République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université MOULOUD MAMMERI de TIZI-OUZOU

Faculté du Génie Electrique et Informatique

Département d'Informatique

Mémoire

De fin d'études

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

Option : Conduite de Projets Informatiques

Thème :

Conception et réalisation d'une application de prise de notes sous Android

Dirigé par :

M[r] DIB Ahmed

Réalisé par :

M[elle] SidAhmed Nora

Promotion 2014/2015

Remerciements

Ce mémoire de Master marque l'aboutissement d'un cycle de la vie, passage de la vie étudiante à la vie professionnelle ; De prime abord je remercie le bon dieu, tout puissant de m'avoir donné la force, la volonté et la patience pour l'élaboration de mon travail.

Je tiens à remercier mon promoteur ***Mr Dib Ahmed*** *pour son encadrement précieux, son encouragement continu et ses conseils.*

Je tiens également à remercier les membres du jury, pour l'honneur qu'ils me font en acceptant d'évaluer et de juger mon travail.

Enfin, mes remerciements vont à tous ceux qui ont contribués de prés ou de loin à l'élaboration de ce travail.

Dédicaces

À tous ceux qui ont su m'apporter aide et soutien

aux moments propices, je dédie ce modeste travail; et

à tous ceux qui m'aiment et que j'aime.

Nora

Sommaire

Chapitre II: Android :

Chapitre III: Analyse et conception :

Liste des figures

Chapitre I : Les applications mobiles :

Chapitre II: Android :

Chapitre III: Analyse et conception:

Chapitre IV: Réalisation et mise en œuvre :

Liste des tableaux

Chapitre I : Les applications mobiles :

Chapitre II: Android :

Chapitre III: Analyse et conception:

Introduction générale

Introduction générale :

Le marché des Smartphones, apparu il y a quelques années, est aujourd'hui un secteur en pleine expansion. Ce nouveau support multimédia permet d'allier tous les avantages d'un téléphone portable (petite taille, connectivité et mobilité) avec les fonctionnalités issues du monde de l'informatique (puissance de calcul, graphisme, géolocalisation, …).

Une des conséquences de cette fusion est l'apparition des « petites » applications pratiques. Ces dernières sont destinées à faciliter la vie quotidienne de l'utilisateur en lui procurant certains services directement sur son Smartphone citant le service de la gestion des taches et la prise des notes. Or, souvent ces types de services existaient déjà sous d'autres supports mais le fait qu'on puisse maintenant tenir dans la poche c'est déjà un avantage.

Certains sites spécialisés proposent des applications de prise de notes comme « GoogleKeep » développée par « Google », « Note EveryThing » développée par « SoftExpirence », « My Day » développée par « Yalantis », etc. Chaque application possède ses propres caractéristiques et ses propres contraintes. C'est dans cet état d'esprit qu'on a choisi le thème de notre projet de fin d'études en décidant de créer l'application «Mes Notes».

Alors, durant ce rapport on va mettre en œuvre quatre chapitres décrivant toutes les étapes à suivre pour arriver en fin à bien réaliser une application adéquate :

- Le premier chapitre s'intitule « Les applications mobiles » : ce chapitre est une introduction aux applications mobiles et leurs multiples typologies.

- Le deuxième chapitre s'intitule « Android » : dans ce chapitre, puisque notre application fonctionne sous le système d'exploitation Android, nous allons présenter une vue approfondie sur ce dernier en analysant son architecture interne et ses versions.

- Le troisième chapitre sous le nom « Analyse et conception » : ce chapitre est consacré à l'analyse et la conception de notre application, afin de réaliser une application adéquate.

Pour cela on a fait recours au langage UML étant le mieux adapté pour les applications mobiles.

- Le quatrième est le dernier chapitre « Réalisation et mise en œuvre », comporte la présentation de l'environnement dont lequel notre application a été réalisée, les outils utilisés et quelques interfaces de notre application.

Problématique :

L'annotation textuelle sous forme de notes manuscrites est apparue avant l'existence des ordinateurs et des Smartphones, le besoin de prendre des notes est évolué avec l'apparition de la technologie mobile, l'utilisateur a eu plus de besoin pour garder ses idées à propos d'une situation précise, pour surmonter ce problème, beaucoup de développeurs d'applications ont pensés à créer des outils comme un aide mémoire, et à ce point là que c'est apparu les outils d'annotation sur les différents systèmes d'exploitation et récemment sur les téléphones intelligents.

D'autre part avec l'explosion rapide de la technologie, les applications d'annotation avait un grand besoin de varier son contenu par des structure de type image, voix et vidéo ce qui a rendu la prise de note très utile et plus efficace.

Actuellement, avec l'apparition des applications mobiles et grâce aux outils d'annotation, nous sommes capables de prendre des notes directement sur un Smartphone pour avoir une vue directe sur nos pensées. C'est dans cet état d'esprit que nous avons pensés à réaliser une application qui permet d'écrire et enregistrer des notes et des listes de notes facilement.

Le service de prise de note s'avère très sobre sans possibilités de mise en forme. Ajouter un titre et le contenu de votre note, et le système sauvegarde votre note avec la date et l'heure actuelle.

Chapitre I

Les applications mobiles

Introduction :

La technologie mobile est en train d'évoluer à grande vitesse, ainsi, le téléphone mobile est à la fois un téléphone et télétexte, baladeur, appareil photo et caméscope, console de jeu, réveil et agenda, terminal Internet, balise GPS, téléviseur, etc.

Ce chapitre présente un bref récapitulatif des applications mobiles fonctionnant sous des téléphones artificiellement intelligents à savoir des « Smartphones ».

On va commencer par introduire l'informatique ubiquiste pervasive pour aboutir à la définition de l'informatique mobile et ses dispositifs appropriés. Ensuite on va aborder la notion des applications mobiles, leurs typologies, l'évolution du marché, ainsi que leurs avantages et leurs inconvénients.

1. L'informatique Ubiquiste :

On appelle informatique ubiquiste ou pervasive (Pervasive Computing) la tendance à l'informatisation et la miniaturisation des dispositifs électroniques en les intégrant dans tout objet du quotidien. [1]

Cette informatique s'est rapidement développée grâce au progrès technique des systèmes de communication (réseaux sans fil) et de la microélectronique.

Assurant plus de liberté, l'informatique ubiquitaire permet à l'utilisateur d'affranchir les contraintes actuelles d'utilisation d'un ordinateur en rendant accessible toutes sortes de services indépendamment de la localisation. De ce fait, la mobilité et la liberté du déplacement semblent être le premier avantage tiré de l'ubiquité.

2. L'Informatique Mobile :

2.1. Présentation :

L'informatique mobile s'inscrit sous l'approche globale de l'informatique ubiquitaire. Historiquement, elle a été autour depuis 1992. Depuis lors, elle constitue un outil de communications très puissant pour les entreprises et les utilisations personnelles. [1]

Les supports qui lui ont étés développés ont repris l'industrie sans fil. Il s'agissait des entités mobiles communicantes et parfois de très petites tailles permettant de se connecter aux différents types de réseaux tant qu'ils disposent des périphériques appropriés. Les Smartphones en font partie.

2.2. Les Smartphones :

2.2.1. Définition :

Un Smartphone est un « téléphone intelligent », appelé aussi « Ordiphone ».C'est un appareil dédié aux communications mobiles, disposant d'un système d'exploitation ouvert et adoptant des applications tierces développées par le fabricant, par l'opérateur ou par un éditeur de logiciel. [1]

Ce téléphone fournit un module de radiocommunication (pour la voix et l'échange de données), des fonctionnalités bureautiques (agenda, calendrier, navigation web, email, messagerie instantanée,

GPS, etc.) ainsi que des fonctionnalités multimédia (photo, musique, vidéo et jeux) comme les présente la figure suivante :

Figure I.1 : Quelques applications installées sur un Smartphone.

2.2.2. Système d'exploitation des smartphones :

En effet, les Smartphones sont dotés d'un système d'exploitation qui gère leurs fonctionnalités. En 2014, les systèmes d'exploitation les plus répandus sont :

- **Android :** Android est un système d'exploitation développé par Google pour Smartphones, tablettes tactiles, PDA et terminaux mobiles. C'est un système open source utilisant le noyau Linux ;
- **Windows Phone :** Windows Phone est un système d'exploitation mobile développé par Microsoft pour succéder à Windows Mobile, sa précédente plateforme logicielle qui a été renommée pour l'occasion en Windows Phone Classic;

- **Black Berry OS:** BlackBerry OS est un système d'exploitation propriétaire pour téléphone mobile de la gamme BlackBerry, conçu par la société canadienne Research In Motion, maintenant connue sous le nom de Blackberry. Il s'agit d'un système multitâche;
- **iOS** (anciennement iPhone OS), est le système d'exploitation mobile développé par Apple pour l'iPhone, l'iPod touch et l'iPad. Il est dérivé d'OS X dont il partage les fondations.

Ils sont illustrés dans la figure suivante :

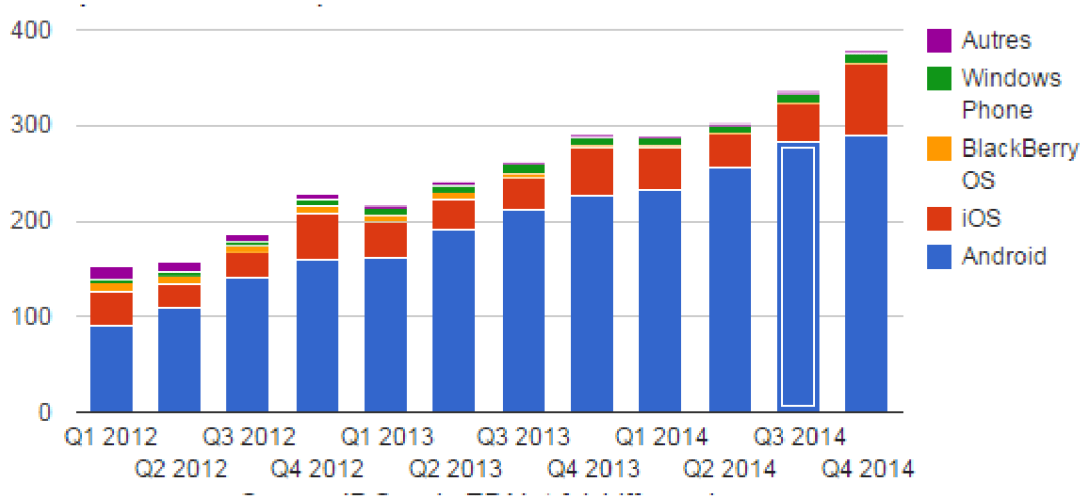


Figure I.2 : Ventes comparées 2012/2014 des Smartphones par système d'exploitation (millions d'unités) [2]

2.3. Les applications mobiles :

2.3.1. Définition :

Une application mobile est un programme téléchargeable de façon gratuite ou payante et exécutable à partir du système d'exploitation de téléphone. Les applications mobiles sont adaptées aux différents environnements techniques des Smartphones/iPhones et à leurs contraintes et possibilités ergonomiques (écran tactile notamment). Elles permettent généralement un accès plus confortable et plus efficace à des sites accessibles par ailleurs en versions mobiles ou web [3].

Une application mobile s'appuie d'une manière générale sur le principe de widgets que nous connaissons sur nos ordinateurs comme le montre la Figure 3 :

Figure I.3: Smartphone et ses applications [4]

Pour télécharger une application sur un téléphone mobile, il existe plusieurs possibilités :

- Transfert depuis un ordinateur via un câble de connexion.
- A partir d'un service mobile.
- Via une boutique de logiciels accessible depuis un téléphone mobile (App Store d'Apple, Windows Market Place, Nokia OVI, Android Market, etc.).
- Le cas échéant, l'application est dite native, elle est déjà dans le téléphone lors de l'achat du téléphone (l'opérateur ou le fabricant l'ayant ajouté comme fonction de base).

Reflet de succès commercial et technique de l'iPhone, une grande majorité des applications mobiles furent créées pour le téléphone mobile d'Apple. Cependant, les applications Android se sont rapidement développées depuis les années 2010/2011 et semblent dépasser désormais en nombre les applications iPhone.

Une application mobile peut avoir une vocation commerciale (m-commerce), marketing et/ou publicitaire [3].

2.3.2. Typologie :

Les types/typologies d'applications mobiles sont souvent confondues avec les catégories de ces dernières, les catégories d'applications (apps) mobiles sont les apps de service telles que :

- Les applications d'actualité (News),
- Les livres,
- Les applications d'opérations bancaires,
- Les applications de divertissement.

Par contre pour les types d'applications mobiles, il en existe trois classes importantes :

- ✓ Les Applications natives ;
- ✓ Les Applications web ;
- ✓ Les Applications hybrides.

a) Application native :

Elle est basée sur un langage propre à sa plateforme ("langage natif"). Une application native iOS (iPhone Operating System) sera écrite en objective-C tandis qu'une application native Android sera écrite en Java. Une application native sera disponible dans l'app store propre à sa plateforme et pourra être lancée en tant que logiciel à part entière. On choisit en général ce type pour exploiter au maximum les capacités du système [5]. Elle peut être installée dans un appareil sans avoir besoin d'aucun transfert de données au serveur. Les apps fonctionnent dans l'appareil sans réseau. Les données de l'application seront stockées dans l'appareil même, comme par exemple pour les « Gaming Applications ». Dans ce cas, la mémoire de l'appareil et sa configuration est tellement importante que l'application dépend complètement de ces fonctionnalités.

b) Application web ou web app :

Il s'agit de sites web optimisés pour mobiles, souvent conçus pour ressembler à de vraies applications. Elle peut être aussi appelée « apps navigateur mobile » car elles ne sont pas installées sur l'appareil. Celles-ci peuvent être accédées en employant le navigateur mobile grâce à l'URL du Web. Dans ce cas, la capacité de la mémoire de l'appareil n'est pas aussi importante comme ni la forme ni les données de l'application ne sont pas stockés dedans. Il est complètement dépendant de la qualité du navigateur. Chaque donnée vient du serveur et rendu dans le navigateur quand vous accédez grâce à l'URL.

c) Application hybride :

Une application hybride est un mélange de code natif et d'affichage de vues HTML/JavaScript. Elles sont nées il y a quelques années avec des Frameworks tels que Phonegap ou Titanium.

Ces frameworks permettent de coder une application une fois en langage web (HTML/CSS/JavaScript) pour être déployée vers plusieurs plateformes différentes. Elles sont dites génériques.

L'application sera également disponible dans l'app store de la plateforme. On choisit ce type d'applications pour leur rapidité d'exécution. Cette solution est controversée pour les pertes de performances constatées sur le résultat [5].

Exemples d'application hybride :

• Linkedln,

• Microsoft Bing pour mobiles

2.3.3. Comparaison entre les typologies d'applications mobiles :

➢ Le tableau ci-dessous décrit les spécificités de chaque type d'application mobile :

Application Native	**Application Web**	**Application Hybride**
• Application dédiée pour chaque OS. • Exécutable téléchargé et stocké sur le terminal. • Développée avec le SDK et le langage spécifique à la plateforme. • Distribuée via un magasin d'applications. • Tire parti de toute la puissance et les possibilités du terminal mobile, de son ″look and feel″.	• Application web optimisée pour les mobiles. • Utilisable via un navigateur du mobile. • Déploiement via une URL. • Utilise les dernières technologies web : HTML5, JS, CSS3, AJAX, JSON, JQuery.	• Mélange de code natif et d'affichage de vues HTML/Javascript. • Peut être distribuée sur les magasins d'applications. • En général basée sur un framework hybride spécifique.

