REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU
• X ●ΘΛ●ΣX [://:V ●X[●Λ[●O
FACULTE DU GENIE ELECTRIQUE
ET INFORMATIQUE

Mémoire de fin d'études

En vue de l'obtention du diplôme de Master en informatique

« Option : Système Informatique »

Thème

Conception et réalisation d'une application mobile sous android offrant une assistance médicale

Dirigé par :

M^elle^ CHAMEK Linda

Réalisé et présenté par :

M^elle^ ***CHERNAI Baya***
M^elle^ ***ARHAB Sofia***

2014/2015

Remerciements

Nous remercions tout d'abord **DIEU** tout puissant de nous avoir donné la santé et le courage d'amener ce projet à terme.

Nous tenons particulièrement à remercier M^{elle} CHAMEK, notre promotrice, pour la finesse de ses attitudes sur le plan aussi bien humain que scientifique.

D'elle, nous avons toujours reçu non seulement les encouragements dont nous avions tant besoin, mais aussi les précieux conseils pratiques que seul une femme, ayant des qualités humaines comme elle, peut amener à prodiguer.

Nous lui sommes infiniment gré.

Nous adressons nos remerciements aux membres de jury, pour l'honneur qu'ils nous font d'accepter de juger notre travail.

Que nos familles et amis trouvent ici nos sentiments de gratitudes les plus sincères, pour avoir financé ce travail sans jamais se défiler pour nous avoir encouragés dans les moments de doute et soutenus quand on s'est cru échouer. Et surtout d'avoir cru en nous.

Nous ne ferons pas le pari de les énumérer tous sans risquer d'en omettre certains.

Nos parents ont tout sacrifié pour nous voir réussir, à travers ce modeste travail nous espérons les rendre fiers et leur prouver que leurs efforts n'ont pas été vains.

MERCI

Dédicaces

Je dédie ce modeste travail :

A ma chère maman et mon cher papa

A mon frère Sid Ali

A mes sœurs Amel et Sandra

A tous mes amis, particulièrement, Cherif, Lynda, Thiziri, layla et Lila avec lesquels j'ai passé mes plus beaux jours.

Et Sofia bien sûre

A tous ceux qui m'aiment de loin ou de prés.

Ch.Baya

Dédicaces

Je dédie ce travail à :

- *Mes chers parents.*
- *Mon frère ,Toute ma famille, mes cousines .*
- *Ma binôme Baya.*
- *Tous mes amis et mes camarades (Lynda, yamina, fetta,) avec lesquelles on a formé un formidable environnement de travail.*

- *Je dédie ce travail à tout ceux que j'aime et j'apprécie.*

A.Sofia

Liste des Figures

SOMMAIRE

Chapitre 2 : Analyse et conception

Chapitre 3 : Réalisation

Introduction générale

Les progrès conjoints de la microélectronique, des technologies de transmission sans fil et des applications embarquées ont permis de produire à coût raisonnable des terminaux mobiles de haute technologie comme les Smartphones et les tablettes PC.

Dans ce contexte, un grand besoins de logiciels pour Smartphone se manifeste de plus en plus. Les utilisateurs de ces systèmes vaudront bien les exploiter au maximum de leurs performances à travers des applications qu'ils téléchargent sur leurs équipements.

Dans le cadre de notre mémoire, il nous a été demandé de faire la conception, le développement et L'intégration d'une application embarquée Android qui permet a son utilisateur de suivre son état de santé à partir de son Smartphone de n'importe ou et sans avoir a se connecter a un réseau.

Pour mener à bien notre travail, nous avons opté pour une démarche qui s'étale sur trois chapitres :

- **Chapitre I :** A pour objectif de voir sommairement une étude de l'univers d'Android.
- **Chapitre II :** dans ce chapitre nous nous intéressons aux différents points nécessaires à la conception et l'analyse de notre application
- **Chapitre III :** ce dernier chapitre est réservé à une description détaillé du processus de réalisation de notre application

Nous clôturerons ce modeste mémoire par une conclusion sur le travail accompli ainsi que ses perspectives.

Chapitre I: Android

I. Introduction:

Dans ce premier chapitre, nous présentons quelques systèmes d'exploitation mobiles, parmi lesquels se trouve le système d'exploitation Android, dont on va détailler l'architecture, les avantages, les différentes versions avec leurs fonctionnalités. Nous aborderons ensuite les différents composants d'une Application Android et on finira par l'introduction du SDK Android et ses outils.

II.1. Définition d'un Système Embarqué:

Un système embarqué dit aussi "Système Enfoui" est défini comme un système électronique et informatique autonome, souvent temps réel, spécialisé dans une tâche bien précise. Le terme désigne aussi bien le matériel informatique que le logiciel utilisé. Ses ressources sont généralement limitées. Cette limitation est généralement d'ordre spatial (encombrement réduit) et énergétique (consommation restreinte).Parmi les systèmes embarqués qui existent figurent les systèmes d'exploitation mobiles.

II.2. les différents systèmes d'exploitation mobiles :

Dans cette section, nous aborderons quelques plateformes existantes, du Palms OS à Android en passant par BlackBerry, Windows Mobile, I phone OS. Cela nous permettra d'avoir une idée assez générale des systèmes d'exploitation tournant sur mobiles, et d'essayer de faire un positionnement d'Android dans l'environnement des systèmes d'exploitation pour mobile.

II.2.1. Palms OS:

Palm OS aussi connu sous le nom de **Garnet OS(** figure 1 **)**, est un système d'exploitation embarqué et propriétaire permettant le fonctionnement d'appareils électroniques (téléphone, PDA etc..). Il a été implémenté sur une large gamme d'appareils électroniques mobiles, dont des smartphones, montres bracelet, consoles de jeux portables.

Figure I.1 : modèle de Palms OS

II.2.2. Windows Mobile :

Windows Phone(icône illustrée dans la figure2), le challenger du marché apparu en 2010, a su s'inspirer de ses concurrents et de son expérience passée dans le domaine mobile tout en innovant. Proposant une interface simple et épurée ainsi que de multiples possibilités de personnalisation, il permet d'arriver rapidement à l'exécution d'une tâche ce qui a particulièrement séduit les utilisateurs. C'est d'ailleurs Windows Phone qui se trouve être l'OS mobile le plus adapté aux utilisateurs.

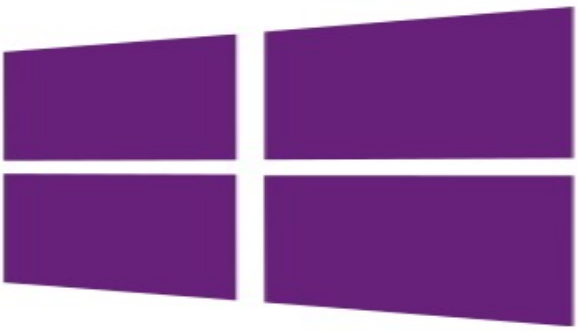


Figure I.2 : icone représentant Windows Mobile

II.2.3. iPhone OS:

iPhone OS d'Apple (icône illustrée dans la figure3), est le premier OS pour téléphones tactiles qui a véritablement lancé la vague des smartphones. Apparu sur le marché en 2007, il a innové dans un domaine qui n'était pas encore développé. Cependant, iOS n'a pas beaucoup évolué depuis sa création. Il se caractérise par une interface peu chargée, qui permet l'exécution de tâches rapidement pour les utilisateurs réguliers, mais qui laisse peu de place à la personnalisation.

Figure I.3 : icone représentant I phone

II.2.4. BlackBerry :

Tout comme l'iPhone, le BlackBerry est aussi un téléphone très en vue. Au départ clairement positionné sur le marché des entreprises, la fonction majeure qui a fait décoller le BlackBerry était le push mail. L'utilisateur n'a alors plus besoin de consulter périodiquement sa boîte pour vérifier s'il n'a pas de nouveaux messages. Ceux-ci lui parviennent directement comme un banal SMS. Cette fonctionnalité est assurée par les serveurs d'infrastructure du fabricant RIM (Research In Motion) avec un protocole propriétaire. Le mail est donc le point fort des BlackBerry qui faisaient rêver de nombreux cadres et dirigeants.

Figure I.4 : modèle de BlackBerry

II.3. Comparaison entre les différents Systèmes d'exploitation mobiles :

Ici nous proposons une comparaison entre les différents systèmes d'exploitation mobiles au niveau de leur popularité et ce en examinant les part de marché de chacun.

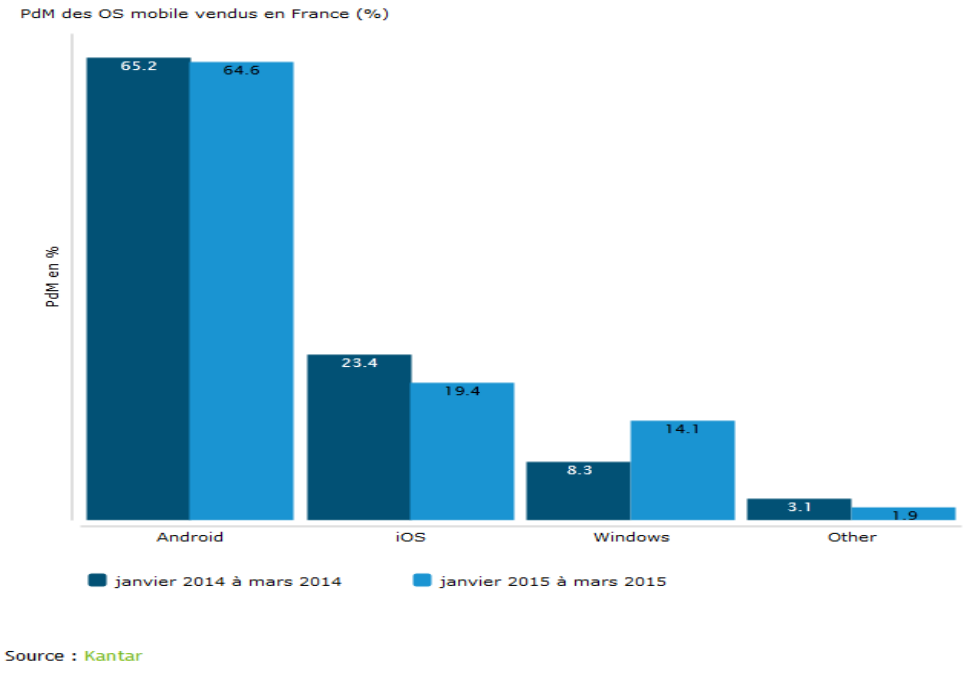


Figure I.5 : Parts du marché des OS mobile pour smartphones vendu en

l'étude **Kantar Worldpanel ComTech** sur les parts de marché des OS pour smartphones nous a livrer ses résultats pour les trois mois se terminant fin février de l'année 2014/2015. Selon les chiffres, en France, Android est toujours en tête le dominant du marché par rapport au autres IOS, Windows OS etc…

III. Le système d'exploitation Android:

Android, quant à lui, n'étant pas porté par un unique constructeur, ne cible pas spécifiquement les particuliers ou le monde professionnel. La plateforme est généraliste, on peut y développer aussi bien des jeux que des applications métiers.
Et vue que nous allons développer notre application sous Android intéressons nous a ce système plus en profondeur.

III.1. Description:

Android est un système d'exploitation pour Smartphones et tablettes tactiles conçu par google 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 acheté 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 un majeur distributeur, Android est une plateforme puissante, moderne, sécurisée et ouverte. Basé sur le noyau Linux et utilisant la plateforme java pour ses applications. Il est entièrement gratuit et sa plateforme très flexible ce qui permet aux développeurs d'intégrer, d'agrandir et d'adapter les applications aux besoins du client ou les remplacer entièrement, l'utilisateur peut donc personnaliser facilement son appareil.

III.2. Open Handset Alliance:

L'Open Handset Alliance (abrégé OHA) est un consortium Regroupant de grands constructeurs et développeurs de logiciels dont le but est de développer des normes ouvertes pour les appareils de téléphonie mobile. Ce consortium a été créé le 5 novembre 2007. Le premier standard annoncé a été Android, une plateforme pour appareils mobiles basée sur un kernel linux 2.6.

Figure I. 6: Le logo de l'OHA

III.3. Bugdroide:

Le personnage nommé Bugdroide est le petit robot vert utilisé par Google pour présenter Android.
Le site Engadget annonce que Bugdroide, le logo d'Android, serait en fait un personnage d'un jeu des années 1990 sur Atari : Gauntlet : The Third Encounter.

Figure I.7: Le logo d'Android

III.4. Historique des versions d'Android:

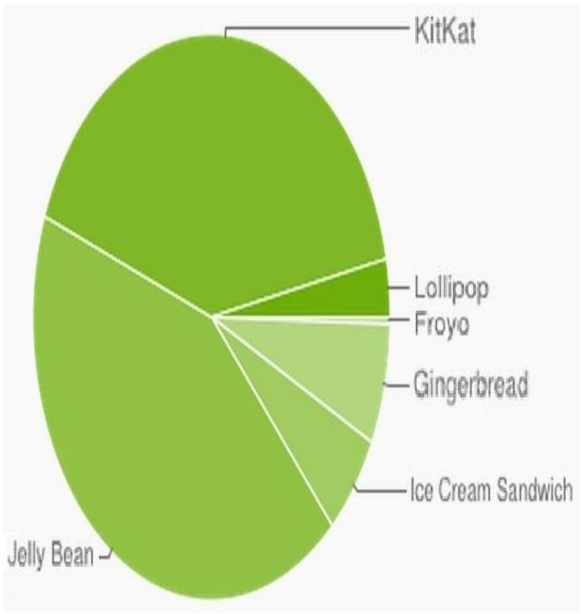
Figure I.8 : évolution des versions d'Android

Android a débuté avec la sortie de la version 1.0 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 nom de codes suivent une logique alphabétique, en voici quelques unes :

- 1.0 : Apple Pie (Tarte aux pommes) ou Alpha, Sdk distribué fin 2007.
- 1.5 : Cupcake (Petit Gâteau), sortie en avril 2009, dernière révision officielle en mai 2010.
- 2.3 (2.3.7) : Gingerbread (Pain d'épice), sortie le 6 décembre 2010, dernière version dédiée uniquement aux Smartphones. Cette version est parfois utilisée sur de petites tablettes.
- 4.2: (Aussi appelée Jelly Bean) introduit photo sphère permettant une prise des photos à 360°etc…..

Pour le moment la pluralité des versions d'Android ne s'amenuise pas étant donné qu'il existe encore cinq versions en circulation sans compter la petite dernière : Android Lollipop, nous avons donc Froyo, Gingerbread, Ice Cream Sandwich, Jelly Bean et KitKat.

