

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU



FACULTE DU GENIE ELECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes de MASTER ACADEMIQUE

Domaine : **Mathématiques et Informatique**

Filière : **Informatique**

Spécialité : **Conduite de Projet Informatique**

Présenté par

Ferroudja MEHOUEL

Idir MAMMAR

Thème

Conception et réalisation d'une application Android pour la surveillance à domicile

Mémoire soutenu publiquement le 30/06/ 2016 devant le jury composé de :

Président : M^{me} R.AOUDJIT

Encadreur : M^{me} R.HADAoui

Co-Encadreur :

Examineur : Mr M.DAOUI

Examineur : M^{me} M.BELKADI

Remerciements

Nous tenons à témoigner notre plus grande reconnaissance au Dieu tout puissant sans qui tout cela n'aurait été possible.

*Nous sommes reconnaissants envers tous ceux qui, par leurs compétences scientifiques ont contribué au bon déroulement de notre mémoire et particulièrement, nous témoignons notre profonde reconnaissance à Mme**HADAoui**, d'avoir accepté de nous encadrer, pour sa patience et son dévouement durant la correction de notre travail.*

Nous Remercions également les membres du jury, qui ont bien voulu nous faire l'honneur de juger notre travail.

Nous adressons nos sincères remerciements à tous ceux
ou celles qui nous ont aidés d'une quelconque manière
soit-elle.

Dédicaces

Au nom de dieu, le Tout Miséricordieux, le Très Miséricordieux

Je dédie ce travail, particulièrement, à mes parents que j'aime.

A mes frères : Moh et Tahar.

A mes sœurs : Nassima, Lynda ,Melissa, Katia

A mon mari :Sadak

A mon binôme Idir.

Sans oublier tous mes amis(es) et mes collègues.

FERROUDJA

Dédicaces

Au nom de dieu, le Tout Miséricordieux, le Très Miséricordieux

Je dédie ce travail, particulièrement, à mes parents que j'aime.

*A mes frères : **Said et yacine.***

*A ma binôme **Ferroudja** et son marie **Sadak.***

Sans oublier tous mes amis(es) surtout ma meilleur amies

Melissa** et mon meilleur ami **Mouloud.

*A la plus merveilleuse des femmes **Taous.***

IDIR

Chapitre I

Figure I.1: Le personnage Bugdroid -----	04
Figure I.2 : Evolution des versions Android-----	04
Figure I.3 : Architecture de l'Android-----	10
Figure I.4 : Architecture de ces médiathèques-----	12
Figure I.5: Transformation d'un fichier .java à un fichier.dex -----	13
Figure I.6: Cycle de vie d'une activité-----	16
Figure I.7 : Une application sur Android-----	18
Figure I.8 : Portail des développeurs Android-----	19
Figure I.9 : Interface d'installation du SDK Android -----	20
Figure I.10 : Interface du simulateur Android-----	21

Chapitre II

FigureII.1: géolocalisation par satellite-----	24
Figure II.2 : géolocalisation par GSM -----	24
Figure II.3 : géolocalisation par wifi-----	25
Figure II.4: géolocalisation indoor -----	26
Figure II.5 : Constellation GPS -----	28
Figure II.6 : Composantesdu système GPS -----	28
Figure II.7 : Segment spatial -----	29
Figure II.8 : Localisation du segment terrestre -----	29
Figure II.9 : Segment de contrôle -----	30
Figure II.10 : Segment utilisateur -----	30
Figure II.11 : Un satellite -----	31
Figure II.12 : Deux satellites -----	31

Figure II.13 : Trois satellites -----	32
Figure II.14 : couche atmosphérique-----	33
Figure II.15 : trajet du signal GPS dans l'atmosphère -----	33
Figure II.16 : Erreurs multitrajet -----	34
Figure II.17 : Erreurs géométriques -----	34
Figure II.18 : Un ellipsoïde-----	35
Figure II.19 : Coordonnées cartésiennes et géographique du point P -----	35
Figure II.20 : Relation entre ellipsoïdes et surface terrestre -----	36
Figure II.21 : Identification par cellules -----	37
Figure II.22 : Identification par le temps -----	37

Chapitre III

Figure III.1 : Principe du fonctionnement de notre l'application -----	41
FigureIII.2 : La démarche adoptée pour la modélisation -----	42
FigureIII.3 : Logo d'UML -----	43
FigureIII.4:Fondateursd'UML -----	44
FigureIII.5 : Diagrammes d'UML 2.0 sous forme de diagrammes de classe-----	45
FigureIII.6 : Cas d'utilisation «Ajouter un surveillant»-----	48
FigureIII.7:Cas d'utilisation «Supprimer un surveillant»-----	49
FigureIII.8 :Cas d'utilisation «Modifier surveillant»-----	49
FigureIII.9 :Cas d'utilisation «configurer l'alarme»-----	49
FigureIII.10 :Cas d'utilisation «Désactiver l'alarme»-----	49
FigureIII.11 :Relation entre acteur et cas d'utilisation -----	50
FigureIII.12 : Relation d'héritage -----	50
FigureIII.13 :Relations d'extension et d'inclusion-----	51

FigureIII.14 :Diagramme de cas d'utilisation de l'application patient-----	51
FigureIII.15 :Diagramme d'activité du cas d'utilisation «Ajouter un surveillant»-----	53
FigIII.16 : Diagramme d'activité du cas d'utilisation « Supprimer un surveillant »-----	54
FigureIII.17 :Diagramme d'activité du cas d'utilisation « configurer l'alarme»-----	55
FigureIII.18 :Formalisme du diagramme de séquence -----	56
FigureIII.19 :Diagramme de séquence simple de cas d'utilisation «Ajouter un surveillant»----	57
FigureIII.20:Diagramme de séquence simple de cas d'utilisation «Supprimer unsurveillant»---	58

Chapitre IV

Figure IV.1 : Interface d'installation d'Android SDK-----	62
Figure IV.2: Création d'un AVD-----	63
Figure IV.3 : Emulateur Android-----	64
Figure IV.4 : Page d'accueil de l'application-----	67
Figure IV.5 : Interface de la configuration de l'alarme -----	68
Figure IV.6 : Interface d'arrêt de l'alarme -----	69
Figure IV.7 : Interface d'ajout de surveillant-----	70
Figure IV.8 : Interface de suppression d'un surveillant -----	71
Figure IV.9 :Interface de modification d'un surveillant -----	72
Figure IV.10 : Interface de la liste des surveillants -----	73

Chapitre I :

Tableau I.1 : Fonctionnalités d'Android -----	6
---	---

Chapitre III :

Tableau III.1 : Spécification des tâches -----	46
Tableau III.2 : Spécification des scénarios-----	47
Tableau III.3 : Table Surveillants -----	59

Introduction générale	1
Chapitre I : Etat de l'art d'Android	
Introduction	2
I.1. Description	3
I.2. Open Handset Alliance	3
I.3. Bugdroïd	3
I.4. Historique des versions d'Android	4
I.6. Différents supports	8
I.6.1. Règles communes	8
I.6.2. Smartphones	9
I.6.3. Téléviseurs	9
I.6.4. Tablette	9
I.6.5. Console de jeux vidéo	9
I.6.6. Montres	10
I.7. Architecture Android	10
I.7.1. Applications	11
I.7.2. Framework de développement	11
I.7.3. Bibliothèques	11
I.7.4. Android Runtime	13
I.7.5. Linux Kernel	14
I.8. les composants principaux d'une application	14
I.8.1. Activités	14
I.8.2. Services	14
I.8.3. Broadcast receivers	15
I.8.4. Content providers	15
I.9. Cycle de vie d'une activité	15

I.10. Développement d'une application Android	17
I.11. Les outils de développement Android	19
I.11.1.Présentation du SDK	19
I.11.2.Le SDK Android :	20
I.11.3.Le débogueur «ADB »	20
I.11.4.Les IDE«Android Developer Tools »et «Android studio »	20
I.11.5.Eclipse.....	20
I.11.6.Android studio.....	20
I.11.7. Emulateur.....	20
Conclusion	22

ChapitreII : Géolocalisation

Introduction.....	23
II.1. Techniques de géolocalisation.....	23
II.1.1. Géolocalisation par satellite.....	23
II.1.2. Géolocalisation via la téléphonie mobile (GSM).....	24
II.1.3. Géolocalisation par Wifi.....	25
II.1.4. Géolocalisation indoor	25
II.1.5. Géolocalisation par adresse IP (sur Internet).....	26
II.2. Technologie de positionnement	26
II.2.1. Global Positionning System (GPS).....	27
II.2.1.1. Historique du système GPS	27
II.2.1.2. Composant du système GPS	28
II.2.1.2.1. Le segment spatial	28
II.2.1.2.2. Le segment contrôle	29
II.2.1.2.3. Le segment utilisateur.....	30
II.2.1.3. A quoi sert le GPS?	30

II.2.1.4.Fonctionnement du GPS.....	31
II.2.1.5. Sources d'erreurs.....	32
II.2.1.5.1. Les erreurs issues du satellite.....	32
II.2.1.5.2. Les erreurs dues à l'atmosphère.....	33
II.2.1.5.3. Erreurs de multitrajets	33
II.2.1.5.4. Erreurs géométriques.....	34
II.2.1.6. Le système des coordonnées GPS.....	35
II.2.1.7. Systèmes de coordonnées locaux.....	35
II.2.1.8. Les applications du GPS.....	36
II.2.2. L'identification par cellules, Cell-ID.....	36
II.2.3. Identification par le temps (E-OTD).....	37
II.2.4. Identification par triangulation	38
II.3. Autre GNSS.....	38
II.3.1. Glonass	38
II.3.2. Galileo	38
II.3.3. Compass	38
Conclusion	39

Chapitre III : Analyse et conception

Introduction.....	40
III.1. Objectif de l'application:.....	40
III.2.Modélisation avec L'UML.....	42
III.2.1. Définition de la modélisation :	42
III.2.2. Langage de modélisation :	42
III.3. Présentation d'UML (Unified Modeling Language):.....	43
III.3.1. Définition :	43
III.3.2. Historique de l'UML.....	43

III.3.3. Diagrammes d'UML :	44
III.4. L'analyse	45
III.4.1. Identification des acteurs :	45
III.4.2. Spécification des tâches	46
III.4.3. Spécifications des scénarios :	46
III.4.4. Les cas d'utilisation :	47
III.5. Conception :	52
III.5.1 Diagramme d'activité:	52
III.5.2. Diagramme de séquence :	55
III.5.3. Diagramme de classe :	59
III.6. Conception de la base de données :	59
Conclusion	60

Chapitre IV : Réalisation

Introduction	61
IV.1. Outils et langage utilisé	61
IV.1.1. Les outils	61
IV.1.1.1. L'environnement matériel	61
IV.1.1.2. L'environnement logiciel	61
A) Android Studio	62
B) Le SDK Android	62
C) La création d'AVD	63
D) L'émulateur	63
E) La base de données SQLite	64
F) Les alarmes	64
G) La localisation	65
IV.1.2. Le langage Java et Xml	65

IV.1.2.1 Le Java	65
IV.1.2.2 Le XML.....	65
IV.2. Implémentation de notre travail	66
IV.3 Quelques interfaces.....	67
Conclusion	74
 Conclusion générale	 75
Référence bibliographique	

Avec l'accroissement démographique en nette progression et le vieillissement d'une bonne frange de la population, la plus part des citoyens ou des patients à leur domicile est ce qui devient pénible de les assister quotidiennement, et ils ne peuvent plus répondre efficacement aux sollicitations de leurs patients. D'où l'assistance des patients à domicile devient un problème épineux et quasi ingérable.

Depuis sa création, l'environnement Android ne cesse de progresser, ajoutant toujours de nouvelles fonctionnalités pour faciliter la vie quotidienne des utilisateurs. La concurrence avec les autres systèmes d'exploitation, tel que l'iOS d'iPhone et le Windows de Nokia a également poussé l'amélioration des produits Android. Selon une étude réalisée par le cabinet américain StratégieAnalytics, le janvier 2015, une personne sur sept aujourd'hui a un Smartphone.

C'est pour profiter de cette généralisation du Smartphone et remédier aux problèmes rencontrés quotidiennement pour la surveillance des patients à domicile que l'idée de notre projet est née.

Notre travail consiste donc à développer une application pour la surveillance à domicile qui s'intéressera à mieux gérer la prise en charge des patients.

Afin d'atteindre cet objectif nous avons structuré notre mémoire en quatre chapitres.

- Le premier chapitre intitulé "GENERALITES SUR ANDROID" traite des concepts de base de système Android.
- Le deuxième chapitre intitulé "LA GEOLOCALISATION" présente la géolocalisation en insistant beaucoup plus sur la technique de localisation par GPS.
- Le troisième chapitre intitulé "ANALYSE ET CONCEPTION", détaille les phases d'analyse et de conception de notre travail en utilisant la modélisation UML.
- Le dernier chapitre intitulé "LA REALISATION" présente les outils utilisés ainsi que quelques interfaces montrant le résultat final de notre application.
- Enfin, nous terminons ce mémoire par une conclusion générale dans laquelle nous avons fait la synthèse du travail accompli et les perspectives de ce travail afin d'améliorer les performances de notre application.

Introduction

Durant cette dernière décennie, les technologies mobiles ont connus de grands progrès, les développements embarqués sont de plus en plus demandés sur le marché.

L'Android est une excellente opportunité pour appréhender le développement d'applications mobiles ambitieuses car la plateforme Android comporte plusieurs points clés :

- Elle est innovante car toutes les dernières technologies de téléphonie y sont intégrées : écran tactile, accéléromètre, GPS, appareil photo numérique etc.
- Elle est accessible car en tant que développeur vous n'avez pas à acheter de matériel spécifique (si vous voulez aller plus loin que l'utilisation d'un émulateur, un téléphone Android pour effectuer vos tests vous sera toutefois nécessaire), ni à connaître un langage plus utilisé ou spécifique : le développement sur la plate-forme Android est en effet réalisé en langage java, un des langages de programmation les plus répandus.
- Elle est ouverte parce que la plateforme Android est fournie sous licence open source, permettant à tous les développeurs et constructeurs de consulter les sources et d'effectuer les modifications qu'ils souhaitent.

Ce chapitre est une introduction à la plate-forme, aux outils et à la configuration de notre environnement de travail.

I.1. Description

Android est un système d'exploitation pour Smartphones et tablettes conçu par Android. D'autres types d'appareils possédant ce système d'exploitation existent, par exemple des téléviseurs, des radioréveils ou des autoradios et même des voitures. Il a été développé par une petite startup qui fut achetée en 2007 par Google qui poursuit activement son développement avec l'Open Handset Alliance, Android est distribué sous licence open source depuis 2008. Ce système est assez nouveau auprès des programmeurs.

Selon Google qui est le majeur distributeur, Android est une plateforme puissante, moderne, sécurisée et ouverte. Elle est basée sur le Kernel Linux 2.6 (noyau Linux 2.6) et utilisant la plateforme Java pour ses applications. Elle est entièrement gratuite et sa plateforme très flexible ce qui permet au développeur d'intégrer, d'agrandir et de remplacer les composants existants et d'adapter les applications aux besoins du client ou les remplacer entièrement, l'utilisateur peut donc personnaliser facilement son appareil.

De plus, il n'y a pas de distinction entre les applications natives et les applications qui sont développées par les développeurs, toutes sont disponibles sur l'**Android Market** (maintenant appelé **Google Play Store**).

En termes d'applications, Android à intégrer plusieurs services de Google pour accéder rapidement aux services d'internet comme Gmail, Youtube, Google Talk, Google Calendar et Google Maps.[1]

I.2. Open Handset Alliance

L'Open Handset Alliance (abrégié OHA) est un consortium regroupant de grands constructeurs et développeurs de logiciels dont le but est de développer des normes ouverts pour les appareils de téléphonie mobile.

Le consortium a été créé le 5 novembre 2007 à l'initiative de Google qui a su fédérer autour de lui 34 compagnies. On note ainsi la présence d'opérateurs comme NTT DoCoMo, Sprint Nextel, Telecom Italia ou Bouygues Telecom. Des équipementiers tels que Samsung Electronics, Motorola Mobility ou LG Electronics. Des semi-conducteurs comme Intel ou Nvidia. Et enfin des incontournables de l'Internet dont eBay.[2]

I.3. Bugdroïd

Le personnage nommé Bugdroïd est le petit robot vert utilisé par Google pour présenter Android. Ce personnage est sous licence « creative commons by (3.0) » et peut donc être utilisé librement.

Le site Engadget annonce que bugdroid, le robot Android, sera en fait un personnage d'un jeu des années 1990 sur Atari : Gauntlet : The Third Encounter.



Figure I.1: Le personnage Bugdroid

I.4. Historique des versions d'Android



Figure I.2 : Evolution des versions Android.

Android a débuté avec la sortie de la version 1.0 « Apple pie » en septembre 2008. Android a connu plusieurs mises à jour depuis sa première version. 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 des codes suivent une logique alphabétique, en voici quelques-unes :

Android 1.5 « Cupcake » a apporté de nombreuses améliorations. La première grande étape fut une mise à jour du noyau vers la version 2.5.27 qui a rendu le système plus stable et corrigea le manque d'API et rend le système plus utilisable.

Android 1.6 « Donut », Android 2.0 et 2.1 « Eclair » ont apporté d'importantes améliorations respectivement sur les fonctionnalités et sur l'interface graphique du système.

Android 2.2 « Froyo » a fortement mis l'accent sur la synergie avec Internet. L'envoi d'applications et de liens instantanés depuis un ordinateur est désormais possible. Aussi Google annonce-t-elle que le navigateur chrome intégré à Android 2.2 est le navigateur mobile le plus rapide au monde grâce à l'intégration du moteur JavaScript V8.

Android 2.3 « Ginerbread » dernière version dédié uniquement aux Smartphones. Cette version est parfois utiliser sur de petites tablettes.

Gingerbread était un système très réussi, qui devint à l'époque le système mobile le plus populaire.

Android 3.0 « Honeycomb » est spécialement étudié pour les tablettes tactiles. Les premiers modèles devraient être annoncés au CES 2011.

On y apprend quelques nouveautés comme la prise en charge de la vidéoconférence via Gtalk, la nouvelle interface Gmail ou encore le lecteur de livre électronique Google.

La refonte graphique de l'interface utilisateur est assez réussie, plus d'informations devraient suivre dont sûrement des éclaircissements sur l'intégration ou non de l'interface de cette version d'Android sur les futurs Smartphones.

Android 4.0 « IceCream Sandwich » devrait arriver très vite pour rajouter encore plus de fonctionnalités aux terminaux. Pour le développement, ces nouvelles versions d'Android devraient proposer de nouveaux composants permettant de réaliser des applications avec une ergonomie plus adaptée aux tablettes tactiles.

Android 3.0 et Android 4.0 devraient apporter plus d'outils aux constructeurs leur permettant de proposer des tablettes tactiles, qui seront capable de rivaliser (surtout au niveau de l'ergonomie) avec Ipad.

Android 4.1 « Jelly Bean » il ajoute un système de notification améliorée, la reconnaissance vocale sans connexion Internet, et le « projet Butter » qui augmentent la fluidité d'Android.

Android 4.4 « KitKat » consommation en ressource moins élevée nécessitant moins de RAM, nouvelles icône plus soignées, la barre du bas et celle de statut deviennent transparente sur certains menus et change de couleur en fonction du contenu affiché.

Android 5.0 « Lollipop » Les changements les plus importants d'Android 5.0 sont sa disponibilité sur les nouvelles plateformes Android TV et Android Auto, le renouveau de l'interface utilisateur nommée Material Design, l'amélioration du système de notifications, l'environnement d'exécution ART qui remplace officiellement Dalvik pour améliorer les performances, ainsi que l'amélioration de l'autonomie de la batterie via le projet Volta.

Android 5.1 « Lollipop » contient de nouvelles fonctionnalités pour les développeurs, avec plus de 5000 nouvelles API ajoutées pour une utilisation par les applications. Les applications Appareil photo, Calculatrice, Contacts, Horloge, Paramètres, Téléchargement et Téléphone ont été mises à jour. L'USB Audio est supporté. Le pack d'extension Android API fournit

également des fonctions graphiques telles que de nouveaux shaders, visant à fournir des graphiques de niveau de PC pour des jeux en 3D sur les appareils Android

Android 6.0 « Marshmallow » se concentrera principalement sur l'amélioration de l'expérience globale de l'utilisateur et apportera quelques fonctionnalités comme un modèle d'autorisation repensé dans lequel les applications ne sont plus accordées automatiquement avec l'ensemble de leurs autorisations spécifiées lors de l'installation, le mode d'alimentation Doze pour une autonomie prolongée lorsque le périphérique n'est pas manipulé par l'utilisateur ainsi qu'un support natif pour la reconnaissance des empreintes digitales.