Tableau I.1 : Les trois types d'application mobile. [6]

- La figure ci-dessous compare les applications natives et les applications web selon plusieurs critères :

Applications natives	Mobiles web applications
Le développement	
Chaque type de Smartphones requiert son propre processus de développement (iOS, Android) et chaque téléphone mobile demande un langage de développement spécifique (Objective-C pour iOS, Java pour Android et Visual C++ pour Windows Mobile)	Cela fonctionne en fonction du navigateur web du Smartphone et donc encore une fois, chaque téléphone aura ses propres spécificités. Les applications web sont développées en langage CSS3, JavaScript ou bien HTML5
Les capacités	
Elles peuvent supporter des caractéristiques comme l'appareil photo ou le flash par exemple	Toutes les informations ne sont pas complètement disponibles (médias, géolocalisation, orientation)
La rentabilité et la monétarisation	
Cela dépend de la plateforme de chaque type de Smartphone, certaines applications seront gratuites et d'autres payantes. A savoir que les « magasins d'applications » (comme App Store) peuvent récolter un pourcentage sur le prix de l'application web	Les applications sont gratuites pour les utilisateurs mais sont rentabilisées par les publicités ou bien les frais d'inscription. Pour les développeurs, cela exige une configuration de plus à installer comme un mode d'abonnement payant
La méthode de livraison	
Il faut les télécharger sur des plateformes et ensuite au niveau de l'utilisation, elle se fait via l'application directement, vous n'avez plus besoin de navigateur web	Vous n'avez pas besoin de télécharger et d'installer quoi que ce soit, vous devez juste avoir un navigateur web pour mobile qui vous permettra d'atteindre votre demande (recherche) et vous guidera vers l'application mobile
Les versions et améliorations	
C'est à l'utilisateur de décider s'il souhaite améliorer les versions en téléchargeant les mises à jour éventuelles	Tout le monde a la même version car les applications web mobiles sont mises à jour directement par le site en question

Figure I.4 : Comparaison entre une application native et une application web. [7]

2.3.4. Le marché des applications mobiles : [8]

En 2014, l'usage des applications mobiles propres à la productivité (Utilities & Productivity) a progressé de 121 %, selon le cabinet Flurry Analytics (désormais propriété de Yahoo) spécialisé dans l'étude des usages mobiles. Soit bien au-delà des 76% de progression moyenne de l'utilisation des applications depuis un smartphone ou une tablette iOS et Android constatée par l'analyste sur l'année passée.

« Les utilisateurs de smartphones ont ouvert plus d'applications Utilitaires et de Productivité en 2014, note le cabinet, confirmant que nos téléphones et tablettes sont devenus des appareils

indispensables qui nous aident à travailler et organiser notre quotidien. ». Le tournant mobile de Microsoft et l'élargissement, à partir de 2013, aux versions iOS et Android de son offre Office mobile a probablement participé significativement à la forte progression des applications à caractère productif.

Si les applications utilisées à des fins professionnelles et d'organisation ont bénéficié d'une adoption remarquable en 2014, elles sont surpassées par celles propres au style de vie et au commerce (Lifestyle & shopping) qui ont progressé de 174%. Elles occupent la première place des logiciels mobiles les plus utilisé et replace dans ce cadre les messageries qui avaient connu la plus forte progression (de 115%) en 2013. En 2014, ces dernières ont néanmoins affiché une belle tenue à 103%. Leur transformation en véritable plateforme de services pour certaine (paiement chez SnapChat et Line, distribution de jeux chez WeChat…) ayant probablement soutenu leurs usages.

Parmi les solutions qui progressent moins que la moyenne, on notera la présence des applications à caractères informatif (sport 74% et news 49%) et de divertissement (music, media & entertainement, 33%). Une chute sévère pour ces dernières en regard des 79% de progression constatées en 2013. Les jeux, applications mobiles historiques s'il en est, connaissent également un fort recul : 30% contre 61%un an plus tôt. « Nous assistons à un tournant où la croissance mobile est passée des application de divertissement aux application qui nous aident à accomplir nos tâches quotidiennes », résume Flurry.

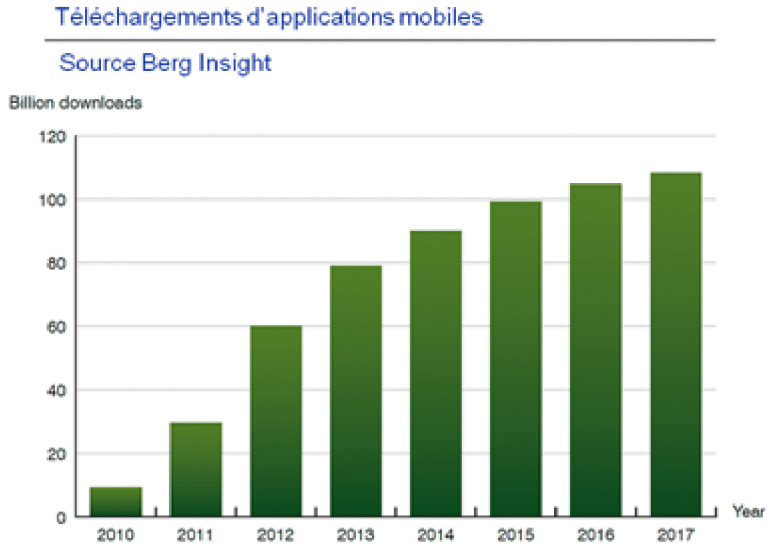


Figure I.5 : Évolution du marché mondial des applications mobiles

2.3.5. Avantages et inconvénients :

Les applications mobiles s'étalent dans des différents domaines, et touchent toutes les générations grâce à leurs avantages et facilitées d'utilisation, malgré quelques inconvénients qu'on leurs trouve.

a) Avantages :

Parmi les avantages des applications mobiles on trouve:

- Contenus adaptés aux spécificités de chaque Smartphone.
- Contenus disponibles hors ligne, sera toujours accessible (plus besoin d'attendre l'ouverture de la page).
- Possibilité de sélectionner des contenus à afficher selon la cible et l'objectif de communication.
- Etablissement d'un lien direct avec le consommateur.
- Rapidité d'exécution (l'application mobile exploite au mieux les capacités du téléphone).
- Facilité d'installation et d'accès : proposez des Qr-code3 pour son téléchargement, la presse papier, des autocollants, etc.
- Possibilité de solliciter un mobinaute (Push).
- Interface plus riche et des meilleures performances
- Facilité de Monétisation (parce qu'elles sont directement disponible sur l'app store)
- L'application a accès aux dispositifs du terminal, ce qu'est ses fonctions spécifiques (exemple appareil photo, GPS).
- Plus besoin de taper l'URL de votre site dans le petit navigateur de votre téléphone.

b) Inconvénients :

Pour développer une application mobile il faut savoir qu'elles ont aussi quelques inconvénients citons:

- **Le respect des règles des plateformes mobiles :**

L'Apple Store, le Play Store et le Windows Store imposent un certain nombre de règles pour les développeurs. Parfois contraignantes, elles sont un passage obligé si l'on souhaite pouvoir distribuer une application Smartphone de manière optimale.

- **Un coût de développement élevé :**

Le coût de création d'une application mobile est relativement élevé et peut décupler les dépenses e-marketing, surtout si l'on souhaite que l'application soit disponible sur tous les systèmes d'exploitation pour mobiles.

- **Contrainte de la mise à jour pour l'utilisateur :**

En cas de mise à jour de l'application, l'utilisateur d'application mobile doit se rendre une nouvelle fois sur les plateformes de téléchargement, Le site mobile peut quant à lui évoluer sans contraintes pour le visiteur.

2.3.6. Les domaines des applications mobiles : [3]

Avec les possibilités matérielles incorporées aux terminaux (caméra, GPS, gyroscope, …), les applications Smartphones et Tablettes peuvent intégrer des fonctionnalités spécifiques et dédiées pour les utilisateurs, permettant ainsi d'enrichir le spectre fonctionnel et imaginer des usages non couvert jusqu'à présent par les systèmes d'information :

- ✓ Géo-localisation, Itinéraires
- ✓ Scan de Code barre, Flash, QR Code
- ✓ Réalité augmentée
- ✓ M-commerce, Paiement mobile
- ✓ Push et notification
- ✓ Gestion de documents, dématérialisation, Workflow
- ✓ Analyse d'Audience
- ✓ Gestion et Sécurisation de parc et de déploiement de terminaux mobiles.

Conclusion :

Les applications mobiles ont pris une place importante dans notre vie quotidienne, vu qu'elles couvrent une grande variété de domaines. Cela nous a conduits à faire une étude des applications mobiles en général dans ce chapitre. Dans le chapitre qui suit on va présenter le système d'exploitation Android.

Chapitre II

Android

Introduction :

L'OS Android est aujourd'hui l'OS le plus populaire au monde, s'exécutent sur différents dispositifs. Des milliards d'appareils électroniques Android sont possédés par le grand public partout dans le monde.

Androïd est une excellente opportunité pour appréhender le développement d'applications mobiles ambitieuse, c'est une plate-forme innovante (car toutes les dernières technologies de téléphonie y sont intégrées : écran tactile, accéléromètre, GPS, appareil photo numérique, etc) et fournie sous licence open-source.

Ce chapitre est consacré à la description du système d'exploitation Android.

1. Généralités:

1.1. Définition :

Android, prononcé Androïde est un système d'exploitation open-source (c'est-à-dire dont les sources sont disponibles librement sur internet) pour smartphones, tablettes tactiles, PDA et autres terminaux mobiles, conçu par Android, développé par l'**Open Handset Aliance**. C'est un ensemble de logiciels destinés à fournir une solution pratique pour les appareils mobiles .Cet ensemble est composé d'un système d'exploitation (comprenant un noyau Linux), d'applications telles que le navigateur web, le téléphone et le carnet d'adresse ainsi que des logiciels intermédiaires entre le système d'exploitation et les applications.

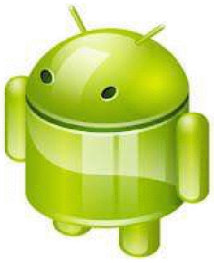


Figure II.1 : Logo officiel du système d'exploitation Android. [9]

Grâce à ses services, Android facilite l'exploitation des réseaux télécom, des technologies sans fil, la manipulation de médias (notamment vidéo, audio et image), l'exploitation des capteurs (capteurs de mouvements, caméra, boussole, récepteur GPS…), le stockage de fichiers, la gestion des images 3D (via le processeur graphique de l'appareil), l'accès à Internet, et bien d'autres applications.

1.2. Historique d'Android:

1.2.1. Naissance d'Android :

L'histoire d'Android commence en octobre 2003, où la société Android Inc. est créée. Officiellement, elle développe des logiciels pour mobiles. Mais en réalité, elle se préparait à sortir un tout nouveau système d'exploitation pour smartphones. En 2005, Google rachète cette entreprise, et sort une première béta en novembre 2007, avant de lancer la version 1.0 en

septembre 2008 avec le HTC Dream. À partir de ce moment là, le rythme des nouvelles sorties est très élevé : pas moins de 11 versions différentes sont sorties en 3 ans ! On remarquera au passage que chaque version d'Android porte le nom d'un dessert.

1.2.2. Les différentes versions d'Android :

La progression des fonctionnalités, amélioration et correction de bogues, ont fait l'apparition de plusieurs versions d'android de plus en plus sophistiquées, parmi lesquelles on cite les versions majeures suivantes :

Les différentes versions d'Android ont toutes des noms de desserts depuis la sortie de la version 1.5 et suivent une logique alphabétique (de A vers Z)

- **La version 1.0 sortie en fin 2007**, caractérisée par une version connue uniquement par des développeurs car c'est la version du SDK distribuée avant la sortie du premier téléphone Android, ″nommée Apple pie″.
- **La version 1.1 sortie le 22 octobre 2008**, caractérisée par la version Beta incluse dans le premier téléphone le HTC G1/Dream, nommée ″Bananas split″.
- **La version 1.5 sortie le 30 avril 2001**, caractérisée par de nouvelles fonctionnalités et la mise à jour de l'interface graphique, nommée ″Cupcake″, basée sur le noyau Linux2.6.27.
- **La version 1.6sortie le 15 septembre 2009**, caractérisée par de nouvelles fonctionnalités et la mise à jour de l'interface graphique, nommée ″Donut″, basée sur le noyau Linux2.6.29.
- **La version 2.0 sortie le 26 octobre 2009**, caractérisée par de nouvelles fonctionnalités et la mise à jour de l'interface graphique, nommée Eclair basée sur le noyau Linux2.6.29.
- **La version 2.2.x sortie le 20 mai 2010**, caractérisée par une vitesse améliorée, nouvelle fonctionnalités et la mise à jour de l'interface graphique, nommée ″Gingerbread″ basée sur le noyau 2.6.35.
- **La version 3.x.x sortie le 22 février 2011**, réservée aux tablettes tactiles et aux téléviseurs connectés, cette mise a jour comprend de nombreux changements dans l'interface, nommée ″Honeycomb″.
- **La version 4.0.x sortie le 19 octobre 2011**, est fortement inspirée d'Honeycomb, unifiée pour Smartphones, tablettes et Google tv apporte de nombreux changements, nommée Ice Cream Sandwich, basée sur le noyau 3.0.1.
- **La version 4.1.x sortie le 9 juillet 2012**, elle ajoute un système de notification améliorée, la reconnaissance vocale sans connexion internet et le projet Butté qui augmente la fluidité d'Android, nommée Jelly Bean, bases sur le noyau 3.0.31.

- **La version 4.2.x sortie le 13 novembre 2012**, caractérisée par une nouvelle interface de l'appareil photo, et par un système multi-compte uniquement sur tablette et de type Gesture permettent d'écrire avec le clavier rien qu'en glissant le doit, nommée Jelly Bean.
- **La version 4.3.x sortie le 24 juillet 2013**, caractérisée par un support de Bluetooth smart, basse consommation et ajout de la norme AVRPC 1.3, gestion multiutilisateur plus poussée ,support d'Open ES 3.0,nouvelle interface de l'appareil photo, mise a jour de sécurité et Slimport, nommée ″Jelly Bean″.
- **La version 4.4.x sortie le 31 octobre 2013**, caractérisée par un support de Bluetooth 4.0, consommation on ressource moins élevée nécessite moins de RAM, nouvelles icones plus soignées, la barre de statut est maintenant transparente sur certain menus et changera de couleurs en fonction du contenu affiche, la barre en bas de l'écran est désormais translucide sur certains menus, nommée Kit Kat.
- **La version 5.x sortie le 15 octobre 2014,** caractérisée par une nouvelle interface (le Material Design) qui fait la part belle aux couleurs et au flat design, soit un rendu au visuel plus « plat », nommée ″Lollipop″.

