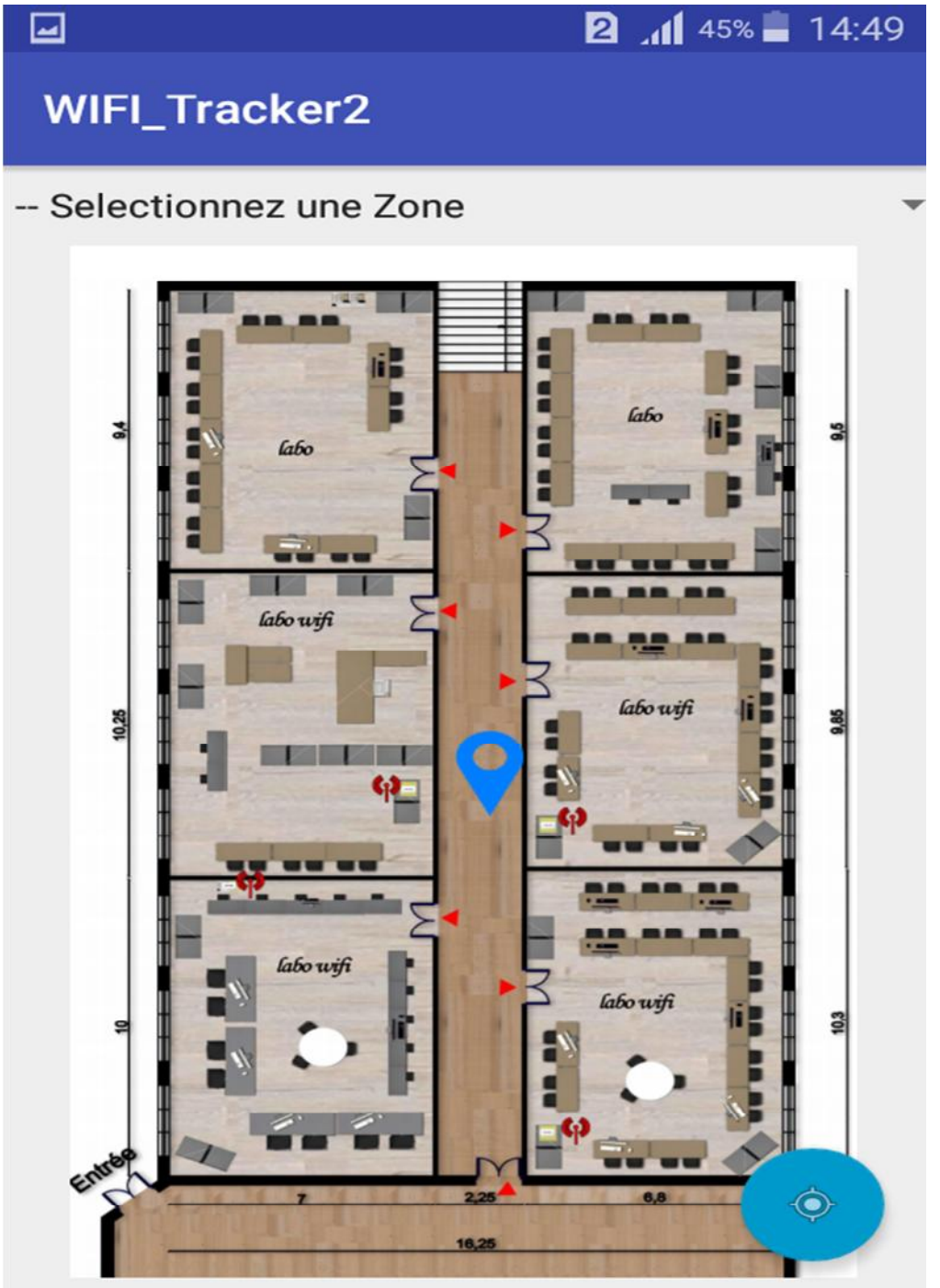


Figure IV.7 : L'objet mobile se trouve au couloire.

L'Objet Mobile se trouve dans le couloire : la position estimée par notre application est similaire à la position réelle avec une marge d'erreur.