I.5. Fonctionnalités d'Android

Android a été conçu pour intégrer au mieux les applications existantes de Google, comme le service de courrier Gmail, l'agenda Calendar ou encore la cartographie Google Maps. [3]

Voici quelques fonctionnalités proposées par Android classées par version :

Version	Nom	API Level	Fonctionnalités
Android 1.0	Apple Pie (tarte aux pommes)	1	<ul style="list-style-type: none"> • Fonctionnalités de base (téléphonie, SMS, réseaux de données, accès au répertoire, ... etc.)
Android 1.1	Banana Split	2	<ul style="list-style-type: none"> • Correction de bogues (alarme, Gmail ...) • Amélioration des fonctionnalités MMS • Introduction des applications payantes sur AndroidMarket
Android 1.5	Cupcake (petit gâteau)	3	<ul style="list-style-type: none"> • Nouveau clavier avec auto complétion • Support Bluetooth • Enregistrement de vidéo • Widgets • Animation améliorées
Android 1.6	Donut (Beignet)	4	<ul style="list-style-type: none"> • Recherche vocale améliorée • Indicateur d'utilisation de la batterie • Apparition de fonctionnalités pour les réseaux privés virtuels(VPN)

Android 2.0 , 2.0.1 , 2.1	Eclair (sorte de gâteau à la crème)	5, 6, 7	<ul style="list-style-type: none"> • Interface graphique améliorée • Support d'HTML 5 • Support Bluetooth 2.1 • Clavier virtuel amélioré • Réduction de la consommation de la batterie • Quelques améliorations graphiques comme l'écran de verrouillage • Fonds d'écran animés • Nouveau effets 3D (notamment sur la gestion des photos) • Amélioration du client Gmail
Android 2.2	Froyo (yaourt glacé)	8	<ul style="list-style-type: none"> • Nouveau Widgets • Gestion des points d'accès au réseau • Support multilingue
Android 2.3	Gingerbread (pain d'épice)	9	<ul style="list-style-type: none"> • Amélioration d'interface graphique • Téléphonie IP • Nouveau clavier virtuel
Android 3.0	Honeycomb (rayon de miel)	11	<ul style="list-style-type: none"> • Optimisé pour les tablettes et équipement à écran large • Multitâche et système de notification amélioré
Android 4.0	Icecream sandwich (sandwich à la crème glacée)	12	<ul style="list-style-type: none"> • Nouvelle interface graphique • Amélioration de la sécurité • Beaucoup de raccourcis (Appareil photo, accès sdcard, ... etc)
Android 4.2	Jelly Bean (dragée)	16	<ul style="list-style-type: none"> • Interface utilisateur plus fluide • Notifications extensibles • Recherche vocale améliorée • Application appareil photo améliorée
Android 4.4	kitkat	19	<ul style="list-style-type: none"> • Nouvelle interface translucide. • Enregistrement séquence vidéo de l'écran. • Amélioration du système de notification. • Gestion système des sous-titres. • Amélioration des performances.

Adroid 5.0	Lollipop	21	<ul style="list-style-type: none"> • Nouvelle interface/design (Material design). • Amélioration de la rapidité. • Amélioration de la gestion de la batterie.
Android 5.1	Lollipop	22	<ul style="list-style-type: none"> • Supporte plusieurs cartes SIM. • Raccourci pour joindre un réseau WIFI ou contrôler un appareil Bluetooth. • Protection par blocage en cas de perte ou de vol. • Appel voix en Haute Définition. • Amélioration de la stabilité et des performances.
Android 6.0	Marshmallow	23	<ul style="list-style-type: none"> • Support de USB type –c. • Support de l'authentification par empreinte digitale. • Amélioration de la durée de batterie avec un mode «deepsleep ». • Panneau pour contrôler les permissions des applications. • AndroidPay. • Amélioration de Google Now.

Tableau I.1 : Fonctionnalités d'Android

I.6. Différents supports

I.6.1. Règles communes

Le projet Android définit un support compatible comme étant un support capable d'exécuter n'importe quelle application Android. Pour permettre aux fabricants d'y parvenir, le projet Android réalise le Compatibility Program (programme de compatibilité). [3]

Il détaille ses objectifs ainsi:

- fournir aux développeurs un environnement cohérent quant aux matériels et aux applications,
- fournir aux utilisateurs un environnement applicatif cohérent,
- permettre aux fabricants de se différencier tout en restant compatibles,
- minimiser les coûts associés au respect de la compatibilité.

Les matériels précisés dans ce programme sont :

- écran.
- Clavier.
- différents capteurs : accéléromètre, magnétomètre, GPS, gyroscope, baromètre, thermomètre (mais il est déconseillé), et détecteur de proximité.
- différents mode de connexion.
- la caméra.
- carte mémoire.
- USB.

I.6.2.Smartphones

Le premier mobile commercialisé sous Android est le HTC G1/Dream produit par la firme Taïwanaise HTC, lancé aux États-Unis sur le réseau T-Mobile le 22 octobre 2008.

I.6.3.Téléviseurs

Le 5 avril 2010, la première télévision sous Android est dévoilée. Celle-ci est développée par l'entreprise suédoise People of Lava et se nomme Scandinavia. Elle possède les applications Facebook, YouTube, Google Maps et Twitter, possède un navigateur Web ainsi qu'un client de messagerie électronique.

Le 20 mai 2010, Google présente lors de sa conférence I/O le concept Google TV, les télévisions présentées précédemment ne faisant pas partie du projet en lui-même Google TV.

I.6.4. Tablette

En septembre 2010, Samsung présente à l'IFA de Berlin le Samsung Galaxy Tab, tournant sous Android 2.2 (FroYo) et sortie fin 2010. Archos avec sa génération 7 de tablettes internet introduit Android (lancée en septembre 2009). Dans la même lignée, les tablettes Archos de la génération 8 (Gen 8) intègrent Android 2.2 (FroYo).

I.6.5.Console de jeux vidéo

DEA commercialise à partir de septembre 2012, sous le nom de MyPlay, une console de jeux vidéo portable sous Android Archos a aussi commercialisé une tablette/console nommée

Game Pad. En mars 2013 ce sera au tour de Boxer8 de sortir leur console Ouya, basé sur le free to play. Nvidia a sorti une machine portable, la Nvidia Shield , qui tourne sous Android et est basée sur son nouveau processeur graphique pour mobile.

I.6.6. Montres

En 2013, le cabinet d'étude Strategy Analytics évalue à 61% les parts de marchés d'Android sur le marché des montres connectées. Le 18 mars 2014, Google dévoile Android Wear, une version modifiée d'Android, dédiée au Smartwatches, le SDK de la première version est disponible depuis la conférence Google I/O 2014, quelques jours plus tard, LG commercialise sa G Watch et Samsung une Gear Live. Motorola a aussi dévoilé sa montre, la Moto 360, qui a su faire parler d'elle pour son cadran rond et son design.

I.7. Architecture Android

Le diagramme suivant illustre les composants principaux du système d'exploitation Android. Chaque section sera décrite dans ce qui suit : [4]

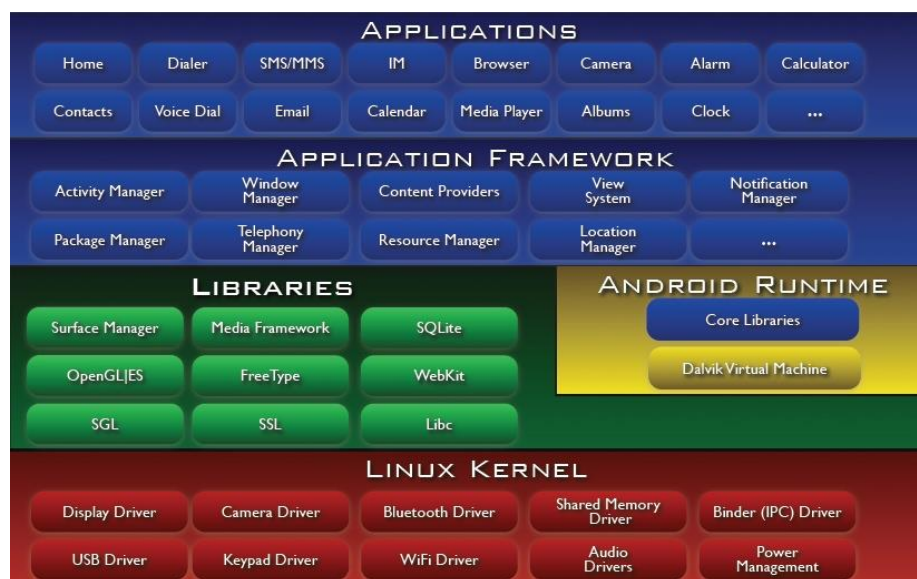


Figure I.3 : Architecture de l'Android

Android est basé sur un Kernel linux 2.6.xx.

Au-dessus de cette couche, on retrouve les librairies C/C++ utilisées par un certain nombre de composants du système Android.

Au-dessus des librairies, on retrouve l'Android Runtime. Cette couche contient les librairies cœurs du Framework ainsi que la machine virtuelle exécutant les applications.

Au-dessus de la couche « Android Runtime » et des librairies cœurs, on retrouve le Framework permettant au développeur de créer des applications. Enfin au-dessus du Framework, il y a les applications.

I.7.1. Applications

Android est fourni avec un ensemble d'applications dont un client email, une application SMS, un calendrier, un service de cartographie, un navigateur...toutes écrites en JAVA.

I.7.2. Framework de développement

En fournissant une plateforme de développement ouverte, Android offre aux développeurs la possibilité de créer des applications extrêmement riches et innovantes. Les développeurs sont libre de profiter du matériel périphérique et informations sur la localisation d'accès, exécuter des services d'arrière-plan, définir des alarmes, ajouter des notifications à la barre d'état, etc.

Les développeurs ont un accès complet au même framework API utilisé par les applications de base. L'architecteur d'application est conçue pour simplifier la réutilisation des composants, n'importe quelle application peut publier ses capacités et n'importe quelle autre application peut alors faire usage de ces capacités (soumis à des contraintes de sécurité appliquées par le framework). Ce même mécanisme permet aux composants d'être remplacés par l'utilisateur.

Toute les applications sous-jacentes forment un ensemble de services et de systèmes, y compris :

- Un jeu extensible de vues qui peuvent être utilisées pour construire une application.
- Des fournisseurs de contenu qui permettent aux applications d'accéder aux données d'autres applications (telles que les Contacts), ou de partager leurs propres données.
- Un gestionnaire de ressources.
- Un gestionnaire de notification qui permet à toutes les demandes d'afficher des alertes personnalisées dans la barre d'état.
- Un gestionnaire d'activité qui gère le cycle de vie des applications et propose une navigation commune.

I.7.3. Bibliothèques

Android dispose d'un ensemble de librairies C / C++ utilisées par les différents composants du système Android. Elles sont offertes aux développeurs à travers le framework Android. En voici quelques-unes :

Système de bibliothèque C : une mise en œuvre dérivée de BSD de la bibliothèque C standard du système (libc), destinés aux systèmes embarqués basés sur Linux.

Comme cela a été dit précédemment, Android ne supporte pas `glibc`, donc les ingénieurs d'Android ont développé une librairie C (`libc`) nommé `bionic libc`. Elle est optimisée pour les appareils mobiles et a été développée spécialement pour Android.

Les ingénieurs d'Android ont décidé de développer une `libc` propre à la plateforme Android car ils avaient besoin d'une `libc` légère (la `libc` sera chargée dans chaque processus) et rapide (les appareils mobiles ne disposent pas de CPU puissant).

La `Bionic libc` a été écrit pour supporter les CPU ARM, bien que le support x86 soit présent. Il n'y pas de support pour les autres architectures CPU telles que power PC ou MIPS. Néanmoins, pour le marché des appareils mobiles, seulement l'architecture ARM est importante.

Cette `libc` est sous licence BSD. Elle reprend une grande partie du code des `glibc` issue d'OpenBSD, FreeBSD et NetBSD.

Ces caractéristiques importantes :

Elle pèse environ 200Ko, soit la moitié de la `glibc`. L'implémentation des `pthread` (POSIX thread) a été complètement réécrite pour supporter les threads de la machine virtuelle Dalvik. De ce fait, la `Bionic libc` ne supporte pas les threads POSIX.

Les exceptions C++ et les « `wide char` » ne sont pas supportés.

Médiathèques : basée sur `PacketVideo` de `OpenCore` ; les librairies permettant la lecture et l'enregistrement audio et vidéo, ainsi que la gestion des fichiers image, y compris MPEG4, H.264, MP3, AAC, AMR, JPG et PNG.

Le schéma ci-dessous décrit tous les éléments de l'architecture de ces médiathèques :

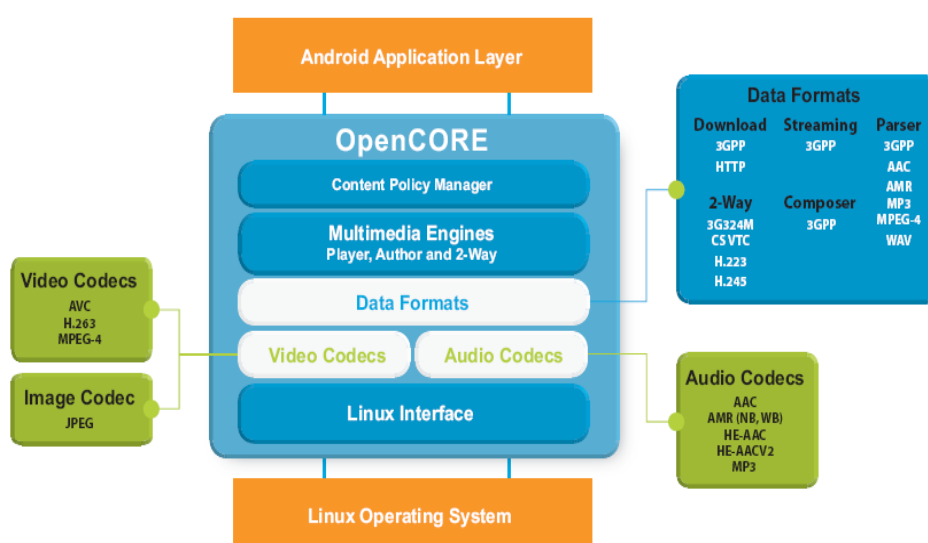


Figure I.4 : Architecture de ces médiathèques.

- Surface Manager : gère l'accès au sous-système d'affichage et de façon transparente.
- LibWebCore : le navigateur web présent dans Android est basé sur le moteur de rendu sous licence BSD WebKit.
- WebKit : est le moteur de rendu, qui fournit une (fondation) sur laquelle on peut développer un navigateur web. Il a été originellement dérivé par apple du moteur de rendu KHTML pour être utilisé par le navigateur web Safari et maintenant il est développé par KDE project, Apple, Nokia, Google et d'autres, WebKit est composé de deux bibliothèques : WebCore et javascriptCore qui sont disponibles sous licence GPL. WebKit supporte le CSS, javascript, DOM, AJAX. La dernière version a obtenu 100% au test Acid 3. La version de WebKit présente dans Android a été légèrement modifiée pour s'adapter aux appareils mobiles. Ainsi, le moteur de rendu basé sur WebKit présent dans Android supporte l'affichage sur une colonne.
- SGL : le moteur graphique 2D.
- Bibliothèque 3D : une implémentation basée sur OpenGL ES 1.0 API ; les bibliothèques utilisent l'accélération 3D matérielle (si disponible).
- FreeType : bitmap et vectoriel de rendu de police.
- SQLite : un moteur de base de données relationnelle puissante et légère, disponible pour toutes les applications.

I.7.4. Android Runtime

Android inclut un ensemble de bibliothèques de base offrant 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 la machine virtuelle Dalvik. Dalvik a été écrit pour que le dispositif puisse faire tourner plusieurs machines virtuelles de manière efficace. La machine virtuelle Dalvik exécute des fichiers dans l'exécutable Dalvik (.DEX), un format optimisé pour ne pas encombrer la mémoire. La machine virtuelle est la base de registres et fonctionne grâce aux classes compilées par un compilateur java et transformées dans le format DEX.

La machine virtuelle Dalvik s'appuie sur le noyau Linux pour les fonctionnalités de base telles que le filtrage et la gestion de la mémoire de bas niveau.

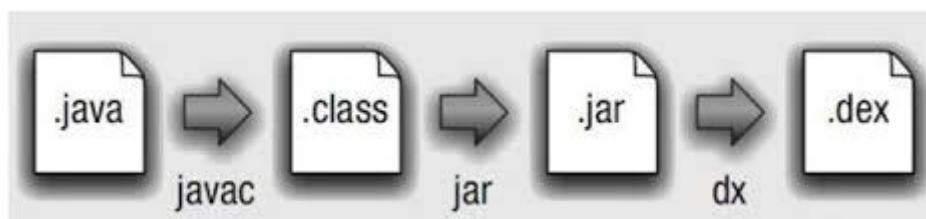


Figure I.5: Transformation d'un fichier .java à un fichier.dex

I.7.5. Linux Kernel

Android est basé sur un Kernel linux 2.6 mais ce n'est pas linux. Il ne possède pas de système de fenêtrage natif. La glibc n'étant pas supportée, Android utilise une libc customisée appelée Bionic libc.

Enfin, Android utilise un kernel avec différents patches pour la gestion de l'alimentation, le partage mémoire, etc. permettant une meilleure gestion de ces caractéristiques pour les appareils mobiles.

Android n'est pas linux mais il est basé sur un kernel linux. Pourquoi sur un kernel linux ?

Le kernel linux est un système de gestion mémoire et de processus reconnu pour sa stabilité et ses performances.

Le model de sécurité utilisé par linux, basé sur un système de permission, est connu pour être robuste et performant. Il n'a pas changé depuis les années 70.

- Le kernel linux fournit un système de driver permettant une abstraction avec le matériel. Il permet également le partage de bibliothèques entre différents processus, le chargement et le téléchargement de modules à chaud.
- Le kernel linux est entièrement open source et il y a une communauté de développeurs qui l'améliorent et rajoutent des drivers.

C'est pour les points cités ci-dessus que l'équipe en charge du noyau a décidé d'utiliser un kernel linux.

I.8. les composants principaux d'une application

Les applications Android sont composées de 4 types de composants : [5]

I.8.1. Activités

Une activité représente un écran de l'application. Une application peut avoir une ou plusieurs activités (par exemple pour une application de messagerie on pourrait avoir une Activité pour la liste des contacts est une autre pour l'éditeur de texte). Chaque Activité est implémentée sous la forme d'une classe qui hérite de la classe Activity.

I.8.2.Services

Les services n'ont pas d'interface graphique et tournent en tâche de fond. Il est possible de s'inscrire à un service et de communiquer avec celui-ci en utilisant l'API Android.

I.8.3. Broadcast receivers

Il se contente d'écouter et de réagir aux annonces broadcast (par exemple changement du fuseau horaire, appel entrant.....).

I.8.4. Content providers

Il permet de partager une partie des données d'une application avec d'autres applications.

I.9. Cycle de vie d'une activité

Une activité n'a pas de contrôle direct sur son propre état, il s'agit plutôt d'un cycle rythmé par les interactions avec le système et d'autres applications. Voici un schéma qui présente ce que l'on appelle le cycle de vie d'une activité, c'est-à-dire qu'il indique les étapes que va traverser notre activité pendant sa vie, de sa naissance à sa mort. [6]

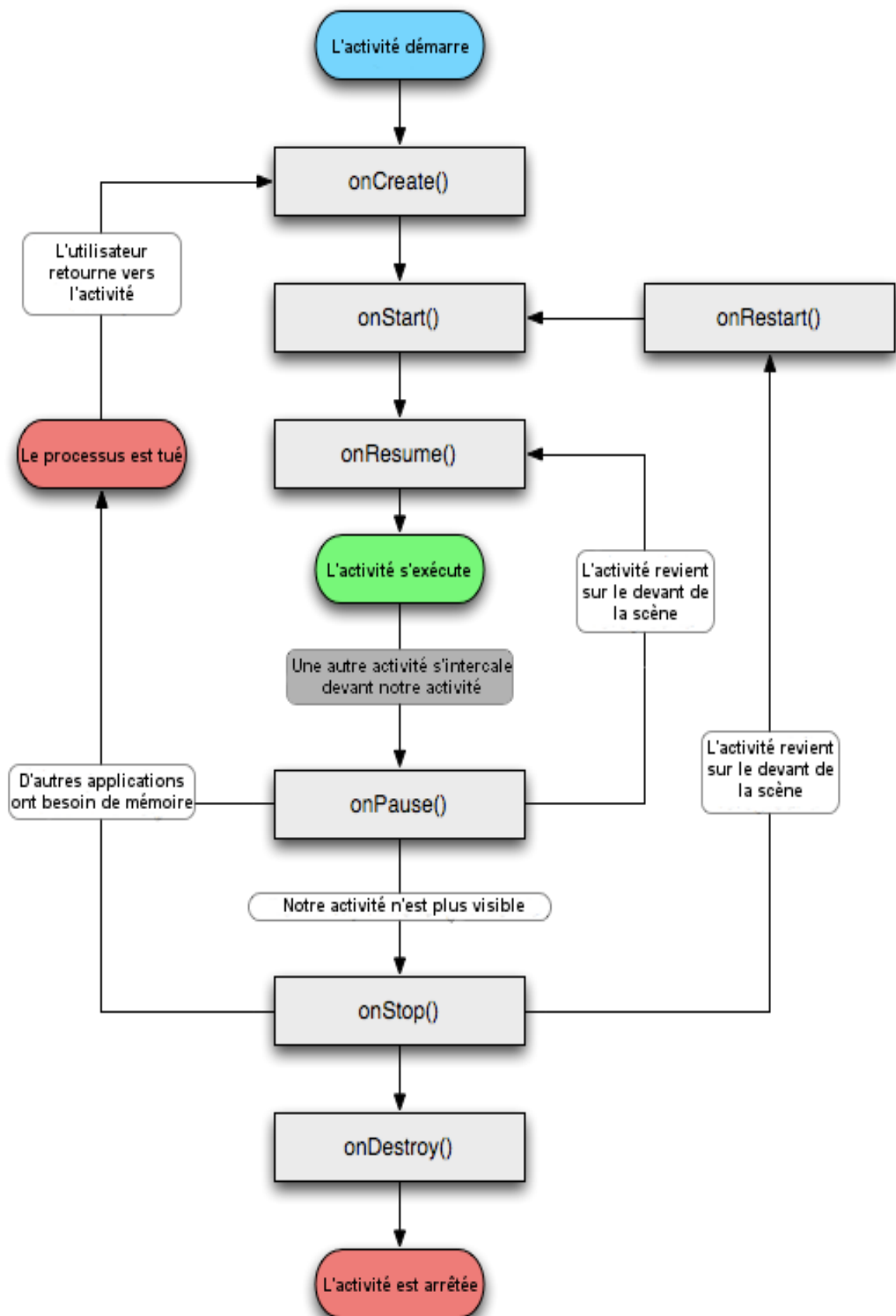


Figure I.6: Cycle de vie d'une activité

Une activité peut se trouver dans quatre états qui se différencient surtout par leur visibilité :

Active (Running) :

L'activité est visible en totalité.