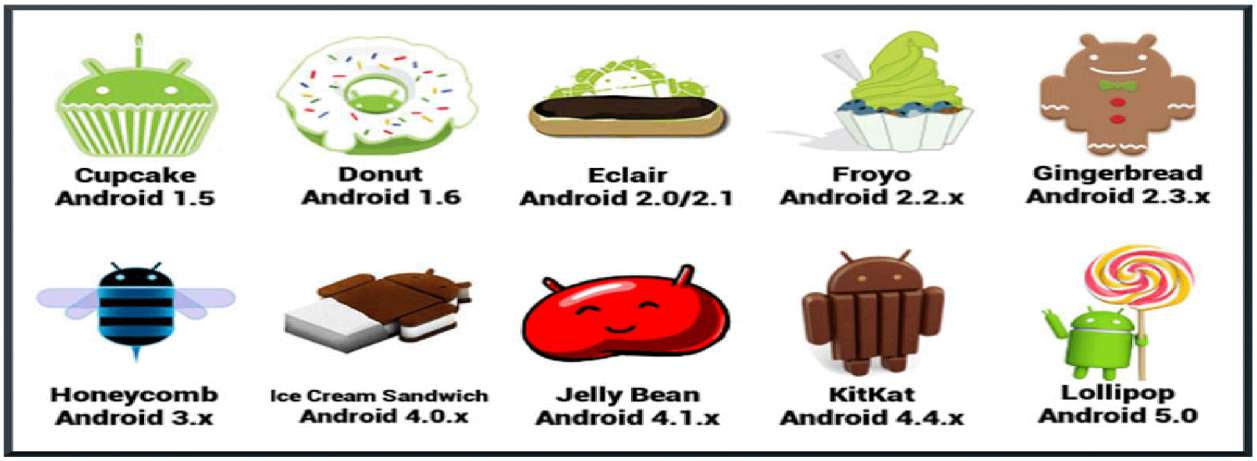


Figure II.2 : Les différentes versions d'Android [10]

1.3. Architecture :

Android bénéficie d'une architecture en couche complète faisant de lui une plateforme riche, dédiée aux appareils mobiles.

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

Figure II.3 : Architecture d'Android [11]

1.3.1. Applications :

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

1.3.2. Le framework (Application Framework) :

En fournissant une plateforme de développement ouverte, Android offre aux développeurs la possibilité de créer des applications extrêmement riches et innovantes. Les développeurs sont libres de profiter du matériel périphérique et informations sur la localisation d'accès, exécutez des services d'arrière-plan, définir des alarmes, ajouter des notifications à la barre d'état, etc. Les utilisateurs ont un accès complet au même framework API utilisé par les applications de base.

L'architecture 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 sécurités appliquées par le framework). Ce même mécanisme permet aux composants d'être remplacés par l'utilisateur.

Toutes 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'activités qui gère le cycle de vie des applications et propose une navigation commune.

1.3.3. Les bibliothèques (Libraries) :

En interne, Android inclut un ensemble de bibliothèques C et C++ utilisées par de nombreux composants de la plateforme Android. Ces bibliothèques sont en réalité accessibles au développeur par l'intermédiaire du framework Android. En effet, le framework Android effectue, de façon interne, des appels à des fonctions C/C++ beaucoup plus rapides à exécuter que des méthodes Java standard. La technologie Java Native Interface (JNI) permet d'effectuer des échanges entre le code Java et le code C et C++. La liste ci-dessous énumère quelques-unes des bibliothèques disponibles dans Android :

- **Bibliothèque système C** : Implémentation (dérivée de BSD) de la bibliothèque standard C (libc), optimisée pour les systèmes Linux embarqués.
- **Bibliothèques multimédias** : Basées sur StageFright, elles permettent le support de nombreux formats audio et vidéo, tels que MPEG4, H.264, MP3, AAC, AMR, JPG et PNG (la liste complète est disponible sur le site des développeurs Android : http://d.android.com/guide/appendix/media-formats.html).
- **Surface Manager** : Permet l'accès au sous-système d'affichage.
- **LibWebCore** : Moteur de rendu de pages Internet basé sur Webkit. Cette bibliothèque est donc principalement utilisée dans le navigateur et dans les vues web embarquées (WebView).
- **SGL** : Moteur graphique 2D.
- **Bibliothèques 3D** : Implémentation basée sur OpenGL ES 1.0 API et plus récemment OpenGL ES 2.0.
- **FreeType** : Rendu des polices de caractères.
- **SQLite** : un moteur de base de données relationnelles ; puissant, léger et disponible pour toutes les applications.

1.3.4. Moteur d'exécution Android (Android Runtime) :

Android inclut un ensemble de bibliothèques qui fournit la plupart des fonctionnalités disponibles dans les bibliothèques de base du langage de programmation Java**.**

Chaque application Android s'exécute dans son propre processus, avec sa propre instance de machine virtuelle Dalvik. Dalvik VM est une implémentation de machine virtuelle ayant été conçue pour optimiser l'exécution multiple de machines virtuelles. Elle exécute du bytecode qui lui est dédié : le bytecode dex. (Format qui est optimisé pour une empreinte mémoire minimale).

Cette particularité d'Android en fait un système unique, loin des systèmes Linux traditionnels que beaucoup avaient pu rencontrer auparavant.

1.3.5. Noyau Linux (Linux Kernel) :

Le noyau Linux version 2.6 est la couche la plus basse d'Android. Cette couche est le niveau d'abstraction matériel qui permet l'interaction des couches supérieures avec la couche matérielle à travers les pilotes des périphériques. Le noyau Linux fournit également les

services les plus fondamentaux tels que la sécurité, la gestion de la mémoire, la gestion des processus ainsi que les fonctionnalités de communication réseau.

Linux est réputé pour sa gestion stable et performante de la mémoire et des processus, ainsi que par son modèle de sécurité robuste basé sur un contrôle d'accès discrétionnaire (DAC) qui n'a pas changé depuis les années soixante-dix.

1.4. Caractéristiques :

Le système d'exploitation Android est doté d'un ensemble de caractéristiques intrinsèques faisant de lui un terrain d'application lucratif. Les principales caractéristiques sont résumées dans le tableau suivant :

Caractéristique	Description
Framework	Framework Java pour le développement d'application pour la plateforme Android.
Machine virtuelle Dalvik	Machine virtuelle spécialement développée pour Android. Cette machine virtuelle permet d'exécuter les applications java développées avec le Framework.
Navigateur web	Navigateur web basé sur le moteur de rendu Webkit.
Graphique	Librairie graphique 2D, librairie graphique 3D basé sur OpenGL ES 1.0. Accélération matériel possible.
Stockage	Base de données SQL : SQLite est utilisé pour le stockage des données.
Média	Android supporte les formats audio/vidéo/image suivants : MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF.
Connectivité	GSM, EDGE, 3G, Bluetooth, Wifi.
Support Matériel	Android est capable d'utiliser Camera, GPS, accéléromètre.
Environnement de développement	Android possède un environnement de

	développement complet contenant : un émulateur, un débuggeur, un analyseur de mémoires et de performances et un plugin Eclipse.

Tableau II.1 : Caractéristiques du système d'exploitation Android. [12]

1.5. Avantages et inconvénients :

❖ Avantages :

✓ Open source

Le contrat de licence pour Android respecte les principes de l'*open source*, c'est-à-dire qu'on peut à tout moment télécharger les sources et les modifier. Android utilise des bibliothèques *open source* puissantes, comme par exemple SQLite pour les bases de données et OpenGL pour la gestion d'images 2D et 3D (pour faire des jeux).

✓ Simple

Android s'est imposé par sa simplicité d'utilisation, sa souplesse, pas besoin d'être un connaisseur expérimenté pour l'utiliser. Par exemple : pour transférer des fichiers, de la musique sur son Smartphone Android, il suffit de brancher l'appareil sur son ordinateur. Comme avec une clé USB. Aucun autre logiciel n'est nécessaire.

✓ Facile à développer

Toutes les API mises à disposition facilitent et accélèrent grandement le travail. Ces APIs sont très complètes et très faciles d'accès.

✓ Facile à vendre

Le Play Store (anciennement Android Market*)* est une plateforme immense et très visitée ; c'est donc une mine d'opportunités pour quiconque veut diffuser une application dessus.

✓ **Flexible**

Le système est extrêmement portable, il s'adapte à beaucoup de structures différentes. Les smartphones, les tablettes, la présence ou l'absence de clavier ou de *trackball*, différents processeurs…etc On trouve même des fours à micro-ondes qui fonctionnent à l'aide d'Android. Non seulement c'est une immense chance d'avoir autant d'opportunités, mais en plus Android est construit de manière à faciliter le développement et la distribution en fonction des composants en présence dans le terminal (par exemple pour une application qui nécessite d'utiliser le Bluetooth, seuls les terminaux équipés de Bluetooth pourront la voir sur le Play Store).

✓ **Ingénieux**

L'architecture d'Android est inspirée par les applications composites, et encourage par ailleurs leur développement. Ces applications se trouvent essentiellement sur internet et leur principe est de pouvoir combiner plusieurs composants totalement différents pour obtenir un résultat surpuissant. Par exemple, si on combine l'appareil photo avec le GPS, on peut poster les coordonnées GPS des photos prises.

❖ Inconvénients :

Malgré le succès qu'a connus Android au bout de quelques années, il reste « incomplet», et on s'aperçoit au fur et à mesure de son utilisation des manques qui ne sont pas en sa faveur, parmi lesquels :

- ✓ Android est un système Open Source, mais toutefois, on peut se demander si Google joue le jeu, le code source de la version 3.0 (Honeycomb), dédiée aux tablettes, n'a jamais été fourni.
- ✓ Lors d'obtention d'une liberté de personnalisation de son téléphone cellulaire et l'accès à des applications téléchargeables illimités, il y a des fortes possibilités de tout détruire. Étant donné qu'Android est open source, n'importe qui peut développer et publier une application sur le magasin. Ce qui augmente les risques que l'appareil tombe sur un bogue avec l'application téléchargée. Un bogue peut donner accès à distance des informations stockées dans l'appareil à un pirate ou à un simple bogue qui peut détruire le téléphone mobile en surchargeant l'utilisation de la batterie par exemple.
- ✓ Le souci de ne pas pouvoir se permettre constamment de mettre à jour son téléphone avec des versions plus récentes. Et le redémarrage impératif de téléphone avec des risques que

l'appareil ne fonctionnera pas correctement après la mise à jour. Les différentes versions et combinaisons n'ont pas été testées.

- ✓ Synchronisation automatique impossible de téléphone Android sur PC,

Il faudrait installer une application de synchronisation tierce ou synchroniser manuellement le téléphone Android avec le bureau.

2. Choix de la plateforme Android :

Android est un système ouvert, contrairement à ses deux concurrents : Microsoft et Apple. C'est un point extrêmement important car, cela signifie que la communauté de développeurs qui contribue à son développement est naturellement plus disposée à investir sur un système ouvert que sur un système rigide et soumis à des cahiers de charges trop contraignants.

De plus, Google est sans cesse en train d'innover et de proposer de nouvelles fonctionnalités gratuites, tant sur les applications web que pour les applications mobiles. Il permet d'intégrer ces dernières dans des applications personnalisées.

2.1. Les composants principaux d'une application Android : [1]

Une application Android consiste en un assemblage de composants liés via un fichier de configuration, qui présente en quelque sorte les briques sur lesquelles se repose l'application.

Ces composants fondamentaux à préciser sont :

- **Les vues (Views) :**

Les Views sont les composants basiques de l'interface graphique. Ce sont les éléments de l'interface que l'utilisateur voit et avec lesquels il agit. C'est de la classe View qu'héritent les widgets (exemple de composants graphiques tel que les boutons), les layouts (le plan sur lequel on organise et on place les composants graphiques) et tous les composants graphiques servant à la création d'une interface graphique interactive.

- **Les contrôles :**

C'est bien la classe des composants graphiques cités dessus. Les contrôles sont tels que les boutons, les champs de saisie de texte, les cases à cocher, etc.

- **Les activités :**

Une Activité représente la fenêtre qui sera affichée à l'utilisateur. C'est un ensemble de vues et de contrôles composant une interface logique. Elle permet également de gérer des fonctionnalités telles que l'appui sur une touche, l'affichage de messages, etc. Ce concept repose essentiellement sur l'interaction de l'utilisateur avec l'écran.

- **Les ressources :**

Les ressources sont des fichiers externes qui ne sont pas des programmes mais qui sont utilisés par le programme et liés à l'application au moment de sa construction.

Les ressources peuvent être de différents types :

- ✓ Des images ;
- ✓ Des gros tableaux qu'on préfère mettre à l'extérieur du programme et des chaines de caractères ;
- ✓ Des fichiers descriptifs d'interfaces graphiques ;
- ✓ D'autres ressources brutes ajoutées sans conversion.

- **Le fichier de configuration :**

C'est un fichier auto généré par l'application, qui lui est indispensable. C'est un fichier XML appelé « AndroidManifest» qui décrit le point d'entrée de l'application (le code à exécuter), les composants du projet ainsi que les permissions nécessaires pour l'exécution du programme.

Une illustration explicative de ces concepts est représentée par le schéma de la figure ci-dessous :

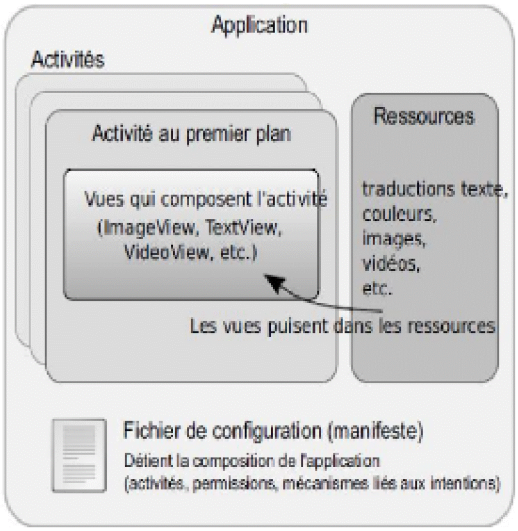


Figure II.4 : Composition d'une application Android

2.2. Les concepts d'une application Android:

Une caractéristique de base d'Android est la possibilité pour une application d'utiliser des éléments d'une autre application (à condition que cette dernière le permette).

Pour ce faire, le système doit être capable de démarré le processus d'une application quand n'importe qu'elle partie de celle-ci est demandée, et instancier les objets Java nécessaire. Pour pouvoir effectuer ceci, les applications Android, contrairement à de nombreuses applications dures d'autres systèmes, ne possède pas un unique point d'entrée pour toute l'application (pas de fonction main par exemple) [13]. A la place, les applications possèdent des composants que le système peut instancier et exécuter au besoin. Une application Android est composée d'un ou de plusieurs des composants suivants :

- Activité (Activity) : Une activité correspond à un écran de l'interface de l'application. Par exemple, une application email peut avoir une activité qui affiche la liste des nouveaux emails, une autre activité pour écrire un nouvel email et une autre activité pour lire un email particulier, …etc

Chaque activité est indépendante et peut constituer un point d'entrée de l'application. Une application peut faire appel à une activité particulière d'une autre application. Par exemple, l'appareil photo peut lancer l'activité de création de nouvel email pour envoyer une photo.

- Services : Un service est un composant qui s'exécute en arrière plan, sans interface graphique. Il a une durée de vie plus longue que l'activité, et peut s'exécuter indépendamment de toute application.

Par exemple, un service peut jouer de la musique en arrière plan pendant que l'utilisateur utilise un autre programme ou télécharge des informations du réseau.

- Fournisseurs de contenu (Content providers) : Les fournisseurs de contenu offrent un moyen de partage des données entre les applications. Les données d'une application peuvent être stockées sur le système de fichier local, sur une base de données SQLite ou sur un réseau (FTP, Web, etc).