Voici un graphique illustrant la part de chaque version d'Android en mars 2015:



Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	6.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	5.9%
4.1.x	Jelly Bean	16	17.3%
4.2.x		17	19.4%
4.3		18	5.9%
4.4	KitKat	19	40.9%
5.0	Lollipop	21	3.3%

Figure I.9 : Répartition des versions d'Android

On remarque que la version KitKat (4.4) est la plus répandue avec plus de 40%, une autre version aussi importante Jelly Been (4.2.x) avec un taux de 19.4% , on a aussi la version IceCream Sandwich(4.0.3) avec 5.9% et c'est justement sur elle que notre choix s'est porté.

III.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 Google Calendrier ou encore la cartographie Google Maps.

Voici quelques fonctionnalités proposées par Android classées par ordre de version et d'API**:**

Évolution d'Android :

Version	Nom	API Level	Fonctionnalités
Android 1.0		1	▪ Fonctionnalités de base (téléphonie, SMS, réseaux de données, accès au répertoire, …etc.)
Android 1.1		2	▪ corrections de bugs (alarme, Gmail...) ▪ Amélioration des fonctionnalités MMS. ▪ Introduction des applications payantes sur l'Android Market
Android 1.5	Cupcake (Coupe gâteau)	3	▪ Nouveau clavier avec auto complétion ▪ Support Bluetooth ▪ Enregistrement de vidéos ▪ Widgets ▪ Animations améliorées
Android 1.6	Donut (Beignet)	4	▪ Recherche vocale améliorée ,Indicateur d'utilisation de la batterie ▪ Apparition de fonctionnalités pour les réseaux privés virtuels (VPN)
Android 2.0	Eclair (sorte de gâteau à la crème)	5	▪ Interface graphique améliorée ▪ Gestion des contacts changée ▪ Support Bluetooth 2.1 ▪ Clavier virtuel amélioré

Android 2.0.1	Eclair	6	▪ Réduction de la consommation de la batterie ▪ Quelques améliorations graphiques comme l'écran de verrouillage
Android 2.1	Eclair	7	▪ Nouveau bureau à 5 écrans virtuels ▪ Fonds d'écran animé ▪ Nouveaux effets 3D (notamment sur la gestion des photos) ▪ Amélioration du client Gmail
Android 2.2	Froyo (yaourt glacé)	8	▪ Nouveaux Widgets
Android 2.3	Gingerbread (pain d'épice)	9	▪ Amélioration d'interface graphique ▪ Nouveau clavier virtuel
Android 3.0	Honeycomb (rayon de miel)	10	▪ Optimisé pour les tablettes et équipement à écran large ▪ Multitâche et système de notification amélioré

Android 4.0	Ice cream Sandwich (Sandwich à la crème glacée)	11	▪ Nouvelle interface graphique ▪ Amélioration de la sécurité ▪ Beaucoup de raccourcis (Appareil photo, accès sdcard)
Android 4.1/4.3	Jelly Bean	16/17/18	▪ Nouvelles fonctionnalités pour les notifications ▪ Nouvelle application pour l'appareil photo ▪ Intégration de l'assistant google
Android 4.4.x	KitKat	19/20	▪ Recherche vocale plus rapide ▪ Affichage web et Chrome ▪ Nouvelle interface pour les applications Téléchargement et E-mail
Android 5.0/5.1	Lollipop	21/22	▪ Refonte totale de l'interface graphique avec un nouveau design nommé "Material Design" ▪ Nouveau moteur d'exécution ART(Android RunTime) qui compile les applications dés leur installation plutôt que la compilation JIT(just-In-Time) de Dalvik ▪ Elle permet d'optimiser la consommation d'énergie .

Figure I.10 : Fonctionnalités des versions d'Android

III.6. Différents équipements utilisant Android:

Le succès d'Android est indéniable, les fournisseurs l'ont bien compris, d'où l'apparition sur le marché de plus en plus d'appareils utilisant ce système d'exploitation:

- **Les Smartphones** :

  Un smartphone, ou téléphone intelligent, est un téléphone mobile disposant d'un grand écran tactile, fonctionnant sur des réseaux haut débit, et offrant de nombreuses fonctionnalités, dont la consultation du courrier électronique et la localisation par GPS. Le premier mobile commercialisé sous Android est le HTC Dream/G1 produit par la firme Taïwanaise HTC, lancé aux États-Unis le 22 octobre 2008.

  Depuis le lancement en 2007 de l'iPhone d'Apple, la progression de leurs ventes est spectaculaire et elles ont dépassé en 2012 celles des téléphones mobiles classiques.

- **Les Tablettes :**

  En septembre 2010, Samsung présente le Samsung Galaxy Tab, tournant sous Android 2.2 (FroYo). Android a été porté aussi sur d'autres appareils comme la HP TouchPad.

- **Les télévisions :**

  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.

- **Audio radios :**

  La société française Parrot a dévoilé en 2011 le premier autoradio tournant sous Android : la Parrot Asteroid. Cet autoradio offre notamment un adaptateur GPS, des ports USB et une connectivité Bluetooth pour contrôler la musique de son téléphone mobile.

- **Consoles de jeux vidéo :**

  une console de jeux vidéo portable sous Android sous le nom de MyPlay est commercialisé depuis septembre 2012.

III.7. Avantages et inconvénients de Google Android

Relativement, nouveau sur le marché de la technologie, le système d'exploitation Android de Google à progresser et est devenu compétitif. Mais ce système d'exploitation est-il fiable ? Quel sont les avantages et les inconvénients de Google Android ?

III.7.1. Les avantages de Google Android:

- Plusieurs applications peuvent être lancées à la fois (écoutez de la musique tout en surfant sur le net).
- Notifications dans le temps des SMS, messagerie de Gmail… par le clignotement d'un indicateur.
- Accessibilité depuis Android Market à des milliers d'autres applications téléchargeables gratuitement (certaines sont payantes).
- Installations de gadgets et raccourcis sur l'écran d'accueil pour accéder rapidement au menu et divers paramètres ou applications.
- De constantes mises à jour sont proposées, ce qui montre une amélioration du système.

III.7.2. Les inconvénients de Google Android:

- Vous êtes connecté en "mode continu" et donc en permanence.
- Les téléphones Android possèdent de faible autonomie. Ils se déchargent vite à cause de la de la connexion.

- Possibilité de chauffage de votre téléphone.
- Certaines applications sont incompatibles à des versions de Google Android.
- Bug de certaines applications.
- Des applications inutiles sont installées, ce qui entraine sur certains modèles une insuffisance d'espace.

Entre les avantages et les inconvénients, il revient à évaluer le côté positif de ce système d'exploitation afin d'en profiter pleinement.

III.8. Présentation de la plateforme Android:

Cette partie présente Android sous un autre angle que celui de l'utilisateur, qui est celui d'une vue de l'intérieur, en explorant les aspects techniques internes. Exploration nécessaire pour tout futur développement de la plateforme.

III.8.1. Architecture d'Android:

Le système d'exploitation Android est un empilement de composants logiciels qui est grossièrement divisé en cinq sections et quatre couches principales, comme illustré dans le schéma suivant :

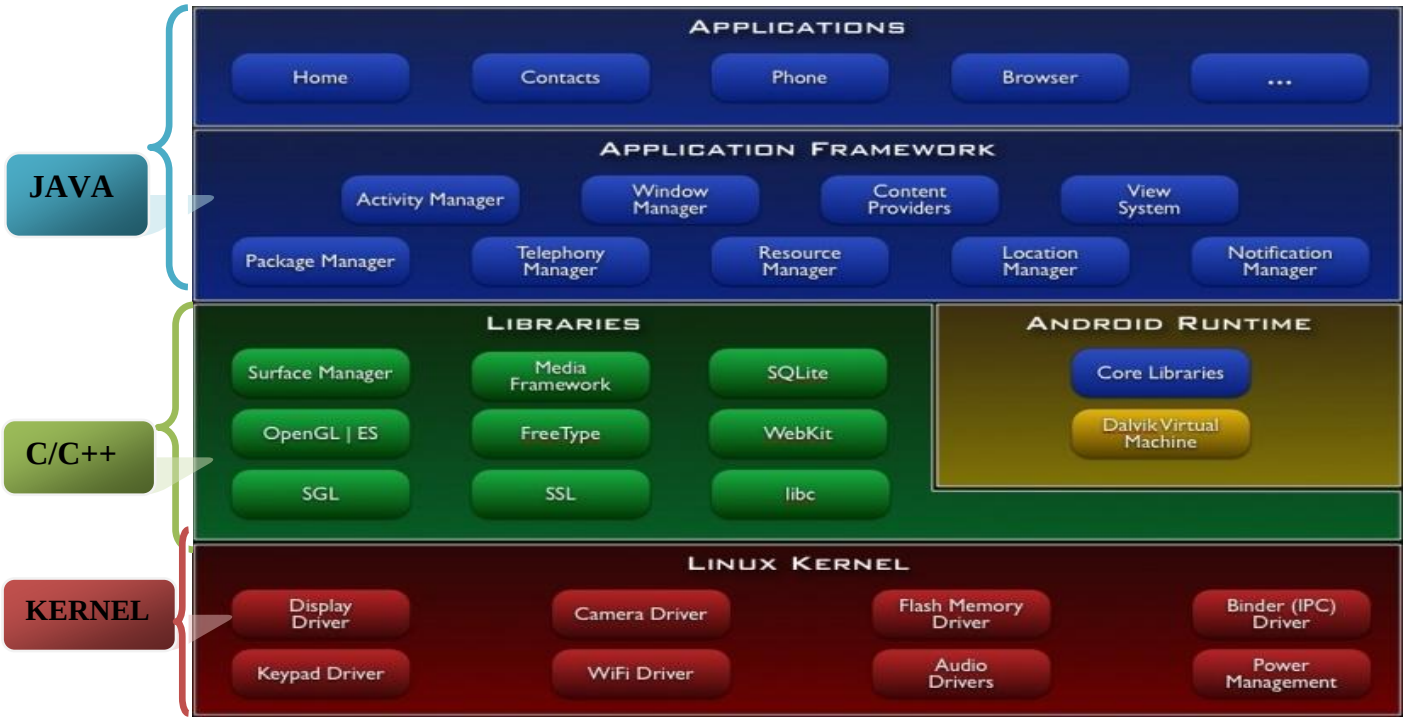


Figure I.11: architecture d'un système Android

On peut y observer toute une pile de composants qui constituent le système d'exploitation. Le sens de lecture se fait de bas en haut, puisque le composant de plus bas niveau (le plus éloigné des utilisateurs) est le noyau Linux et celui de plus haut niveau (le plus proche des utilisateurs) est constitué par les applications.
Voyons de plus prés chacun de ces niveaux:

- **1er niveau : Linux Kernel (noyau):**

Android se base sur le noyau Linux 2.6 pour les services systèmes de base tels que la sécurité, la gestion de la mémoire, la gestion des processus, ce qui permet une meilleure gestion des caractéristiques des appareils mobiles.
Le noyau agit également comme une couche d'abstraction entre le matériel et le reste de la pile logicielle

- **2 ème Niveau : Bibliothèques et environnement :**

Au dessus du kernel, il y a les librairies ainsi que le moteur d'exécution Android Runtime.

1) 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.

2) Le moteur d'exécution d'Android (Android Runtime) :

L'environnement d'exécution d'Android est la machine virtuelle Dalvik avec ses propres bibliothèques adaptées aux équipements mobiles.
Dalvik est une machine virtuelle, c'est-à-dire des émulateurs, qui permettent aux applications de tourner sur des appareils, indépendamment des différences matérielles. En d'autres termes, Dalvik permet de faire tourner les applications sur différents smartphones Android, quel que soit le modèle. Dalvik a été développé notamment pour permettre aux appareils peu puissants de faire tourner plusieurs applications simultanément.
Dans la machine virtuelle Dalvik, les programmes sont écrits en Java et compilés avec les outils de java pour obtenir un byte code qu'on recompile encore avec un outil spécifique

dex[(2)] pour obtenir un code adapté à la machine Dalvik.

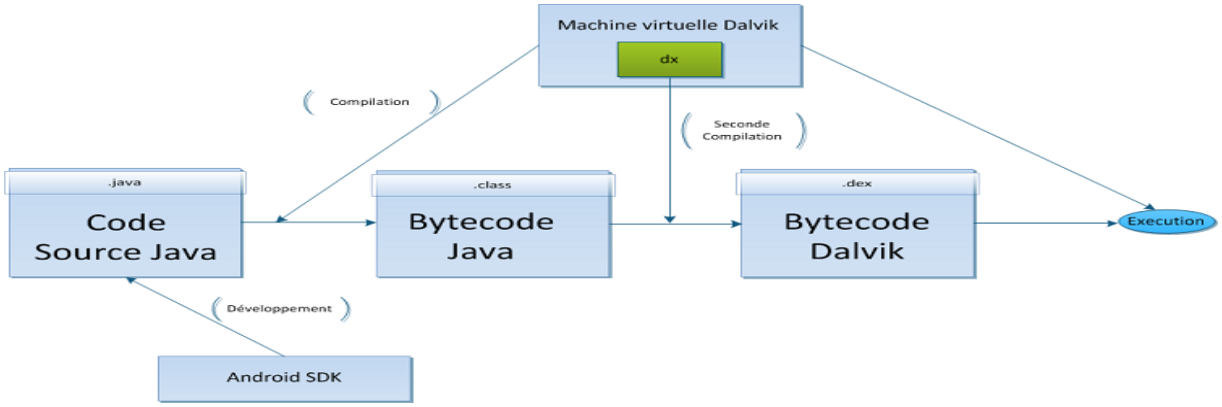


Figure I.12 : Compilation de machine virtuelle Dalvik

Sur Android, par défaut chaque application tourne dans son propre processus Linux, c'est-à-dire sa propre VM. L'un des points forts d'Android étant ses capacités multitâches. À chaque fois qu'une application est démarrée, une VM se crée, puis le processus meurt à la fermeture du programme.

- **3eme Niveau : développement de l'application (Application Framework) :**

Le Framework est situé au dessus de l'Android Runtime et des librairies. Il fournit des API (Application Programming Interface) permettant aux développeurs de créer des applications riches.

- **4eme Niveau : Application:**

Le système Android peut exécuter un grand nombre d'applications qui marchent sous la plate forme Android incluant les clients de messagerie, les programmes de gestions d'SMS, gestionnaires de rendez-vous, les contacts, la caméra, calculatrice, navigateurs, …etc. Toutes ces applications sont écrites en Java, et elle sont présentes sous le format d'un archive **APK** (Android Package File), qu'on peut installer sur les appareils mobiles.

(2)- DEX : C'est un Compilateur Android qui transforme le bytecode java en code Dalvik pour optimiser les ressources comme CPU, RAM, et pour l'application portable

III.8.2. Les Composants d'une Application Android:

Une application Android est un assemblage de composants liés grâce à un fichier de configuration. Nous allons présenter chacun des composants en détails, et nous préciserons comment le fichier de configuration de l'application la décrit et comment toutes les pièces interagissent entre elles.

Avant de rentrer dans le détail d'une application Android et de son projet associé, différents concepts fondamentaux sont à préciser :

- Les activités (présentées ultérieurement)
- Les vues et contrôles (et leur mise en page)
- Les ressources
- Le fichier de configuration appelé également manifeste.