Elle est sur le dessus de la pile, c'est elle qui a le focus. C'est ce que l'utilisateur consulte en ce moment même et il peut l'utiliser dans son intégralité et agir directement dessus.

Paused (en pause) :

L'activité est partiellement visible à l'écran.

C'est le cas lors de la réception d'un SMS et qu'une fenêtre semi-transparente se pose devant l'activité pour afficher le contenu du message et permettre d'y répondre par exemple.

Ce n'est pas sur cette activité qu'agit l'utilisateur. L'application n'a plus le focus, c'est l'application sous-jacente qui l'a. Pour que notre application récupère le focus, l'utilisateur devra se débarrasser de l'application qui l'obstrue, puis il pourra à nouveau interagir avec.

Stopped (Arrêtée) :

L'activité n'est pas visible à l'écran. Si une activité est complètement masquée par une autre activité, il est arrêté. Il conserve tous les états membres et de l'information, cependant il n'est plus visible pour l'utilisateur que sa fenêtre est cachée et il sera souvent tué par le système lorsque la mémoire est nécessaire ailleurs.

Dead (Mort) :

Cette activité a terminé ou il n'a jamais été démarré. Si une activité est en pause ou arrêtée, le système peut chuter l'activité de la mémoire, soit par lui demandant de se terminer, ou tout simplement tuer le processus. quand il est affiché de nouveau à l'utilisateur, il doit être redémarré et restauré à son état antérieur.

Les transitions d'états sont captées par les méthodes suivantes :

onCreate0 : l'activité est en création.

onStart0 : l'activité va devenir visible.

onResume0 : l'activité est maintenant visible.

onPause0 : l'activité va être mise en pause.

onStop0 : l'activité sera plus visible.

onDestroy0 : l'activité va être détruite.

I.10. Développement d'une application Android

Voici quelques étapes principales dans le processus de développement d'une application sur Android :

- Faire la conception de la base de données.

- Créer des classes pour représenter les données physiques (couche Mapping) et pour définir des actions comme : supprimer, ajouter, modifier des données.
- Dessiner des interfaces en les fichiers XML ou en codage :
 - ✓ Les vues (View) : Text, Edit, List, Image, Web, Map, etc.
 - ✓ Les arrangements (layout) : Frame, Linear, Relative, Table, Absolute.
- Choisir des arrangements (layout) : les layouts sont les ressources qui indiquent les interfaces des activités. On utilise les fichiers XML pour exprimer les interfaces. Mais il existe d'autre technique pour dessiner l'interface. Dans cette technique, on programme directement les composants graphiques en utilisant le codage.
- Organiser des ressources : les constantes globales (string.xml), les icônes, les images, etc.
- Créer et mettre à jour le fichier de configuration : AndroidManifest.xml. AndroidManifest.xml (configuration de l'application) est utilisé pour stocker les dispositions (settings) globales comme les permissions de l'application, les activités, les filtres de l'intention.

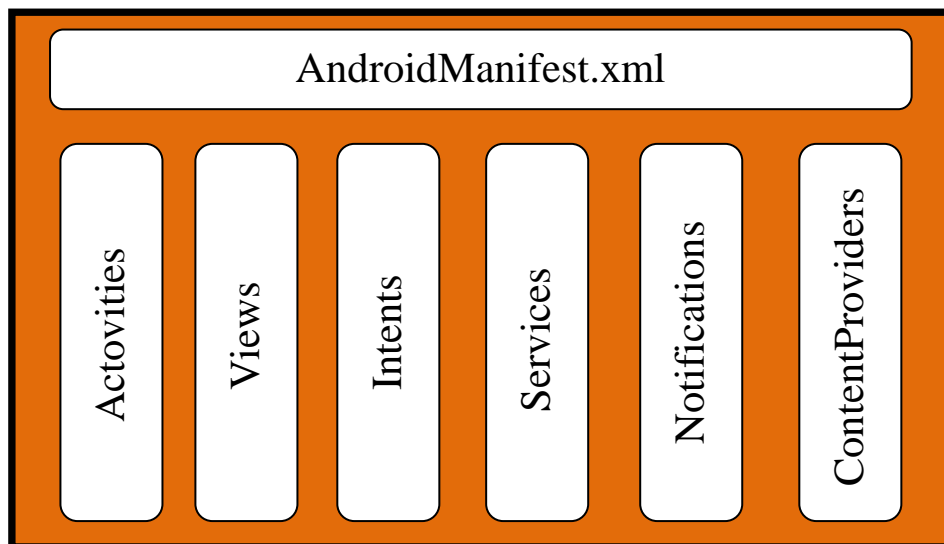


Figure I.7 : Une application sur Android.

- Créer des activités (créer les classes pour exécuter les fonctions avec la base de données (ajouter, supprimer, modifier, mettre à jour, etc.) :
 - ✓ Chaque activité peut correspondre avec un écran ou une fonction de cette application.
 - ✓ Il existe quelques activités qui s'occupent des méthodes pour communiquer avec la base de données (Couche Mapping).
 - ✓ Les Intents sont utilisées pour orienter des activités (CALL, ACTION_MAIN, ACTION_VIEW, etc).

I.11. Les outils de développement Android

Les différents outils de développement Android sont :

I.11.1.Présentation du SDK

Google a mis en place un grand nombre d'outils pour aider les développeurs Android.

❖ Le portail des développeurs

La première chose à visiter est le portail des développeurs Android, mis en place par Google. [7]

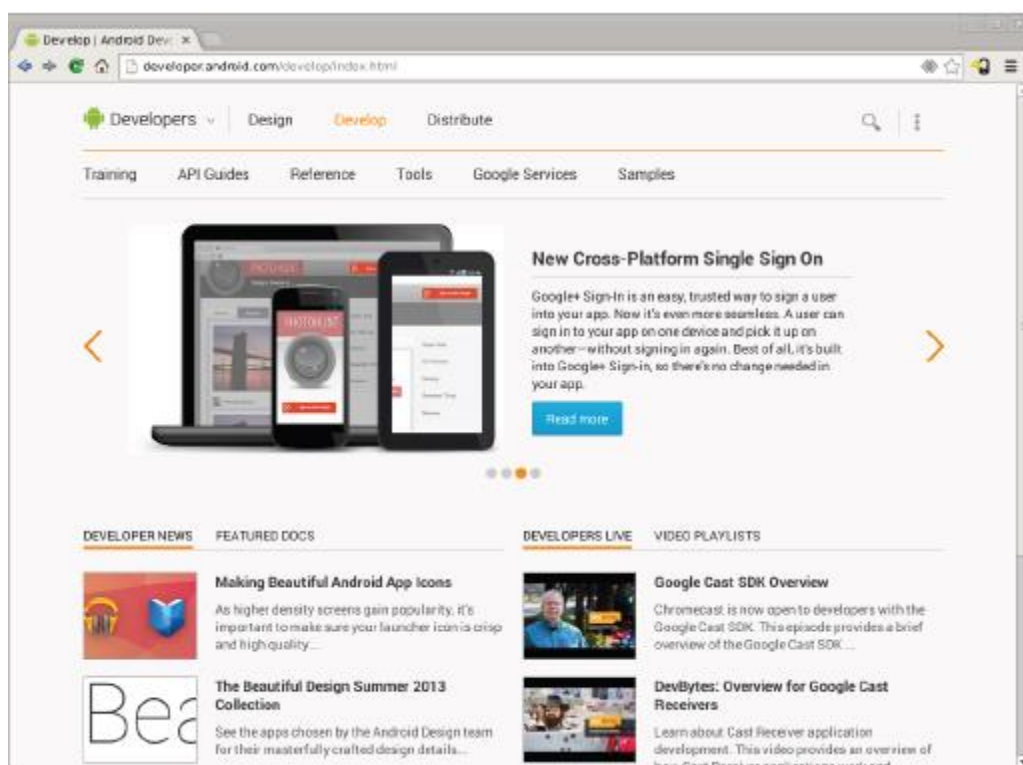


Figure I.8 : Portail des développeurs Android.

Très complet, ce site présente Android, explique comment installer et utiliser les différents outils (SDK, NDK etc.), propose un ensemble de tutoriels et articles concernant le développement d'applications Android, expose la référence de l'API Android ainsi que les actualités liées à Android.

Le tout est très bien fait et permet de rapidement être confortable vis-à-vis du développement sur Android.

I.11.2. Le SDK Android :

L'outil le plus important est le SDK Android. Facile à installer, il permet de télécharger tous les outils indispensables aux développements d'applications. Un petit logiciel permet d'abord de télécharger les différentes versions du SDK (une version du SDK par version d'Android : 1.4, 1.5, 1.6, 2.0 ect.). Il permet également de télécharger les différentes versions d'APIs (APIs pour intégrer des fonctionnalités liées aux services Google tels que Maps etc.) ou de la documentation JavaDoc. Son fonctionnement est similaire aux gestionnaires de paquets de Linux. [8]

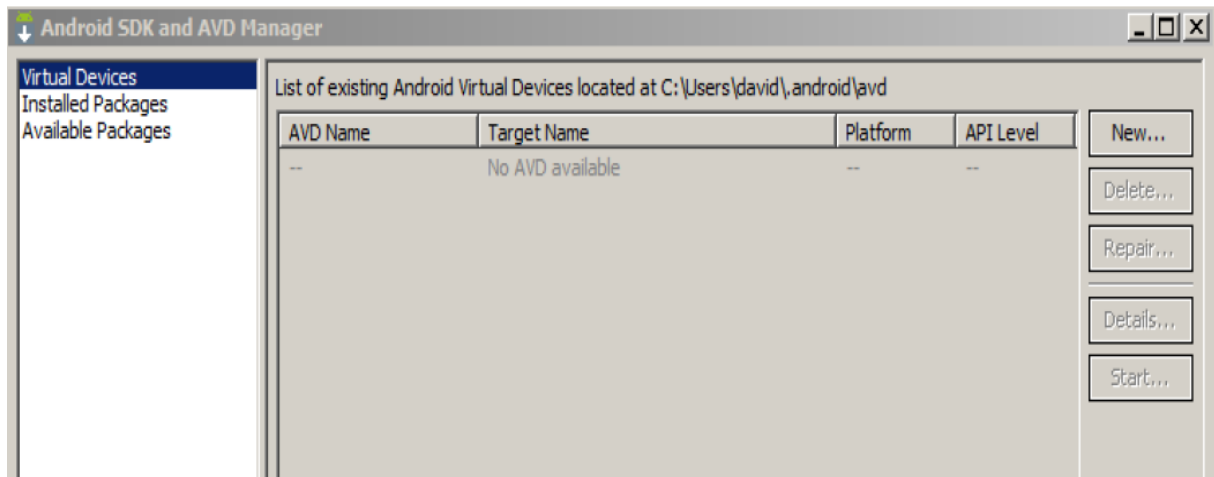


Figure I.9 : Interface d'installation du SDK Android.

I.11.3. Le débogueur « ADB »

Le SDK Android contient un débogueur appelé « Android debug bridge » ou aussi « adb », qui permet de connecter un appareil Android virtuel ou réel, dans le but de gérer le périphérique ou de déboguer votre application.

I.11.4. Les IDE « Android Developer Tools » et « Android Studio »

Google propose deux environnements de développement intégrés (IDE) pour développer de nouvelles applications.

I.11.5. Eclipse

Les outils de développement Android sont basés sur l'IDE Eclipse. ADT est un ensemble de composants (plug-ins), qui étendent l'IDE Eclipse avec des capacités de développement Android.

I.11.6. Android Studio

Google propose également cet IDE appelé pour la création d'applications Android quel que soit le terminal sous-jacent (smartphone, tablette, montre, TV...) et qui est basé sur l'IDE IntelliJ.

I.11.7. Emulateur

Nous l'avons évoqué plus haut, le SDK propose un émulateur Android. Il permet de lancer sur la machine de du développeur un terminal virtuel représentant à l'écran un téléphone embarquant Android. C'est bien évidemment un outil indispensable pour le développement mobile. A chaque version d'Android est associée une version de l'émulateur, permettant au développeur de voir exactement à quoi ressemblera son application sur un matériel réel.

Rappelons cependant que l'émulateur ne propose pas toutes les fonctionnalités d'un vrai téléphone. Il ne permet par exemple pas d'émuler la gestion du Bluetooth.



Figure I.10 : Interface du simulateur Android

Conclusion

Dans ce chapitre, nous avons fait une étude de l'état de l'art d'Android tout en présentant un bref historique, les fonctionnalités que nous pouvons trouver sur ce système d'exploitation et l'architecture d'Android, à savoir les principaux composants du système.

Enfin nous avons présenté l'environnement logiciel et matériel utilisé pour la programmation Android.

Nous allons réaliser une application de localisation, donc notre prochain chapitre se portera sur la géolocalisation.

Introduction

La géolocalisation est une technologie avancée qui permet de collecter des informations permettant de localiser un objet ou une personne sur une carte, à l'aide de coordonnées géographiques.

Ce concept a vu le jour en Amérique, en 1993. Créée pour les besoins de l'armée américaine, la géolocalisation a tout d'abord servi à localiser les objets et les personnes. L'utilisation de la géolocalisation s'est modernisée depuis quelques années.

Les possibilités en termes de géolocalisation ont connu un développement inouï au cours des dix dernières années révolutionnant ainsi de nombreux domaines. L'alliance des nouvelles technologies telles que les Smartphones, les tablettes numériques et les GPS (Global Positioning System) a su optimiser l'usage de la géolocalisation. C'est à présent un outil majeur de communication personnelle et professionnelle qui est très en vogue actuellement.

La géolocalisation fait usage des plusieurs technologies différentes, touchant aux domaines d'activités complémentaires. Dans la partie qui suit, on va donner une description de cette nouvelle technologies qui fait parler d'elle de plus en plus et de regarder chacune des techniques de géolocalisation les plus importantes .

II.1. Techniques de géolocalisation

Pour géolocaliser l'utilisateur grâce à son Smartphone, il existe plusieurs techniques que l'on recoupe afin d'obtenir un résultat à la fois rapide et précis.

II.1.1. Géolocalisation par satellite

Le principe de la localisation par satellite est de déterminer la position géographique d'un terminal par la connaissance des distances entre 3 satellites de la constellation GPS et l'observateur. Ces distances sont calculées à partir de la mesure des temps de parcours des signaux émis par les satellites (horloge de très grande précision nécessaire !). La précision de cette technique est estimée de 15 à 100 mètres pour le réseau GPS.

Cette méthode permet d'obtenir la longitude, la latitude, mais aussi l'altitude de l'observateur qui peut être représentée physiquement sur une carte, et est utilisé aussi pour la navigation GPS en voiture

Pour qu'un terminal soit capable de se géolocaliser grâce au réseau GPS, celui-ci doit être équipé d'une puce électronique GPS, puce que l'on retrouve dans tous les Smartphones actuels.[9]



Figure II.1 : géolocalisation par satellite

II.1.2. Géolocalisation via la téléphonie mobile (GSM)

Le Global System Mobiles (GSM) est le réseau le plus utilisé. Ce moyen de communication sans fil fonctionne par transmission d'ondes entre une base relais et le téléphone portable de l'utilisateur, couvrant une zone de plusieurs kilomètres.

Avec cette technique de géolocalisation, on trouve la position d'un GSM en se basant sur certaines informations relatives aux antennes GSM auxquels ce dispositif est connecté. La précision va dépendre, ici, de l'environnement, du milieu où se situe l'appareil. En milieu urbain, la densité d'antennes est supérieure par rapport au milieu rural.

Il existe plusieurs techniques de géolocalisation par GSM mais la méthode du Cell ID (identification de la cellule radio), reste la plus utilisée. Cette méthode consiste à récupérer l'identifiant de l'antenne GSM à laquelle le terminal est connecté. Ensuite, grâce à des bases de données, l'identifiant sera relié à la position géographique connue de l'antenne. Sur base de cela, on va pouvoir faire une estimation de l'emplacement du terminal.[10]

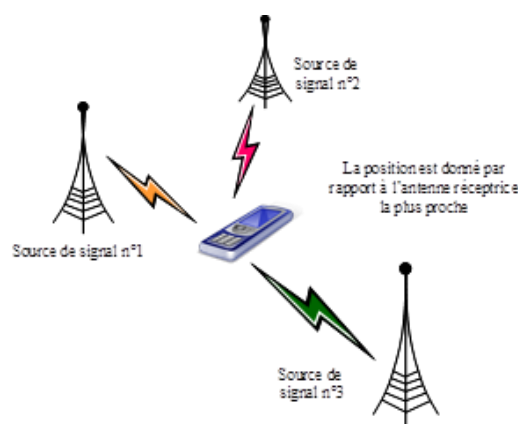


Figure II.2 : géolocalisation par GSM.

II.1.3. Géolocalisation par Wifi

Tout comme la géolocalisation par GSM, un terminal WiFi peut être localisé grâce à la méthode Cell ID sur un réseau GSM, . Dans ce cas-ci, c'est l'identifiant des bornes WiFi qui est détecté. Des bases de données reprenant ces identifiants et leur position géographique sont communiquées par des entreprises privées et d'autres communautés.



Figure II.3 : géolocalisation par wifi

II.1.4. Géolocalisation indoor

Certains composants GPS peuvent recevoir un signal suffisamment élevé pour déterminer une position à l'intérieur d'un bâtiment. Cependant, le résultat n'est généralement pas assez précis pour être utilisable. De plus le GPS ne permet pas de se localiser dans un bâtiment à plusieurs étages.

La géolocalisation *indoor* est devenue le Saint Graal du marketing géolocalisé car elle permet d'assister le consommateur de son domicile au centre commercial. Elle le guide à l'intérieur du lieu, déclenche l'envoi de messages et lui permet de régler ses achats à partir de son mobile. Un des objectifs principaux de cette nouvelle méthode de marketing est d'éviter la fuite des consommateurs vers des sites e-commerce.

L'intérêt de la géolocalisation *indoor* ne se limite pas au marketing. De nombreux aéroports, gares, centres de congrès et d'exposition s'équipent de ces technologies pour aider les utilisateurs en leur fournissant des informations utiles.

Le terme générique de géolocalisation indoor couvre 3 principaux niveaux de services :

- **La géolocalisation indoor**, véritable GPS d'intérieur, aide les utilisateurs à trouver leur chemin, leur permet d'optimiser leur visite, de localiser leurs amis et collègues;

ainsi que de fournir des analyses comportementales aux commerçants sur le parcours des visiteurs.

- **La micro-localisation** permet aux voyageurs, visiteurs ou consommateurs d'interagir avec un élément spécifique : un produit sur une étagère en magasin, une œuvre d'art exposée dans un musée... La présence du consommateur est identifiée seulement quand il est à proximité d'une balise. Sa géolocalisation est perdue lorsqu'il s'en éloigne.
- **Le Geofencing** envoie des informations spécifiques lorsque un utilisateur entre ou sort d'une zone prédéfinie. Le géomarketing ou les programmes de fidélisation comptent parmi les cas d'utilisation les plus courants de cette méthode de localisation.[11]



Figure II.4: géolocalisation indoor

II.1.5. Géolocalisation par adresse IP (sur Internet)

Grâce à cette technique, on peut déterminer la position géographique d'un ordinateur ou de n'importe quel autre terminal connecté à Internet. On se base ici sur l'adresse IP (numéro d'identification attribué à un appareil connecté à un réseau informatique). Ces adresses IP sont gérées par l'Internet Assigned Numbers Authority (IANA), une organisation qui se charge de distribuer les adresses IP disponibles aux pays demandeurs. Les attributions sont donc très bien documentées et il est possible de déterminer facilement dans quel pays se trouve un terminal.

II.2. Technologie de positionnement

Il y a plusieurs méthodes pour localiser un objet. Ces méthodes sont basées sur la transmission de certains signaux et leur réception à leur bout.

La technologie de positionnement utilisée est choisie selon les besoins (par exemple le temps de réponse) des applications. Parmi les différentes technologies de positionnement existantes, nous avons :

II.2.1. Global Positionning System (GPS)

Le GPS (Global Positionning System en anglais et il s'appelle système de positionnement globale en français) est le système américain de positionnement par satellite capable de donner n'importe où sur le globe, de jour comme de nuit, avec précision, en temps réel:

- la position (x, y);
- L'altitude (z);
- L'heure;
- La vitesse;
- L'orientation.

Le système GPS actuel est un système de navigation basé sur 31 satellites disposés en réseau de 6 orbites différentes. [12]

II.2.1.1. Historique du système GPS

- En 1968, Ministère de la Défense des USA imagine un système de localisation terrestre composé d'une constellation de satellites en orbites autour de la terre qui pourrait leur fournir la position d'un point sur la planète en temps réel et à tout moment.
- En 1973, le système NAVSTAR (Navigation System based on Time and Ranging) a été conçu pour un usage strictement militaire.
- Le 1er satellite a été lancé en 1978
- Depuis 1995, le système est opérationnel et comporte 24 satellites en orbite autour de la terre.

À l'origine, le GPS a été destiné pour répondre aux besoins des militaires, mais la *Presidential Decision Directive* du 28 mars 1996 s'achevait par la déclaration suivante :
« *Nous continuerons de fournir en permanence le service de positionnement standard GPS à des fins pacifiques d'utilisation civile, commerciale et scientifique à l'échelle mondiale et sans imposer de redevances directes aux utilisateurs* ». [Traduction]

Le GPS offre toutefois deux niveaux de service : un service de positionnement standard (SPS) accessible à tout utilisateur, et un système de positionnement précis (PPS) dont l'accès est réservé principalement aux militaires américains. Le SPS offre une précision de l'ordre de 20 mètres dans le plan horizontal, 95% du temps.