Les autres applications peuvent accéder à ces données (ou les modifier si l'application le permet) en utilisant un content provider. Par exemple, le système Android fournit un content provider fournissant la liste des contacts téléphoniques qu'une autre application peut utiliser

- Receveurs de diffusion (Broadcast receivers) : Un receveur de diffusion est un composant qui répond à un message diffusé par le système Android ou une autre application, il est aussi appelé Intent (ou intention) et signale un événement particulier, par exemple :

✓ L'écran est passé en mode veille
✓ Le niveau de la batterie est faible
✓ l'appareil photo à pris une photo
✓ …etc

Les applications peuvent émettre leurs propres intents qui seront captés par les Boradcast receivers d'autres applications.

2.3. 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'autre application, ce que l'on appelle le cycle de vie d''une activité.

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

- **Active :**

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.

- **Suspendue :**

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 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.

- **Arrêtée :**

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.

Le schéma suivant indique le cycle de vie d'une activité et les méthodes appelées lors des changements d'état :

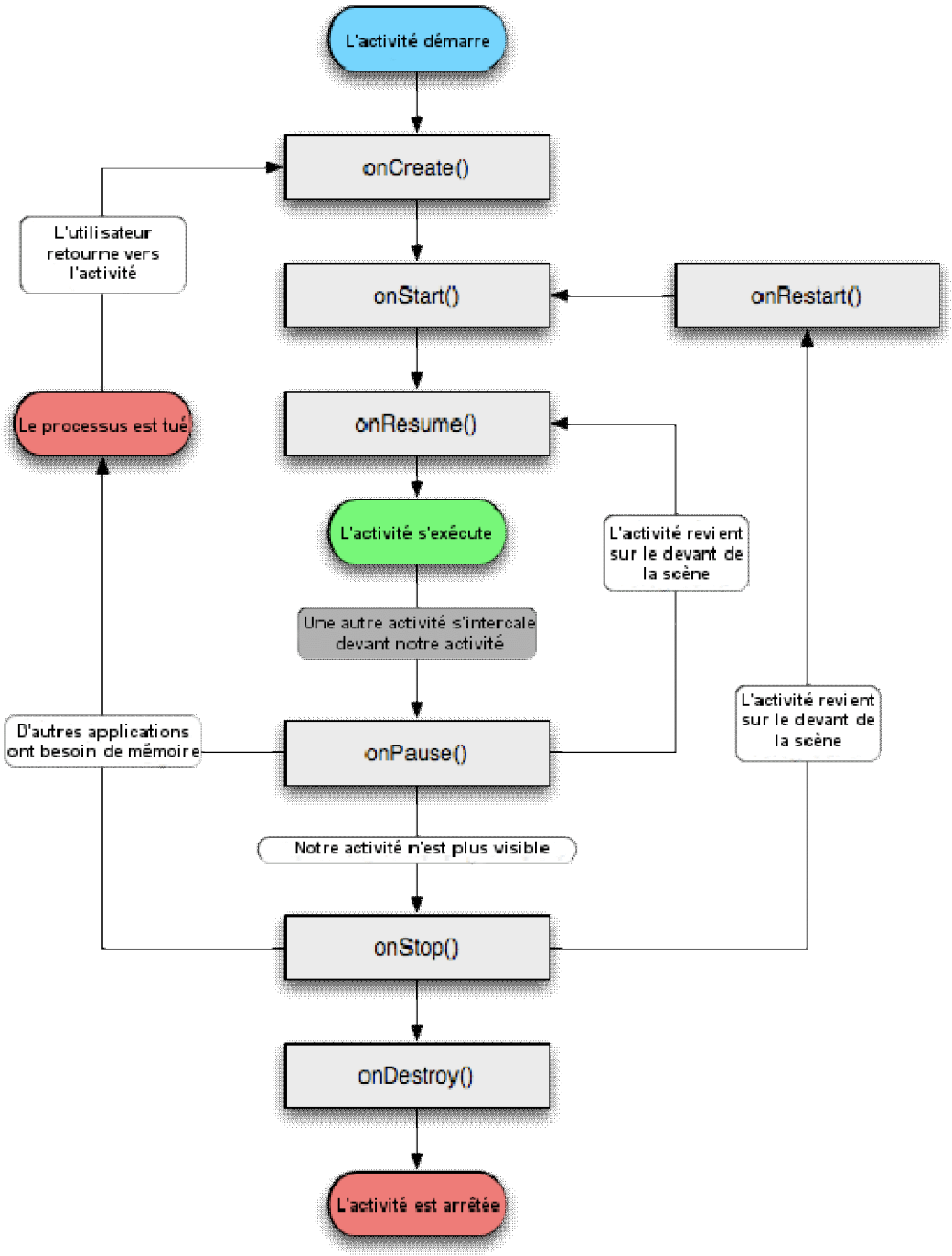


Figure II.5 : Cycle de vie d'une activité [14]

Ainsi, les différentes méthodes sont présentées ci-dessous :

- **onCreate( ) :** l'activité est en création.
- **onStart( ) :** l'activité va devenir visible.
- **onResume( ) :**l'activité est maintenant visible

- **onPause( ):** l'activité va être mise en pause.
- **onStop( ):** l'activité ne sera plus visible.
- **onDestroy( ):** l'activité va être détruite.

3. Applications de prise de notes sur le marché Android :

En bénéficiant de la plateforme Android gratuite, les développeurs ne cessent de se conquérir pour satisfaire les besoins de la communauté en offrant des solutions aux divers problèmes quotidiens, dont la prise des notes.

Quoique leur but soit le même, les applications disponibles pour les téléphone Android, traitant ce problème, sont assez nombreuses et variées. Elles se diffèrent en termes des outils employés. Plusieurs applications exigent la connexion internet, ainsi qu'elles comportent une sorte de complicité dans leur utilisation et surtout qu'elles ne sont pas ni sécurisées sur le support mobile ni gratuites à 100%.

On a présenté dans le tableau ci-dessous quelques exemples des outils de prise de notes pour Android :

Application	Description
 Evernote	Evernote permet d'enregistrer instantanément une idée pour pouvoir s'en souvenir plus tard dans la journée. Prendre des notes, des photos, créer une liste de tâches à accomplir, Evernote propose une palette d'outils très étendue.
 Google Keep	Google Keep est l'application gratuite pour Android développée par Google qui permet de prendre des notes rapidement. Il est possible de prendre des notes à l'écrit, à l'oral ou encore d'annoter une photographie. Des widgets à ajouter sur le bureau ou sur l'écran de déverrouillage sont disponibles.

 PushBullet	PushBullet est une application qui permet d'envoyer des adresses, mémos ou fichiers depuis votre PC vers le volet de notifications de votre appareil mobile.
 Note Everything	Note Everything est une application qui permet de dessiner un schéma, ou de sauvegarder une information urgente. Ce bloc-notes intègre
 OneNote	OneNote est l'outil de prise de notes pensé par le géant Microsoft. Cette application nous donne la possibilité de synchroniser des notes entre plusieurs ordinateurs (sous Windows) et appareils Android (smartphones et tablettes). Trois types de notes sont utilisables : l'inévitable note manuscrite, la capture d'un son ou encore la prise d'une photo.
 Note Anytime Lite	Note Enytime Lite est une application de prise de notes qui peut être manipulée avec un stylet. Il est possible de créer des notes en écrivant directement ces dernières à l'aide d'un stylet.
 seNotes Plus	seNotes Plus est un gestionnaire de Post-it pour Android.

Tableau II.2 : Description de quelques applications de prise de notes. [15]

Conclusion :

Dans ce chapitre, on a fait une étude de l'art du système d'exploitation Android tout en présentant un bref historique, et en mettant en évidence ses différentes versions, ces caractéristiques et son architecture. Ensuite on a étudié Android en tant que plate-forme, où on a mis en place les différents composants et concepts d'une application Android. Et enfin, nous avons étudiés quelques exemples d'applications du domaine d'étude.

Dans le prochain chapitre, on passera à l'étape analyse et conception pour pouvoir modéliser notre application, à travers différents diagrammes, permettant une meilleure compréhension des fonctionnalités du système.

Chapitre III

Analyse & Conception

Introduction :

Dans le but d'une meilleure organisation et une bonne maitrise de travail, tout processus de développement d'application ou de systèmes informatiques doit suivre une méthode ou une démarche bien définie.

Dans ce chapitre, on va entamer le processus par une analyse qui mettra en évidence les différents acteurs intervenant dans le système cible ainsi que leurs besoins. La phase conception, s'appuyant sur les résultats de la phase analyse, donnera la modélisation des objectifs à atteindre. Pour ce faire, notre démarche va s'appuyer sur le langage UML, conçu pour la visualisation, la spécification et la construction des systèmes logiciels.

1. Objectif de l'application :

L'objectif de notre application est de permettre aux utilisateurs de se souvenir de tout, en enregistrant leurs idées, les organiser, et les regrouper dans des listes. Ils auront également la possibilité de chercher des utilisateurs, les ajouter à fin de pouvoir partager des notes en toute sécurité.

2. Modélisation du système :

La modélisation consiste à créer une représentation simplifiée d'un problème, un concept et le simuler. Elle comporte deux composantes :

- L`analyse, consiste à l`étude du problème ;
- La conception, soit la mise au point d`une solution au problème.

2.1. Présentation du langage UML : [20]

UML (Unified Modeling Language) est un langage de modélisation formel et normalisé, né de la fusion de plusieurs méthodes existantes. Il permet de modéliser informatiquement un ensemble d'éléments cohérents du monde réel en un ensemble d'entités informatiques. Ces entités informatiques sont appelées objets. Ces objets sont décris par des vus statiques et dynamiques, incluant un ensemble de diagrammes, qui collaborent pour représenter diverses projections d'une même représentation d'un système d'objets.

UML possède 9 diagrammes de modélisation, repartis sur trois axes du niveau conceptuel :

- ✓ Statique
- ✓ Dynamique
- ✓ Fonctionnel

2.2. Modélisation avec le langage UML : [21]

UML permet de représenter des modèles, mais il ne définit pas de processus d'élaboration de modèles. Les auteurs d'UML conseillent tout de même une démarche pour favoriser la réussite d'un projet, cette démarche met en œuvre :

- **Une démarche itératif et incrémentale :** Pour comprendre et représenter un système complexe, pour analyser par étapes, pour favoriser le prototypage et pour réduire et maitriser l'inconnu.
- **Une démarche guidée par les besoins des utilisateurs :** Tout est basé sur le besoin des utilisateurs du système, le but du développement lui-même est de répondre à leur besoin. Chaque étape sera affinée et validée en fonction des besoins des utilisateurs.
- **Une démarche centrée sur l'architecture logicielle :** c'est la clé de voute de succès d'un développement, les choix stratégiques définiront la qualité du logiciel.

2.3. La démarche de modélisation avec l'UML :

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

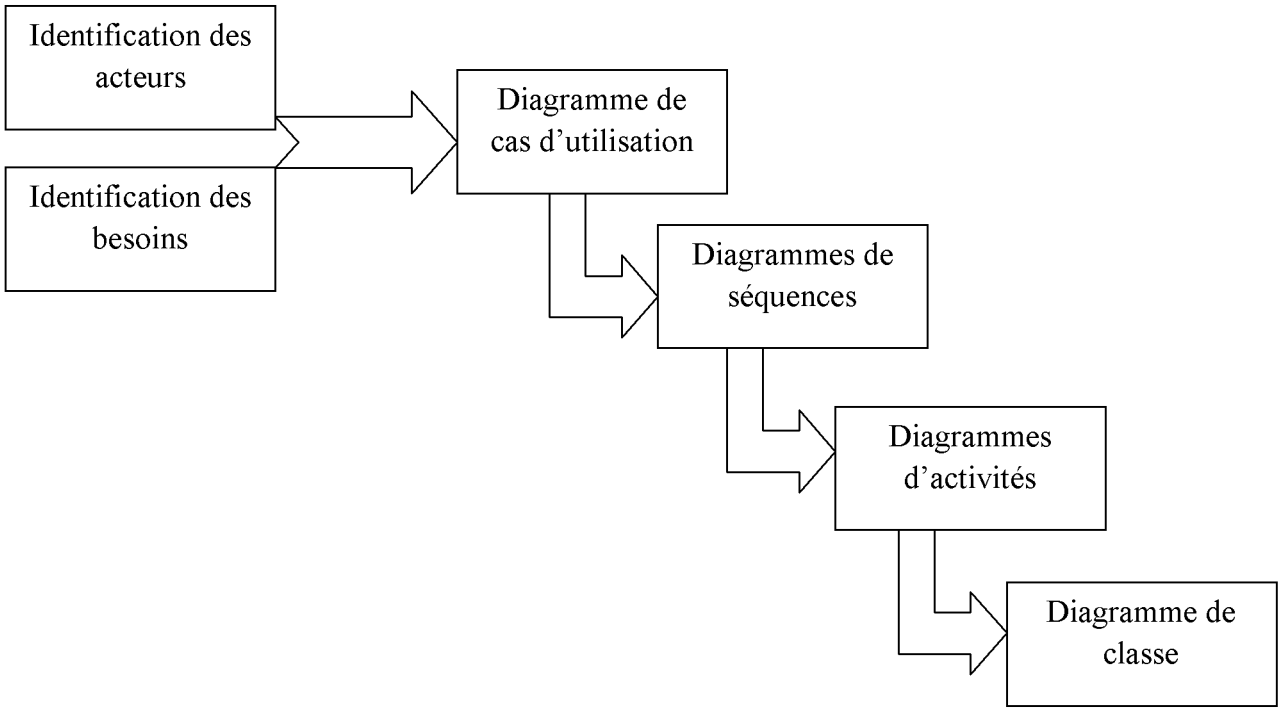


Figure III.1 : Démarche adoptée pour la modélisation

3. Analyse :

Cette partie comprend l'identification des besoins du système, des acteurs et leurs interactions avec le système ainsi que les cas d'utilisation:

3.1. Identification des besoins :

L'utilité d'un système logiciel est déterminée par ses exigences fonctionnelles et ses caractéristiques non-fonctionnelles, ainsi que ses contraintes. A titre de rappel que :

- Une exigence fonctionnelle est une exigence définissant une fonction du système à développer : Ce que le système doit le faire.
- Une exigence non-fonctionnelle est une exigence qui caractérise une propriété (qualité) désirée du système telle que sa performance, sa robustesse, sa convivialité, sa maintenabilité, etc.
- Une contrainte est une restriction sur une ou plusieurs valeurs d'une partie du système ou de tout le système.

3.1.1. Les besoins fonctionnels :

Cette section a pour objet de présenter les besoins fonctionnels aux quels doit répondre notre projet. Alors, « Mes Notes » doit permettre à l'utilisateur de :

- **Gérer des notes :**

- Un utilisateur peut créer, supprimer, afficher et modifier une note.
- Un utilisateur peut partager une note avec un ou plusieurs contacts.

- **Gérer des listes des notes :**

Un utilisateur peut créer, supprimer, afficher et modifier une liste dont il peut y regrouper ses notes.

- **Gérer des contacts :**

Un utilisateur peut ajouter, afficher et supprimer un contact.

- **Gérer des groupes des contacts :**

Un utilisateur peut ajouter, modifier, afficher et supprimer un groupe à fin de regrouper ses contacts.

- **Modification de mot de passe :**

Un utilisateur peut, à chaque fois, modifier son mot de passe.

3.1.2. Les besoins non fonctionnels :

Concevoir un service à la fois riche, intuitif et adapté à la taille d'un écran de Smartphone n'est pas une chose aisée. Les besoins non fonctionnels spécifient les propriétés du système telles que les contraintes d'environnement et d'implémentation, la performance, la maintenance, l'extensibilité et la flexibilité. Certains besoins non fonctionnels sont généraux et ne peuvent pas être rattachés à un cas d'utilisation particulier.