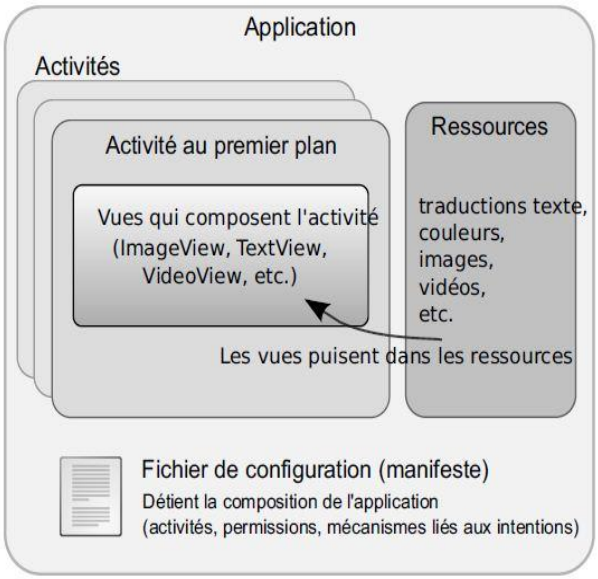


Figure I.13 : Composants d'une application Android

- **Les vues :** sont les éléments de l'interface graphique que l'utilisateur voit et sur lesquels il pourra agir. Les vues contiennent des composants, organisés selon diverses mises en page (les uns à la suite des autres, en grille, etc.).

- **Les contrôles :** (boutons, champs de saisie, case à cocher, etc.) sont eux-mêmes un sous-ensemble des vues. Ils ont besoin d'accéder aux textes et aux images qu'ils affichent. Ces textes et ces images seront puisés dans les fichiers "ressources" de l'application.
- **Les ressources** : sont les éléments visuels, notamment les images, les gabarits (layouts), des variables stockées... exploitables par l'application.

À côté de ces éléments se trouve un fichier XML le Manifest : le **fichier de configuration** de l'application. C'est un fichier indispensable à chaque application qui décrit entre autres :

- Le point d'entrée de votre application (quel code doit être exécuté au démarrage de l'application)
- Quels composants constituent ce programme
- Les permissions nécessaires à l'exécution du programme (accès à Internet, accès à l'appareil photo, etc.).

Comme on vient de voir, une application Android est composée de plusieurs éléments qu'il faut assembler pour obtenir un tout cohérent. Plus une application est complexe, plus le nombre de pièces utilisées sera grand. Ces pièces sont divisées en deux catégories :

➢ Composants applicatifs : activité, service, fournisseur de contenu et gadgets

- **L'activité** (présentée ultérieurement dans le chapitre).
- **Le service** est un composant qui fonctionne en tâche de fond, de manière invisible. Ses principales utilisations sont la mise à jour de sources de donné et le déclenchement de notifications.
- **Le fournisseur de contenu** permet de gérer et de partager des informations. Un même fournisseur permet d'accéder à des données au sein d'une application et entre applications.
- **Le gadget** est un composant graphique qui s'installe sur le bureau Android. Le calendrier qui affiche de l'information ou la météo qui affiche les détails de la journée sont deux exemples de gadgets que l'on trouve souvent sur un écran d'accueil.

➢ Composants d'interaction : Intents, récepteurs, notifications

Les éléments suivants permettent l'interaction entre les différents composants du système, entre les applications installées sur l'appareil ou avec l'utilisateur.

- **L'Intent :** Les Intents sont des objets permettant de faire passer des messages contenant de l'information entre composants principaux. La notion d'Intent peut être vue comme une demande de démarrage d'un autre composant, d'une action à effectuer. La raison d'être des Intents provient du modèle de sécurité d'Android. Grâce aux Intents, les applications ont la possibilité de fournir leurs services ou données si elles le souhaitent. Nous les utiliserons dans notre application pour dialoguer à l'intérieur de celle-ci pour:

  - Naviguer entre les activités.
  - Surveiller les clicks sur les boutons.

Trois composants d'application sont lancés par les Intents : Les activités, Les services, Les receveurs de diffusion.

- **Récepteur d'Intents :** il permet à une application d'être à l'écoute des autres afin de répondre aux objets Intent qui lui sont destinés et qui sont envoyés par d'autres composants applicatifs.
- **Notification :** une notification signale une information à l'utilisateur sans interrompre ses actions en cours.

III.8.3. Cycle de vie d'une application (gestion des processus)

Les applications Android ont un fonctionnement particulier : elles réagissent à des changements d'état imposés par le système (démarrage, pause, reprise, arrêt, etc.).
Néanmoins elles n'ont aucun contrôle direct sur leur propre cycle de vie. En cas de besoin, le système peut mettre en pause ou alors complètement arrêter une activité s'il juge nécessaire de le faire, par exemple si l'application consomme trop de ressource processeur ou mémoire. Ce comportement est dit non déterministe. Chaque application fonctionne dans son propre processus. Le système Android est responsable de la création et de la destruction des processus et gère ses ressources avec comme objectif la sécurité mais aussi la disponibilité et la réactivité de l'appareil. Ainsi, un processus consommant beaucoup de ressource processeur, ne pourra entraver d'autres fonctionnalités, comme la réponse à un appel entrant. Ceci implique qu'un processus peut être tué à n'importe quel moment sans son consentement pour libérer des ressources nécessaires à d'autres applications.

Les activités étant le bloc de base d'une application. Elles sont tellement indispensables, qu'il est tout simplement impossible de construire notre application sans passer par ses activités. Voyons tous cela en profondeur :

III.8.3.1. Les Activités:

Une activité correspond à la partie présentation de l'application et fonctionne par le biais des vues qui affichent des interfaces graphiques et répondent aux actions utilisateurs. Elle peut être assimilée à un écran qu'une application propose à son utilisateur. Pour chaque écran d'application, on doit donc créer une activité. La transition entre deux écrans correspond au lancement d'une activité par les intents ou au retour sur une activité placée en arrière-plan.

Une activité est composée de deux volets :

- La logique de l'activité et la gestion du cycle de vie de l'activité qui sont implémentées en Java dans une classe héritant de "Activity".
- L'interface utilisateur, qui pourra être définie soit dans le code de l'activité soit de façon plus générale dans un fichier XML placé dans les ressources de l'application.

```java
import android.app.Activity;
import android.os.Bundle;

public class ActiviteSimple extends Activity {
    /**
    * Méthode appelée à la création de l'activité
    * @param savedInstanceState permet de restaurer l'état
    * de l'interface utilisateur
    */

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }
}
```

Figure I.14 : Squelette minimal pour créer une activité

III.8.3.2. Cycle de vie d'une activité

Tout ce que nous avons vu en parlant du cycle de vie d'une application, notamment sur la gestion des processus en fonction des ressources, a un impact direct sur les activités et notamment sur leur cycle de vie.

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

Active (Resumed) :

L'activité est visible en totalité, elle est sur le dessus de la pile, c'est elle qui détient le focus (en exécution). C'est ce que l'utilisateur consulte en un moment donnée et il peut l'utiliser dans son intégralité et agir directement dessus.

Suspendue (Posed) :

L'activité en pause est partiellement visible à l'écran en arrière plan. 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, ou lorsque une notification s'affiche. Ce n'est pas sur cette activité qu'agit l'utilisateur. L'application n'a plus le focus, c'est l'application sus-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.
L'activité en pause est encore en vie, elle est encore en mémoire et rattaché au gestionnaire des fenêtre du système Android. Elle peut être tuée par le système Android en cas de manque sévère de ressources mémoires.

Arrêtée (Stopped) :

L'activité est tout simplement oblitérée par une autre activité, on ne peut plus la voir du tout.
L'application n'a évidemment plus le focus, et puisque l'utilisateur ne peut pas la voir, il ne peut pas agir dessus. Le système retient son état pour pouvoir reprendre, mais il peut arriver que le système tue l'application pour libérer de la mémoire système.

Ci-dessous, le diagramme qui représente ces principaux états et les transitions y menant :

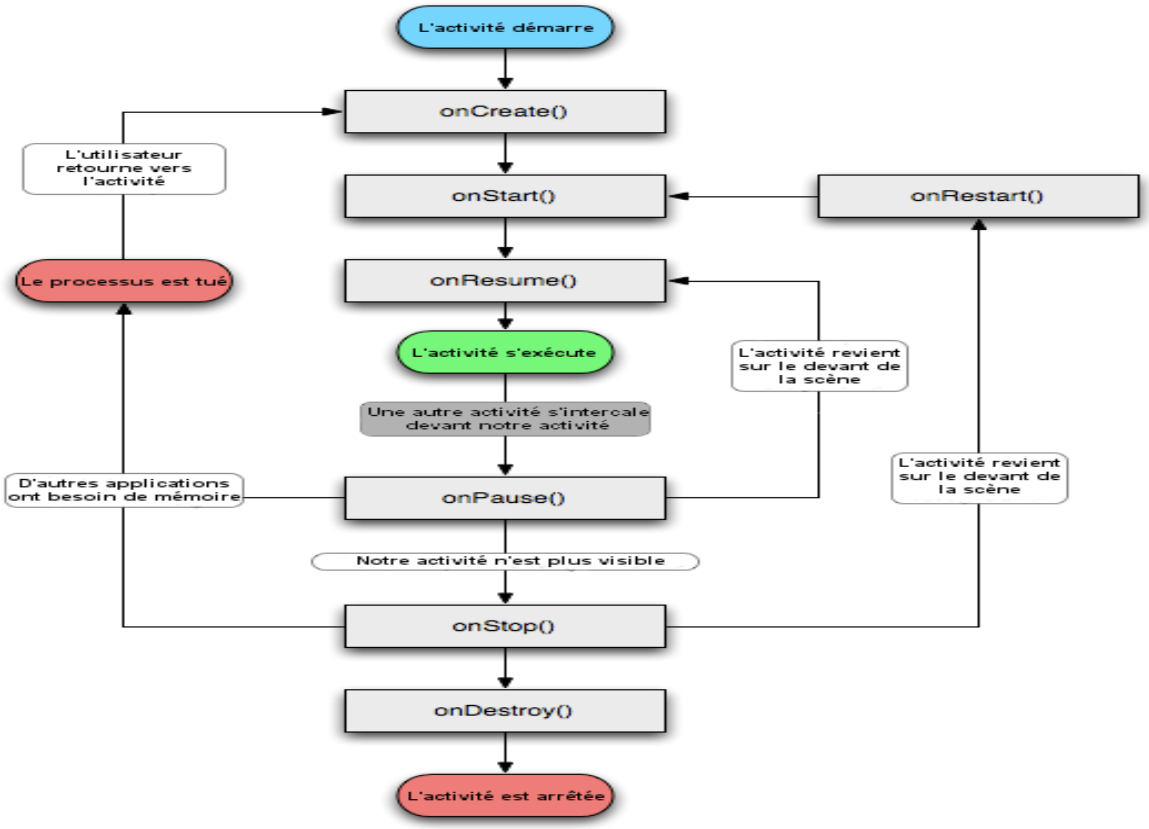


Figure I.15 : cycle de vie d'une activité

Le cycle de vie d'une activité est parsemé d'appels aux méthodes relatives à chaque étape de sa vie.

- **onCreate() :** Cette méthode sera exécutée à la création de l'activité par le système (avant son démarrage). C'est une méthode d'initialisation des vues, des paramètres et d'autres données.
- **onStart() :** l'activité va devenir visible Cette méthode sera exécutée lors du démarrage de l'activité.
- **onRestart() :** appelée à un nouveau démarrage de la même activité (quand l'activité était arrêtée).
- **onPause() :** méthode qui sert à arrêter une activité temporairement.
- **onResume() :** l'activité est maintenant visible, Cette méthode sera exécutée lorsque l'activité résume son exécution après la suspension (pause) et que l'activité commence à interagir avec l'utilisateur.

- onStop() : cette fonction est utilisée quand l'activité n'est plus visible à l'utilisateur. Elle est cachée soit à cause d'une nouvelle activité lancée, soit parce que l'activité en cours s'apprête à être détruite.

- onDestroy(): cette méthode est invoquée quand l'activité est détruite. La destruction opère quand quelqu'un appelle la méthode finish() ou quand c'est le système qui décide de tuer l'activité pour économiser de l'espace.

IV. SDK Android (Software Développent Kit):

SDK signifie **S**oftware **D**evelopment **K**it, c'est un ensemble d'outils d'aide à la programmation pour concevoir des logiciels, jeux, applications mobiles, pour un terminal .

un SDK contient du code, permettant de concevoir une interface ou une partie d'une interface numérique (web, mobile, jeux, logiciels de recherches, widget météo…). Ce code est conçu avec le langage de programmation correspondant au terminal (ordinateur, téléphone, tablette…) et au système de navigation ciblés.
Par exemple, pour développer une application mobile (logiciel) pour iPhone (terminal mobile avec système d'exploitation iOS), il faut utiliser le SDK iOS mobile. Ce code est organisé sous forme de librairies (ou bibliothèques logicielles) c'est-à-dire des collections de fonctions prédéfinies, de points d'accès à du matériel et à des fonctionnalités système d'un terminal. Le SDK Android permet donc de développer des applications pour Android et uniquement pour Android.

IV. 1. les outils que fournit le SDK:

En parallèle de ces librairies, de nombreuses ressources peuvent accompagner un SDK voici quelques une :

- **L'ADB (Android Debug Bridg) :**

ADB est un outil disponible en ligne de commande polyvalent. Il permet de communiquer avec une instance de l'émulateur directement ou avec un téléphone physique à l'aide d'un câble USB standard.
Il donne la possibilité d'installer ou de désinstaller des applications, d'envoyer ou de récupérer des fichiers et de se connecter en ligne de commande sur l'appareil. Il est aussi utilisé par le SDK pour publier et débugger des applications.

- **Le DDMS (Dalvik Debug Monitor Server):**.

La vue DDMS ouvre une perspective sur un ensemble d'interfaces permettant de suivre l'activité de l'émulateur : détail des tâches et de la pile des applications, explorateur de fichier, liste des applications qui s'exécutant dans l'émulateur et console d'administration de l'émulateur (simulation d'appels, d'envoi de SMS, etc.).
La perspective DDMS propose également la vue **LogCat** qui agit comme un véritable filtre d'entrées de journal (erreur, avertissement, information et débogage) pour les applications.

V. Concepts et innovations :

Android possède aussi un environnement très riche, ainsi que des fonctionnalités tout aussi intéressantes, voici quelques unes auxquelles nous feront appel dans la réalisation de notre projet :

- **Le bureau virtuel :**

Le bureau virtuel est l'écran d'accueil du Smartphone, il est composé de 5 parties (ou plus). Chacune est personnalisable, il est possible d'y mettre des raccourcis (vers des applications, des fichiers, des dossiers, des contacts) ou des widgets (calendrier, horloge, notes, …etc.).
L'image de fond s'étend aussi sur tout le bureau, et bouge dés qu'on passe sur une partie à une autre ce qui donne l'impression que le contenu fait partie du décor. Cependant et dans le but de faire la différence, les constructeurs ajoute leurs touches personnelles par exemple : le constructeur HTC possédant l'interface Sence qui est dotée de 7 bureaux.