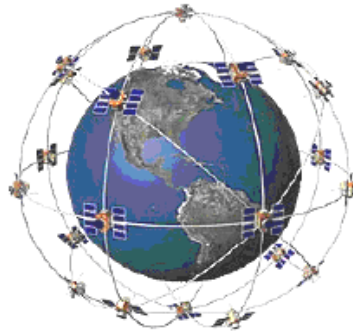


Figure II.5 : Constellation GPS.

II.2.1.2. Composant du système GPS

La configuration complète du GPS comporte trois segments distincts :

- **le segment spatial** : les satellites en orbite autour de la Terre.
- **le segment de contrôle** : des stations positionnées à proximité de l'équateur terrestre pour contrôler les satellites.
- **le segment utilisateur** : quiconque capte et utilise le signal GPS. [13]

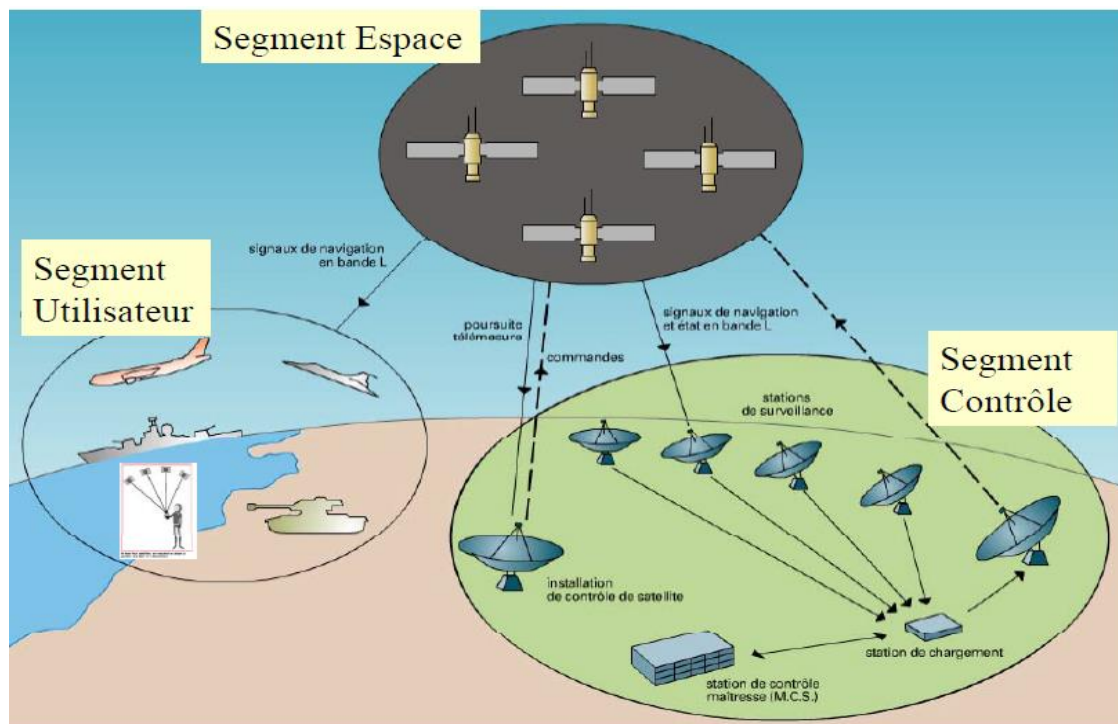


Figure II.6 : Composantes du système GPS.

II.2.1.2.1. Le segment spatial

Ce segment se compose de 24 satellites opérationnels répartis sur six orbites circulaires à 10 900 milles marins au-dessus de la terre. De ces 24 satellites, 21 sont des satellites de navigation (SV) et 3 satellites de secours qui prennent le relais en cas de panne

d'un satellite. Les orbites sont inclinées à 55° par rapport au plan équatorial et leur période est d'environ 12 heures. Cette configuration permet à un récepteur à la surface ou au-dessus de la terre de recevoir les signaux de cinq à huit satellites, 24 heures par jour. Les satellites transmettent continuellement leurs données de position et l'heure, qui sont reçues et traitées par les récepteurs GPS afin de déterminer la position tridimensionnelle de l'utilisateur (latitude, longitude, altitude), sa vitesse et l'heure.



Figure II.7 : Segment spatial

II.2.1.2.2. Le segment contrôle

Constitué de cinq stations terrestres :



Figure II.8 : Localisation du segment terrestre.

Ces stations sont en quelque sorte les yeux et les oreilles du GPS, elles reçoivent les informations fournies par les satellites. On assure ainsi le contrôle et la mise à jour des paramètres du système (données orbitales, performance des horloges embarquées...) par un suivi permanent du segment spatial. La disposition des stations tout autour de la terre permet de suivre chaque satellite pendant 92% du temps. Les informations sont centralisées au «Master Control » du Colorado Springs. Des données de corrections peuvent être émises en direction des satellites par les stations Ascension, Diego Garcia et Kwajalein.

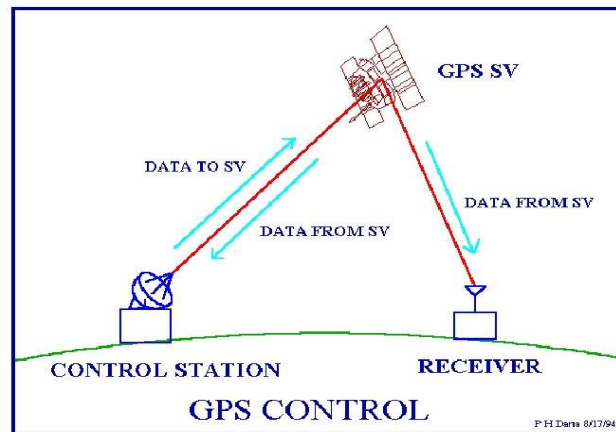


Figure II.9 : Segment de contrôle.

II.2.1.2.3. Le segment utilisateur

Le secteur utilisateur inclut toute personne se servant d'un récepteur GPS pour capter le signal GPS et déterminer sa position et/ou connaître l'heure avec précision. Les applications les plus courantes au sein du secteur utilisateur sont la navigation terrestre pour les randonneurs, la localisation de véhicule, la topographie, la navigation maritime, la navigation aérienne ou encore le guidage de machines.



Figure II.10 : Segment utilisateur

II.2.1.3. A quoi sert le GPS?

« Quelle est l'heure, quelle est la position, quelle est la vitesse ? »

Le système GPS est utilisé pour déterminer :

- Le temps
- La position
- La distance
- La vitesse

- L'orientation
- La cartographie

II.2.1.4. Fonctionnement du GPS

Il existe plusieurs méthodes différentes pour obtenir une position en utilisant le GPS. La méthode employée dépend de la précision requise par l'utilisateur et du type de capteur GPS disponible.

Dans le GPS, les transmissions proviennent du satellite et contiennent des données qui permettent au récepteur de calculer sa distance par rapport au satellite. Cette mesure de distance situe alors le récepteur à la surface d'une sphère dont le centre est le satellite et qui a un rayon égal à la distance du satellite. Si les émissions de plusieurs satellites sont reçues et traitées, le récepteur peut alors être placé à l'intersection de trois sphères, ce qui donne un positionnement tridimensionnel en latitude, longitude et altitude.

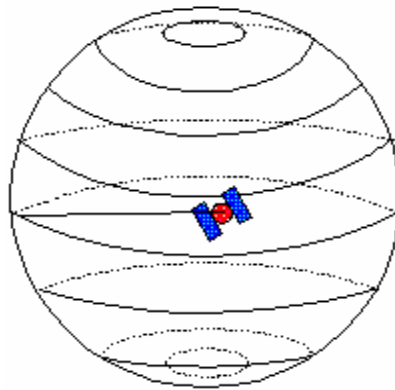


Figure II.11 : Un satellite

En recoupant les informations de **2 satellites**, le lieu géométrique du récepteur devient un cercle

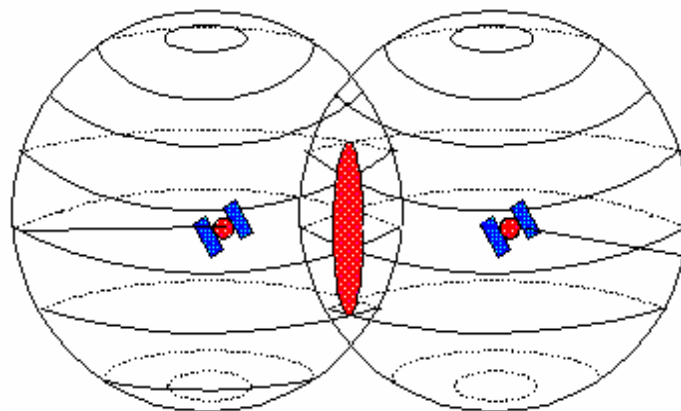


Figure II.12 : Deux satellites

Avec 3 satellites, l'intersection se réduit à un (ou 2) points

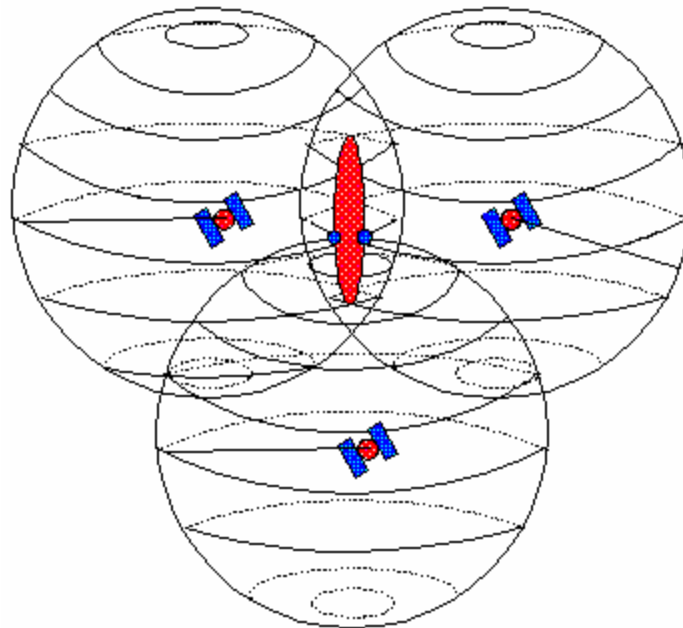


Figure II.13. Trois satellites

L'intervalle entre l'émission et la réception de l'impulsion, est mesurée à sa réception pour établir la distance. Le temps absolu de transmission n'est pas requis, seuls le sont les intervalles entre les impulsions émises et reçues. Dans le système de positionnement global, l'émission est générée par le satellite et arrive au récepteur avec un certain retard correspondant à la distance entre le récepteur et le satellite. À la différence des objets proéminents cartographiés, les satellites sont des cibles en déplacement. Pour que le récepteur puisse déterminer sa position avec précision, on doit connaître le moment exact de la transmission du signal et la position du satellite sur son orbite à ce moment.

II.2.1.5. Sources d'erreurs

Il l'est presque, mais pas tout à fait. La précision du système lorsque son signal est émis sur la fréquence L1 atteint 20 mètres. Il existe d'autres sources d'erreurs qui peuvent introduire des inexactitudes dans la position finale, allant de un mètre à des centaines de mètres. Ces sources d'erreurs sont les suivantes :

- ❖ Les erreurs non corrigées de l'horloge du satellite
- ❖ Les erreurs de données des paramètres orbitaux
- ❖ Les erreurs dues à l'atmosphère (ionosphère, troposphère).
- ❖ Les erreurs de multitrajets
- ❖ Les erreurs géométriques

II.2.1.5.1. Les erreurs issues du satellite

Quoique les horloges des satellites soient extrêmement précises et stables, et malgré la vérification régulière de leur précision, de légères erreurs temporelles restent toujours

possible. Ces erreurs, couplées à de faibles erreurs dans le message de position diffusé du satellite, peuvent entraîner des erreurs de distance d'environ trois mètres.

II.2.1.5.2. Les erreurs dues à l'atmosphère

Lors de leur parcours du satellite au récepteur, les ondes GPS traversent deux couches atmosphériques:

- l'ionosphère,
- la troposphère.

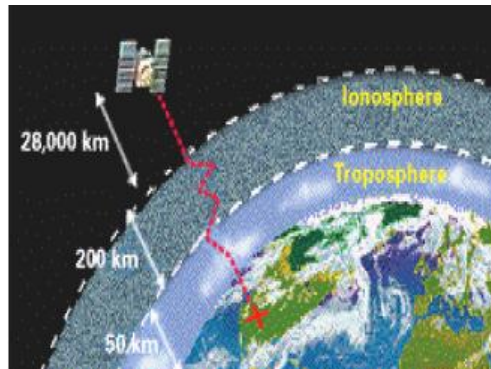


Figure II.14 : couche atmosphérique

La mesure de distance d'un satellite repose sur l'hypothèse fondamentale que la vitesse d'émission du signal du satellite est constante. Ce fait n'est vrai que dans l'espace libre car cette vitesse change avec le déplacement du signal à travers des particules électriquement chargées de l'ionosphère terrestre, et ensuite à travers la vapeur d'eau de la troposphère. Ce changement peut entraîner des erreurs dans la distance mesurée par rapport au satellite de l'ordre de 10 à 12 mètres.

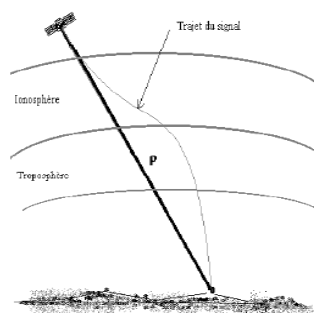


Figure II.15 : trajet du signal GPS dans l'atmosphère

II.2.1.5.3. Erreurs de multitrajets

La vraie distance du satellite est la distance oblique, c'est-à-dire la distance en ligne directe. S'il existe des obstacles à proximité de la source d'émission du signal, à l'intérieur ou

à l'extérieur du bâtiment, le signal peut atteindre l'antenne qu'après une ou plusieurs réflexions.

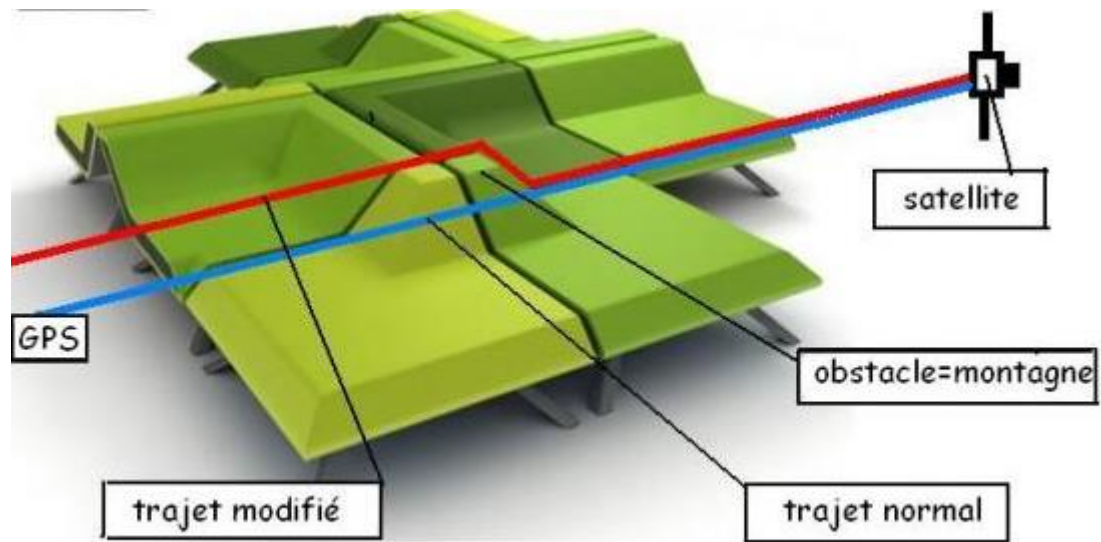
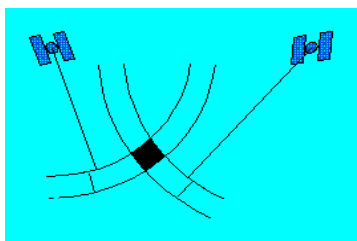


Figure II.16 : Erreurs multitrajet

II.2.1.5.4. Erreurs géométriques

Il existe des erreurs qui se produisent dans le GPS et d'autres systèmes de positionnement lorsque l'angle d'incidence entre les lignes de position est très faible. Une erreur minime dans les données mesurées peut créer une importante région d'incertitude à un faible angle d'incidence. À l'inverse, la même erreur dans les lignes de position qui se coupent à 90° produit une faible région d'incertitude. Dans le GPS, les satellites se trouvant presque sur le même relèvement placeront le récepteur sur les surfaces de sphère qui se coupent à angle étroit. Les erreurs de distance accumulées conduiront à un déplacement important de la position.

Satellites largement espacés. Bon angle d'incidence.



Satellites presque alignés. Mauvais angle d'incidence

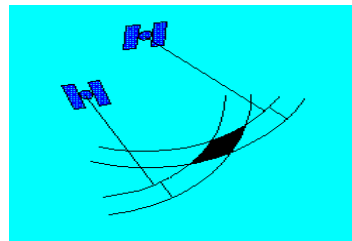


Figure II.17 : Erreurs géométriques

II.2.1.6. Le système des coordonnées GPS

Si vue de l'espace, la Terre peut apparaître comme une sphère uniforme, sa surface est en réalité loin d'être régulière. Le GPS devant attribuer des coordonnées à tout point de la surface terrestre, il utilise un système de coordonnées géographiques basé sur un ellipsoïde (parfois assimilé à un sphéroïde), lequel est une sphère aplatie. Le choix se porte sur un ellipsoïde approchant la forme de la Terre au plus près. Cet ellipsoïde est dépourvu de toute réalité physique, c'est une surface définie mathématiquement.



Figure II.18 : Un ellipsoïde

En fait, il existe bon nombre d'ellipsoïdes différents ou de définitions mathématiques de la surface terrestre, comme nous le verrons plus loin. L'ellipsoïde utilisé par le GPS est appelé le WGS84, World Geodetic System 1984 (système géodésique mondial de 1984). Un point de la surface terrestre (notez bien qu'il ne s'agit pas de la surface de l'ellipsoïde) peut être défini en utilisant la latitude, la longitude et l'altitude ellipsoïdique. La position d'un point peut également être définie dans un système de coordonnées cartésiennes, par des distances selon les axes X, Y et Z par rapport au centre ou à l'origine du sphéroïde. Cette méthode est privilégiée par le GPS pour définir la position d'un point dans l'espace.[14]

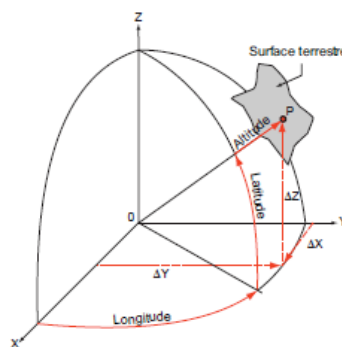


Figure II.19: Coordonnées cartésiennes et géographique du point P

II.2.1.7. Systèmes de coordonnées locaux

Comme pour les coordonnées GPS, les coordonnées locales ou les coordonnées utilisées dans le système cartographique d'un pays donné sont basées sur un ellipsoïde local, destiné à correspondre au mieux au géoïde dans la zone concernée. En règle générale, ces coordonnées sont projetées sur un plan de manière à obtenir des coordonnées planes.

Les ellipsoïdes utilisés pour la plupart des systèmes de coordonnées locaux dans le monde ont été définis il y a fort longtemps, bien avant l'avènement des techniques spatiales. Ces ellipsoïdes sont généralement bien adaptés à la zone qu'ils doivent couvrir mais ne peuvent pas être employés dans d'autres régions du globe. Chaque pays a donc défini un système de référence basé sur un ellipsoïde local pour répondre à ses propres besoins.

Lorsque vous utilisez le GPS, les coordonnées des positions calculées sont basées sur l'ellipsoïde WGS84. D'ordinaire, les coordonnées existantes sont exprimées dans un système de coordonnées local, les coordonnées GPS doivent donc être transformées dans ce système local.

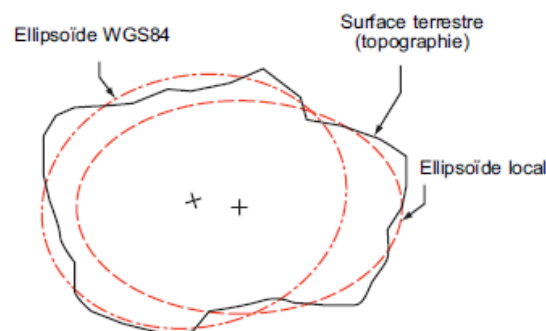


Figure II.20 : Relation entre ellipsoïdes et surface terrestre

II.2.1.8. Les applications du GPS

➤ Les premières applications sont militaires

- Des essais de guidage de bombes (les bombardiers utilisant le système NAVSTAR en déterminant la position de leurs cibles, pouvaient potentiellement détruire de 400 à 600% d'unités ennemies de plus qu'en utilisant les systèmes de localisation habituels).
- Dans l'utilisation des missiles de croisière.

➤ Les applications civiles

- La circulation automobile;
- La circulation aérienne
- La circulation navale
- La cartographie
- Le cadastre et l'urbanisme
- La gestion des ressources naturelles
- Etc.

II.2.2. L'identification par cellules, Cell-ID

Cette méthode simple d'identification va s'effectuer à partir de l'adresse de la BTS (Base Transceiver Station) à laquelle le mobile est connecté. La BTS repère le mobile pour pouvoir prendre la communication, il y a identification de la carte SIM avant de démarrer la communication. Chaque cellule BTS sait donc quels portables sont dans son champ de

fonctionnement, ces données sont automatiquement transmises à la BSC (Base Station Controller) puisque que c'est le BSC qui décide quelle BTS est affecté à chaque mobile. Ces données sont ensuite transmises à une base de données, qui sait donc quelles cartes SIM sont dans le champ de chaque cellule. Or cette base de données sait aussi l'adresse exacte de chaque antenne. L'on peut donc connaître la localisation approximative d'une carte SIM.