3.1.2.1. Contraintes ergonomiques :

Les contraintes ergonomiques sont les contraintes liées à l'adaptation entre les fonctionnalités de l'application, leurs interfaces et leur utilisation.

Pour notre projet, on doit obéir aux contraintes ergonomiques suivantes :

- Permettre un accès et une recherche rapide de l'information.

- Interface simple et compréhensible.

3.1.2.2. Contraintes techniques :

-L'accès à la base de données doit être souple et rapide.

-L'application doit être toujours fonctionnelle.

-Espace de stockage des données suffisant.

-Temps de réponse est minimum.

-L'application doit garantir la sécurité.

-Communiquer des données entre deux environnements hétérogènes : Protocole de communication, format des données…

3.1.2.3. Contraintes du matériel :

-L'application sera installée que sur un téléphone mobile possédé d'un OS Android.

3.1.2.4. Contraintes de déploiement :

- L'application doit être téléchargeable à partir de Google Play.

3.2. Identification des acteurs :

On va maintenant énumérer les acteurs susceptibles d'interagir avec le système. Tout d'abord, on commence par définir ce qui est un acteur.

Définition : Un acteur représente un ensemble de rôles joués par des entités externes (utilisateur humain, dispositif matériel ou autre système) qui interagissent directement avec le système étudié. [22]

Dans le cas de notre application, il y'a un seul acteur qui interagit avec le système qui est :

- ✓ **Utilisateur :** cet acteur représente les propriétaires des téléphones Android.

3.3. Les cas d'utilisation:

3.3.1. Définition : [22]

Un cas d'utilisation (**user case**) représente un ensemble de séquence d'actions qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier. Un cas d'utilisation modélise un service rendu par le système.

3.3.2. Description des cas d'utilisation :

Nous procéderons aux descriptions des cas d'utilisation dans le système :

Cas d'utilisation**: **Authentification

***Rôle**: utilisateur*

***Description principale** :*

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. Apres chargement de l'application, une interface d'authentification s'affiche.
3. L'utilisateur saisit son nom et son mot de passe.
4. le système vérifie si le nom d'utilisateur et le mot de passe sont corrects ou non.

Description alternative :

5. Le nom d'utilisateur et le mot de passe sont corrects, alors affichage du menu principal.
6. Le nom d'utilisateur et/ou le mot de passe sont incorrects, alors affichage d'un message d'erreur.

Figure III.2 : Cas d'utilisation « Authentification »

***Cas d'utilisation**: Gérer des notes*
***Rôle**: utilisateur*
***Description 1**:*

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « note ».
5. L'utilisateur choisi l'icône « ajouter une note ».
6. Le système affiche l'interface « ajouter note ».
7. l'utilisateur saisit ses données nécessaires.
8. l'utilisateur confirme la création d'une nouvelle note.
9. une notification d'ajout avec succès s'affiche.

Description 2 :

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « note ».
5. L'utilisateur choisi l'icône « chercher une note ».
6. L'utilisateur clique sur la note affichée.
7. L'utilisateur peut modifier la note affichée.

Description 3 :

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « note ».
5. L'utilisateur choisi l'icône « chercher une note ».
6. L'utilisateur clique sur la note affichée.
7. L'utilisateur clique sur l'icône ″Supprimer″.
8. Une boite de dialogue s'affiche pour confirmer ou annuler.

Figure III.3: Cas d'utilisation « Gérer des notes »

***Cas d'utilisation**: Gérer des listes de notes*
***Rôle**: utilisateur*
***Description 1**:*

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « liste ».
5. L'utilisateur choisi l'icône « ajouter une liste ».
6. Le système affiche l'interface « ajouter liste ».
7. l'utilisateur confirme la création d'une nouvelle liste.
8. une notification d'ajout avec succès s'affiche.
9. l'utilisateur peut ajouter des notes à sa liste.

Description 2 :

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « liste ».
5. L'utilisateur choisi l'icône « chercher une liste ».
6. L'utilisateur clique sur la liste affichée.
7. L'utilisateur peut modifier la liste des notes affichée.

Description 3 :

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « liste ».
5. L'utilisateur choisi l'icône « chercher une liste ».
6. L'utilisateur clique sur la liste des notes affichée.
7. L'utilisateur clique sur l'icône ″Supprimer″.
8. Une boite de dialogue s'affiche pour confirmer ou annuler.

Figure III.4: Cas d'utilisation « Gérer des listes de notes »

***Cas d'utilisation**: Gérer des contacts*

***Rôle**: utilisateur*

***Description 1**:*

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « contacts ».
5. L'utilisateur choisi l'icône « ajouter un contact ».
6. Le système affiche une liste des contacts.
7. l'utilisateur choisi et confirme l'ajout d'un contact.

Description 2:

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône «contact ».
5. L'interface de « contacts » s'affiche.
6. L'utilisateur choisit et clique sur un parmi les contacts affichés.
7. L'utilisateur demande l'interface « supprimer contact »
8. Une boite de dialogue s'affiche pour confirmer ou annuler

Figure III.5: Cas d'utilisation « Gérer des contacts»

***Cas d'utilisation**: Gérer des groupes de contacts*

***Rôle**: utilisateur*

***Description 1**:*

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « groupe ».
5. L'utilisateur choisi l'icône « ajouter un groupe ».
6. Le système affiche l'interface « ajouter groupe ».
7. l'utilisateur confirme la création d'un nouveau groupe.

Description 2 :

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « groupe ».
5. L'utilisateur choisi l'icône « chercher un groupe».
6. L'utilisateur clique sur le groupe affiché.
7. L'utilisateur peut effectuer des modifications sur le groupe affiché.

Description 3 :

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi l'icône « groupe ».
5. L'utilisateur choisi l'icône « chercher un groupe ».
6. L'utilisateur clique sur le groupe affiché.
7. L'utilisateur peut supprimer le groupe affiché.

Figure III.6: Cas d'utilisation « Gérer des groupes de contacts »

***Cas d'utilisation**: Changement du mot de passe.*

***Rôle**: utilisateur*

Description :

1. L'utilisateur lance l'application après avoir cliquer sur l'icône.
2. L'utilisateur s'authentifie.
3. Le système affiche le menu principal.
4. L'utilisateur choisi le sous menu « Changement du mot de passe ».
5. Le système affiche l'interface « Changement du mot de passe ».
6. L'utilisateur saisie l'ancien mot de passe.
7. L'utilisateur saisie le nouveau mot de passe et clique sur le bouton ″Valider″.
8. Le système affiche un message de confirmation.
9. L'utilisateur clique sur le sous menu ″Déconnexion″.

Figure III.7: Cas d'utilisation « Changement du mot de passe ».

4. Conception :

Le processus de conception de notre projet se caractérise par deux niveaux : le niveau Applicatif et le niveau Données.

Le niveau applicatif s'appuie essentiellement sur quelques diagrammes de l'extension du langage de modélisation UML. A cet effet nous avons adopté la démarche suivante :

- ✓ Après l'identification des différents acteurs ainsi que les cas d'utilisation qui sont mis en œuvre par ces acteurs, le diagramme de cas d'utilisation est élaboré.
- ✓ Chaque cas d'utilisation se traduit par un ou plusieurs scénarios. Chaque scénario fait l'objet d'une description sous forme graphique à l'aide d'un diagramme de séquence et un diagramme d'activité.
- ✓ Une identification des classes est fournie par la synthèse des diagrammes de séquence, ainsi le diagramme de classe sera élaboré.

Le niveau Données concerne l'organisation conceptuelle, logique et physique des données manipulées. Durant la partie analyse nous avons pu identifier les données nécessaires et indispensables au

bon fonctionnement de l'application et à travers la conception du niveau applicatif nous allons dégager les classes significatives, dès lors on peut élaborer la conception de la base de données.

4.1. Le niveau Applicatif :

4.1.1. Le diagramme des cas d'utilisation :

Lors de la phase d'analyse nous avons pu identifier les acteurs ainsi que les cas d'utilisation associés à ces derniers. Ce qui nous donne l'opportunité d'élaborer le diagramme des cas d'utilisation :

Définition : Les diagrammes de cas d'utilisation permettent de représenter un ensemble de cas d'utilisation, d'acteurs et leurs relations. Ils présentent la vue statique des cas d'utilisation d'un système et sont particulièrement importants dans l'organisation et la modélisation des comportements d'un système. [23]

- ✓ **La relation d'inclusion (include) :** Elle indique que le cas d'utilisation source contient aussi le comportement décrit dans le cas d'utilisation destination. Cette relation permet de décomposer des comportements et de définir les comportements partageables entre plusieurs cas d'utilisation. [23]
- ✓ **La relation d'extension (extend) :** Elle indique que le cas d'utilisation source ajoute son comportement au cas d'utilisation destination. L'extension peut être soumise à des conditions. [23]

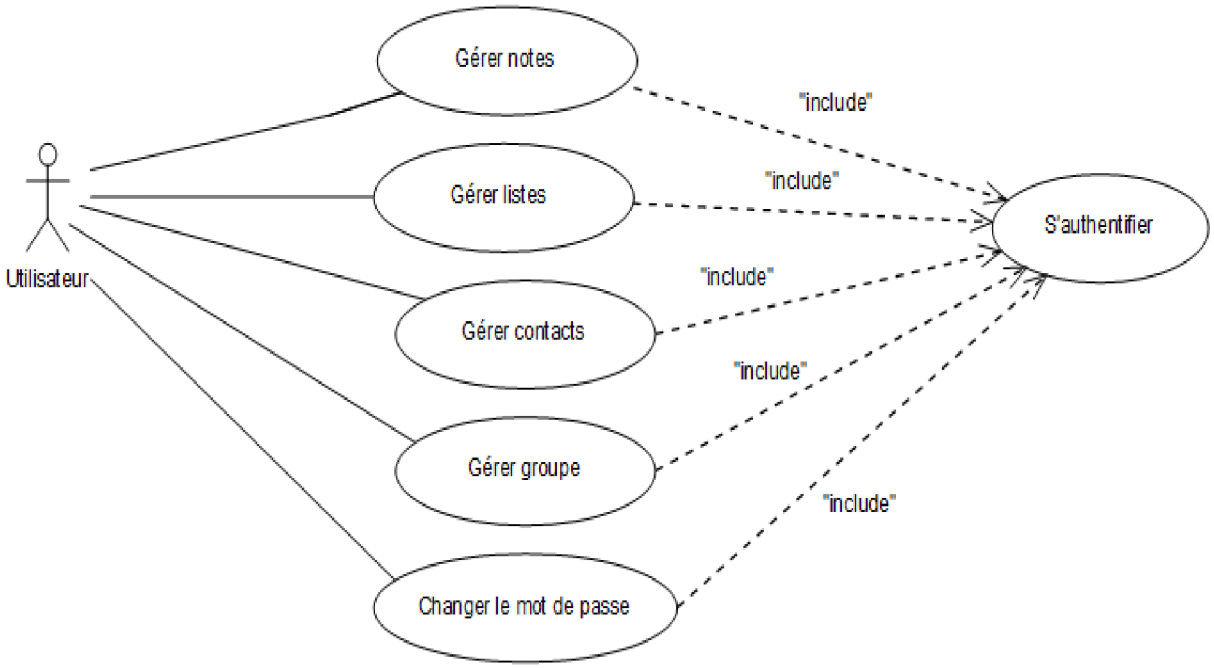


Figure III.8 : Diagramme de cas d'utilisation général.

4.1.2. Les diagrammes de séquences:

Un diagramme de séquence permet de spécifier les interactions qui existent entre un groupe d'objet selon un point de vue temporel, on y met l'accent sur la chronologie des envois de message. Le diagramme de séquence est toujours de haut vers le bas, il illustre l'ordre dans lequel les messages sont envoyés entre les objets [24]. Il peut servir à illustrer un cas d'utilisation.

Les composants d'un diagramme de séquence sont les suivants :

Les objets : ils apparaissent dans la partie supérieure, ce qui facilite l'indentification des classes qui participent à l'interaction.

Les messages : ils sont représentés par des flèches directionnelles. Au-dessus des flèches directionnelles figurent un texte nous informant du message envoyé entre les objets.

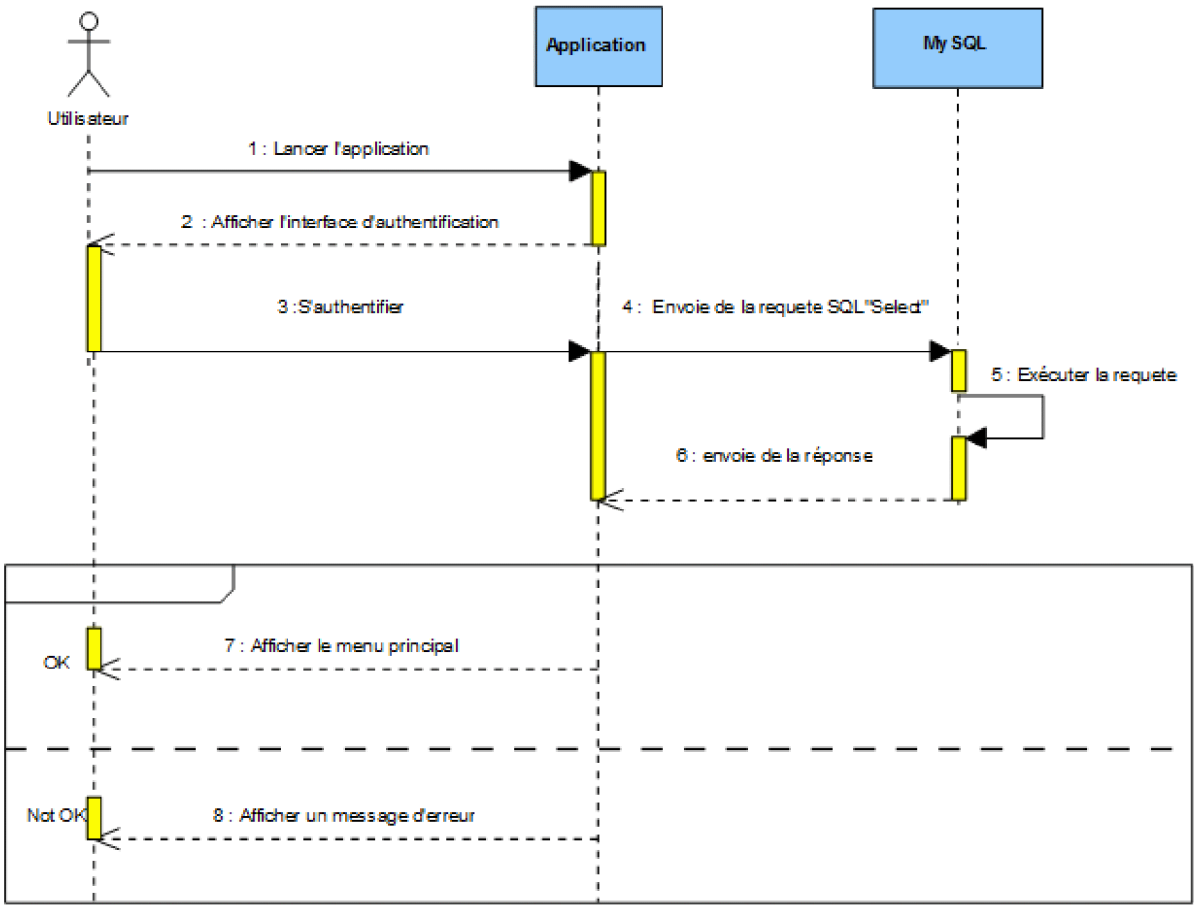


Figure III.9 : Diagramme de séquence de cas d'utilisation « Authentification ».