Figure I.16: le bureau virtuel

❖ **Les notifications :**

Une notification est une indication qui s'affiche sur la barre qui se situe en haut d'un téléphone Android. Cette notification sert à prévenir un utilisateur de certains évènements, comme la réception d'un message ou d'un appel manqué, le niveau de batterie, ainsi que les connectivités: réseau, Bluetooth, GPS …etc.

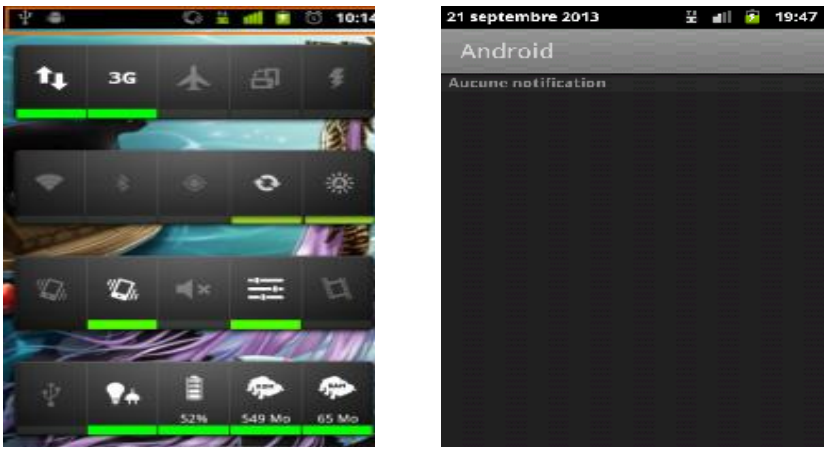


Figure I.17 : Exemple de notification

❖ **SQLite :**

Android ne fournit aucune base de données de son propre chef. C'est donc au développeur que revient la tâche d'en créer un type SQLite.

SQLite est une bibliothèque écrite en C, et comme son nom l'indique, utilise un dialecte de SQL pour effectuer des requêtes, des manipulations de données, et des définitions de données. Il est si efficace en termes de mémoire que le moteur d'exécution d'Android qui peut l'inclure dans son intégralité.

❖ L'AppWidget :

Une des forces d'Android est son côté personnalisable. Un des exemples les plus probants est qu'il est tout à fait possible de gérer les éléments qui se trouvent sur l'écran d'accueil.

Les AppWidgets font partie de ses éléments, ce sont des applications miniatures destinées à être utilisées dans d'autres applications. Ils permettent d'améliorer une application à peu de frais en lui ajoutant un compagnon permanent. De plus, mettre un AppWidget sur l'écran d'accueil permet à l'utilisateur de se rappeler de l'existence de l'application principale et par conséquent d'y accéder plus régulièrement.

Dans cette grande famille de Widget, on peut en distinguer deux types :

- **Les widgets natifs :** sont Ceux qui sont installés nativement avec le téléphone, comme votre barre de recherche Google, l'horloge, la météo (sur HTC par exemple)..
- **Les widgets non natifs** : Ces widgets sont placés lors de l'installation de nouvelles applications rajoutés par l'utilisateur.

Android donne la possibilité à tous les utilisateurs de personnaliser leurs bureaux, en y disposant des widgets sur les différents bureaux.

❖ Android Market

Le Play Store (anciennement Android Market) est une plateforme immense et très visitée qui permet d'acheter et d'installer des applications Android (jeux vidéo et utilitaires) mais aussi de louer ou acheter des films, des séries TV, de la musique, des livres. C'est donc une mine d'opportunités pour quiconque qui peut y accéder par un accès web classique ou le plus souvent directement à partir du smartphone.

Figure I.18 : icône représentant le android market

VI. Conclusion:

Dans ce chapitre nous avons présenté ce que on appelle Android , le systéme dans sa jeunesse. De ce fait nous avons conclu que cette Plateforme est un système d'exploitation riche et complet. Il s'est développé tellement vite en quelques années qu'il est devenu l'OS le plus utilisé pour les smartphone dans le monde avec plus d'un milliard d'utilisateurs. Dans le prochain chapitre, Nous allons passer à la modélisation de notre application puis nous détaillerons les étapes de la conception afin de d'en assurer une utilisation pratique et surtout intuitive.

I. Introduction:

Afin d'aboutir à une meilleure organisation et une bonne maîtrise du travail et donc arriver à déployer de meilleures applications, il est nécessaire de suivre une démarche méthodologique rigoureuse. Pour cela le choix d'une méthode d'analyse et de conception est d'une très grande importance. notre choix s'est posé sur la modélisation UML.
Dans ce chapitre nous allons donc expliquer le processus de développement et de conception de notre application, pour ce faire notre démarche va se dérouler en deux étapes cruciales :

- **Étape 1 :** Mettre en évidence les différents acteurs intervenants au sein de celui-ci, ainsi que de leurs besoins en illustrant les différents cas d'utilisation de l'application.
- **Étape 2 :** c'est la dernière étape de la démarche , la conception de l'application.

II. Objectif de l'Application:

L'objectif principal de l'application est d'offrir aux utilisateurs un suivi quotidien de leurs états de santé et ce n'importe où et à n'importe quel heure. Disposant d'une liste de maladies avec les traitement correspondant afin que l'utilisateur puisse enregistrer son état , surveiller ses progrès au fil de sa vie(garder trace de toute les maladies atteintes et leurs médicaments).
A l'image de la plupart des applications proposées désormais, l'application que nous cherchons à développer disposera d'une alarme réveil pour se rappeler de l'heure de prise du médicament en question.

III. Processus de développement:

III.1. Première étape : Analyse:

L'analyse est la partie étude du système c.à.d. le recueil et l'organisation des besoins auxquels le système doit répondre, en d'autres termes, l'analyse c'est la recherche de toutes les connaissances et informations pour définir ce que doit faire le système. Et afin de formaliser les besoins, UML met à notre disposition différents diagrammes qui permettent de représenter de manière simple les grandes fonctionnalités du système, dans notre cas, et pour cette partie on va élaborer un diagramme de cas d'utilisation général et des descriptions textuelles pour certains cas d'utilisations.

III.1.1. Présentation de l'UML :

Définition:

UML, c'est l'acronyme anglais pour « Unified Modeling Language ». On le traduit par «Langage de modélisation unifié ». La notation UML est un langage visuel constitué d'un ensemble de schémas, appelés des diagrammes, qui donnent chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour représenter le logiciel à développer : son fonctionnement, sa mise en route, les actions susceptibles d'être effectuées par le logiciel, etc… Réaliser ces diagrammes revient donc à modéliser les besoins du logiciel à développer.

Cela dit, on peut tout à fait s'en servir pour décrire de futures applications, sans pour autant déjà être fixé sur le type de développement. C'est exactement ce à quoi UML sert dans des projets de réalisation de logiciels !

III.1. 2. Les acteurs du systéme :

On emploi le terme d'acteur pour toute entité externe qui interagit avec le système en d'autre terme un acteur est l'idéalisation d'un rôle joué par une personne externe.

pour notre cas d'étude on distingue un seul acteur le voici :

Utilisateur :

toute personne disposant d'un Smartphone Android version 4.0.3 et plus.

C'est lui qui doit lancer l'application, choisir une activité à faire dans le menu principal.

III.1.3. Un cas d'utilisation :

Un cas d'utilisation est une unité cohérente d'une fonctionnalité visible de l'extérieur. Il réalise un service de bout en bout, avec un déclenchement, un déroulement et une fin, pour l'acteur qui l'initie. Un cas d'utilisation modélise donc un service rendu par le système, sans imposer le mode de réalisation de ce service. Elle permet aussi de représenter l'ensemble des besoins et des exigences que notre application doit respecter.

Dans notre cas nous distinguons quelques cas d'utilisation dont on va élaborer un cas d'utilisation générale.

III.1. 4. Diagramme de cas d'utilisation général :

Un diagramme de cas d'utilisation général capture le comportement d'un système, d'un sous-système, tel qu'un utilisateur extérieur le voit. Il scinde la fonctionnalité du système en unités cohérentes, les cas d'utilisation ayant un sens pour les acteurs. Les cas d'utilisation permettent d'exprimer le besoin des utilisateurs d'un système, ils sont donc une vision orientée utilisateur de ce besoin au contraire d'une vision informatique.

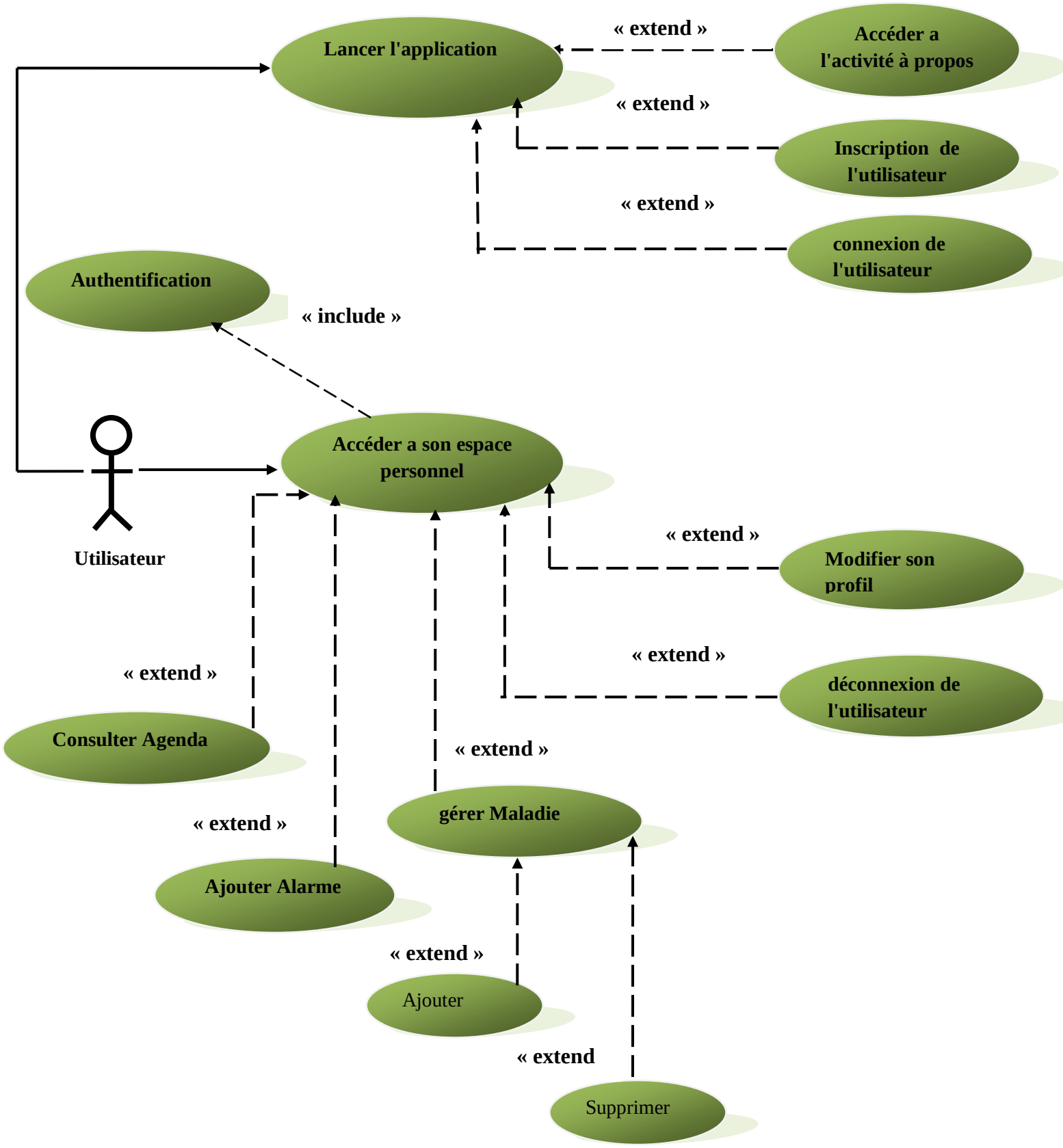


Figure II.1 : Diagramme de cas d'utilisation général

III.1. 5. Description textuelle de quelques cas d'utilisation:

Le diagramme de cas d'utilisation décrit les grandes fonctions d'un système du point de vue des acteurs, mais n'expose pas de façon détaillée le dialogue entre les acteurs et les cas d'utilisation. Bien que de nombreux diagrammes UML permettent de décrire un cas, il est recommandé de rédiger une description textuelle car c'est une forme souple qui convient pour toute les situations.

Dans le cadre de notre travail, nous avons choisi d'étudier et de modéliser les cas d'utilisation les plus importants dont voici la description textuelle.

1) Inscription de l'utilisateur :

cas d'utilisation : inscription de l'utilisateur .

acteurs : Utilisateur

Description :

1- L'utilisateur lance l'application en cliquant sur l'icône.

2- Le système affiche l'activité principale (l'accueil).

3- L'utilisateur clique sur le bouton « inscription ».

4- Le système affiche l'activité contenant le formulaire d'inscription.

5- L'utilisateur remplit les champs du formulaire d'inscription et clique sur le bouton «valider ».

6- Le système lui affiche une autre activité lui proposant de s'authentifier pour accéder à son espace .

2) Identification :

acteurs : Utilisateur

cas d'utilisation : identification de l'utilisateur.

Description :

1- L'utilisateur atteint l'activité principale de l'application ensuite il clique sur le bouton « connexion ».
2- Le système répond en affichant le formulaire d'authentification.
3- L'utilisateur remplit le formulaire en saisissant son login et son mot de passe et clique sur le bouton « connexion ».
4- Le système charge l'espace approprié à l'utilisateur.

3) Ajouter une maladie :

acteurs : Utilisateur

cas d'utilisation : Ajouter une maladie

Description :

1- Après authentification l'utilisateur atteint son espace personnel.
2- l'utilisateur clique sur le bouton « maladie».
3- Le système lui affiche l'activité maladie qui contient le bouton «ajouter».
4- l'utilisateur appuis sur le bouton « ajouter».
5- Le système lui affiche l'activité correspondante , celle qui contient un formulaire d'ajout.
6- l'utilisateur remplit les champs et clique sur « valider».

4) Programmer une alarme pour médicament:

acteurs : Utilisateur

cas d'utilisation :Programmer une alarme.

Description :

1- L'utilisateur atteint l'activité principale de l'application ensuite il clique sur le bouton « connexion » pour s'authentifier.

2- Une fois l'authentification s'est achevé Le système android charge l'espace approprié de l'utilisateur.

3- L'utilisateur clique sur le bouton « Alarme».dans le menus.