Cette localisation dépend donc fortement de la densité d'antenne, si un récepteur est dans plusieurs champs différents, on commence à pouvoir de localiser de manière précise. En ville l'on peut repérer un portable à 250 mètres près, en zone rurale on peut arriver à une précision de seulement 10 km. [15]

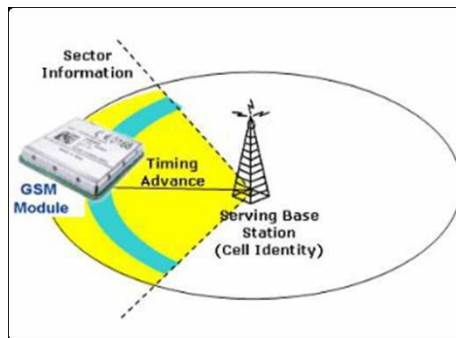


Figure II.21 : Identification par cellules

II.2.3. Identification par le temps (E-OTD)

La méthode E-OTD (Enhanced Observed Time Difference) nécessite l'envoi d'un signal par le portable. Il faut donc que le mobile soit équipé pour pouvoir être localisé. Le BTS envoie des signaux régulièrement, dès que le mobile reçoit un de ces signaux, il réémet. Le BTS peut donc calculer la distance en mesurant le temps d'aller-retour.

Pour avoir un temps plus précis, on utilise plusieurs cellules BTS pour repérer un mobile (même principe que le 4e satellite pour le GPS). L'idéal serait d'avoir trois cellules dans la portée du mobile pour avoir une localisation optimale.

Toutes ces différentes géolocalisations par GSM sont toute fois moins précises que le GPS, surtout en campagne, elles dépendent essentiellement de la densité d'antenne autour du mobile.

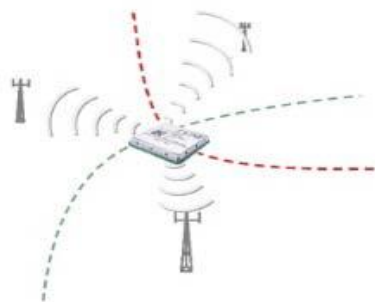


Figure II.22 : Identification par le temps

II.2.4. Identification par triangulation

Cette identification se base non plus sur les informations des BTS mais de celles de la BSC. Il s'agit d'établir un rapport de distance d'un mobile entre trois stations BTS. On peut établir une comparaison entre la puissance du signal émis dans chaque BTS, ce qui permet d'évaluer la distance qui le sépare des trois BTS. Cela permet de préciser la position par rapport à une simple Cell-Id.

II.3. Autre GNSS

Il existe d'autre système de navigation globale par satellite tel que :

II.3.1. Glonass

Le système Glonass (GLObal NAVigation Satellite System en anglais) est le système de positionnement global par satellite développé par l'URSS (Union des républiques socialistes soviétiques) durant la guerre froide et maintenu par la Russie depuis la chute du bloc soviétique. Tout comme GPS, ce système a été développé dans un cadre militaire.

II.3.2. Galileo

Galileo est le futur système européen de navigation par satellite, lancé par l'UE (Union européenne) et l'ESA (European Space Agency, l'agence spatiale européenne). L'objectif pour l'Europe est donc de disposer de son propre système global de navigation par satellite et donc d'être indépendant vis à vis des systèmes GPS et GLONASS .

Galileo fournira des services de localisation précis, sécurisés et certifiés à l'échelle du globe. Le système sera compatible avec GPS et GLONASS et offrira (sous abonnement) une précision de localisation en temps réel de l'ordre du mètre, ce qu'aucun autre système public ne permet d'obtenir actuellement.

Notons enfin qu'une différence majeure de Galileo vis à vis de GPS et Glonass est son placement sous le contrôle d'autorités civiles et non militaires.

II.3.3. Compass

Compass, aussi appelé Beidou-2, est le GNSS actuellement développé par la Chine. C'est l'évolution du système régional Beidou-1 mis en place depuis le début des années 2000.

Deux types de services seront disponibles : l'un civil, l'autre militaire (de précision et de fiabilité plus intéressantes).

Signalons que ce système est actuellement peu documenté et peu utilisé.

Conclusion

Dans ce chapitre, nous avons fait une étude sur la géolocalisation plus précisément sur GPS, tout en présentant un bref historique ainsi les techniques utilisées tels que la géolocalisation par satellite, par wifi, et par GSM, nous avons présenté aussi quelque technologie de positionnement et d'autre système de navigation telque Glonass, Galileo et Compass .

Nous allons essayer dans la prochaine étape de notre travail, à savoir l'analyse et conception, de mettre en place un système d'information s'adaptant au mieux à l'atteinte des objectifs.

Introduction

Dans la vie quotidienne l'assurance de l'assistance des patients est pénible au lieu de confier un membre de famille, il est plus pratique de concevoir et de réaliser une application mobile qui permet de surveiller et d'offrir une meilleure assistance des patients à distance.

Pour cela nous proposons une application Android permettant de réaliser cette tâche, et elle sera constituée de deux interfaces (patient, surveillant).

Celle du patient à une alarme qui se déclenche à un intervalle de temps régulier (selon la configuration du patient), à chaque déclenchement, si le patient ne la désactive pas alors l'application localise le plus proche des surveillants, puis déclenche une alarme chez ce dernier.

Celle du surveillant permet d'envoyer les coordonnées GPS à l'interface patient et d'arrêter l'alarme déclenchée par le patient.

III.1. Objectif de l'application:

Notre but est de créer une application Android, qui permet de surveiller la situation d'un patient à distance.

Pour mieux comprendre notre application, voici un schéma global qui montre son fonctionnement :

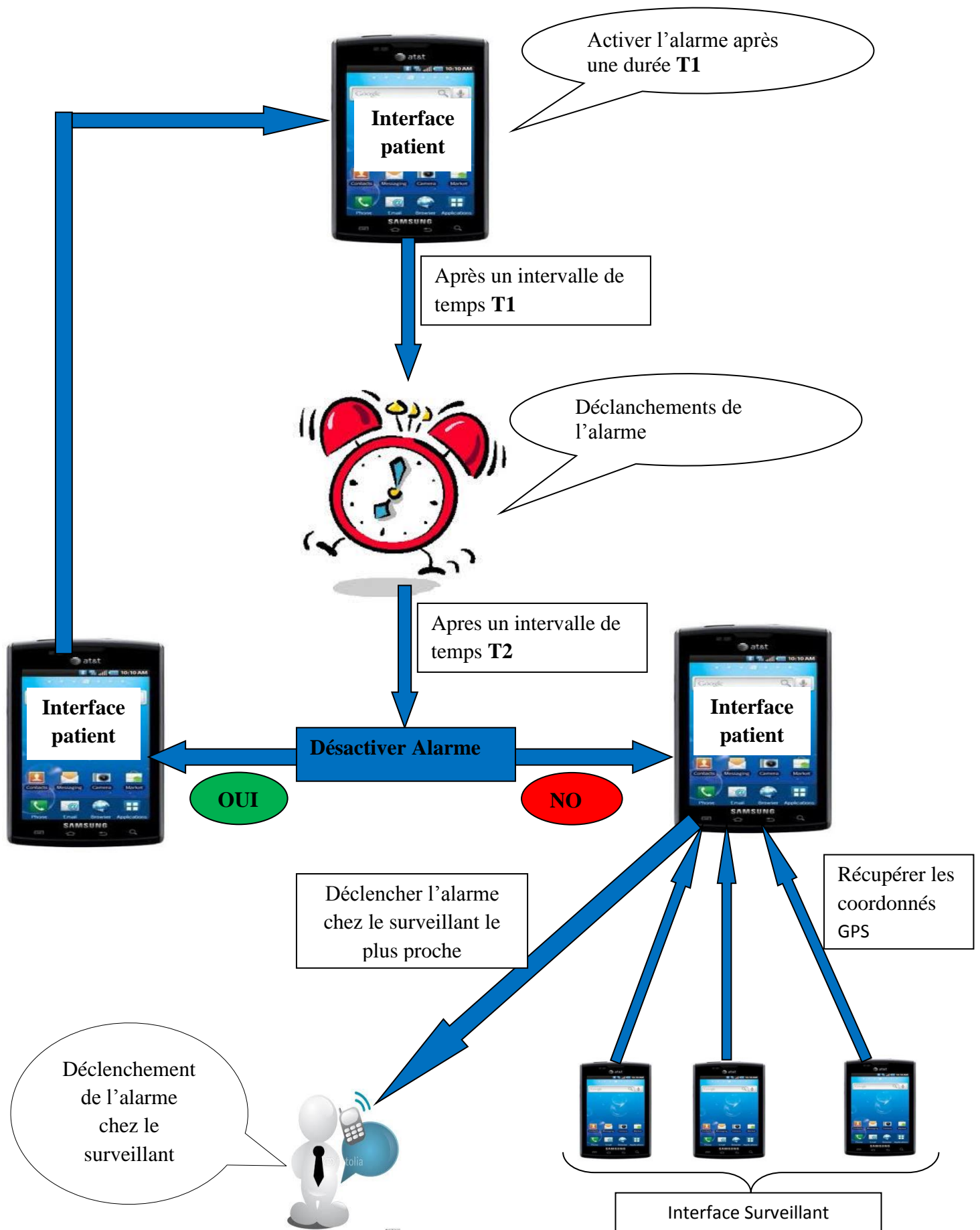


Figure III.1 : Principe du fonctionnement de notre l'application

Afin d'atteindre cet objectif nous allons suivre une méthode normalisée UML qui nous a aidé à bien visualiser l'organisation de notre application sur lequel se base le prochain point.

III.2. Modélisation avec L'UML

III.2.1. Définition de la modélisation :

La modélisation est la représentation abstraite d'un système, elle est destinée à faciliter l'étude et de mieux comprendre le système à développer.

L'objectif principal de la modélisation est de maîtriser la complexité du système : chaque intervenant du groupe, utilise et enrichit le modèle différemment. La construction d'un modèle abstrait aide à y remédier. [16]

La démarche de modélisation choisie pour concevoir notre application peut être représentée graphiquement comme suite :

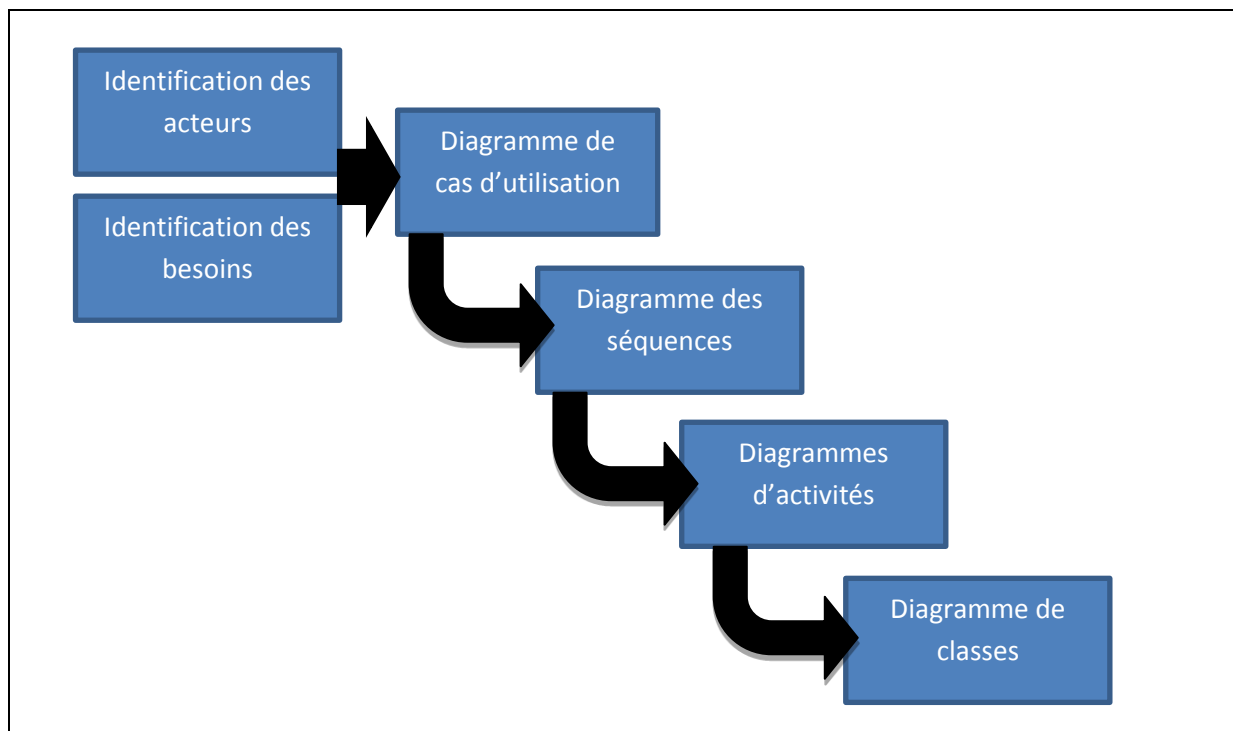


Figure III.2 : La démarche adoptée pour la modélisation

III.2.2. Langage de modélisation :

❖ Définition :

« Un langage de modélisation est un langage artificiel qui peut être utilisé pour exprimer de l'information ou de la connaissance ou des systèmes dans une structure qui est définie par un ensemble cohérent de règles. Les règles sont utilisées pour l'interprétation de la signification des composants dans la structure ».

Un langage de modélisation peut être graphique ou textuel

- **Langages de modélisation graphiques** : utilisent des diagrammes avec des symboles associés à des noms qui représentent les concepts, et des lignes représentant des relations.
- **Langages de modélisation textuels** : utilisent des mots-clés standardisés, accompagnés des paramètres pour rendre l'expression compréhensible par l'ordinateur. [17]

III.3. Présentation d'UML (Unified Modeling Language):

III.3.1. Définition :

UML (en anglais Unified Modeling Language ou « langage de modélisation unifié ») est un langage de modélisation graphique à base de pictogrammes conçu pour fournir des procédés normalisés pour visualiser la conception d'un système. Il est couramment utilisé en développement de logiciel et en conception orientée objet, il peut être appliqué à toutes sortes de systèmes ne se limitant pas au domaine informatique. [18]



Figure III.3 : Logo d'UML

III.3.2. Historique de l'UML

Au début des années 90, une cinquantaine de méthodes d'analyse et de conception objet existaient. Parmi elles, trois étaient considérées comme les plus importantes : BOOCH de Grady Booch, OMT de James Rumbaugh et enfin OOSE d'Ivar Jacobson. Ces trois auteurs ont ensuite décidé d'unir leurs efforts au sein de la société Rational Software et en 1996 la version 0.9 d'UML est proposée.

Deux éléments importants sont à noter :

1) Le terme « unified » signifie que les auteurs ont essayé de regrouper les éléments importants des concepts objets.

2) Le terme langage montre qu'il s'agit d'un langage de modélisation et non d'une méthode.

Les principaux acteurs du secteur informatique ont ensuite participé à cet effort, et UML 1.0 a été proposé à l'Object Management Group (OMG). Cet organisme international chargé de définir des standards dans le domaine de l'objet normalise UML 1.1 en 1997. Cette norme a depuis continué d'évoluer et nous en sommes aujourd'hui à la norme 2.5 bêta depuis Août 2013. UML est un langage qui permet de modéliser non seulement des applications informatiques ou des structures de données, mais également les activités d'un domaine : mécanique, biologie, processus métier, etc. C'est la dernière version que nous allons utiliser pour l'analyse et conception de notre application. [19]

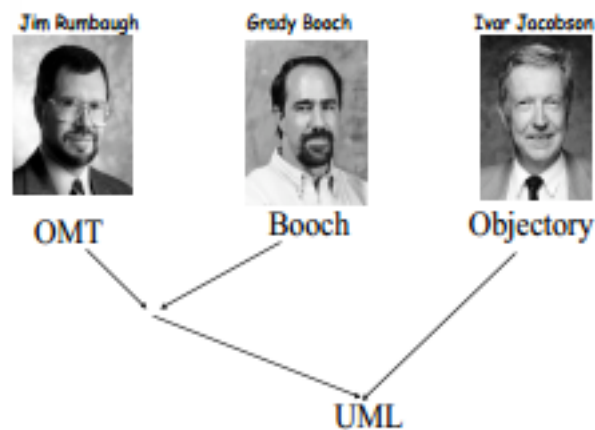


Figure III.4:Fondateurs d'UML

III.3.3. Diagrammes d'UML :

Un diagramme UML est un modèle sous forme de schéma servant à définir ou simuler le fonctionnement d'un système, on doit être alors le plus précis possible dans le contenu du modèle pour s'approcher du code.

UML compte aujourd'hui 14 diagrammes, chacun d'eux est dédié à représenter des concepts particuliers d'un système logiciel, ils sont répartis en 2 grands groupes : Diagrammes structurels et Diagrammes comportementaux, comme le montre la figure ci-dessous.

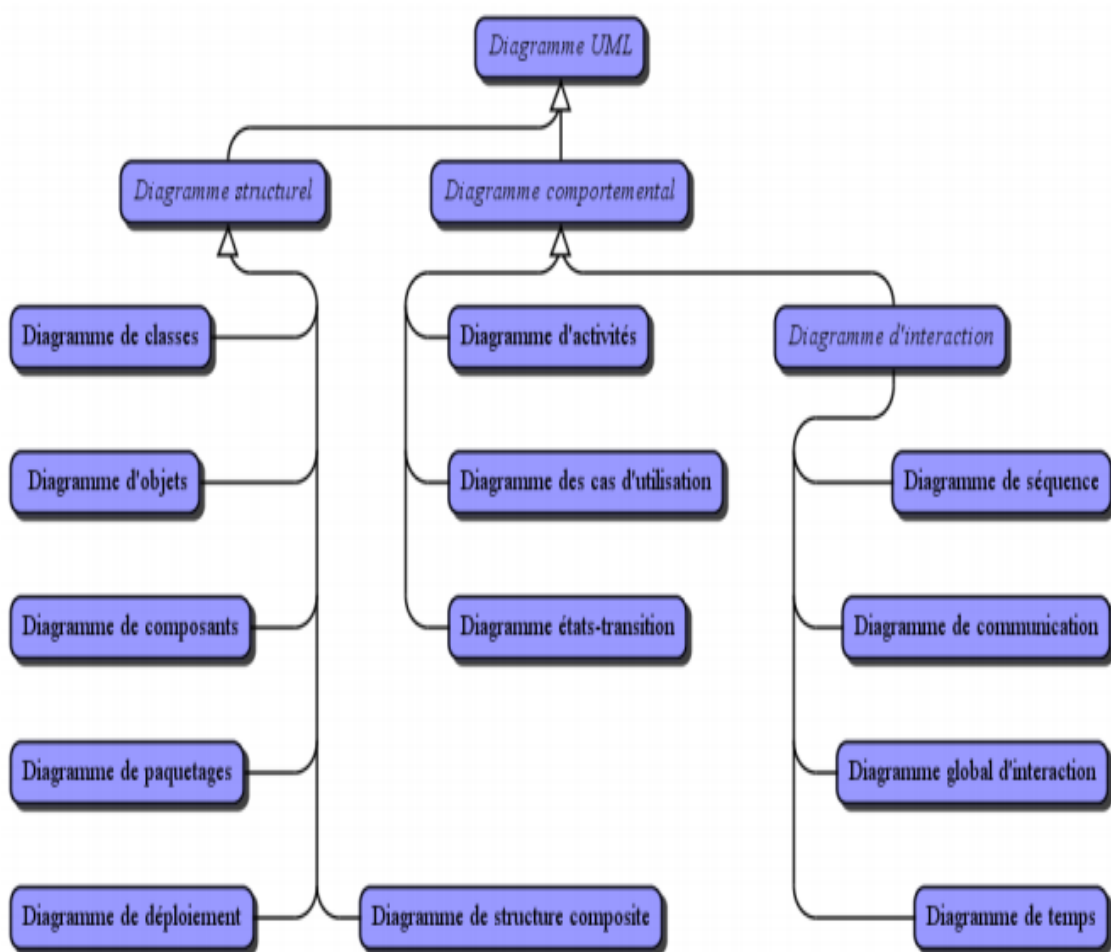


Figure III.5 Diagrammes d'UML 2.0 sous forme de diagrammes de classe

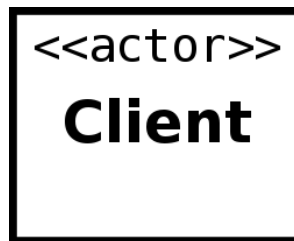
III.4. L'analyse

La phase analyse est l'activité des plus importantes du développement, elle consiste à définir les besoins fonctionnels du système, des acteurs et leurs interactions avec le système avant de conclure sur les cas d'utilisation.

III.4.1. Identification des acteurs :

- ✓ **Définition** : est une entité externe (personne ou machine) agissant sur le système en échangeant de l'information.
 - Il possède un rôle par rapport au système ;
 - Il peut consulter ou modifier l'état d'un système.

Il est possible de représenter les acteurs sous forme d'un « bonhomme » comme ci-dessous à gauche ou sous forme d'un classeur comme ci-dessous à droite et sont identifiés par un nom. [20]



Dans le cas de notre application, Nous avons identifié les acteurs suivants :

Patient : Cet acteur a pour responsabilité de lancer l'application patient, et d'accéder au menu des tâches et le cœur de notre application et delà, entre autre, localiser tous ses surveillants.

Surveillant : a pour responsabilités de lancer l'application, et d'envoyer les coordonnées GPS (altitude, longitude) Chez l'application du patient.

III.4.2.Spécification des tâches

- ✓ **Définition** : Une tâche est l'ensemble des différentes fonctions accessibles pas un acteur bien spécifié.