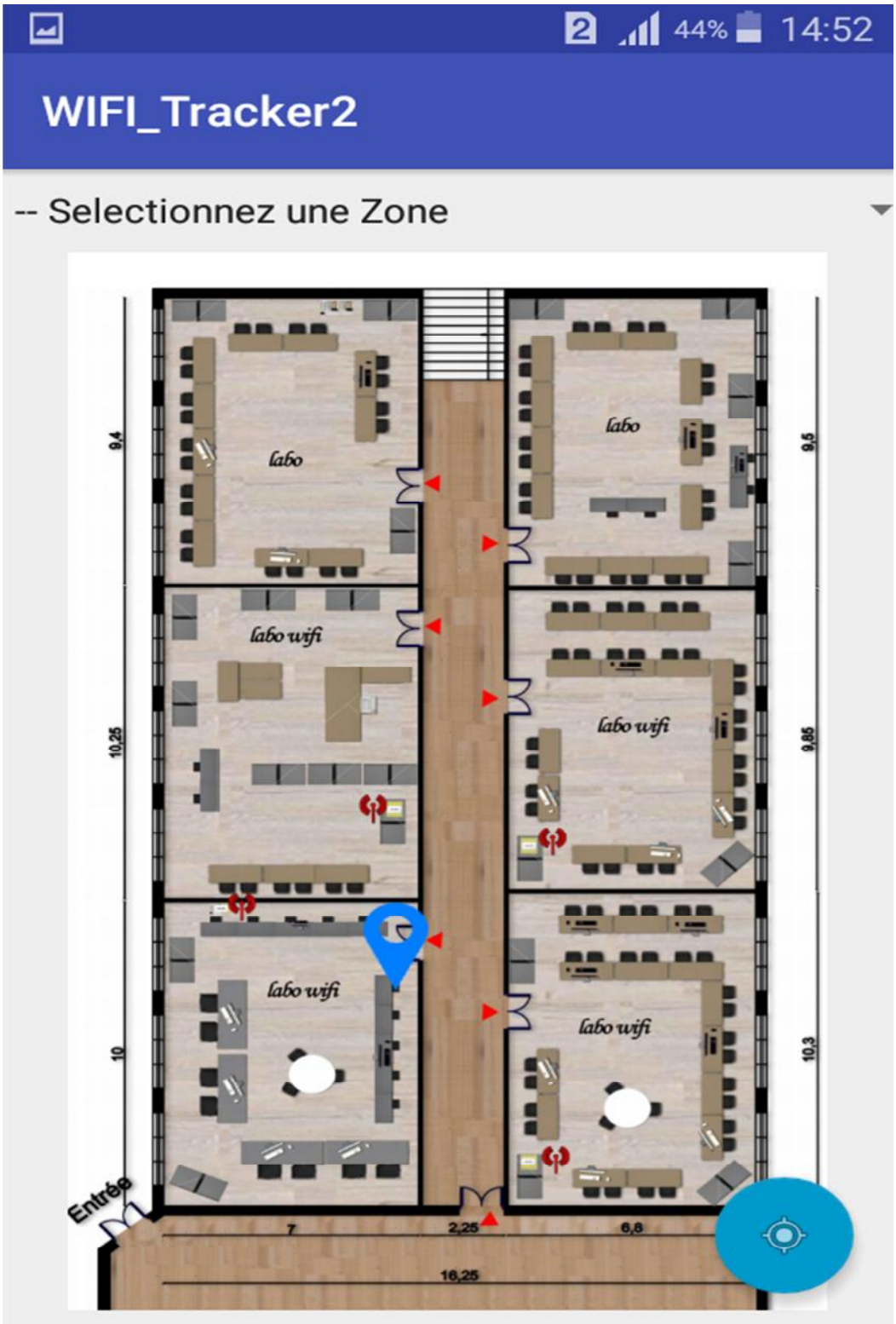


Figure IV.8: L'objet mobile se trouve au labo 4.

L'Objet Mobile se trouve dans le Labo 4 : la position estimée par notre application est similaire à la position réelle avec une petite marge d'erreur.

Conclusion

Dans ce chapitre nous avons, en premier lieu, présenté l'environnement, le langage de programmation et les outils que nous avons utilisé pour implémenter notre application. Par la suite, nous avons expliqué le fonctionnement de notre application et enfin nous avons terminé par la présentation de quelques interfaces de notre application mobile.

Le positionnement en intérieur basé sur les empreintes digitales wifi existantes devient de plus en plus fréquent. Cette méthode parait efficace en intérieur dans des espaces vides ou homogènes, la précision se dégrade rapidement lorsque le milieu devient hétérogène avec la présence de murs se qui induit des interférences, l'atténuation du signal, se qui entraine une faible précision. C'était le problème majeur rencontré lors du développement de notre application.

Conclusion générale

L'objectif du présent travail étant de concevoir et de développer une application sous android permettant de se localiser à l'intérieur. Notre attention s'est orientée vers l'utilisation de la méthode statique basée sur les puissances des signaux émis par les points d'accès. Afin de mieux cerner et comprendre le bon fonctionnement de cette méthode, nous avons présenté dans le premier chapitre des généralités sur la géolocalisation, ses différentes méthodes, et aussi ses techniques de mesure terminant par les systèmes de positionnement.

Ayant utilisé le système android comme outil de développement de notre application, des généralités de se dernier ont été introduites au deuxième chapitre.

Pour le développement de notre application, nous avons suivi une méthodologie typique: étude, analyse, conception, codage et tests. Les trois premières phases ont fait objet du troisième chapitre.

Le quatrième chapitre est donc réservé pour les deux dernières phases. Les outils de développement utilisés ont été mis en avant et différents tests confortés de figures ont été présentés.