4- Le système affiche l'activité correspondante qui contient deux boutons « R.D.V » « Médicament ».qui désigne tout simplement la nature de l'alarme.

5- L'utilisateur clique par exemple sur le bouton « Médicament ».

6- Le systéme android affiche un formulaire associé au médicament (traitement, durée, heure).

7- L'utilisateur remplit les champs et clique sur le bouton « Valider».

5) Décaler un RDV :

acteurs : Utilisateur

cas d'utilisation : Décaler un RDV(Rendez-vous).

Description :

1- A près authentification l'utilisateur atteint son espace personnel.

2- L'utilisateur clique sur le bouton « Agenda » se trouvant dans le menus principal.

3- Le systéme affiche l'activité qui lui est associé contenant ces deux boutons « RDV » et «Médicament».

4- L'utilisateur appuis sur le bouton « RDV ».

5- Le système affiche une autre activité qui contient le bouton « Décaler RDV».

6- L'utilisateur appuis sur le bouton « Décaler RDV».

7- Le système affiche une fenêtre qui permet d'envoyer un message contenant plusieurs icones parmi eux ilya SMS, Email, etc..

8- L'utilisateur clique sur l'icône SMS par exemple.

9- Le système affiche une autre fenêtre proposant de saisir le message.

10- L'utilisateur rédige le message et clique sur « envoyer ».

c'est la même procédure pour envoyer un émail .

III.2. Deuxième étape : Conception:

La conception est la dernière étape avant la réalisation technique de la plate forme de l'application, elle consiste donc à réaliser le modèle qui va être implémenté, un modèle ou les contenus sont classés et hiérarchisés. on apporte plus de détails à la solution et on cherche à clarifier des aspects techniques,

Tout comme l'étape d'analyse , UML utilise différents diagrammes pour la conception, et dans notre cas on va élaborer des diagrammes de séquence ainsi que les diagrammes de classes pour les cas d'utilisation étudié et un diagramme de classe générale.

III.2.1. Diagrammes de Séquences:

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre.

- **Diagramme de séquence de conception du cas d'utilisation « Inscription de l'Utilisateur ».**

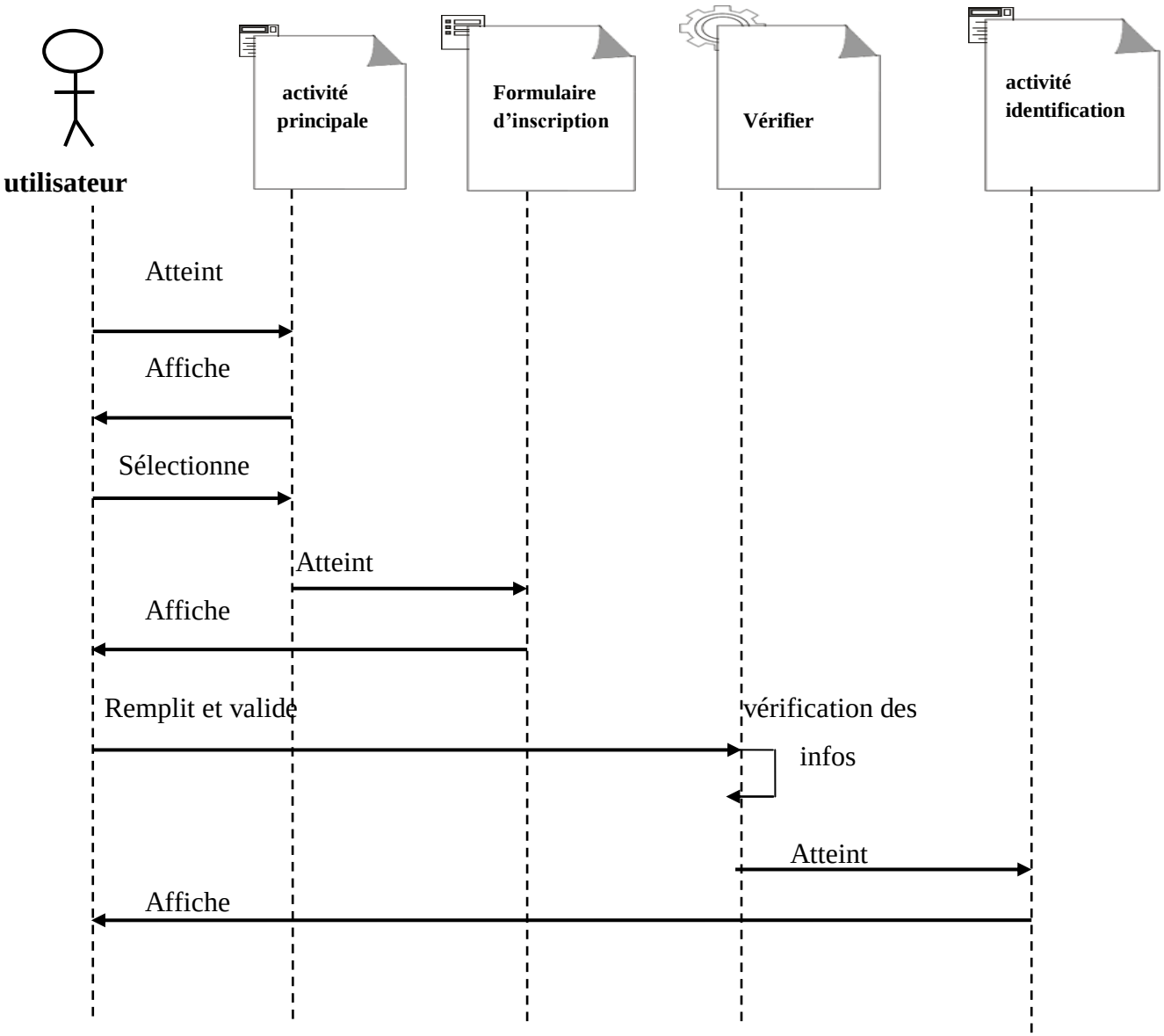


Figure II. 2 : Diagramme de séquence du cas d'utilisation «Inscription de l'utilisateur »

- **Diagramme de séquence de conception du cas d'utilisation « Identification ».**

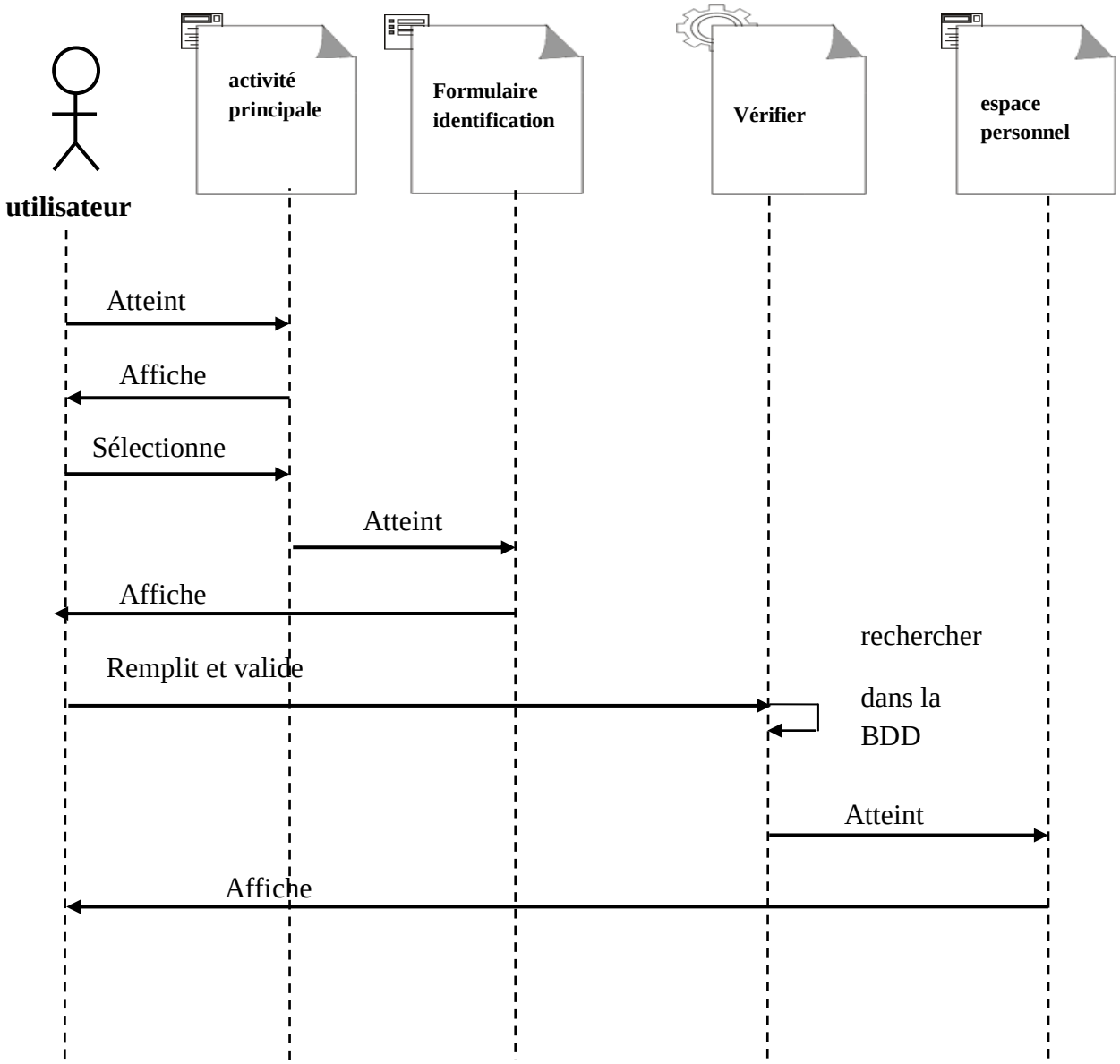


Figure II. 3: Diagramme de séquence du cas d'utilisation «Identification »

- **Diagramme de séquence de conception du cas d'utilisation «Ajouter maladies»**

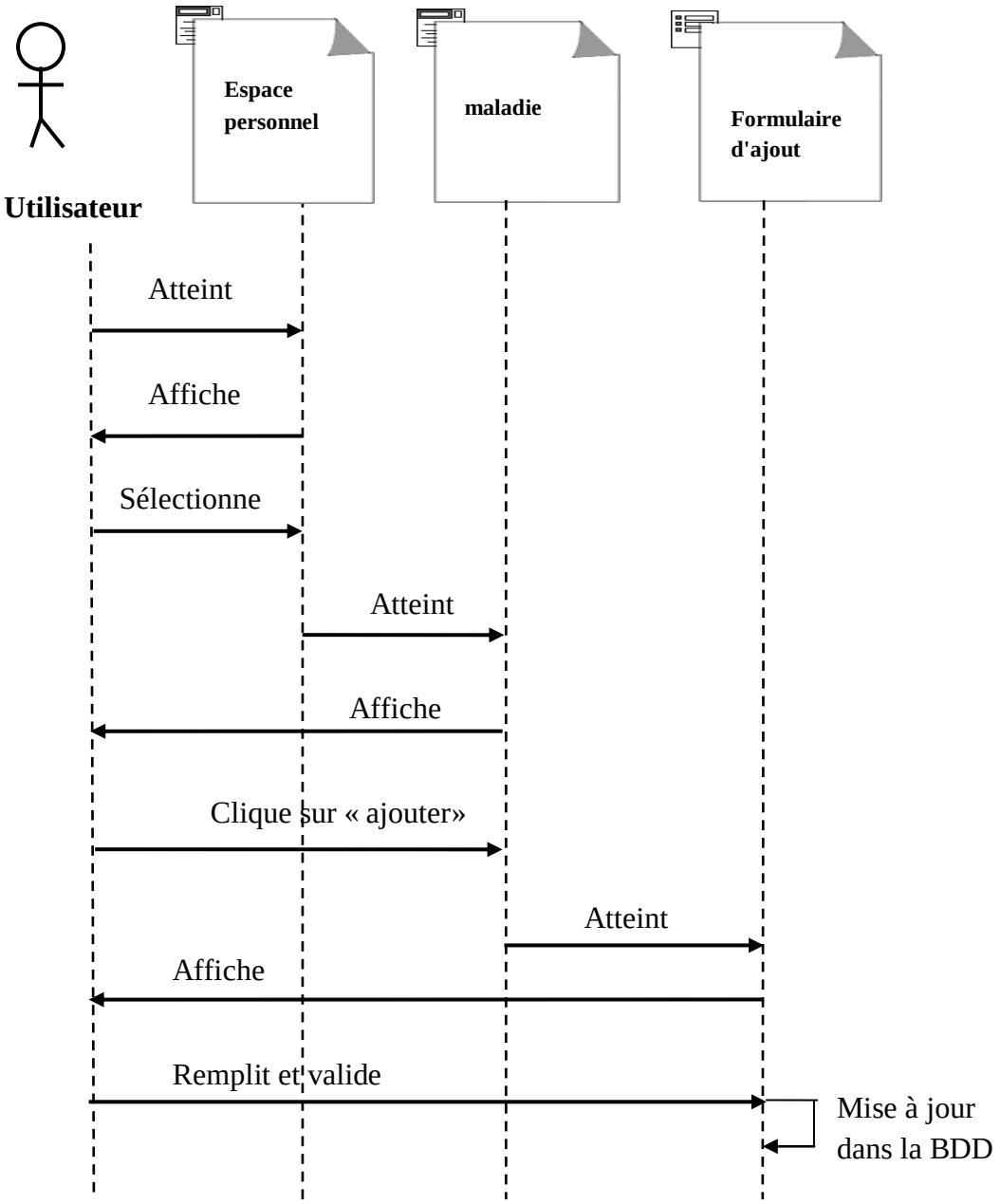


Figure II.4: Diagramme de séquence du cas d'utilisation «Ajouter une maladie»

- **Diagramme de séquence de conception du cas d'utilisation «programmer une alarme pour médicament ».**