Les acteurs définis précédemment effectuent un certain nombre de tâches, ces tâches sont résumées dans le tableau ci-dessous :

Acteurs	Tâches
Patient	T1 : Lancer l'application T2 : Ajouter un surveillant T3 : Supprimer un surveillant T3 : Modifier un surveillant T5 : Afficher la liste des surveillants T6 : Activer l'alarme. T7 : Désactiver l'alarme. T8 : Arrêter alarme.
Surveillant	T9: Lancer l'application. T10 : Arrêter l'alarme.

Tableau III.1 : Spécification des tâches.

III.4.3.Spécifications des scénarios :

- ✓ **Définition** : est une séquence spécifique d'actions qui illustre des comportements. Il représente une instance de cas d'utilisation.

- Guide l'analyse et la conception du système.
- Sert de support pour organiser les phases de tests.

Acteurs	Tâches	Scénarios
Patient	T1 : lancer l'application	S1 : Cliquer sur l'icône de l'application.
	T2 : ajouter un surveillant	S2 : demande du formulaire d'ajout. S3 : Saisir le numéro de téléphone et le nom du surveillant. S4 : Valider.
	T3 : Supprimer surveillant	S5 : Afficher la liste des surveillants. S6 : Sélectionner le surveillant à supprimer. S7 : Effectuer la suppression.
	T4 : Modifier surveillant	S8 : Choisir le surveillant à modifier. S9 : Saisir le nouveau nom et le numéro de téléphone S10 : valider les nouvelles informations.
	T5 : Afficher la liste des surveillants	S11 : Accéder à la liste des surveillants.
	T6 : Activer l'alarme	S12 : Accéder au formulaire de configuration de l'alarme. S13 : Remplir l'intervalle d'alerte. S14 : activer l'alarme.
	T7 : Désactivé l'alarme.	S15 : Accéder au formulaire de configuration de l'alarme S16 : désactiver l'alarme.
	T8 : Arrêter alarme	S17:Arrêter l'alarme
Surveillant	T9 : lancer l'application du surveillant	S17 : Cliquer sur l'icône de l'application

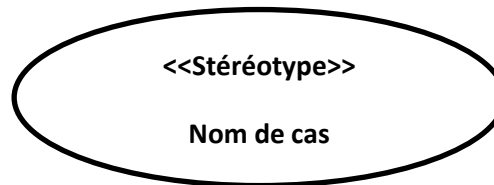
Tableau III.2 : Spécification des scénarios.

III.4.4. Les cas d'utilisation :

III.4.4.1.Définition : (Use Case) Un cas d'utilisation représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produit un résultat observable intéressant pour un acteur particulier, il permet de décrire ce que le système devra faire, sans spécifier comment le faire.

- Décrit plusieurs chemins possibles ;
- Le nom du use case doit se composer d'un verbe à l'infinitif qui décrit une action.

Le cas d'utilisation se représente par une ellipse contenant un nom décrivant la fonctionnalité et éventuellement un stéréotype.



III.4.4.2. Description des cas d'utilisations :

Un cas d'utilisation peut être représenté de 2 façons :

III.4.4.2.1. Représentation textuelle : donne une description du comportement, des actions et réactions des cas d'utilisation. Cette représentation contient plusieurs points dont :

- Le nom du cas d'utilisation ;
- Les intervenants (les acteurs) ;
- La description des scénarios ;
- Les opérations du scénario principal.

Nous procéderons aux descriptions des cas d'utilisation de notre application :

Cas d'utilisation « Ajouter un surveillant »

Use case : Ajouter un surveillant

Scenarios : S1, S2, S3, S4

Acteurs : Patient

Description :

- Accéder à l'application du patient.
- Demande du formulaire d'ajout.
- Saisir le nom et le numéro de téléphone du surveillant à ajouter.
- Valider.

Figure III.6 : Cas d'utilisation «Ajouter un surveillant»

Cas d'utilisation « Supprimer un surveillant »

Use case : Supprimer un surveillant.

Scenarios : S1, S5, S6, S7

Acteur : Patient

Description :

- Accéder à l'application.
- Afficher la liste des surveillants.
- Effectuer la suppression.

Figure III.7 : Cas d'utilisation «Supprimer un surveillant»

Cas d'utilisation « Modifier un surveillant »

Use case : Modifier un surveillant.

Scenarios : S1, S8, S9,S10

Auteur : Patient

Description :

- Accéder à l'application.
- Choisir le surveillant à modifier
- Saisir le nouveau nom et le numéro de téléphone.
- Valider les nouvelles informations.

Figure III.8 : Cas d'utilisation «Modifier un surveillant»

Cas d'utilisation «Activer l'alarme »

Use case : Activer l'alarme

Scenarios : S1, S12, S13,S14

Acteur : Patient

Description :

- Accéder à l'application.
- Accéder au formulaire de configuration de l'alarme.
- Remplir l'intervalle d'alerte.
- Activer l'alarme

Figure III.9 : Cas d'utilisation «Activer l'alarme»

Cas d'utilisation «Désactiver l'alarme »

Use case : Désactiver l'alarme.

Scenarios : S1, S11, S14

Acteurs : Patient

Description :

- Accéder à l'application.
- Accéder au formulaire de configuration de l'alarme.
- Désactiver Alarme.

Figure III.10 : Cas d'utilisation «Désactiver l'alarme»

III.4.4.2.Représentation à l'aide d'un diagramme : Il permet d'identifier les possibilités d'interaction entre le système et les acteurs (intervenants extérieurs au système), c'est-à-dire toutes les fonctionnalités que doit fournir le système.

-Relation entre acteur et cas d'utilisation :

L'interaction est une association, elle représente le lien entre un acteur et un cas d'utilisation qu'il peut exécuter.

Un cas d'utilisation doit être relié à au moins un acteur.

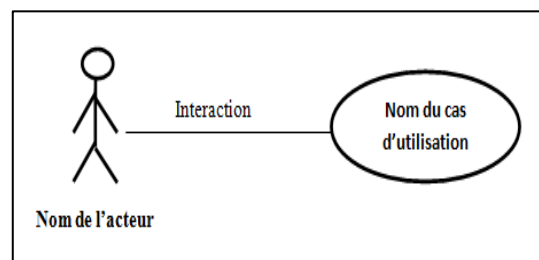


Figure III.11 : Relation entre acteur et cas d'utilisation

-Relation entre cas d'utilisation :

Trois types de relation standard entre cas d'utilisation sont proposés par UML :

- i. **Relation de généralisation (héritage) :** le cas d'utilisation *fils* hérite de tous les caractéristiques du cas d'utilisation *parent*.

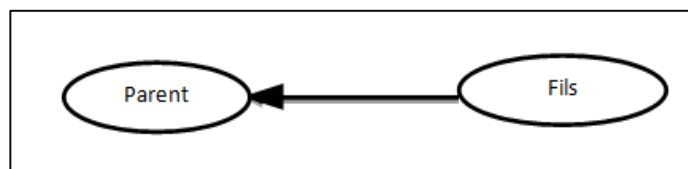


Figure III.12 : Relation d'héritage

- ii. **Relation d'inclusion (include)**

Une relation d'inclusion est une relation dans laquelle un cas d'utilisation (le cas d'utilisation source) inclut les fonctionnalités d'un autre cas d'utilisation (le cas d'utilisation destination).

Pour représenter cette relation, on dessine donc une flèche en pointillé partant du cas d'utilisation source vers le **cas d'utilisation destination**. Puis, on indique le **stéréotype** « include » sur la flèche (voir la figure III.13).

iii. Relation d'extension (extends) :

Une relation d'extension est la deuxième forme de relation stéréotypée. Cette relation est utilisée pour indiquer que le cas d'utilisation destination n'est pas toujours nécessaire au cas d'utilisation source.

L'ajout de cette relation se fait en dessinant une flèche en pointillé partant du cas d'utilisation destination vers le cas d'utilisation source. Puis, on indique le stéréotype « extend » sur la flèche. [21]

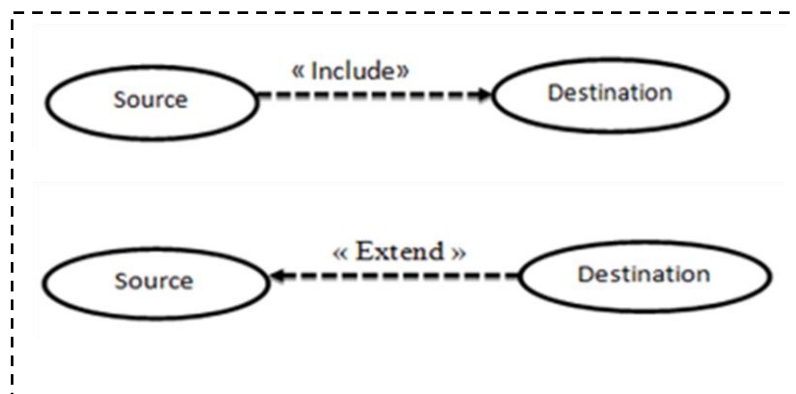


Figure III.13 : Relations d'extension et d'inclusion

Le diagramme de cas d'utilisation de notre application : Les figures suivantes illustrent le diagramme de cas d'utilisation de notre application

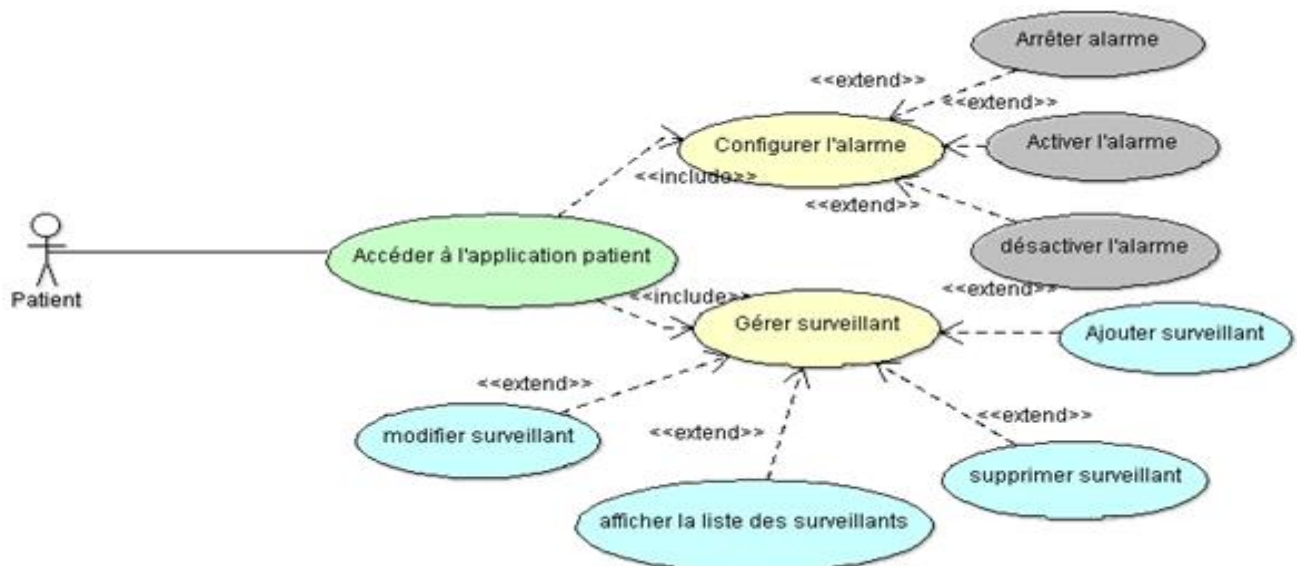


Figure III.14 : Diagramme de cas d'utilisation de l'application patient







III.5. Conception :

La phase de conception consiste à décrire le fonctionnement futur de notre application. On utilise pour cela divers diagrammes, dont le diagramme d'activité, de séquence et de classe.

III.5.1 Diagramme d'activité:

Le diagramme d'activité représente le déroulement des actions, sans utiliser les objets. En phase d'analyse, il permet de modéliser le traitement et de représenter graphiquement le comportement d'une méthode ou l'exécution d'un scénario de cas d'utilisation.

Concepts spécifiques au diagramme d'activité :

-  Nœud (état) initial
-  Nœud final
-  Nœud de fin flot (état de sortie)
-  Nœud de décision (choix)
-  Nœud de bifurcation (fourche) ou d'union (synchronisation)
-  Transition

Nous présenterons dans ce qui suit, quelques diagrammes d'activité de quelques cas d'utilisations.

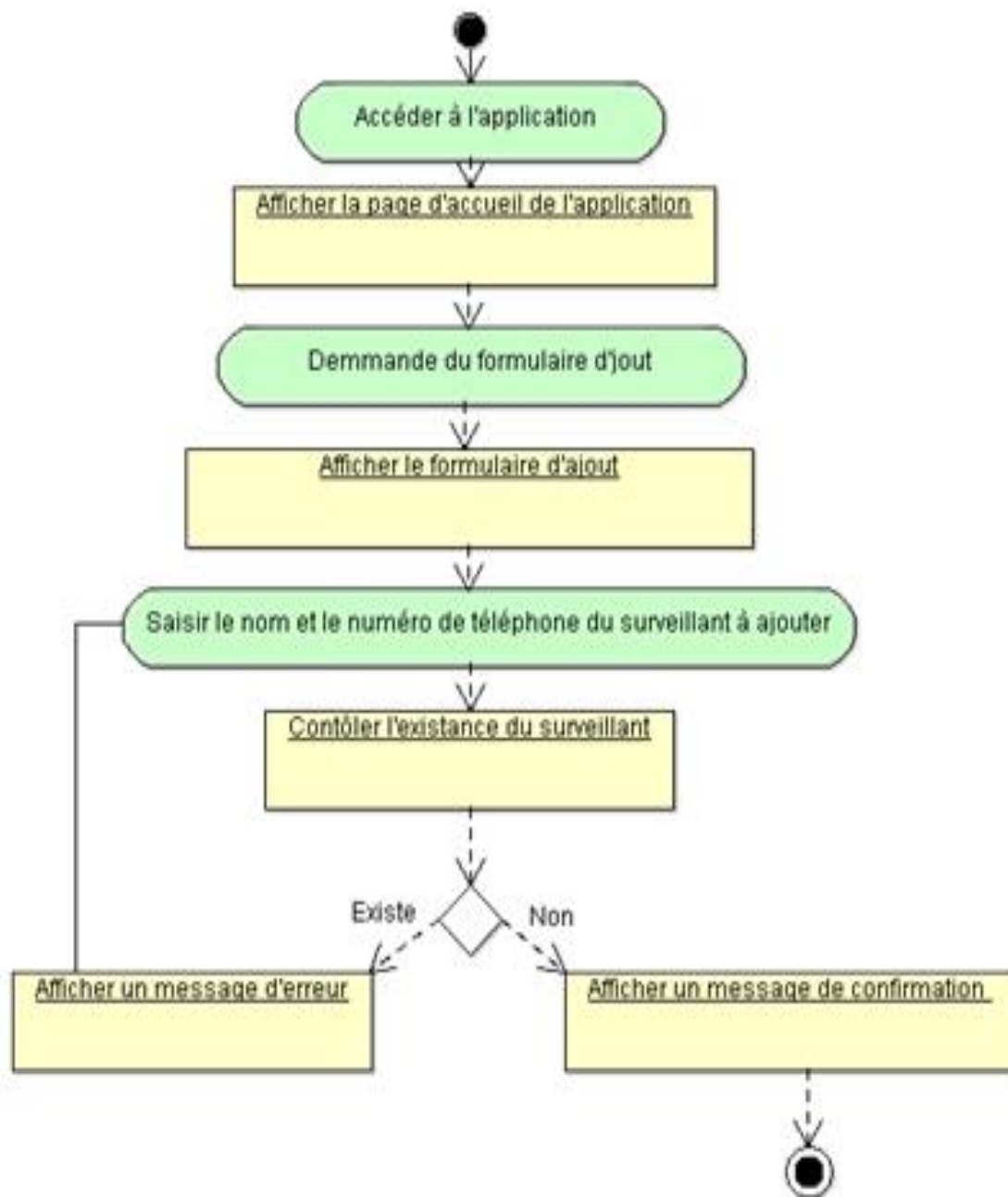
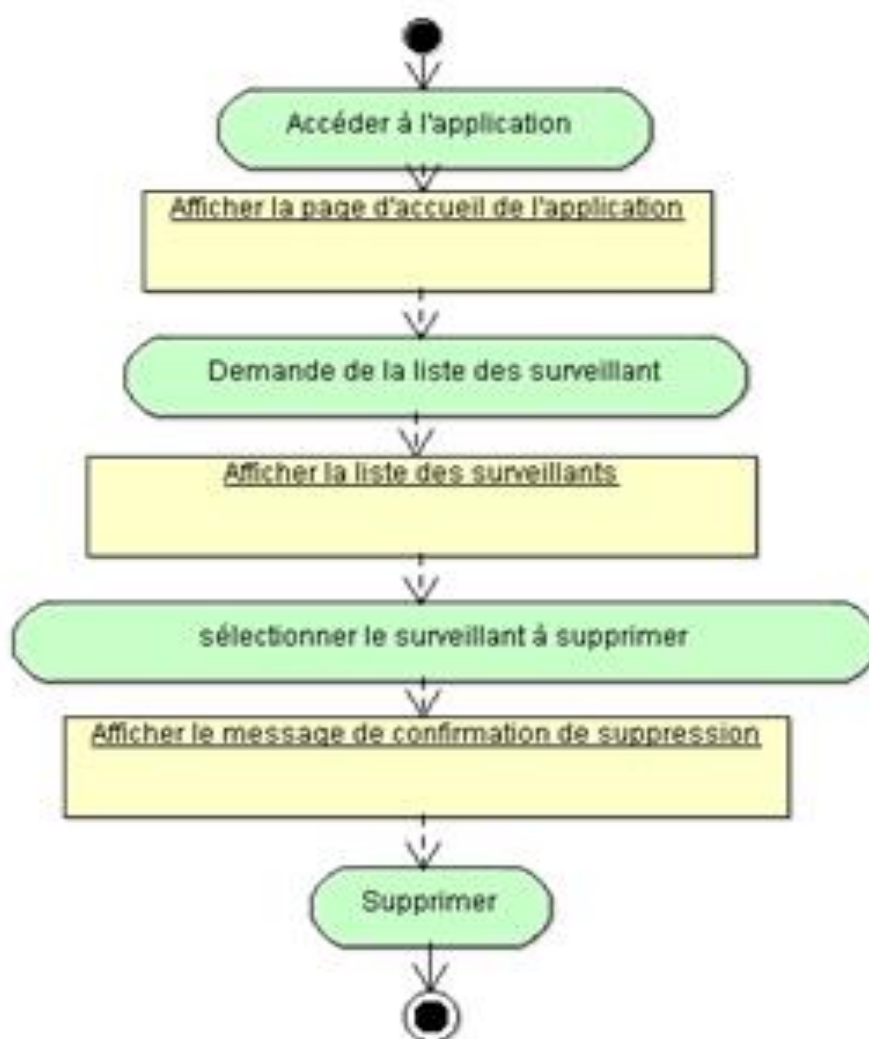


Figure III.15 : Diagramme d'activité du cas d'utilisation «Ajouter un surveillant»



FigIII.16 : Diagramme d'activité du cas d'utilisation « Supprimer un surveillant »