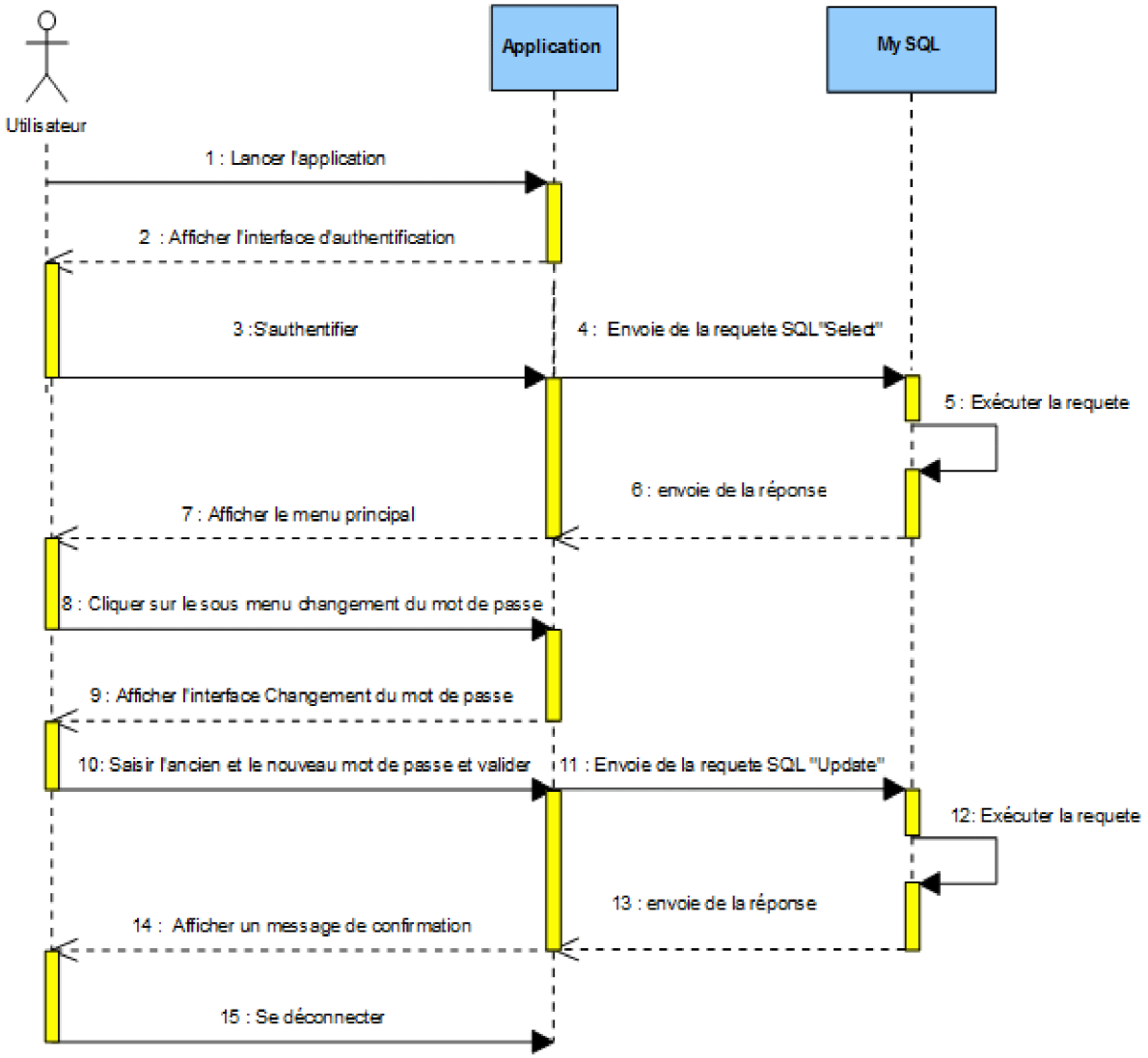


Figure III.10 : Diagramme de séquence de cas d'utilisation « Changement du mot de passe».

4.1.3. Les diagrammes d'activités:

Un diagramme d'activités apporte un point de vue complémentaire à l'aspect dynamique de la modélisation. Il offre un pouvoir d'expression très proche des langages de programmation objets [24]. Il est donc bien adapté à la spécification détaillée des traitements en phase de réalisation. Un diagramme d'activités se concentre plutôt sur les activités entre les objets, c'est-à-dire, il met en évidence l'activité qui a lieu dans le temps, donc les opérations transmises entre les objets.

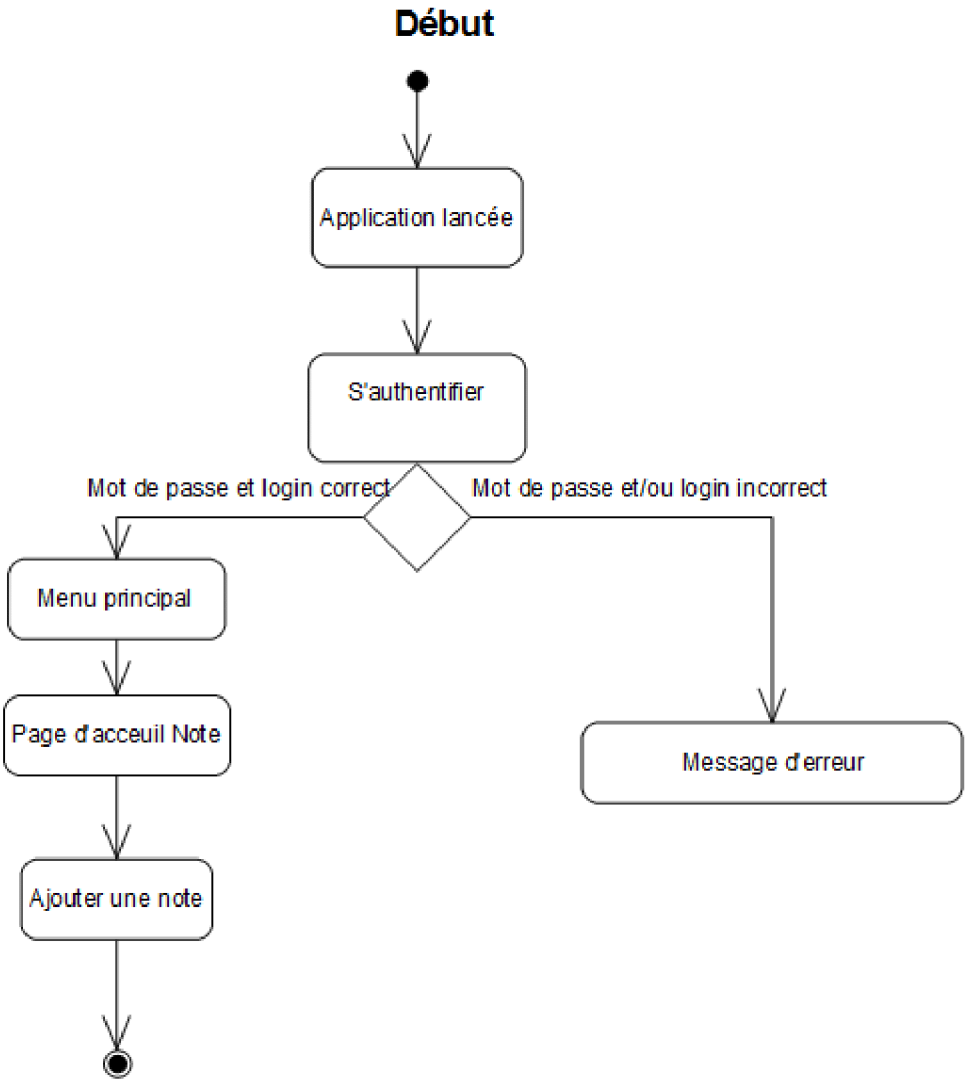


Figure III.11 : Diagramme d'activité « Ajouter note ».

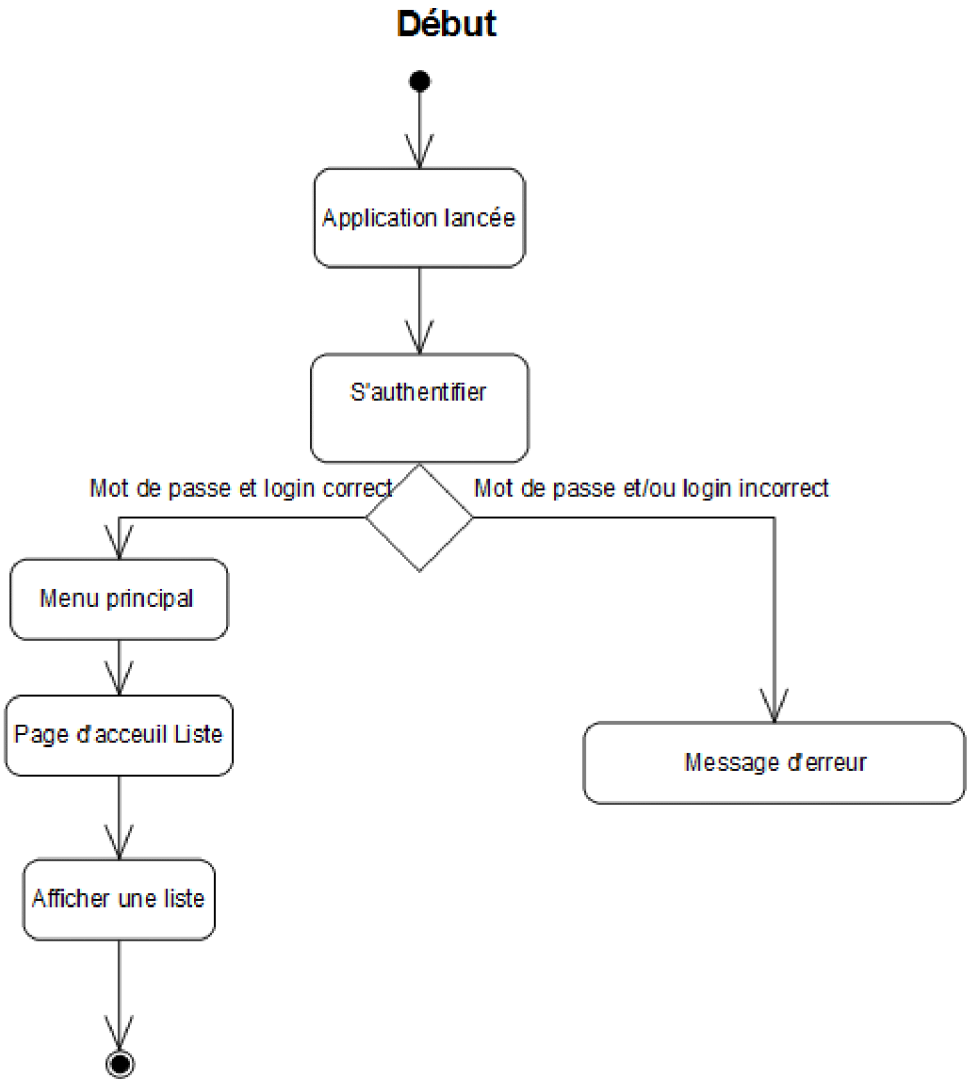


Figure III.12 : Diagramme d'activité « Afficher liste ».

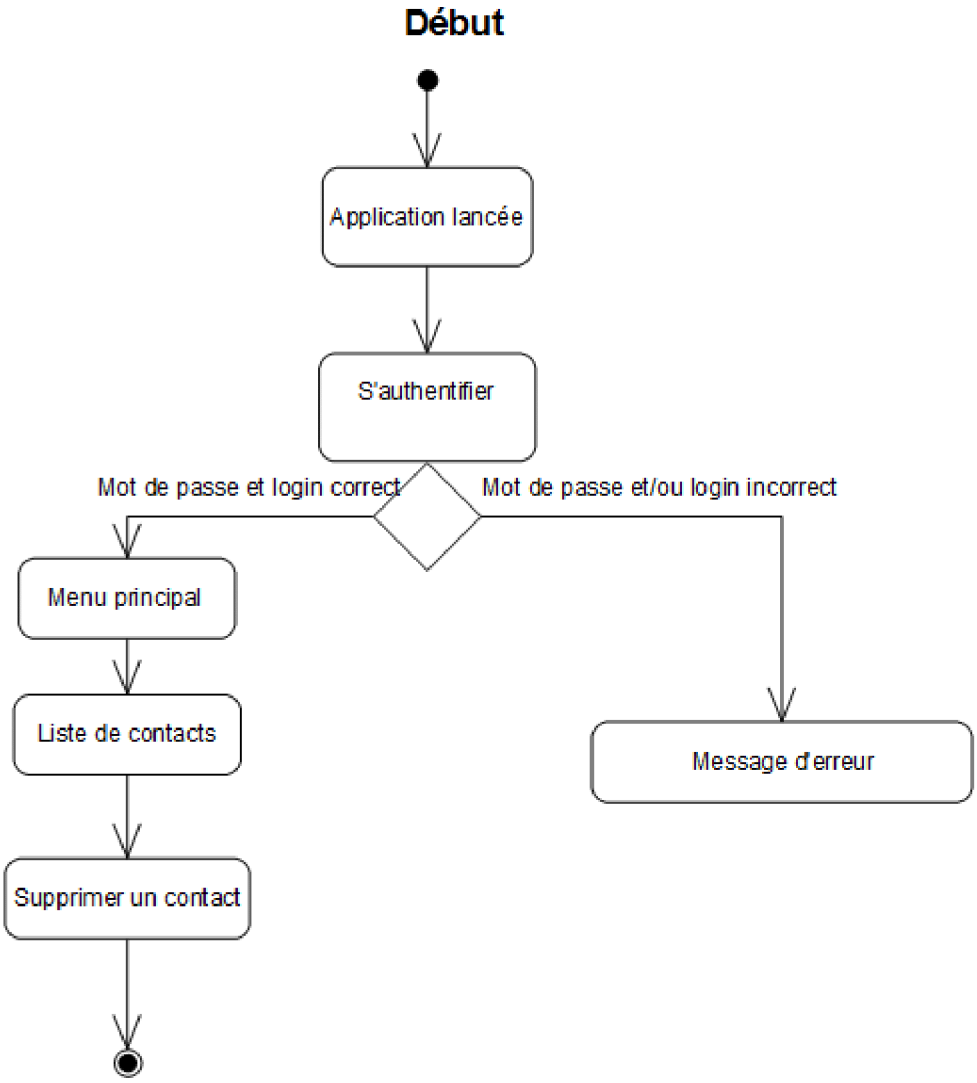


Figure III.13 : Diagramme d'activité « Supprimer contact ».