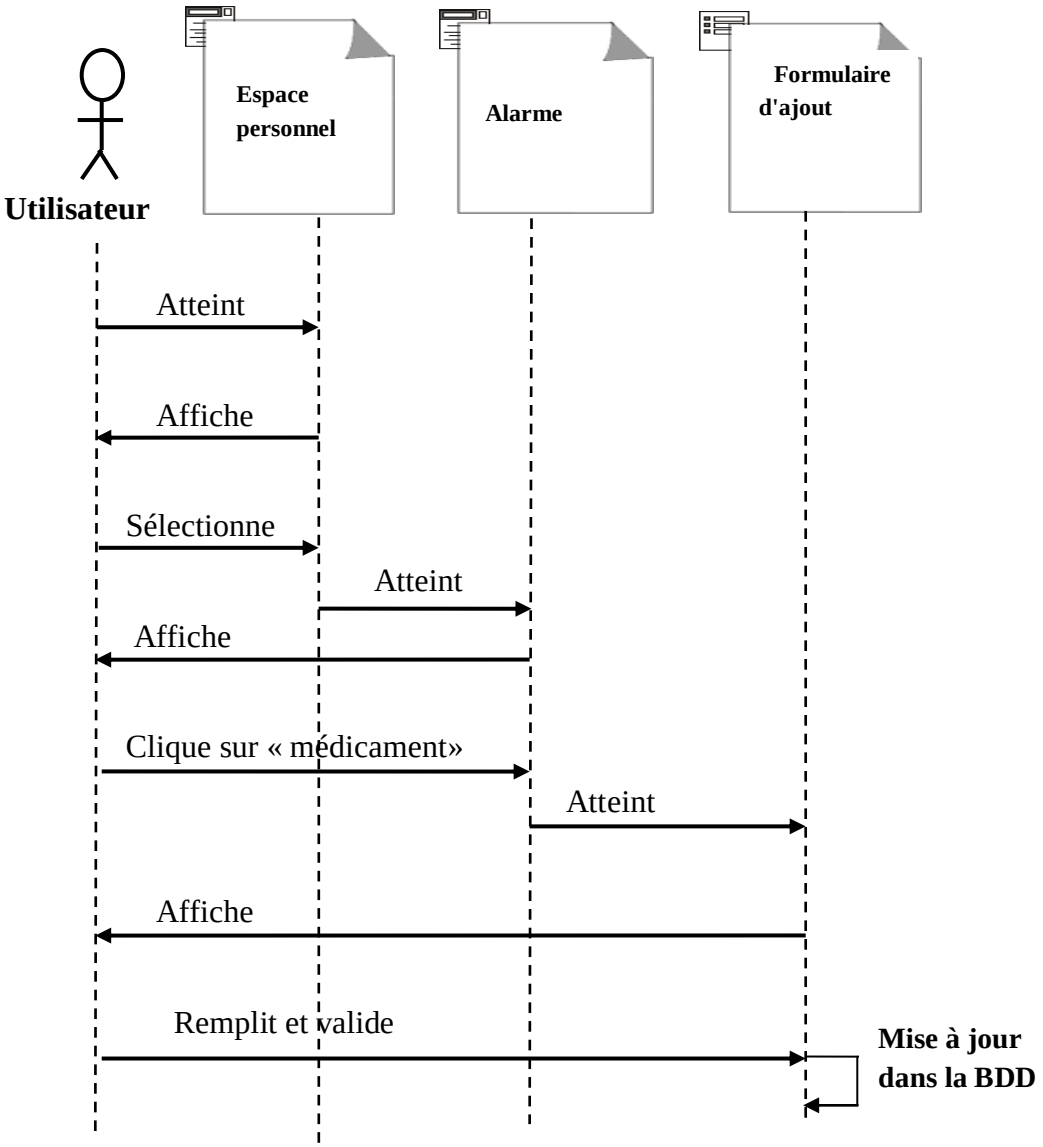


Figure II.5: Diagramme de séquence du cas d'utilisation « programmer une alarme pour médicament ».

- **Diagramme de séquence de conception du cas d'utilisation «Décaler RDV».**

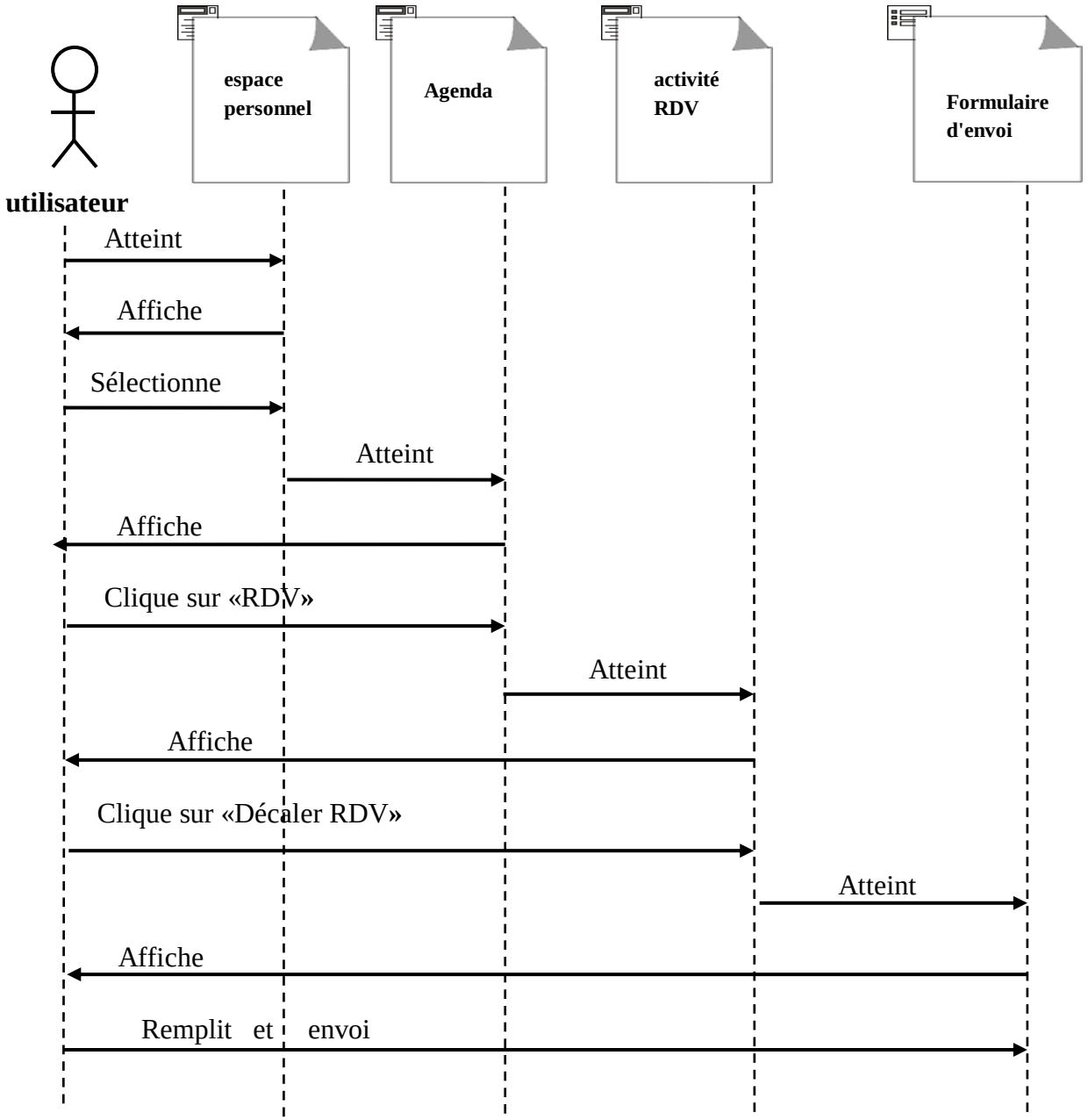


Figure II. 6: Diagramme de séquence du cas d'utilisation «Décaler RDV»

III.3.2. Diagrammes de Classes (Class Diagram):

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il 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. Il permet de fournir une représentation abstraite des objets du système qui vont interagir ensemble pour réaliser les cas d'utilisation.
Il s'agit donc d'une vue purement statique car on ne tient pas compte du facteur temporel dans le comportement du système. Le diagramme de classes modélise les concepts du domaine d'application ainsi que les concepts internes créés de toutes pièces dans le cadre de l'implémentation d'une application. Il permet aussi de modéliser les classes du système et leurs relations indépendamment d'un langage de programmation particulier.

voici quelques digrammes de classes que nous avons élaborer:

- **Diagramme de classe du cas d'utilisation « Identification » :**

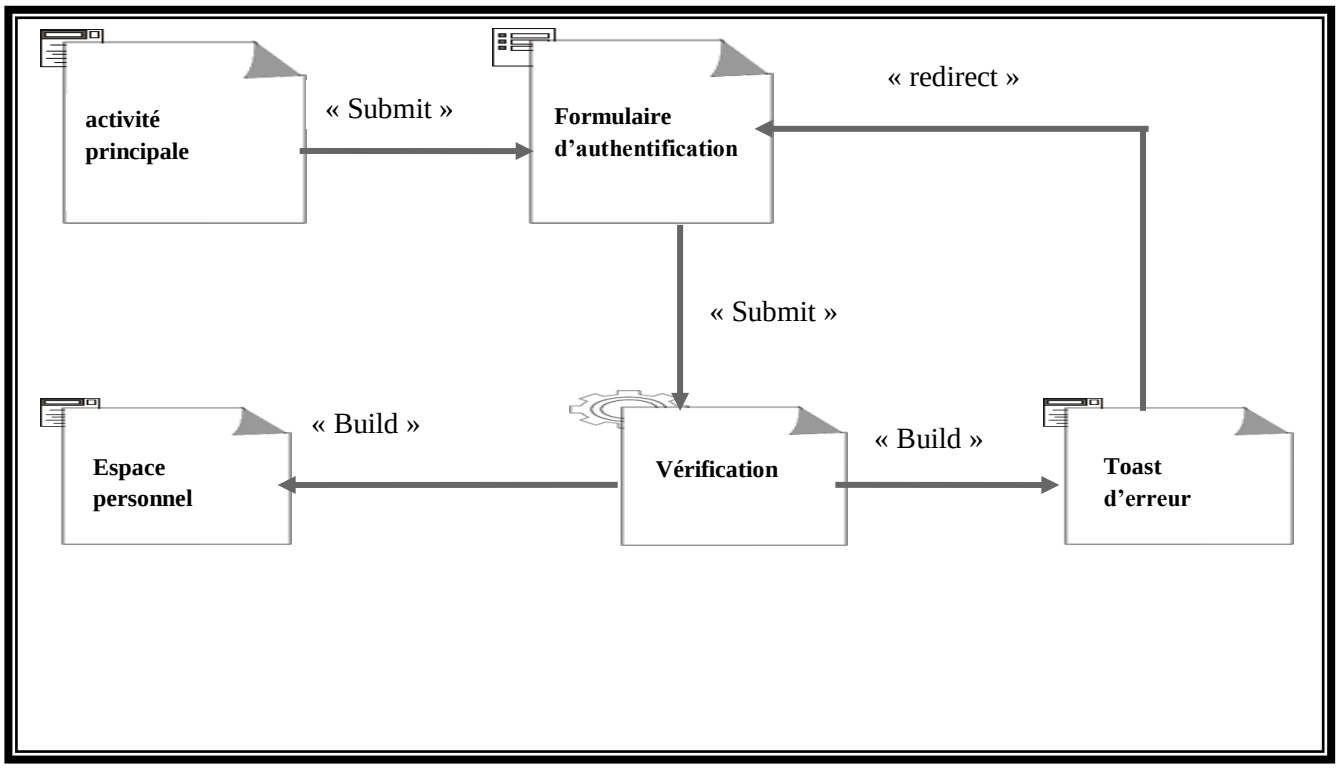


Figure II.7: Diagramme de classe du cas d'utilisation « Identification »

- **Diagramme de classe du cas d'utilisation « Inscription de l'utilisateur »**

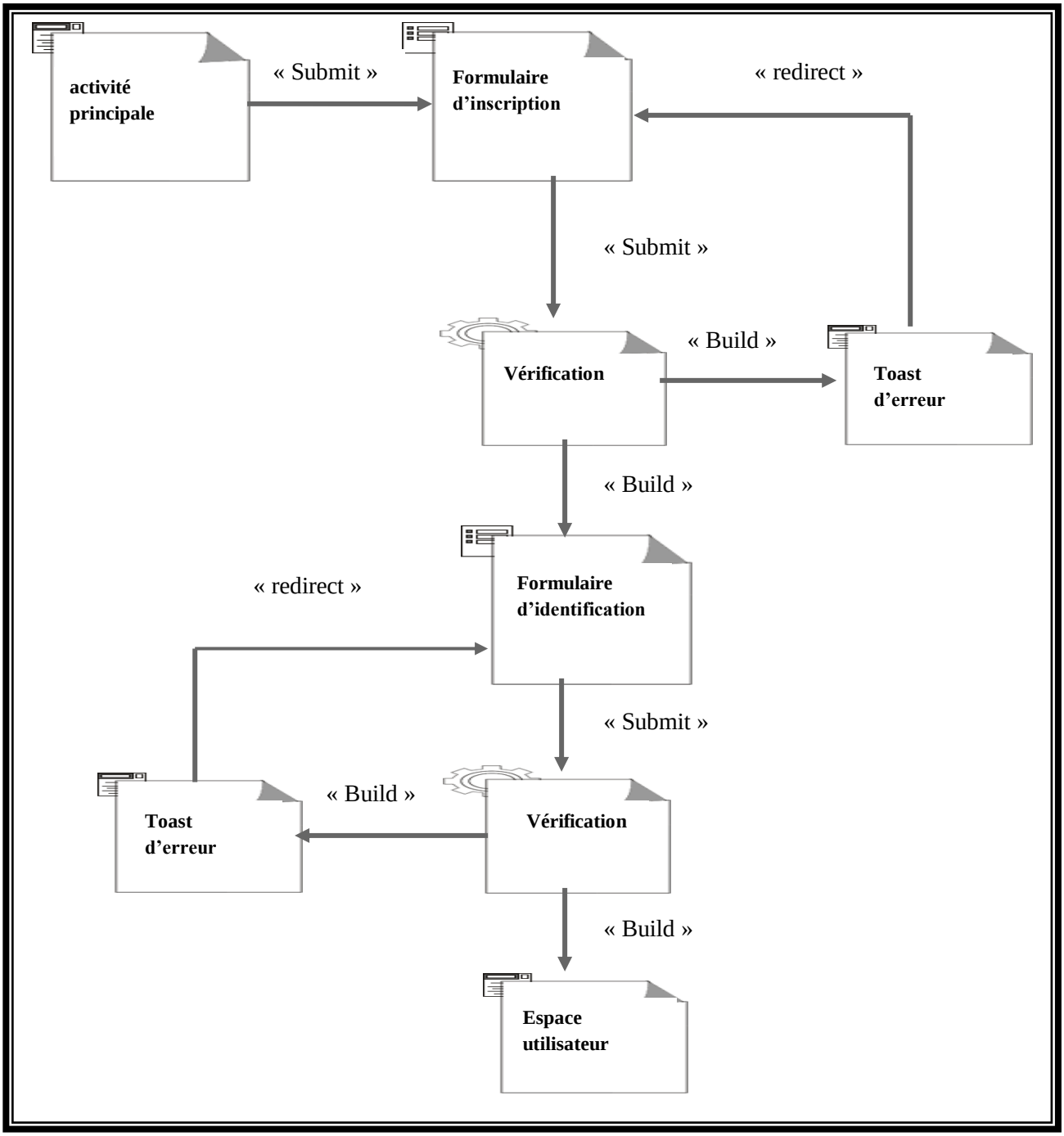


Figure II.8 : Diagramme de classe du cas d'utilisation « Inscription de l'utilisateur »