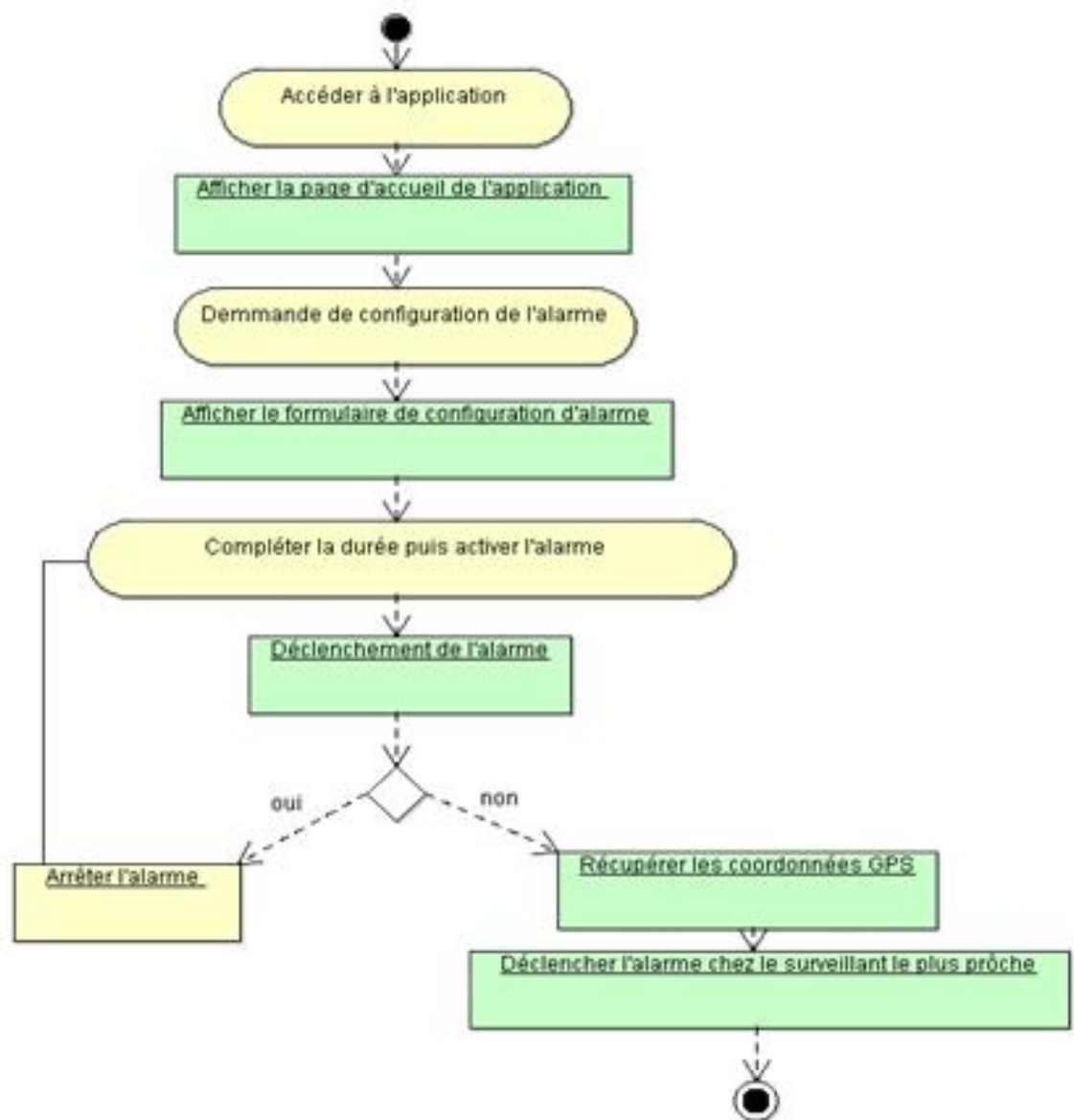
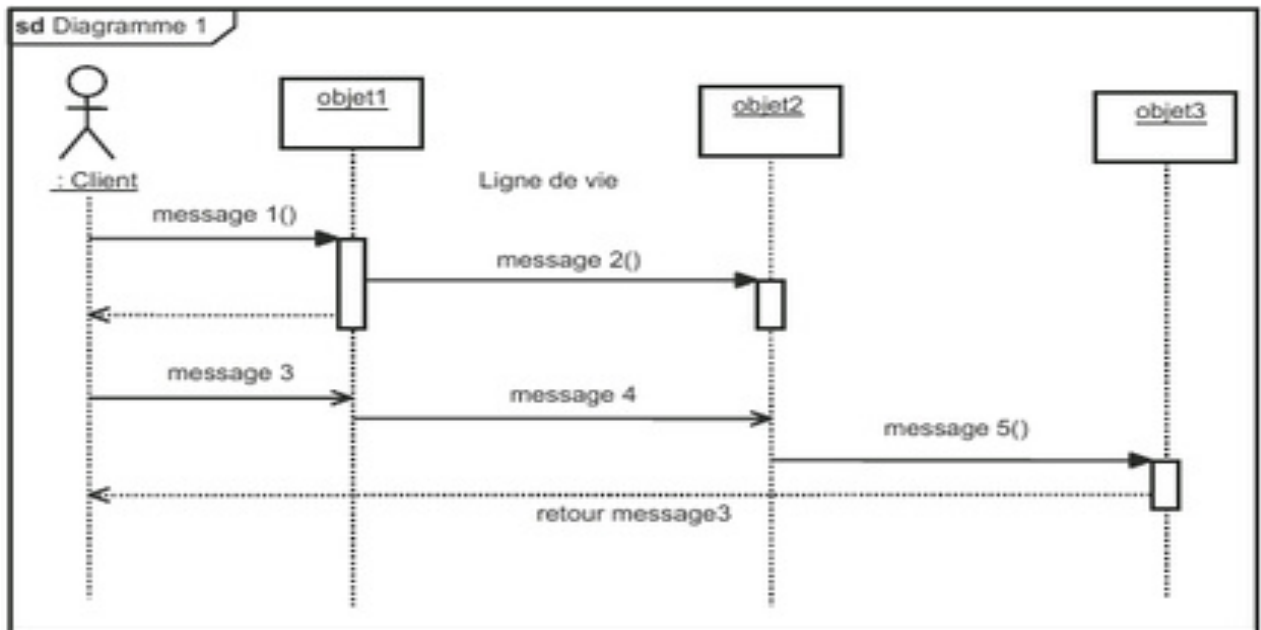


Figure III.17 : Diagramme d'activité du cas d'utilisation « configurer l'horloge »

III.5.2. Diagramme de séquence :

Le diagramme de séquence est une représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique. Il permet de :

- Visualiser l'aspect temporel des interactions ;
- Connaître le sens des interactions (acteur vers système ou contraire), ainsi que les étapes des scénarios.

Formalisme :**FigIII.18 :** Formalisme du diagramme de séquence

Nous présenterons dans ce qui suit, quelques diagrammes de séquences de quelques cas d'utilisations.

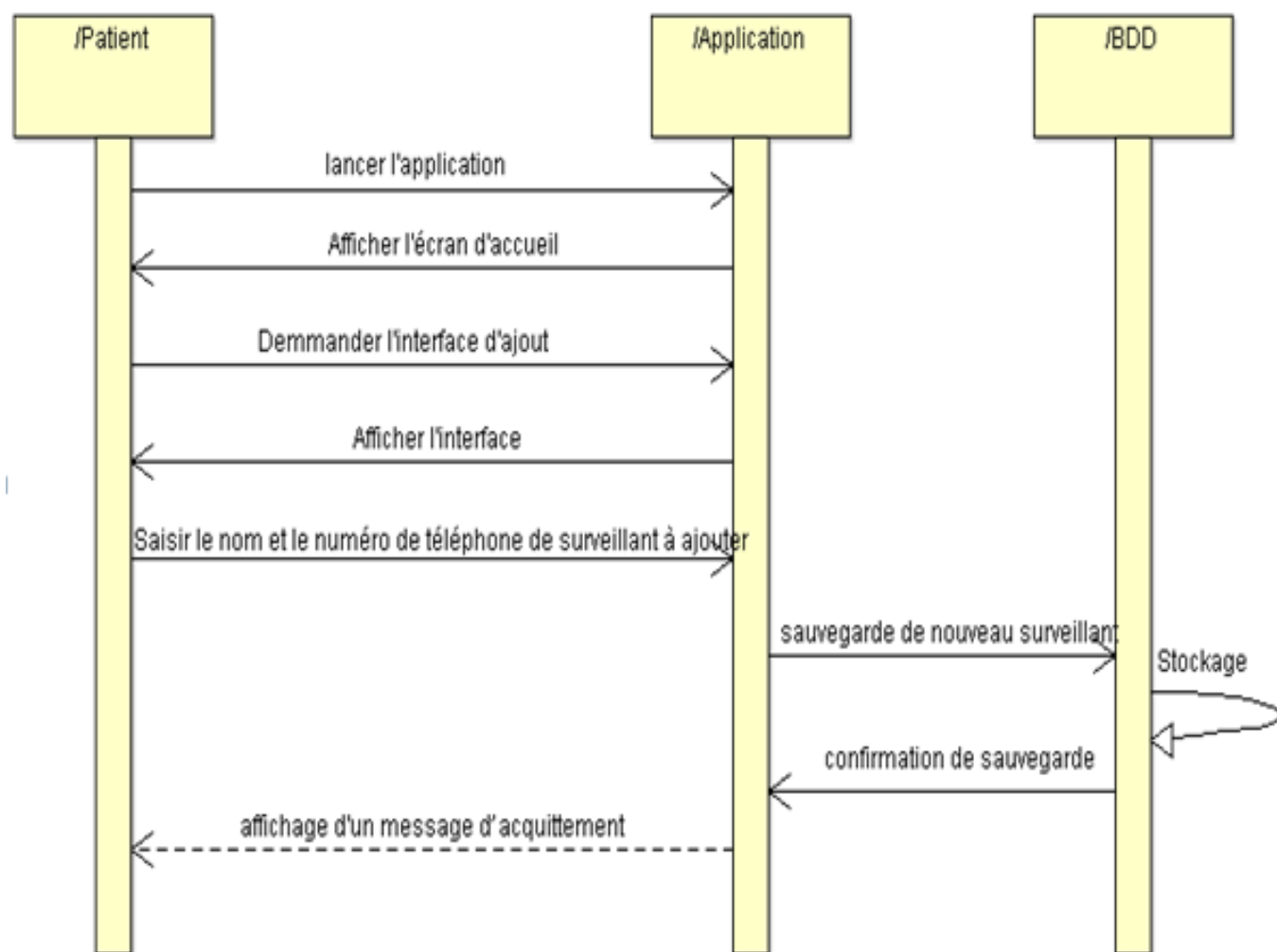


Figure III.19 : Diagramme de séquence simple de cas d'utilisation « Ajouter un surveillant ».

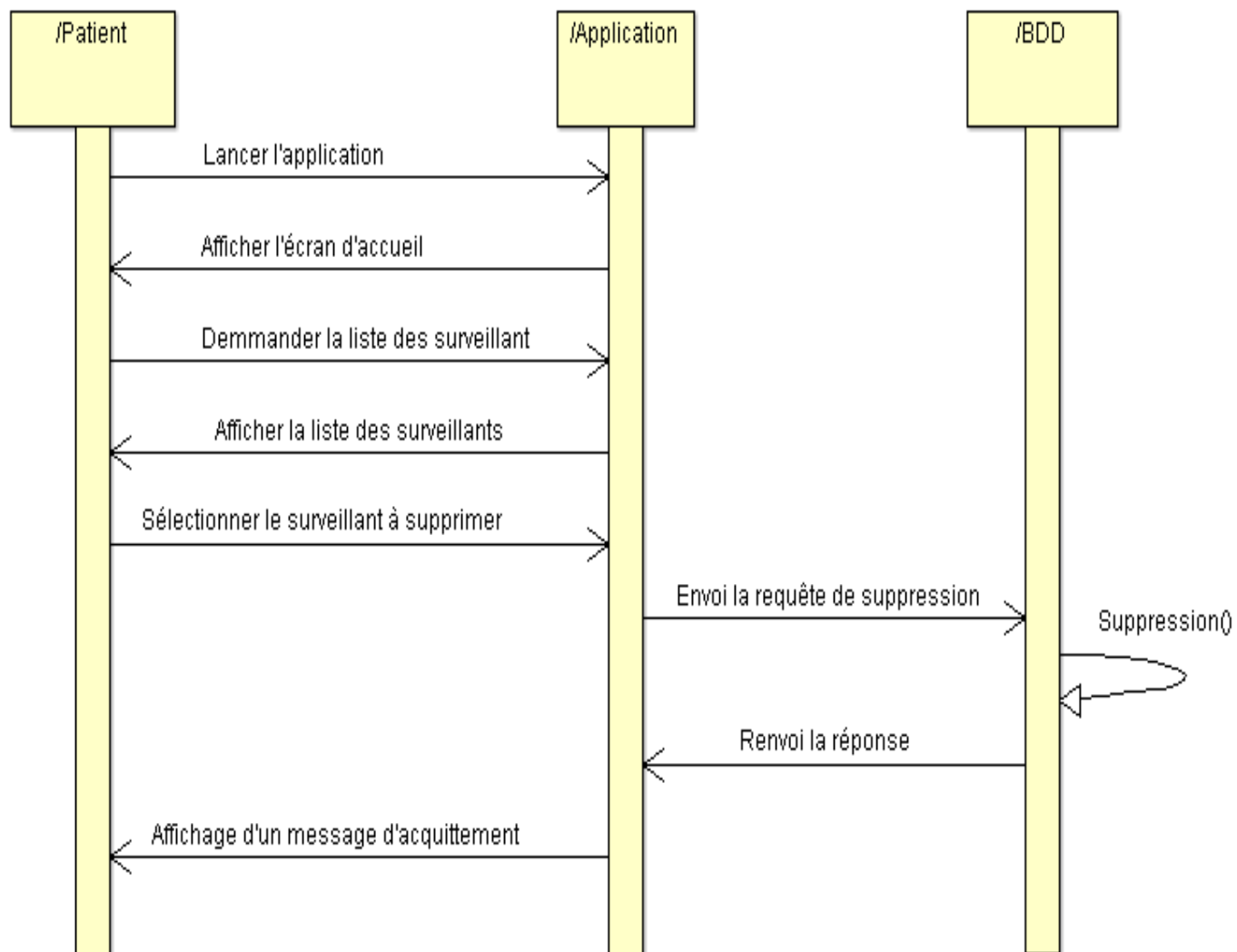


Figure III.20: Diagramme de séquence simple de cas d'utilisation « Supprimer un surveillant »

III.5.3. Diagramme de classe :

Le diagramme de classe est une représentation statique des éléments qui composent un système et de leurs relations. Il est considéré comme un pilier important de la modélisation orientée objet, qui est le seul obligatoire lors d'une telle modélisation.

Alors que le diagramme de cas d'utilisation montre un système du point de vue des acteurs, le diagramme de classes en montre la structure interne du logiciel. Il est surtout utilisé dans la conception, mais peut être utilisé en analyse.

La description d'un diagramme de classe est fondée sur

- Le concept objet,
- Le concept de classe comprenant les attributs des opérations,
- Les différents types d'associations entre classes.

III.6. Conception de la base de données :

Les diagrammes de l'UML nous ont apporté une grande aide dans la conception de la base de données, toutefois ce ne sont que des modèles théoriques, qu'on ne peut pas implémenter. La solution serait de traduire ces diagrammes en modèles physiques, on a implémenté pour cela une seule table sous SQLite .

La table de la base de donnée « bdd_surveillance_adomicille » :

Table : Surveillant

champ	Type	Clé	Signification
Id_surveillant	integer	Primaire(PK)	Numéro d'identifiant
Num_tel	String		Numéro de téléphone
Nom_surveillant	String		Nom du surveillant

TableauIII.3 : Table Surveillant

Conclusion

Dans ce chapitre, nous avons introduit les fonctionnalités de notre application. Pour les atteindre, 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 les acteurs de notre application, les tâches qu'ils assurent et les scénarios associés à chaque tâche. Nous avons élaboré les diagrammes de cas d'utilisation de chaque acteur, élaboré les diagrammes de séquences, de classe de quelque cas d'utilisation. Enfin, nous avons représenté la table 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 utilisés et les différentes fonctionnalités de notre application à travers ses différentes interfaces.

Introduction

Après avoir décortiqué les parties analyse et conception dans le chapitre précédent, nous présenterons, dans ce dernier chapitre, la réalisation de notre application. Nous procéderons dans ce chapitre à des spécifications techniques en parlant des outils du développement et le choix du langage qui entrent en jeu pour concrétiser l'implémentation de notre application. Nous terminons ce chapitre par présenter et décrire l'architecture de notre application et expliquer son fonctionnement par les différentes interfaces.

IV.1.Outils et langage utilisé

IV.1.1. Les outils

Le travail de création d'une application nécessite des outils adaptés à la technologie visée et surtout les versions les plus récentes de différents logiciels de développement pour un meilleur résultat. Nous citons ci-dessous les divers outils utilisés pendant la réalisation de notre application

IV.1.1.1. L'environnement matériel

Pour la réalisation de l'application, nous avons utilisé :

- ❖ Un ordinateur HP :
 - Edition Windows → Windows 7 Professionnel
 - Type du système → Système d'exploitation 32 bits
 - Processeur → Intel® Core™ i3-4005U CPU@ 1.70 GHz
 - Mémoire installer (RAM) → 4.00 Go (2.45 Utilisable)

Nous avons utilisé deux Smartphones pour les tests, un pour l'interface patient et l'autre pour le surveillant.

- ❖ Un Smartphone condor c4+ portant la version 4.2.2 (Jelly Bean) d'Android comme système d'exploitation.
- ❖ Un Smartphone HTC DESIRE 620 portant la version 4.4 (kit kat) d'Android comme système d'exploitation.

IV.1.1.2 .L'environnement logiciel

Pour pouvoir réaliser une application dans des bonnes conditions il est de bien entendu de choisir son environnement de travail selon les besoins. Nous avons opté pour la réalisation d'une application mobile sous le système Android pour son développement, afin d'obtenir un fichier à extension *.apk* qui sera par la suite installé sur des terminaux mobiles de différents types fonctionnant avec le système Android 4.2 ou plus.

Le développement d'applications pour Android se fait entièrement en Java, ce dernier est un langage puissant orienté objet, utilisé très largement dans le monde du développement, en utilisant des IDE tels que : Eclipse , NetBeans et Android Studio.

A) Android Studio

Android Studio est un environnement de développement des applications Android. Il est basé sur la version *IntelliJ*..

B) Le SDK Android

Signifie *Software Development Kit*, c'est un ensemble d'outils d'aide à la programmation pour concevoir des logiciels, jeux, application mobiles, etc. pour un terminal et/ou, un système d'exploitation spécifique un SDK contient du code, permettant de concevoir une interface ou une partie d'interface numérique(web ,mobile, jeux, logiciels de recherches...).Ce code est conçu avec un langage de programmation correspondant au terminal(ordinateur, téléphone ,tablette...)et au système de navigation cibles..

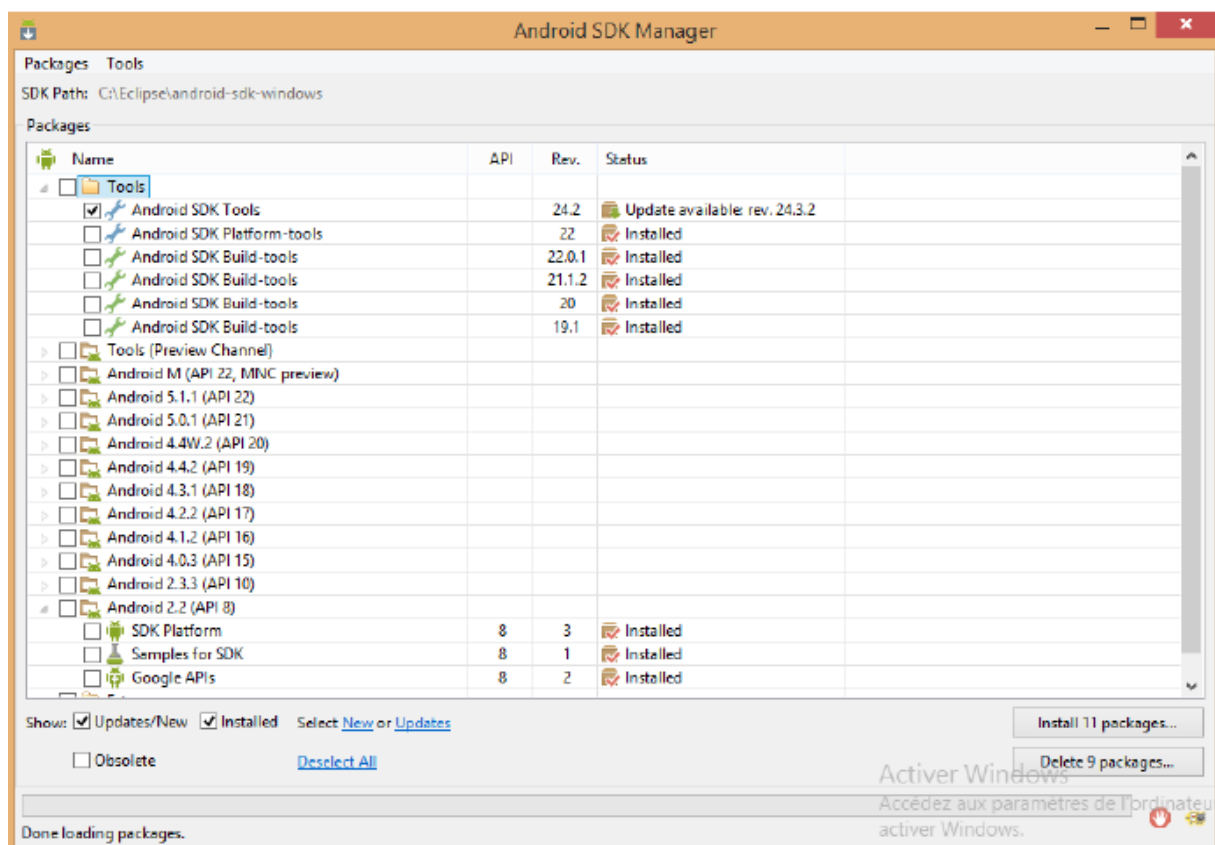


Figure IV.1 : Interface d'installation d'Android SDK.

C) La création d'AVD

L'Android Virtual Device, Emulator en anglais, aussi appelé AVD, est un émulateur de terminal, c'est-à-dire un logiciel qui fait croire à notre ordinateur qu'il est un appareil sous le système Android et pour chaque version d'Android est associée une version de l'émulateur. C'est la raison pour laquelle on n'a pas besoin d'un périphérique sous Android pour développer et tester la plupart de nos applications ! En effet, une application qui affiche un calendrier par exemple peut très bien se tester dans un émulateur, mais une application qui exploite le GPS doit être éprouvée sur le terrain pour que l'on soit certain de son comportement.