4.1.4. Le diagramme de classe:

Dans cette partie nous allons passer à la modélisation de l'aspect statique de notre application, c'est-à-dire nous allons modéliser l'intérieur de notre système. Pour ce faire, nous allons utiliser le diagramme de classes :

Définition : Le diagramme de classe est considéré comme le plus important de la modélisation orientée objet. Alors que les diagrammes précédents (diagrammes de séquences et diagrammes d'activités) montrent le système du point de vue dynamique, le diagramme de classe en montre

la structure interne. Il contient principalement des classes reliées par des associations et chaque classe contient des attributs et des opérations. [24]

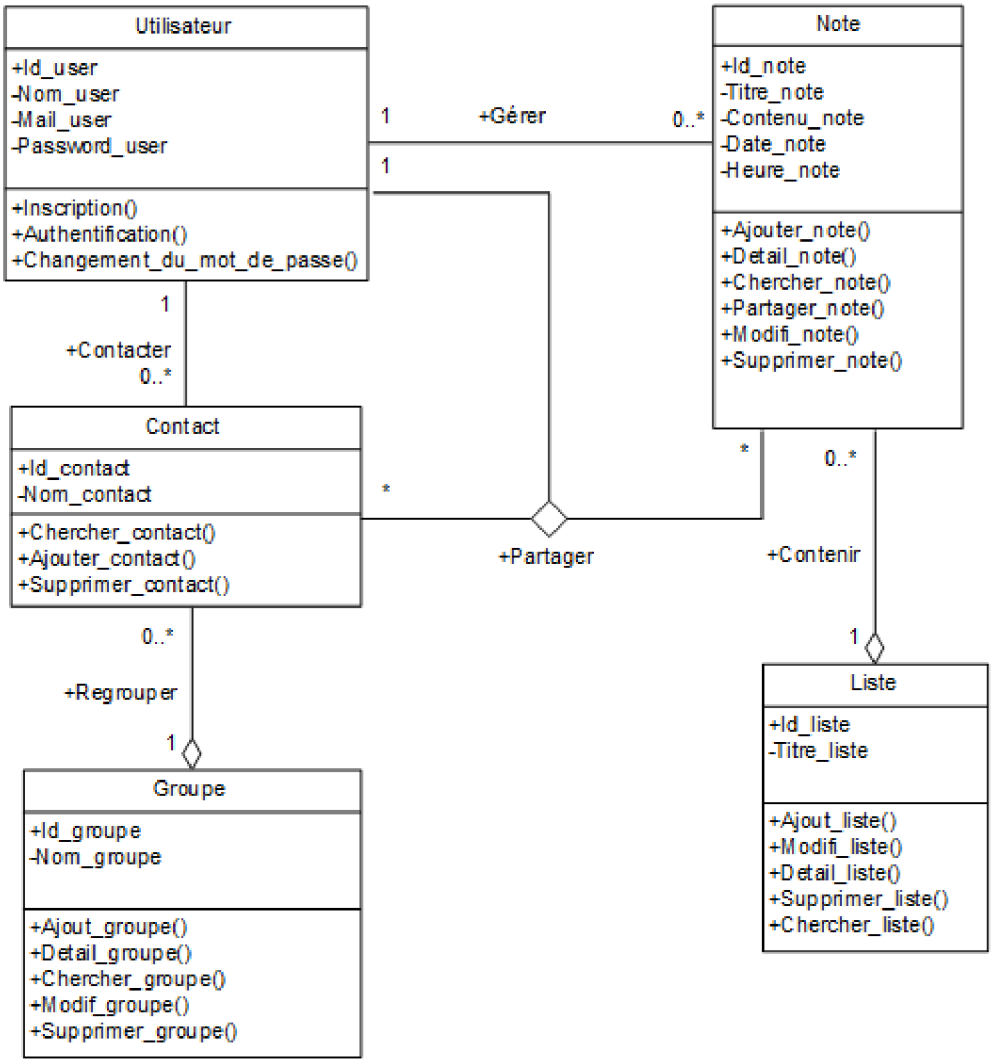


Figure III.14 : Diagramme de classe.

4.2. Le niveau Données :

4.2.1. Les tables :

L'application dans tout ces cas d'utilisation manipule les données stockées dans la base de données, et elle contient les tables suivantes :

La table Liste :

Champ	Signification	Type	Clé
Id_liste	L'identifiant unique pour chaque liste, auto-incrémenté.	INT	Primaire (Primary Key)
Titre_liste	Le titre de la liste.	VARCHAR	

Tableau III.1 : Table Liste.

La table Groupe :

Champ	Signification	Type	Clé
Id_groupe	L'identifiant unique pour chaque groupe, auto-incrémenté.	INT	Primaire (Primary Key)
Nom_groupe	Le nom du groupe.	VARCHAR	

Tableau III.2 : Table Groupe.

La table Contact :

Champ	Signification	Type	Clé
Id_contact	L'identifiant unique pour chaque contact, auto-incrémenté.	INT	Primaire (Primary Key)
Nom_contact	Le nom du contact.	VARCHAR	
Id_groupe	L'identifiant de groupe qui contient le contact.	INT	Etrangère (Foreign Key)

Tableau III.3 : Table Contact.

La table Utilisateur :

Champ	Signification	Type	Clé
Id_user	L'identifiant unique pour chaque utilisateur, auto-incrémenté.	INT	Primaire (Primary Key)
Nom_user	Le nom de l'utilisateur.	VARCHAR	
Mail_user	L'adresse e-mail de l'utilisateur.	VARCHAR	
Password_user	Le mot de passe de l'utilisateur.	VARCHAR	

Tableau III.4 : Table Utilisateur.

La table Note:

Champ	Signification	Type	Clé
Id_note	L'identifiant unique pour chaque note, auto-incrémenté.	INT	Primaire (Primary Key)
Titre_note	Le titre de la note.	VARCHAR	
Contenu_note	Le contenu de la note.	VARCHAR	
Date_note	La date de création de la note.	Date	
Heure_note	L'heure de création de la note.	VARCHAR	
Id_liste	L'identifiant de la liste qui contient la note.	INT	Etrangère (Foreign Key)
Id_user	L'identifiant de l'utilisateur qui a créé la note.	INT	Etrangère (Foreign Key)
Id_contact	L'identifiant de contact avec qui l'utilisateur a partagé la note.	INT	Etrangère (Foreign Key)

Tableau III.5 : Table Note.

Conclusion :

Dans ce chapitre nous avons proposé une démarche de modélisation pour développer notre application, en commençant par la spécification des cas d'utilisations suivie d'une description complète des différents diagrammes pour une meilleure compréhension du système.

La réalisation du projet sera détaillée dans le prochain chapitre.

Chapitre IV

Réalisation et mise en œuvre

Introduction :

La tâche de réalisation ou d'implémentation est la phase finale de l'élaboration d'un système. Elle permet aux matériels et logiciels d'entrer en fonction pour passer de l'expression d'un besoin informatique à un système fonctionnel fiable. Cette phase consiste à traduire la conception exprimée à l'aide d'un formalisme en un code source écrit dans un langage donnée.
Pour ce faire, on va exposer dans ce qui suit, l'environnement matériel et logiciel dans lesquels notre application a été développée en indiquant les technologies utilisées. Nous clôturons ce chapitre par quelques captures d'écran traduisant le déroulement de l'application.

1. Environnement de travail :

1.1. Environnement matériel :

Pour la réalisation de notre application, on a utilisé un PC portable de marque "Packard Bell", ayant les caractéristiques suivantes :

- ✓ Micro processeur : Intel® Core™ i3.
- ✓ CPU: M370 @ 2.40 GHZ.
- ✓ Disque dur: 500 GO.
- ✓ RAM: 4.00 GO.
- ✓ Clavier AZERTY, 105 touches.

1.2. Environnement logiciel :

L'environnement logiciel employé s'illustre en :

- ✓ Un Système d'exploitation Windows 7 professionnel.
- ✓ Eclipse comme environnement de développement JAVA, sur lequel on a installé le plugin ADT.
- ✓ Android SDK 4.4.2 (version KitKat).
- ✓ Pacestar UML Diagrammer, pour la création des différents diagrammes élaborés.

2. Les technologies utilisées :

2.1. Le langage de programmation JAVA :

Java est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C++.

Approche objets :

La programmation orienté objet est définie comme un paradigme de programmation informatique basé sur l'interaction de briques appelées objets, qui sont eux-mêmes des abstractions d'objets réels, via leurs relations et les propriétés qui leur sont accordées.

Ce paradigme assure :

Une modularité des programmes résolvant le problème de la complexité des codes.

Une vitesse d'exécution.

Intégration des traitements aux données.

Encapsulation.

Une souplesse de réutilisation et d'évolution du code.

Parmi les caractéristiques Java qui ont contribués au succès de ce langage, on retrouve la

propriété de portabilité : le code des applications développées dans ce langage est indépendant de la machine sur laquelle il s'exécute. Le code passe par une phase de compilation pour générer du « byte code », pour pouvoir être exécuté par la JVM (Java Virtuel Machine) qui le transformera en code natif pour la machine.

Il est donc possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtuel Machine, il s'agit du principe « Compile Once, run everywhere ». Comme le montre la figure suivante :

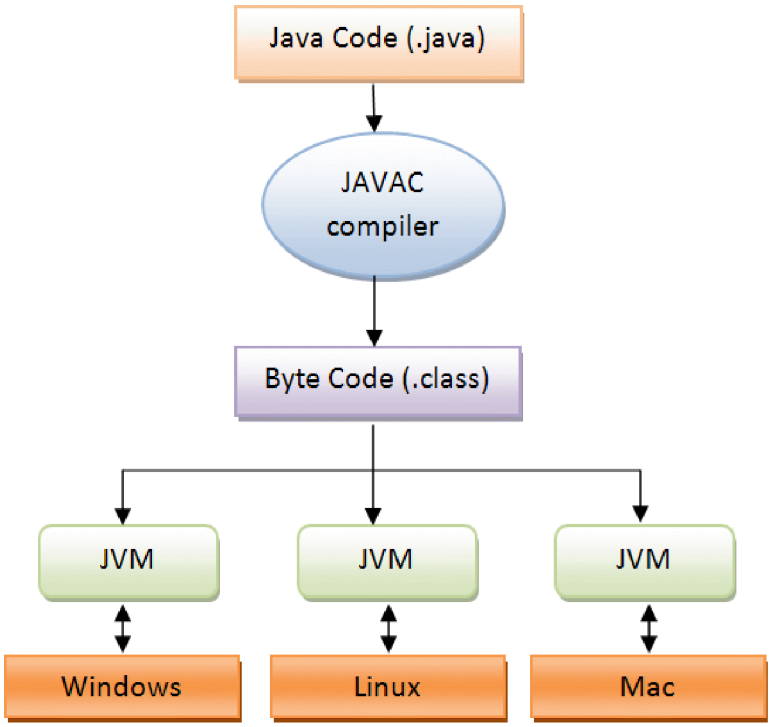


Figure IV.1 : Le concept de machine virtuelle java

2.2. L'IDE Eclipse:

Eclipse est un environnement de développement open source et extensible, développé par IBM en 2004, initialement destiné à Java, mais par sa gestion de plugins il est aujourd'hui possible d'utiliser Eclipse avec presque toutes les technologies. Lui-même écrit en Java, il est disponible pour toutes les plateformes. Eclipse est libre (sous licence Eclipse Public Licence) et donc gratuit.

Pour le développement de notre application, on a choisi Eclipse JUNO, son interface principale est donnée dans la figure suivante :

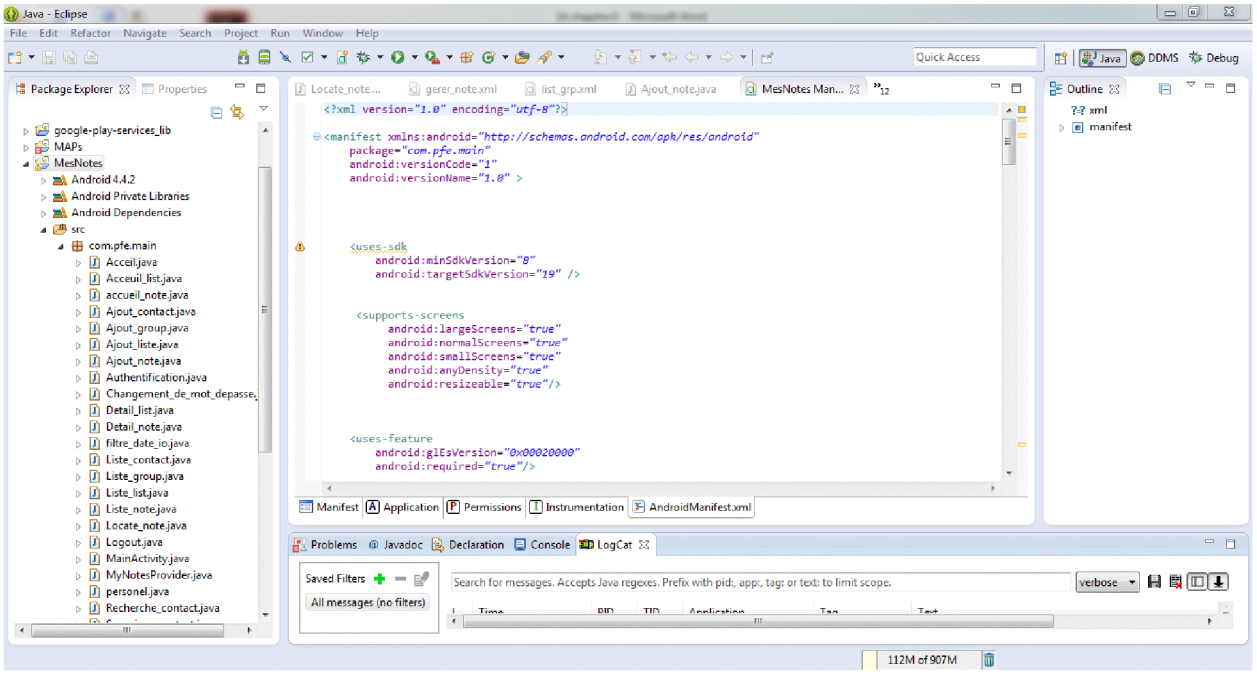


Figure IV.2 : Plateforme Eclipse

2.3. Le plugin ADT :

Le plugin ADT pour Eclipse permet de créer et de déboguer facilement et rapidement les applications :

- Il fournit un nouveau gestionnaire de projet, qui aide à initialiser tous les fichiers de base dont on a besoin pour une nouvelle application Android.
- Il fournit un éditeur Android qui aide à écrire des fichiers XML valides pour Android.
- Il permet aussi d'exporter un projet dans un fichier APK signé. Il est important de noter que pour être installé sur un système Android, tout package APK doit être préalablement signé.

2.4. Le SDK Android (Software Development Kit) :

Un SDK est un kit de développement, propre à chaque système d'exploitation. Il est entre autre, utilisé pour développer des applications sous Android, il est gratuitement mis à la disposition des développeurs par Google. Le développement pour la plate-forme Android nécessite une connaissance optimale en programmation Java et se fait via d'autres logiciels tels qu'Eclipe IDE. [20]

2.5. Le formalisme XML :

XML (Extensible Markup Language) est un langage informatique de balisage générique. Il sert essentiellement à stocker/transférer des données de type texte Unicode structurées en champs arborescents.

Sous Android, et grâce à ce langage, les interfaces sont décrites dans un format spécial, et Android les convertit automatiquement en objets Java qui seront par la suite disponibles comme tout autre objet du code de l'application. Il offre ainsi plus de souplesse de développement, facilite les modifications du code et assure la séparation entre la présentation et le comportement des objets.