- **Diagramme de classe du cas d'utilisation « Ajouter maladie »**

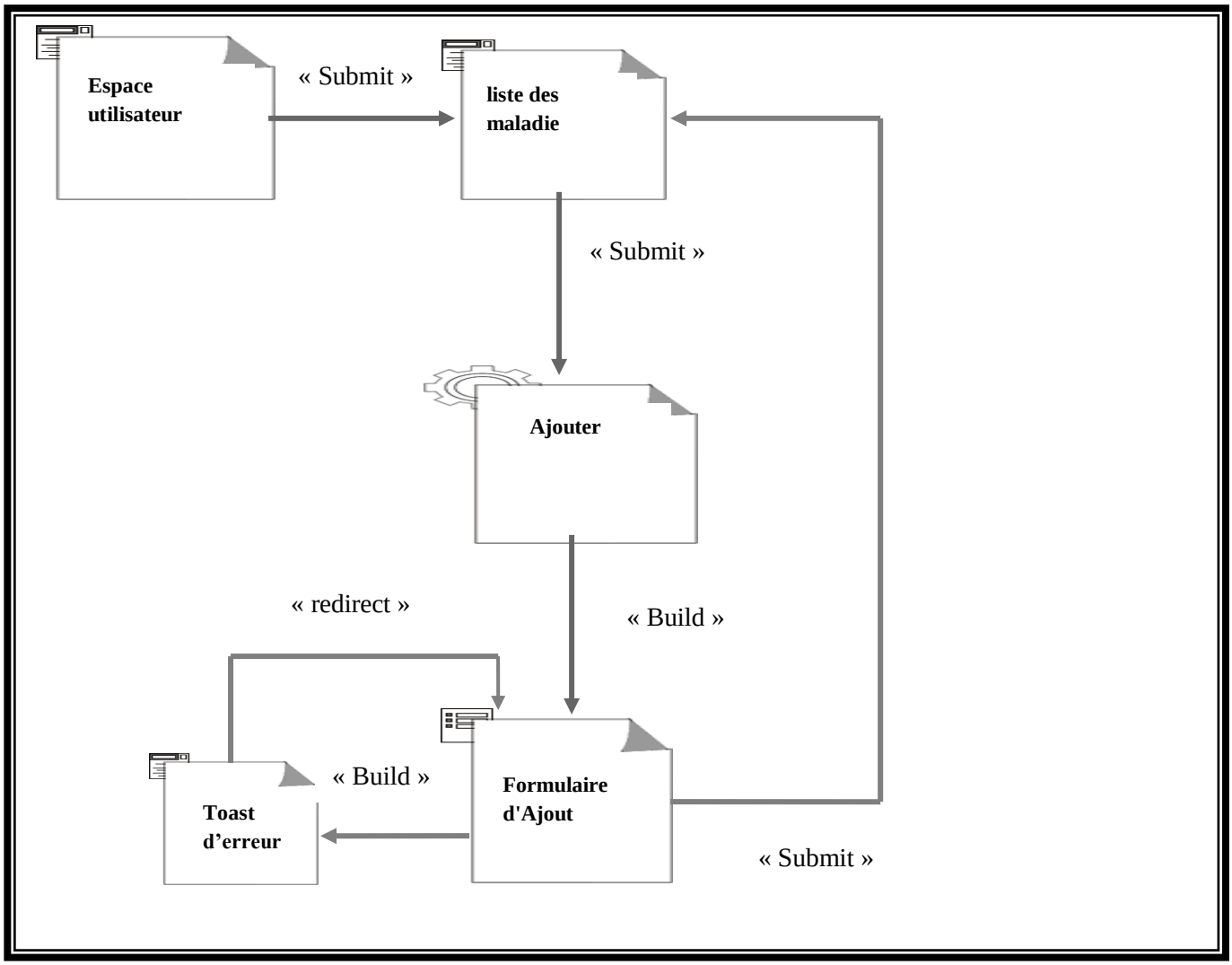


Figure II.9: Diagramme de classe du cas d'utilisation « Ajouter maladie»

- **Diagramme de classe du cas d'utilisation « Décaler RDV »**

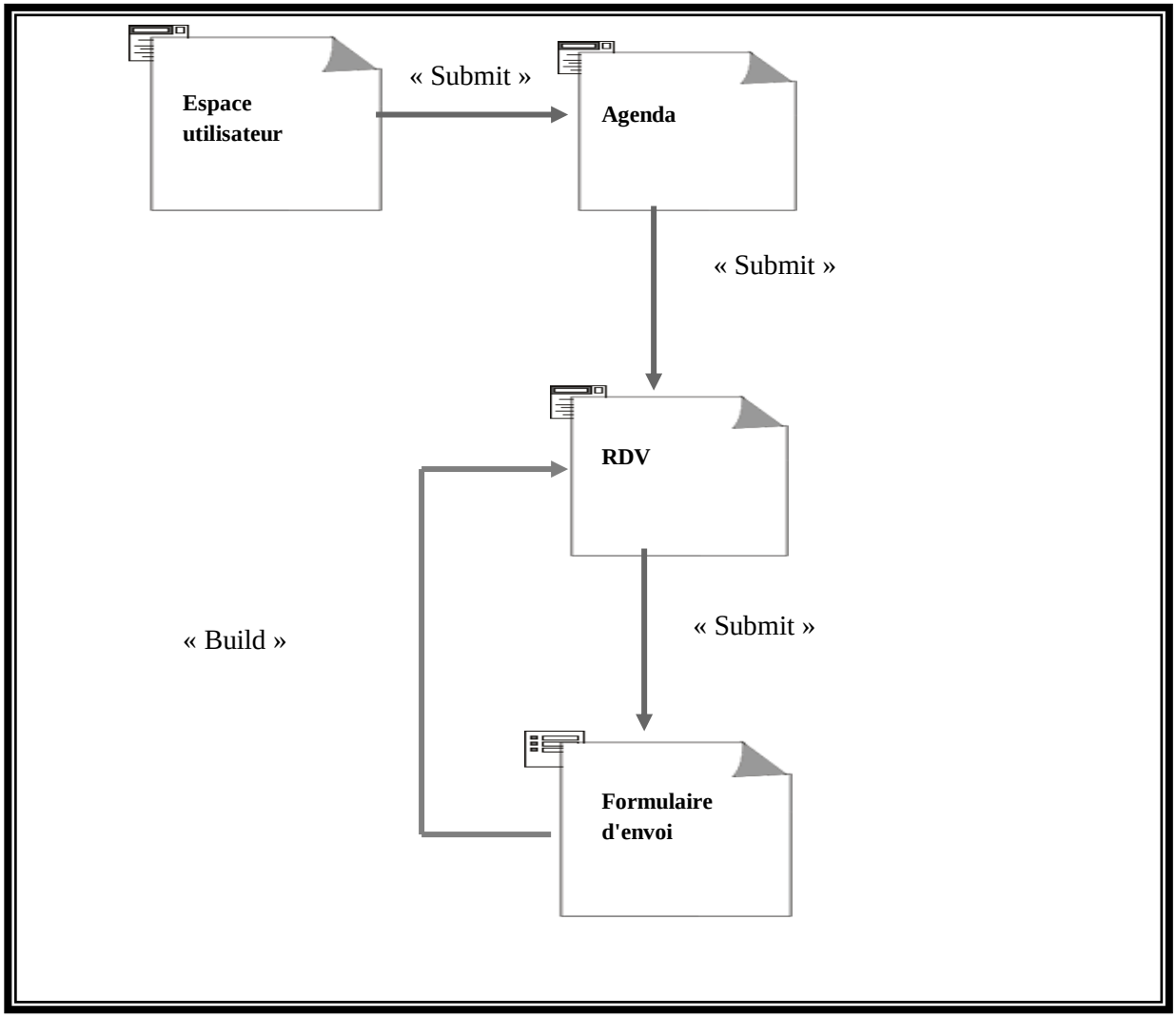


Figure II.10: Diagramme de classe du cas d'utilisation «Décaler un RDV»

III.3.3. Diagrammes de Classe Générale:

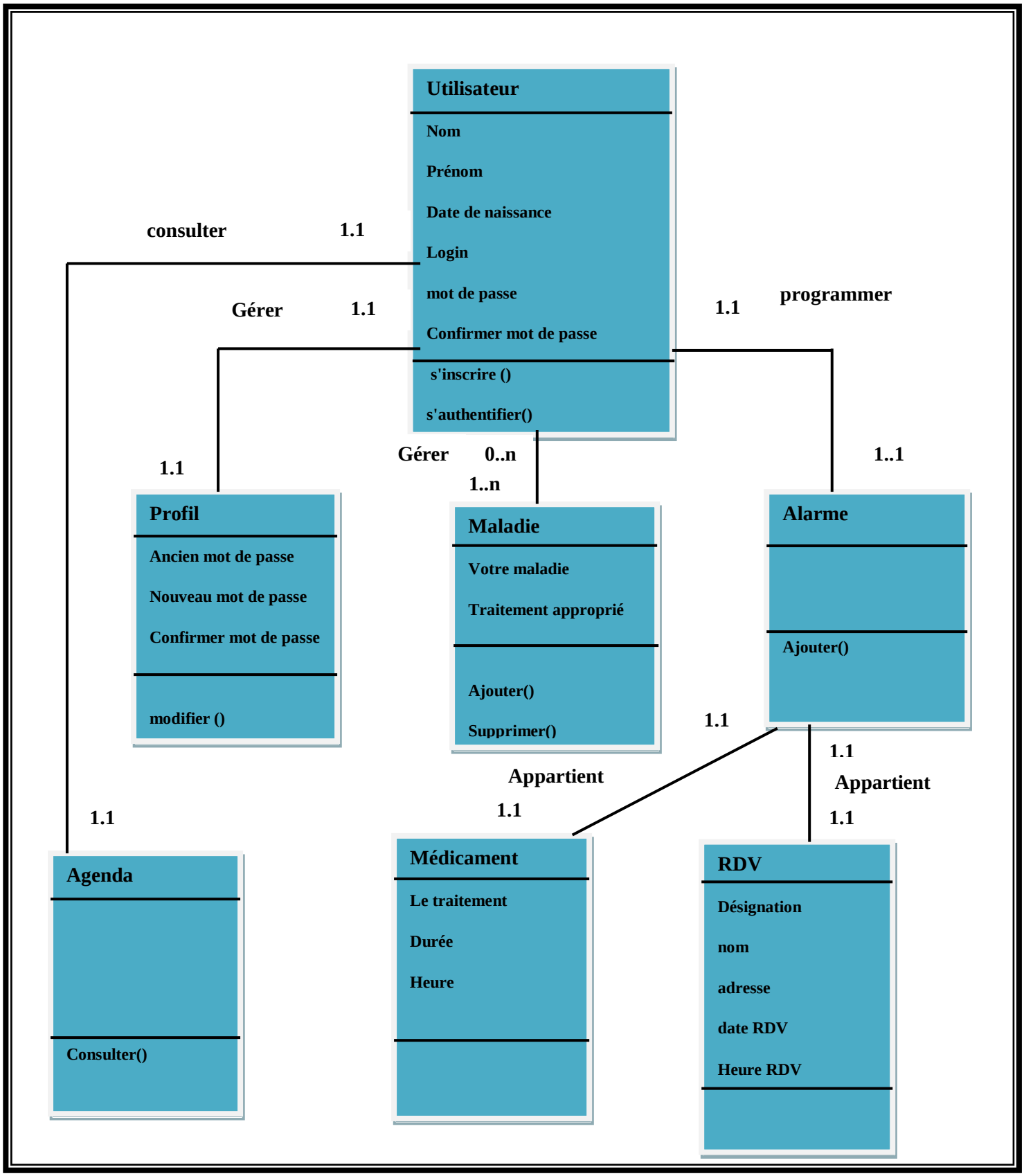


Figure II.11: Diagramme de classe général

IV. Bases de données:

Une base de données est un grand ensemble de données structurées et mémorisées sur un support permanent. Elle peut être manipulée par plusieurs utilisateurs aillant des vues différentes de ces données, du point de vue physique, une BDD est le regroupement de plusieurs fichiers partagés par plusieurs utilisateurs différents.

IV.1. Modèle relationnel :

Le modèle relationnel représente l'information dans une collection de relations, et dans notre cas, et dans ce qui suit, on va déterminer l'ensemble des informations manipulées par notre application.

Client (**id-client**, user_login, user_pwd1, user_nom, user_prénom, user_naiss, user_pwd2)

Maladie (**id-client-maladie**, user_maladie, user_traitement)

Médicament (**id-client-médicament,** user_médicament, user_heure, user_dureé).

RDV (**id-client-rdv,** user_désignation, user_nommed, user_adrss, user_ daterdv, user_heurerdv)

IV.2. Codification :

Toutes les informations manipulées par notre application doivent être stockées dans différentes tables d'une base de données, les noms attribués à ces tables et à leurs différents champs doivent respecter certaines règles.

Table «maladie» :

Nom	*Type de données*	*Description*	*Clef*	*Observation*
Id-client-maladie	Integer	Identificateur du client	Primaire	
user_maladie	Text	Nom de la maladie		
user_traitement	Text	Le nom du traitement		

Table « client » :

Nom	*Type de données*	*Description*	*Clef*	*Observation*
Id-client	Integer	Identificateur du client	primaire	
user_nom	Text	Nom de l'utilisateur		
user_prénom	Text	Prénom de l'utilisateur		
user_login	Text	Login de l'utilisateur		
user_naiss	Datepicker	Date de naissance		JJ/mm/aaaa
user_pwd1	Text	Mot de passe de l'utilisateur		Doit être unique
user_pwd2	Text	Confirmation du mot de passe		

Table «Médicament »

Nom	*Type de données*	*Description*	*Clef*	*Observation*
id-client-médicament	Integer	Identificateur du client	Primaire	
user_médicament	Text	Nom du médicament		
user_heure	Timepicker	L'heure de prise du médicament		
user_durée	Datepicker	La durée pendant la quelle l'utilisateur prend le médicament		

Table «RDV »

Nom	*Type de données*	*Description*	*Clef*	*Observation*
id-client-rdv	Integer	Identificateur du client	primaire	
user_désignation	Text	Nom de la désignation du RDV		
user_nommed,	Text	Le nom du médecin		
user_adrss	Text	L'adresse physique de la désignation(analyse ou médecin)		
user_ daterdv	Datepicker	La date prévu pour le rdv		jj/mm/aaaa
user_heurerdv	Timepicker	L'heure prévu pour le rdv		

V. Conclusion :

Dans ce chapitre, nous avons proposé une démarche de modélisation pour développer notre application. Nous avons d'abord commencé par la spécification du cas d'utilisation dans un premier temps, suivi avec une étude de conception de notre application.

La réalisation du projet sera détaillée dans le prochain chapitre.