Nous estimons que l'objectif fixé au début du travail est atteint, néanmoins des améliorations peuvent être apportées. En plus de la localisation, notre système pourrait être amélioré pour jouer le rôle d'un traceur permettant de voir l'itinéraire suivi par un utilisateur en envoyant sa position d'une manière répétitive.

Au cours de la réalisation de notre travail, des difficultés ont été rencontrées auxquelles nous avons essayé de trouver des solutions.

D'autre part, la réalisation de ce travail nous a permis d'acquérir et d'enrichir nos connaissances et nos compétences aux travers l'immense variété d'outils technologiques disponibles dans le monde informatique à savoir les logiciels de base de données (MySQL), les langages et techniques de programmation (Android et Java), les environnements de simulations (émulateur), les technologies sans fil (WIFI) .

Étant donné que l'application réalisée est développée sous android, cette dernière bénéficie de toute la richesse d'un tel système comme la portabilité, la sécurité et la performance.

Bibliographie

[2] Djennane Nabila, Belkacemi Dihia, Localisation des mobiles par WIFI, mémoire de master, UMMTO 2012/2013.

[3] Jinane Sayah, contribution à la modélisation, à la simulation et à l'évaluation d'application nomades à intelligence répartis- application à l'assistance aux voyageurs aveugles dans les transports public et les pôles d'échanges, thèse en ligne, le 18 décembre 2009.

[4] Samah Kahina, Zaghbib Yamina, conception et réalisation d'une application mobile sous android de géolocalisation indoor (cas : super marché priba de AZAZGA), mémoire de master UMMTO, 2014/2015.

[5] Evanaska Maria Barbosa Nogueira, Conception d'un système d'antennes pour la localisation en temps réel avec réseau de capteurs sans fil, thèse en ligne de doctorat université de GRENOBLE, 13 Décembre 2013.

[6] Truer Pierre, Damien Viatour Olivier, Géopositionnement par WiFi indoor, article de l'université de FRANCHE-COMTE, Jeudi 10 décembre 2009.

[8] Ismahène Hadj Khalifa, Approches de modélisation et d'optimisation pour la conception d'un système interactif d'aide au déplacement dans un hypermarché, thèse en ligne de doctorat Ecole Doctorale SPI 072 PRES Université Lille Nord de France ,16 juin 2011.

[9] Patrick DESSALLE, Conception et réalisation d'une plateforme de déploiement de services géolocalisés, mémoire de fin d'étude en vue de l'obtention du grade d'Ingénieur Civil Informatique Université libre de Bruxelles, 2005/2006.

[10] Iness Ahriz Roula, Application de technique d'apprentissage à la géolocalisation par radio fingerprint, thèse de Doctorat université de Piérre et Marie Curie, 2010.

[11] SEGHIER Nor El Houda, Localisation d'un mobile dans un réseau UMTS, Thèse de Magister université des Sciences et de la Technologie d'Oran Mohamed BOUDIAF,, 29/01/2013.

[12] Jean Pierre HAUET, L'identification par radiofréquences (RFID) – Techniques et perspectives, article.

[13] Jana KALAWOUN, Localisation Indoor à base d'un Réseau de Capteurs Sans Fil, thèse d'ingénieur, Juillet 2012.

[14] Séoul Corée, OCDE : organisation de coopération et de Développement Economiques, direction de la science RFID identification par radio fréquence réunion ministérielle de l'OCDE le futur de l'économie internet, 17,18 juin 2008

[15] BRUN -MUROL Pierre, Vers une méthodologie normalisée d'évaluations des solutions RFID en application de sécurité, mémoire présenté en vue de l'obtention du diplôme de maitrisées sciences appliquées (génie informatique), Ecole polytechniques de Montréal, avril 2013

[16] Département Informatique de l'Institut Universitaire de technologie de l'Université bordeaux 1 Analyse et Conception des systèmes d'information - Méthodes Objet Le langage de modélisation objet UML Olivier

[17] Olivier Sigaud, Introduction à la modélisation orientée objets avec UML

[18] Joseph Gabay, Davis Gabay, uml 2 analyse et conception ,2008

[19] pascal Rpques, uml 2 par la pratique ,6eme édition EYPOLLES

[22] 802.11 Les réseaux sans fils Ebook, Mars 2003

Webliographie

[1] http://www.timcod.fr/solutions/geolocalisation-indoor-outdoor/

[7] http://www.cnil.fr/listitution/actualite/article/la-geolocalisationa-partir-despointsdacces-wi-fi

[20] https://fr.wikipedia.org

[21] https://www.linksys.com/be/r/amplificateur-de-signal/qu'est-ce-qu'un-point-d'accès/

[27] Tutorielandroid.francoiscolin.fr/structproj.php

[28] ANDROID-DEV.FR

[23] http://www.fandroid.com/quest-ce-que-android

[24] http://socialcompare.com/fr/comparison/android-versions-comparison

[25] https://search?q=architecture+android

[26] http://mathias-seguy.developpez.com