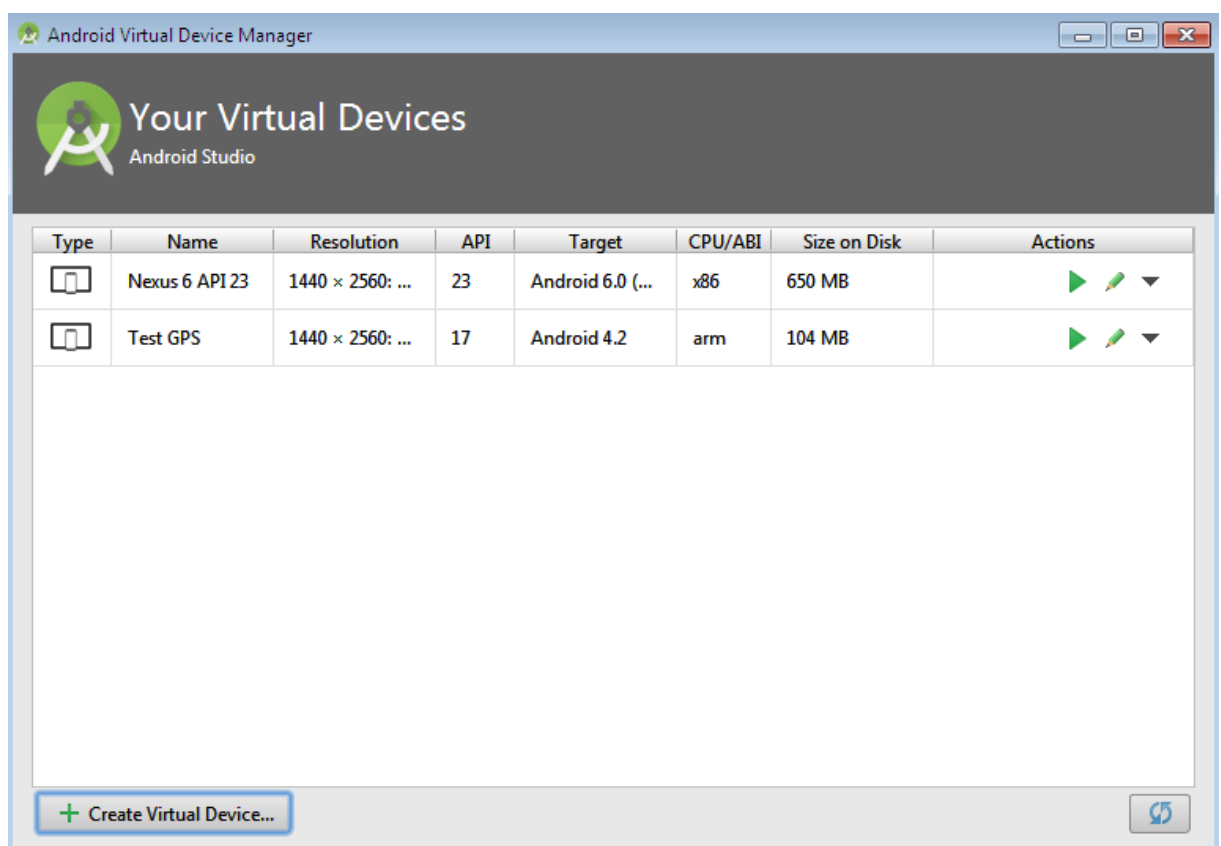


Figure IV.2 : création d'un AVD.

D) L'émulateur

L'émulateur permet de lancer sur la machine du développeur un terminal représentant à l'écran un téléphone embarquant Android.

Il est indispensable pour le développeur qui ne dispose pas un Smartphone Android et il est associé aux différentes versions d'Android. En général, il peut remplacer un Smartphones tous en proposant de tas des fonctionnalités. Et on peut également télécharger un émulateur prédéfini sur le site de Google « DevelopersAndroid ».



Figure IV.3 : Emulateur Android.

E) La base de données SQLite

Dans notre travail nous avons utilisé SQLite qui est un moteur de base de données très léger, écrit en C, qui implémente le langage SQL. IL fait partie d'Android. Ce SGBD (Système de Gestion de Base de Données) intègre la presque quasi-totalité des fonctionnalités du langage SQL à quelques exceptions près. SQLite n'est pas exploitable directement, il faut instancier une classes de base, à savoir `SQLiteDatabase` qui est incluse dans le package `android.database.sqlite`. Ensuite nous avons besoin d'une classe qui va nous permet de gérer l'insertion, la modification, la suppression d'un surveillant dans la base de données ainsi que de faire des requêtes pour récupérer un surveillant contenu dans la base. Et cette classe s'appelle **surveillant** dans notre cas.

F) Les alarmes

Contrairement aux services qui s'exécutent en permanence en arrière-plan, l'objectif des alarmes est uniquement de lancer une ou plusieurs Intents à un moment précis.

- Une alarme ne fait donc pas réellement partie intégrante de l'application et celle-ci s'exécutera d'ailleurs en dehors de l'application.
- Une alarme pourra très bien déclencher un objet Intent pour votre application, même après sa fermeture. En cela, les alarmes restent un concept à part, n'ayant que peu ou pas de lien avec l'application que vous développez (mais partie intégrante de la logique de votre application).
- Les alarmes sont particulièrement intéressantes lorsqu'elles sont combinées avec des Broadcast Receivers puisque de cette façon vous n'avez pas besoin que votre application soit en cours d'exécution pour traiter l'intention de l'alarme. Les alarmes fonctionnent même lorsque l'appareil est en veille et permettent justement de le sortir de la veille.

G) La localisation

Pour pouvoir géocaliser un utilisateur, il faut posséder une instance de la classe `LocationListener` ou que l'activité incluant votre carte implémente l'interface `LocationListener`.

Ainsi afin de pouvoir s'abonner à la mise à jour des coordonnées de l'utilisateur, il faut utiliser la méthode **`requestLocationUpdates(String, long, float, LocationListener)`**. Cette méthode possède 4 arguments :

- Le provider utiliser pour recevoir les mises à jour des coordonnées utilisateurs (GPS / NETWORK ...)
- Un intervalle minimum entre deux notifications (en millisecondes)
- Un intervalle minimum entre deux notifications (en mètre)
- L'instance de votre `LocationListener`

IV.1.2. Le langage Java et Xml

IV.1.2.1 Le Java

C'est un langage de programmation orienté objet, développé par Sun Microsystems. Il permet de réaliser des logiciels compatibles avec de nombreux systèmes d'exploitation. Java offre aussi la possibilité de réaliser des programmes pour téléphones mobiles et assistants personnels PDA. Il possède plusieurs caractéristiques qui le rendent parmi les meilleurs outils de développement des logiciels tels que : simple, distribué, interprété, indépendant de l'architecture, portable, robuste, sûr, dynamique, multithread.

IV.1.2.2 Le XML

Le XML (eXtensibleMarkupLanguage, « langage à balise extensible » en français) est un langage informatique de balisage générique qui dérive du SGML10. Cette syntaxe est dite « extensible » car elle permet de définir différents espaces de noms, c'est-à-dire des langages avec chacun leur vocabulaire et leur grammaire, comme XHTML, XSLT, RSS, SVG. Il est couramment utilisé comme format générique pour l'échange de données entre serveurs et applications, dans les processus de communication entre couches applicatives ou pour le stockage de données complexes. Le XML a été largement adopté dans tous les secteurs d'industrie et par la majorité des langages de programmation.

IV.2. Implémentation de notre travail

Notre application permet de surveiller les patients à distance, elle est composée de deux interface (patient et surveillant).

❖ L'interface qui concerne le patient

Elle permet de :

- Ajouter un surveillant.
- Supprimer un surveillant.
- Modifier un surveillant.
- Afficher la liste des surveillants.
- Activer l'alarme.
- Désactiver Alarme.
- Arrêter alarme.

L'alarme se déclenche régulièrement après un intervalle de temps **T1** (selon la configuration du patient). Et à chaque déclenchement après un temps **T2**, si le patient ne la désactive pas alors l'application localise le plus proche des surveillants puis déclenche une alarme chez ce dernier.

❖ L'interface qui concerne le surveillant

Cette application permet de :

- se localisé et d'envoyer les coordonnées GPS (latitude et longitude) à l'interface patient.
- Arrêter l'alarme déclenchée par le patient.

IV.3 Quelques interfaces

Dans cette section nous allons vous présenter les écrans de notre application, ainsi que leurs fonctionnalités sous forme de capture d'écran.

II.3.1 Page d'accueil



Figure IV.4 : Page d'accueil de l'application

1 : Bouton de configuration d'une alarme

2 : Bouton pour gérer les surveillants.

IV.3.2. La fenêtre configuration de l'alarme

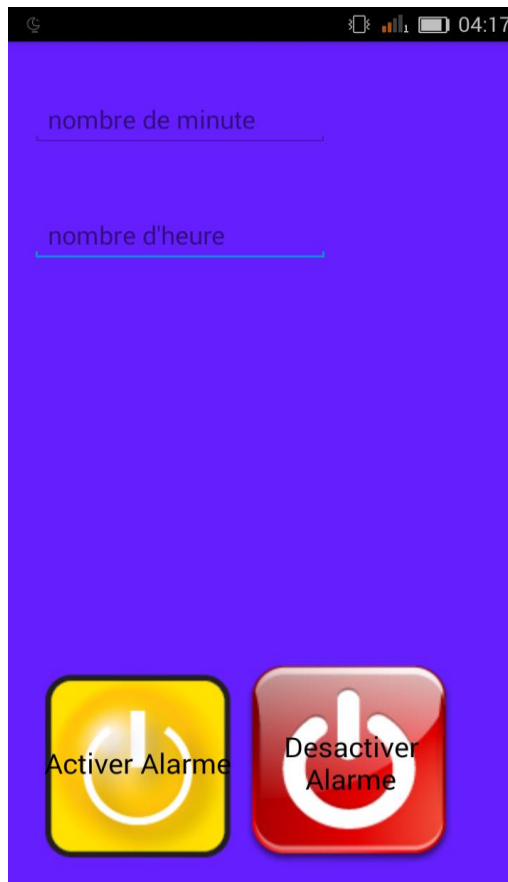


Figure IV.5 : interface de la configuration de l'alarme

Lors d'un clic sur le bouton « Configurer Alarme » de la page d'accueil, le patient se redirige vers la page de configuration de l'alarme.

1 : Zone de texte pour saisir la durée en nombre de minute.

2 : Zone de texte pour saisir la durée en nombre d'heure.

3 : Bouton pour activer l'alarme.

4 : Bouton pour désactiver l'alarme.

IV.3.2. La fenêtre d'arrêt de l'alarme



Figure IV.6 : interface d'arrêt de l'alarme

1 : Alerte généré par l'application après un temps T.

IV.3.3. Fenêtre Ajouter surveillant

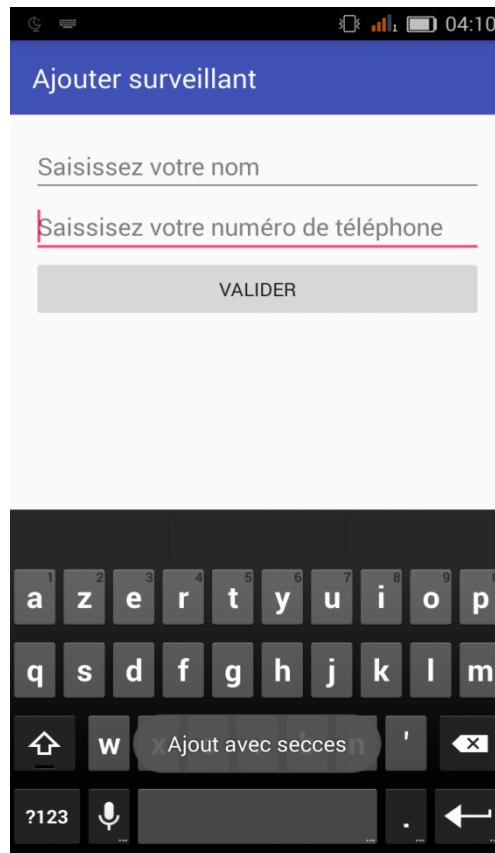


Figure IV.7 : interface d'ajout de surveillant

Lors d'un clique sur le bouton «Gérer surveillant» de la page d'accueil, l'interface d'ajout s'affiche.

1 : Zone de texte pour saisir le nom du surveillant.

2 : Zone de texte pour saisir le numéro de téléphone du surveillant.

3 : Bouton pour valider l'ajout.

IV.3.4. Fenêtre Supprimer surveillant

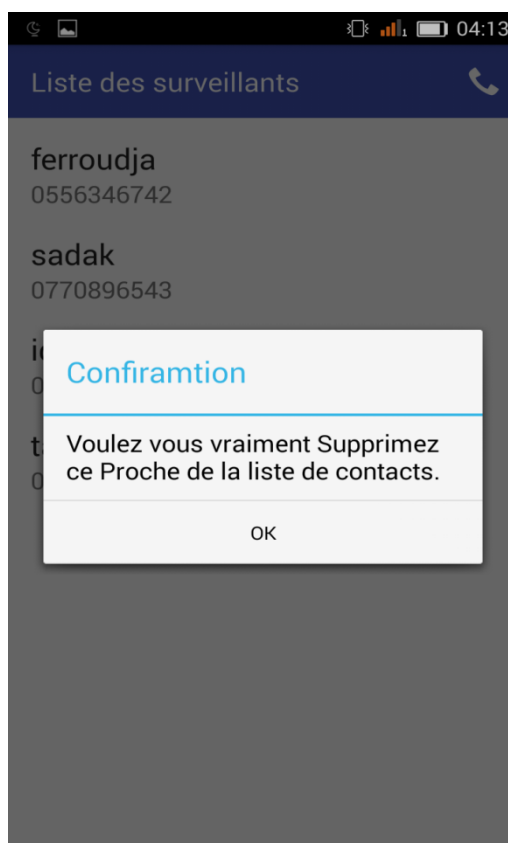


Figure IV.8 : interface de suppression de surveillant

Après avoir sélectionné le surveillant à supprimer, l'application nous fournit un message de confirmation de la suppression.

IV.3.5. Modifier surveillant

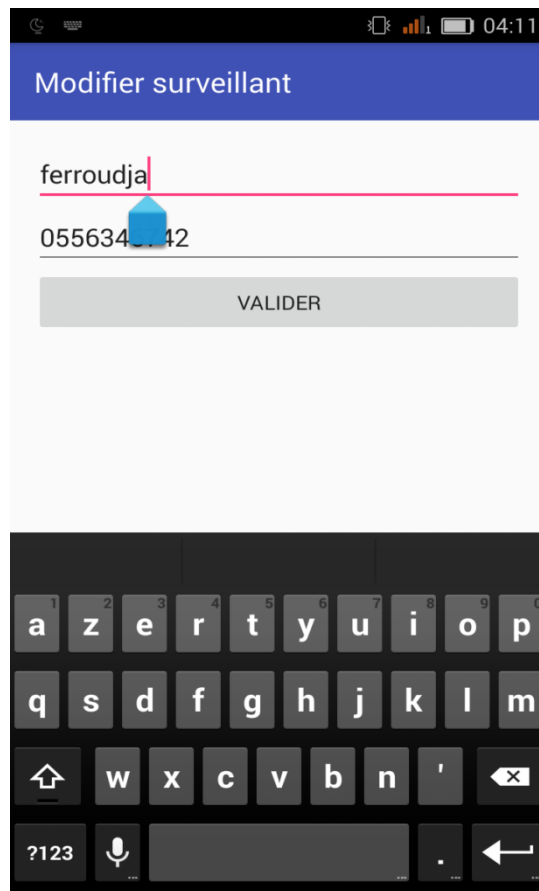


Figure IV.9 : Interface de modification d'un surveillant

Après avoir cliqué sur le surveillant à modifier on aura :

- 1 : zone de texte pour modifier le nom du surveillant.
- 2 : zone de texte pour modifier le numéro de téléphone.

IV.3.6. Liste des surveillants

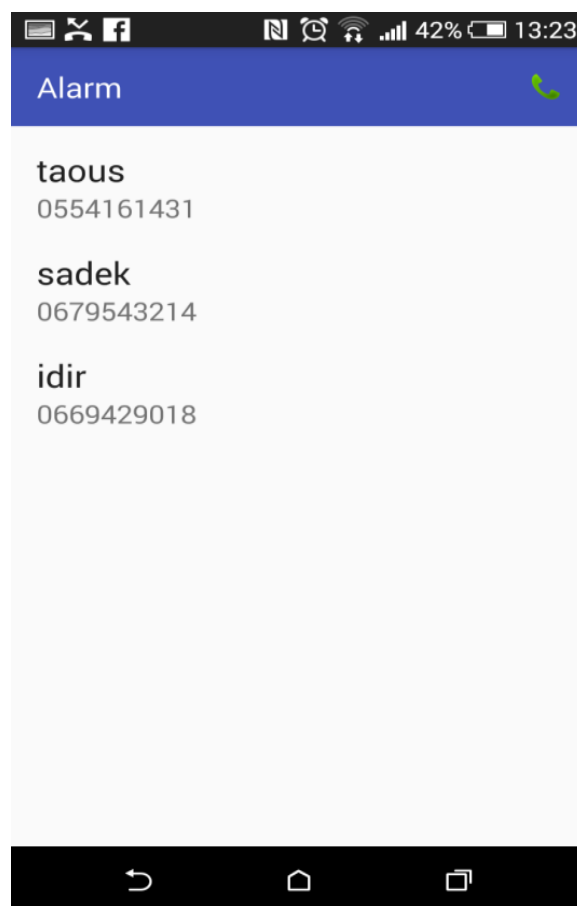


Figure IV.10 : Interface de la liste des surveillants

Après avoir cliqué sur le bouton “Gérer surveillant” de la page d’accueil

Conclusion

Dans ce chapitre nous avons fait une description de notre application, ou nous avons présenté l'environnement de développement et les différents outils de programmation, ensuite nous avons décrit quelques interfaces d'utilisation.

Notre projet a permis de concevoir et de réaliser une application mobile sous Android servant à la surveillance à domicile, elle permet en cas d'urgence de localiser et de sélectionner le surveillant qui se trouvent tout près de la localisation du patient pour pouvoir passer un appel.

Dans le cadre de ce projet, nous avons eu l'opportunité de nous familiariser avec des technologies récentes telles qu'Android et la modélisation par l'UML, à enrichir notre savoir-faire et notre expérience par rapport à la programmation. Un autre aspect de ce projet était de pouvoir travailler sur la technologie et des logiciels comme Android studio ainsi que le Système de Gestion de Base de Données SQLite et d'apprécier la qualité des services offerts par ce type de logiciels, en fin nous avons aussi l'occasion de manipuler la géolocalisation.

Toutefois, cette solution peut présenter des imperfections du fait des difficultés de localiser des personnes via leur Smartphones équipés de GPS, si celles-ci se trouvent indoor.

Et pour cela, et comme perspectives, nous suggérons de combiner cette solution avec la combinaison de plusieurs techniques de localisation.

Il serait aussi intéressant de remplacer la base de données interne par une base de données externe.

Référence bibliographique

- [1]:<http://www.maxisciences.com/android/>
- [2]:https://fr.wikipedia.org/wiki/Open_Handset_Alliance
- [3] :<https://fr.wikipedia.org/wiki/Android>
- [4]:<https://openclassrooms.com/courses/creez-des-applications-pour-android/l-architecture-d-android>
- [5] :<http://javamind-fr.blogspot.com/2012/04/les-composants-dune-application-pour.html#!/2012/04/les-composants-dune-application-pour.html>
- [6] :<http://supertos.free.fr/supertos.php?page=1076>
- [7]: <http://developer.android.com/>
- [8]: [http:// developer.android.com/sdk/1.5_r1/index.htm](http://developer.android.com/sdk/1.5_r1/index.htm)
- [9] :<http://anr-prodige.com/index.php?n=Technologies.Geolocalisation>
- [10] :<http://www.geolocalisation-vehicule.be/techniques-de-geolocalisation-le-gsm-le-wifi-ladresse-ip-et-le-rfid>
- [11] :<http://blog.clever-age.com/fr/2015/05/21/les-technologies-de-geolocalisation-indoor/>
- [12] : http://www.foretcommunale-cameroun.org/download/COURS_GPS.PDF
- [13] : Jean Luc Cosandier : Principe généraux de la localisation par satellites
- [14] : <http://www.utdallas.edu/~aiken/GPSCLASS/GPSBasics.pdf>
- [15] :<http://telecom.insa-lyon.fr/sites/default/files/cgt/promo-20092012%20-%20geolocalisation.pdf>
- [16]: <http://laurent-audibert.developpez.com/Cours-UML/?page=introduction-modelisation-objet#L1-2-1-b>
- [17] : https://fr.wikipedia.org/wiki/Langage_de_mod%C3%A9lisation
- [18] : <http://www.additeam.com/SSII/uml/>
- [19] : [http://fr.wikipedia.org/wiki/UML_\(informatique\)#Histoire/](http://fr.wikipedia.org/wiki/UML_(informatique)#Histoire/)
- [20] : <http://www.commentcamarche.net/contents/1138-uml-cas-d-utilisation-use-cases>
- [21] :<https://openclassrooms.com/courses/debutez-l-analyse-logicielle-avec-uml/les-relations-stereotypes>

Avec l'accroissement démographique en nette progression et le vieillissement d'une bonne frange de la population, la plus part des citoyens ont des patients à leur domicile et ça devient pénible de les assister quotidiennement, et ils ne peuvent plus répondre efficacement aux sollicitations de leur patients. D'où l'assistance des patients à domicile devient un problème épineux et quasi ingérable.

Depuis sa création, l'environnement Android ne cesse de progresser, ajoutant toujours de nouvelles fonctionnalités pour faciliter la vie quotidienne des utilisateurs. La concurrence avec les autres systèmes d'exploitation, tel que l'iOS d'i Phone et le Windows de Nokia a également poussé l'amélioration des produits Android. Selon une étude réalisée par le cabinet américain Stratégie Analytiques, le janvier 2015, une personne sur sept aujourd'hui a un Smartphone.

C'est pour profiter de cette généralisation du Smartphone et remédier aux problèmes rencontrés quotidiennement pour la surveillance des patients à domicile que l'idée de notre projet est née.

Notre projet a permis de concevoir et de réaliser une application mobile sous Android servant à la surveillance à domicile, elle permet en cas d'urgence de localiser et de sélectionner le surveillant qui se trouvent tout près de la localisation du patient pour pouvoir passer un appel.

Dans le cadre de ce projet, nous avons eu l'opportunité de nous familiariser avec des technologies récentes telles qu'Android et la modélisation par l'UML, à enrichir notre savoir-faire et notre expérience par rapport à la programmation. Un autre aspect de ce projet était de pouvoir travailler sur la technologie et des logiciels comme Android studio ainsi que le Système de Gestion de Base de Données SQLite et d'apprécier la qualité des services offerts par ce type de logiciels, en fin nous avons aussi l'occasion de manipuler la géolocalisation.

Toutefois, cette solution peut présenter des imperfections du fait des difficultés de localiser des personnes via leur Smartphones équipés de GPS, si celles-ci se trouvent indoor.

Et pour cela, et comme perspectives, nous suggérons de combiner cette solution avec la combinaison de plusieurs techniques de localisation.