I. Introduction :

Une fois le travail de modélisation achevé, vient l'étape du codage (Implémentation ou programmation), qui consiste en la traduction dans un langage de programmation des différentes fonctionnalités définies lors des phases d'analyse et de conception. Et afin d'avoir en résultat une application qui réalise correctement toutes les tâches attendues par ses utilisateurs, il faut bien choisir les logiciels ainsi que les langages adaptés pour la mise en place de l'application.

II. Outils utilisés :

II.1. Matériels :

- PC portable
- Smartphone Samsung Galaxie

II.2. Logiciels :

Pour pouvoir réaliser une application dans de bonnes conditions il faut en tout premier lieu bien choisir son environnement de travail. Nous avons opté pour la réalisation d'une application mobile sous système Android ce qui nous impose de travailler sous Eclipse avec le langage JAVA et le SDK Android pour son développement, afin d'obtenir un fichier.APK qui sera par la suite installé sur les terminaux mobiles de types Smartphones fonctionnant sous système Android version 4.0.3 et plus.

II.2.1. L'IDE Eclipse :

L'IDE Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, visant à fournir un environnement de production de logiciels libres qui soit extensible, universel et polyvalent, permettant de programmer dans différents langages (principalement Java), et ce grâce à ses nombreux plug-ins et notamment le plug-in Android.

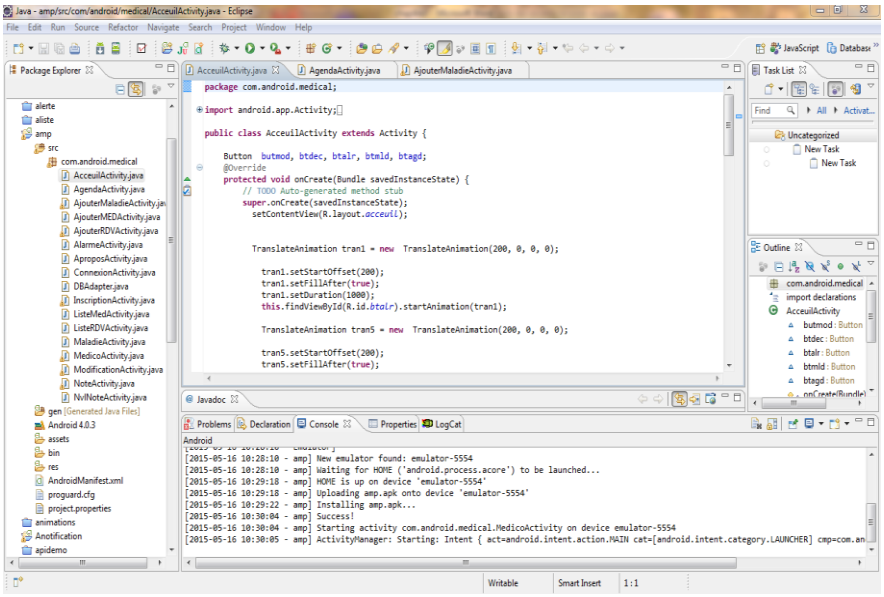


Figure III.1 : capture d'écran d'une fenêtre Eclipse

II.2.2. Le langage JAVA :

Java est à la fois un langage de programmation informatique orienté objet et un environnement d'exécution informatique portable créé par James Gosling et Patrick Naughton employés de Sun Microsystems avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld. Le langage Java a la particularité principale que les logiciels écrits avec ce dernier sont très facilement portables sur plusieurs systèmes d'exploitation tels que Unix, Microsoft Windows, Mac OS ou Linux avec peu ou pas de modifications... C'est la plate-forme qui garantit la portabilité des applications développées en Java.

II.2.3. Les fichiers XML:

Notre application devra à plusieurs reprises, nous afficher des activités avec les quelles l'utilisateur interagis, les interfaces graphiques de ses activités sont réalisable par le biais des fichiers XML.

voyons les caractéristiques de ce fichier :

Définition :

XML (eXtensible Markup Language) soit « Langage de balisage extensible » est un langage de balisage définissant un format universel de représentation des données.

XML n'est pas un langage de programmation, il n'y a pas de boucle for, de if, de while. Il est presque exclusivement utilisé pour stocker des données (du texte) de façon structurée. C'est un langage qui utilise des balises tout comme le HTML.

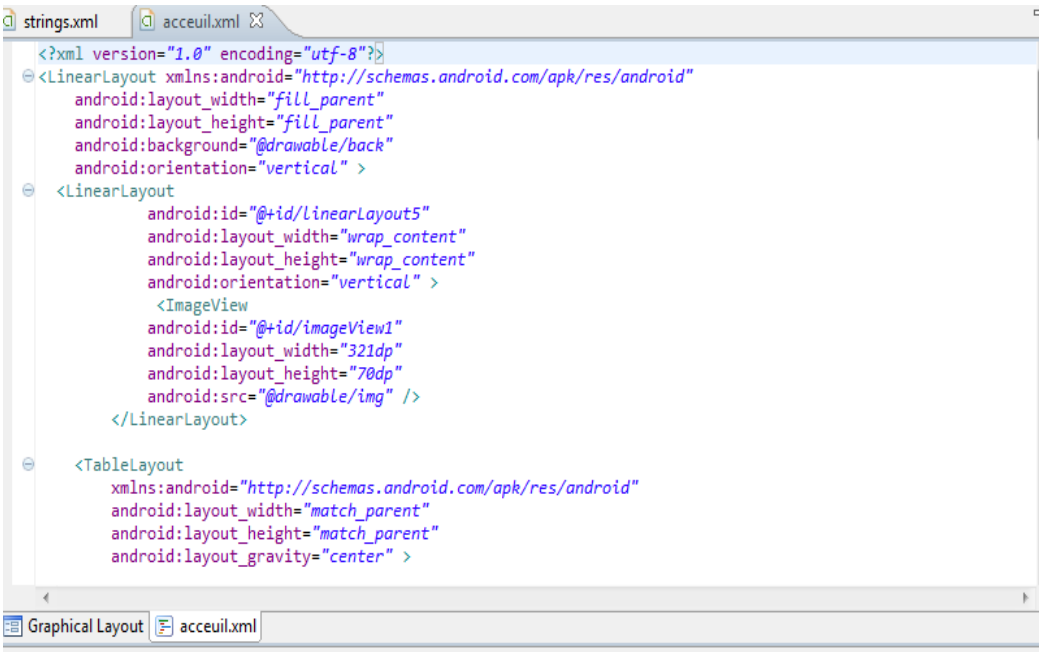


Figure III.2 : capture d'écran d'une fenêtre d'un fichier XML

II.2.4. AVD (Android Virtual Device) :

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

- Un AVD est un appareil Android virtuel accompagnant le système de développement.
- il permet de tester des applications Android sur des appareils avec des caractéristiques variables avant de les déployer sur les véritables terminaux

Figure III.3: Capture d'écran de l'émulateur

III. Mode de fonctionnement de l'application :

Nous allons désormais expliquer le processus de fonctionnement de l'application et monter son rendu visuel sous forme de zones de texte représentant les actions réalisées dans différents cas de figures.

III.1. La fenêtre principale de l'application :

En premier lieu l'utilisateur lance l'application en appuyant sur l'icône lui correspondant dans le menu du Smartphone, ce qui donne ceci :

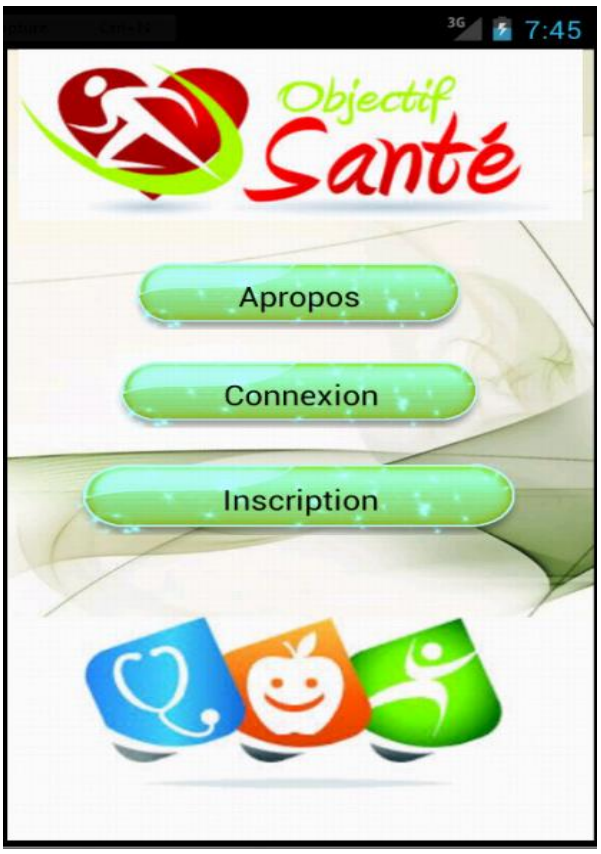


Figure III.4 : capture d'écran de la fenêtre principale de l'application

Cette fenêtre dispose de trois boutons :

- L'un permet de s'informer sur le fonctionnement l'application.
- L'autre ouvre un formulaire d'identification.
- Et enfin le dernier ouvre un formulaire d'inscription .

III.2. Le menu principal de l'application :

En appuyant sur connexion on aura le menu principal de l'application :

Figure III.5 : capture d'écran du menu principal de l'application

Cette fenêtre dispose de cinq boutons :

- Le premier ouvre la fenêtre liste des maladies..
- Le deuxième ouvre l'activité alarme permettant de programmer une alarme.
- Le troisième sert uniquement pour consultation de l'agenda.

  Le quatrième permet de modifier le mot de passe de son profil.
- Le dernier permet de se déconnecté du compte et revenir à la page d'accueil.

III.3. La fenêtre « Ajouter maladies » :

Figure III.6 : capture d'écran de La fenêtre Ajouter maladie

Cette fenêtre dispose deux champs texte:

- Le premier permet de sélectionné une maladie.
- Le second permet de spécifier le traitement de la maladie.
- et un bouton valider pour confirmer l'ajout.

III.4. Suppression d'une maladie de la base de données :

Appuyer longtemps sur un item de la liste dans la fenêtre maladie et cliquer sur « Supprimer», cela effacera définitivement ce dernier de la base de données.

Figure III.7 : capture d'écran de la fenêtre Supprimer maladie

III.5. La fenêtre « Alarme pour médicament » :

Celle-ci permet de programmer une alarme pour un médicament.

Figure III.8 : capture d'écran de la fenêtre alarme pour médicaments

Cette fenêtre dispose de trois champs texte :

- Le premier permet de mentionner le nom du médicament.
- Le second permet de spécifier la durée pendant laquelle l'alarme va se déclenchée.
- le dernier permet de préciser l'heurte de l'alarme

Le bouton valider permet d'activer l'alarme

III.6. La fenêtre « A propos » :

Celle-ci contient les informations concernant le fonctionnement de l'application.

Figure III.9: Capture d'écran de la fenêtre à propos de l'application

IV. Conclusion :

Dans ce dernier chapitre nous avons présenté une vue générale sur l'application que nous avons eu pour but de réaliser. Nous avons montré les différents rendus visuels que nous offrait l'application ainsi que ses nombreux composants, le rôle de chacun.

Étant donné les résultats obtenus, nous pouvons en déduire que la plupart de nos objectifs de départ ont étés réalisés.

Conclusion Générale

Au cours de ce projet, nous avons développé une application mobile permettant à son utilisateur de suivre son état de santé (des maladies, des traitements ainsi que des alarmes, toutes ses informations sont enregistrées dans une base de données).

La mobilité de notre application, du fait qu'elle soit directement téléchargeable sur un Smartphone fonctionnant sous système Android (version 4.0.3 et plus) permet à son utilisateur une grande liberté d'utilisation. (L'utilisateur n'aura pas a ce connecter a un serveur ou un réseau quelconque pour s'en servir).

Cette réalisation nous a aussi permis d'enrichir notre savoir et de développer nos connaissances informatiques, notamment dans le domaine de la programmation et de la mobilité. En effet, l'application a exigé des connaissances du langage JAVA et des outils de développement indispensables à sa réalisation. La mise en œuvre de notre travail a exigé des connaissances très approfondies en la matière ainsi qu'une bonne maitrise de la configuration d'éclipse et de l'environnement Android. Android s'est aussi avéré d'une importance capitale lors de nos recherches.

Quoi qu'il en soit cette application est loin de convenir parfaitement aux besoins réels de l'utilisateur, dans un futur proche et des connaissances plus enrichies, nous espérons bien l'étendre afin qu'il puisse y avoir plus de choix, Par ailleurs ce projet de fin d'étude est une expérience humaine très enrichissante.

Pour finir, nous espérons que notre application répondra aux besoins de futurs utilisateurs et que notre mémoire apportera un soutien considérable pour de prochains développeurs désireux de créer une application mobile pour le système d'exploitation Android.

Perspective en vue d'une extension vers une application plus importante

Actuellement, nous nous sommes juste limités à une simple application de suivi médical (garder trace de tous les médicaments déjà pris) afin de répondre aux spécifications attendues en termes de fonctionnalités par l'application.

Une extension pour rendre l'application plus performante serait de permettre un appwidget sur le bureau .Ce dernier affiche les dates des rendez-vous, celle des analyses deux jour avant la date prévu par exemple.

Par rapport a l'alarme on aimerais bien que au bout de trois minute si l'utilisateur ne clique pas sur le bouton d'arrêt de l'alarme, une alerte par SMS sera envoyé automatiquement chez une personne bien spécifique que l'utilisateur devra déterminer, cela serra plus adéquat pour les personnes âgées atteintes de maladie chronique qui ont vraiment besoin de prendre leurs médicament a temps réel .

Notre application est expliqué en français afin d'être à la portée des personnes les plus novices en la matière. Tout fois il peut être utile de décliner son application dans plusieurs langues (anglais par exemple) d'autant plus que le market android lui est disponible dans de nombreux pays.

On pourrait aussi rendre l'application souple de manière à ce qu'elle intègre des fonctionnalités simples et rapides. L'utilisateur ayant accès seulement aux écrans, il faut avoir un graphisme exemplaire.

Références Bibliographiques

Livres :

- Programmation Android, De la conception au déploiement avec le SDK Google Android 2 (Edition EYROLLES)
- Programmer en Java (Edition EYROLLES)
- Développer des applications mobiles pour les Google Phones (Edition DUNOD)
- L'art du développement Android (Edition PEARSON)
- Reto Meier, Professional Android 2 : Application Development, Wrox
- Florent Garin, Android : Développer des applications mobiles pour les Google Phones, Dunond
- Damien Guignard, Julien Chable, Emmanuel Robles, Programmation Android : De la conception au déploiement avec le SDK Google Android 2, Eyrolles
- W. Franck Ableson, Robi Sen, Chris King**,** Android In Action, Second Edition, Manning

Sites internet :

- http://www.android.com/
- http://www.planet-android.com/
- http://developer.android.com/
- http://android.smartphonesfrance.info/
- B http://www.editions-eyrolles.com
- http://www.android-le-livre.fr