2.6. La plateforme WampServer :

WampServer est une plate-forme de développement Web sous Windows pour des applications Web dynamiques à l'aide du serveur Apache2, du langage de scripts PHP et d'une base de données MySQL. Il possède également PHPMyAdmin pour gérer plus facilement vos bases de données. [21]

- **Le langage PHP :**

PHP (officiellement, ce sigle est un acronyme récursif pour PHP : Hypertext Preprocessor) est un langege de scripts généraliste et Open Source, spécialement conçu pour le développement d'applications web. Il peut être intégré facilement au HTML. [22]

- **Le SGBD MySQL :**

Le SGBD MySQL est un système de gestion de base de données relationnelles SQL développé dans un souci de performances élevées, il est multi-thread, multi-utilisateurs. C'est un logiciel libre développé sous double licence en fonction de l'utilisation qui est faite dans un produit libre (open-source) ou dans un produit propriétaire.

Le couple PHP/MySQL est largement utilisé pour la création de sites web dynamiques et proposé par la majorité des hébergeurs.

Le SGBD MySQL est fréquemment utilisé avec une interface de gestion appelée ″phpMyAdmin″ qui permet à un non-initié de générer des requêtes SQL sur une base de données.

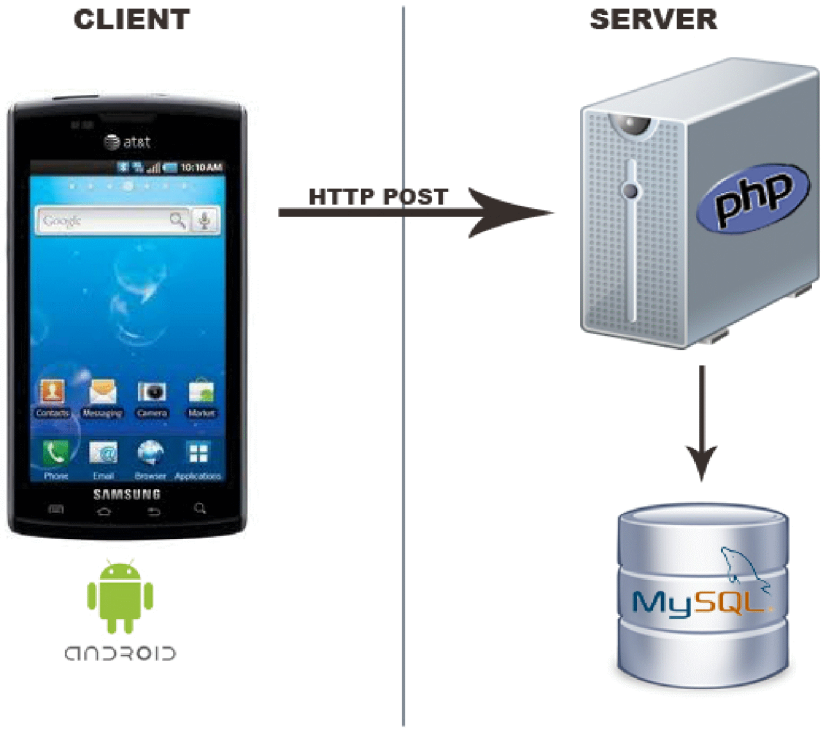


Figure IV.3 : Plateforme client/serveur [23]

3. Fonctionnement de l'application :

On va vous présenter quelques interfaces graphiques (captures d'écran) de l'application réalisée, ainsi que leurs descriptions :

3.1. La fenêtre d'authentification :

Figure IV.4 : Interface d'authentification

L'interface d'authentification est affichée directement après le chargement de l'application afin de garder la sécurité des données personnelles.

3.2. La fenêtre d'inscription :

Figure IV.5 : Interface d'inscription.

L'interface d'inscription est affichée après avoir cliqué sur le bouton "S'inscrire". Si l'utilisateur n'est pas encore inscrit, il doit remplir les champs pour pouvoir connecter à l'application.

3.3. La fenêtre « menu principal » :

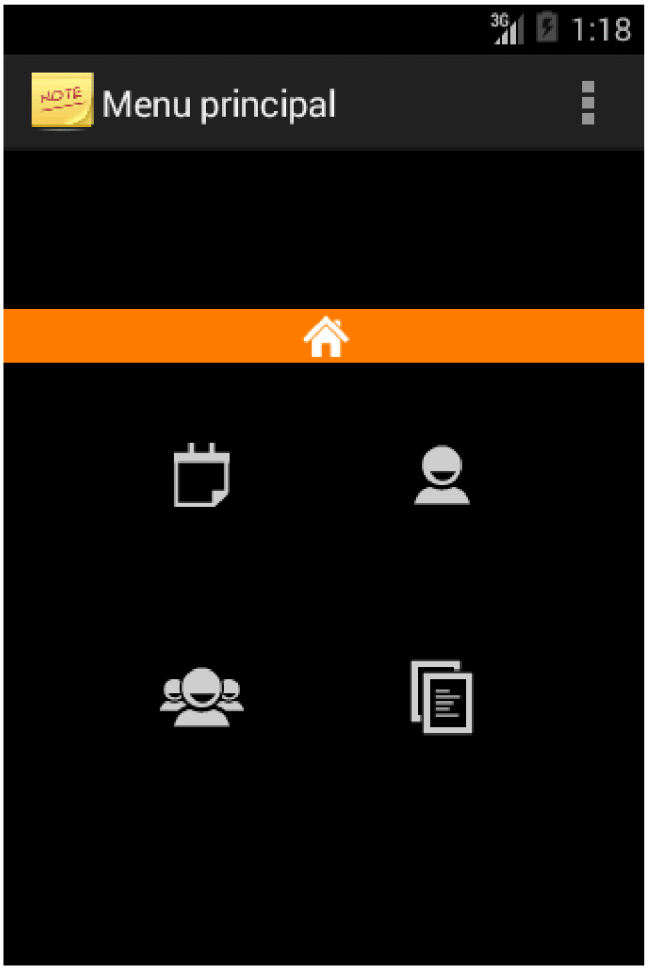


Figure IV.6 : Interface d'accueil.

Cette interface présente l'activité d'accueil après l'authentification.

Elle comporte un menu relatif aux différentes parties de l'application. Alors :

- *L'icone en haut à gauche mène à l'interface de gestion des notes*
- *L'icone en haut à droite mène à l'interface de gestion des contacts.*
- *L'icone en bas à gauche mène à l'interface de gestion des groupes.*
- *L'icône en bas à droite mène à l'interface de gestion des listes.*

3.4. La fenêtre "chercher une note" :

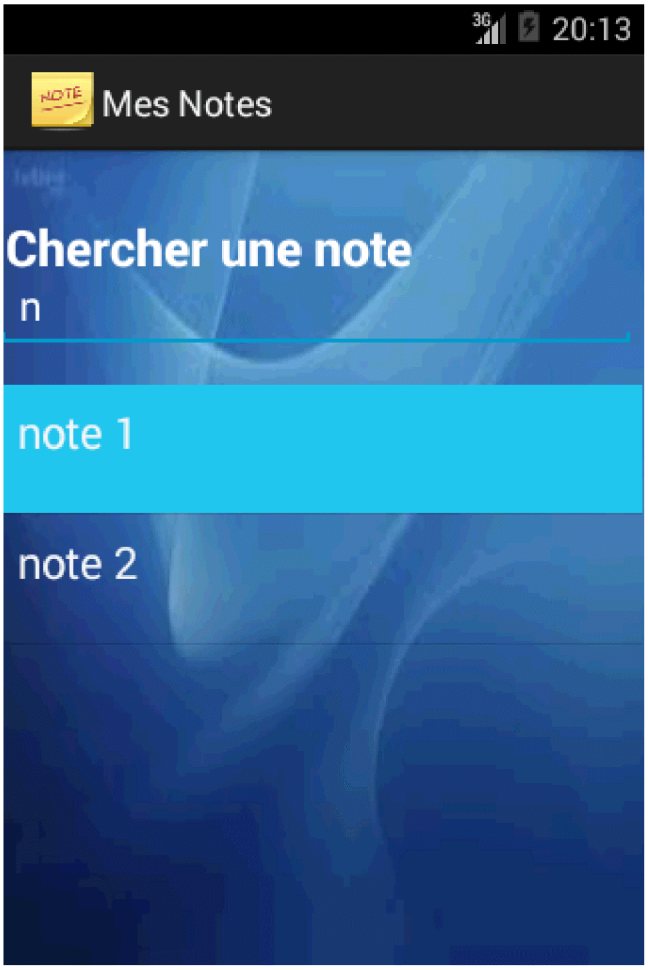


Figure IV.7 : Chercher une note.

Cette interface s'affiche pour trouver une note recherchée. Les notes sont affichées par ordre alphabétique.

3.5. La fenêtre "Détail note" :

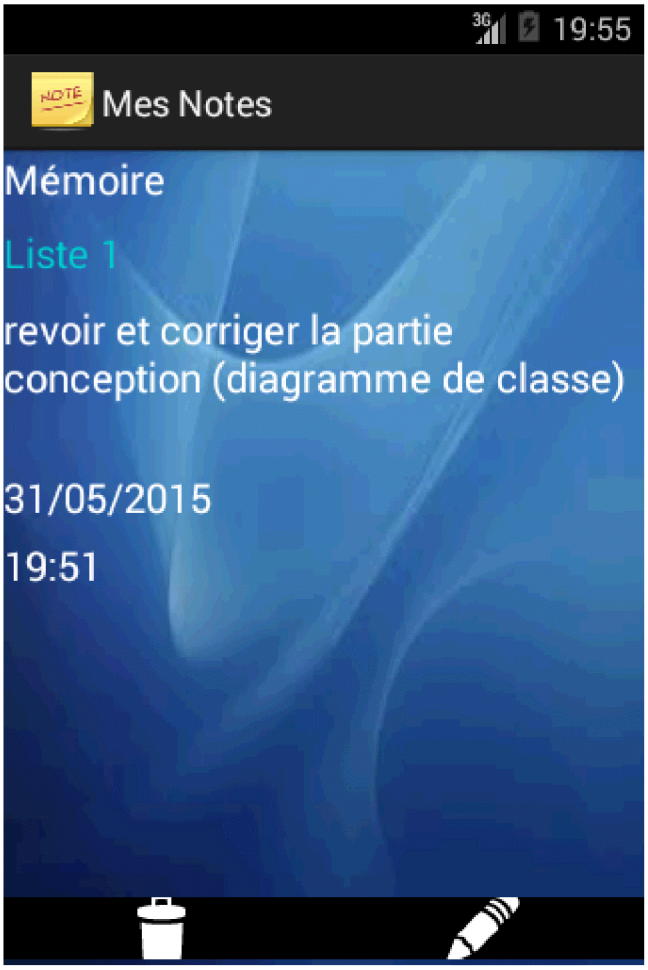


Figure IV.8 : Interface "Détail note"

En cliquant sur une note alors son détail s'affiche comme le montre cette figure .On outre cette interface comporte deux autres boutons afin de pouvoir modifier ou bien supprimer la note affichée.

3.6. La fenêtre de modification d'une note :

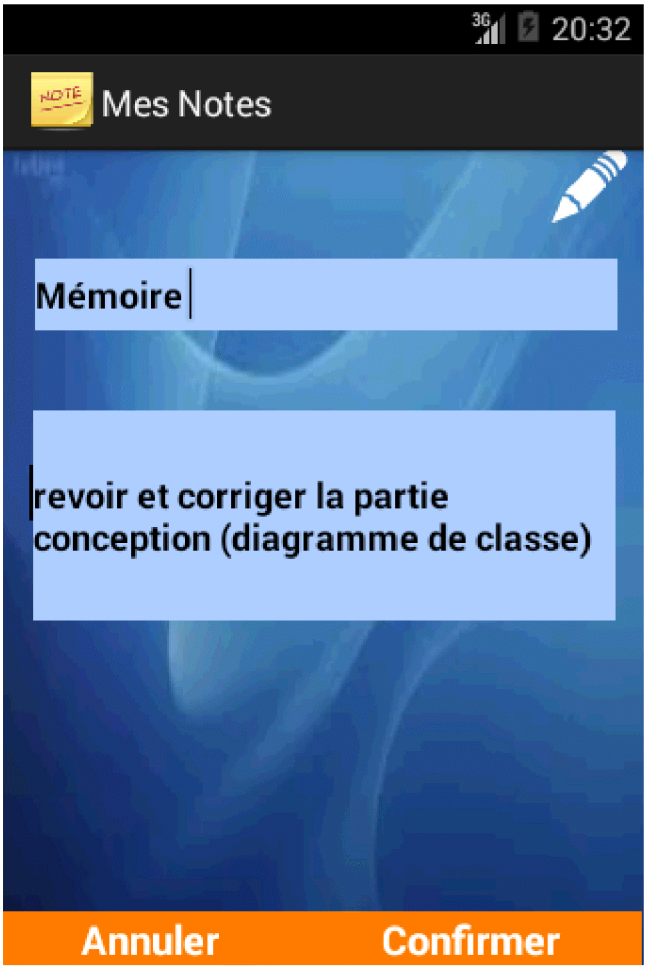


Figure IV.9 : Modifier une note

En désirant modifier une note cette interface s'affiche, l'utilisateur modifie les données souhaitées et confirme les changements à l'aide du bouton confirmer. Par contre le bouton annuler annule la modification en fermant cette interface et retourner à l'interface « Accueil note ».

3.7. L'écran d'une liste de notes :

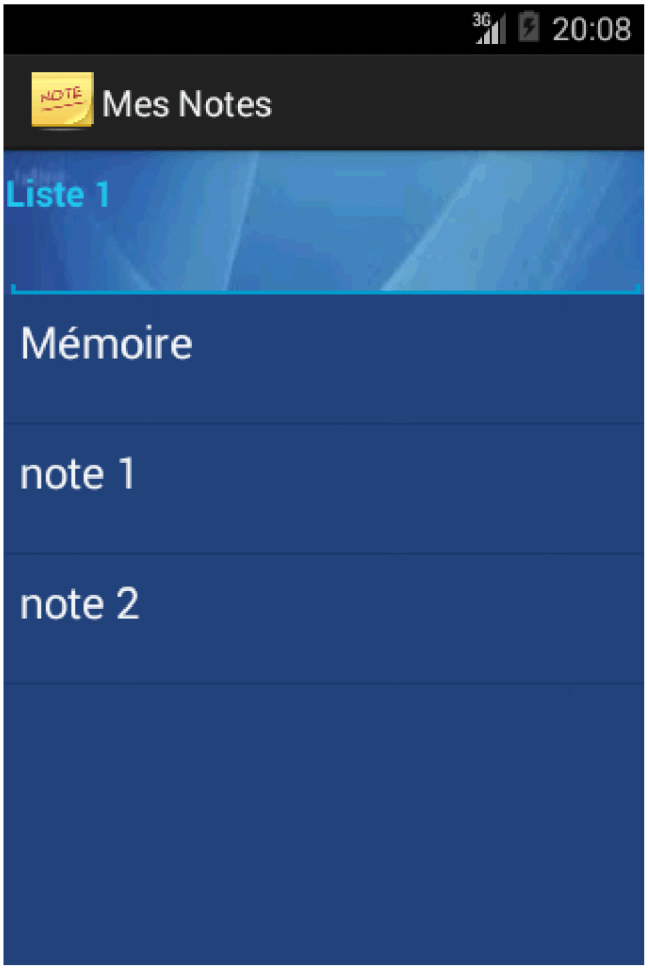


Figure IV.10 : Interface liste.

Cette interface représente une liste qui contient des notes. Le nom de la liste est en haut en bleu.

3.8. La fenêtre de modification de mot de passe :

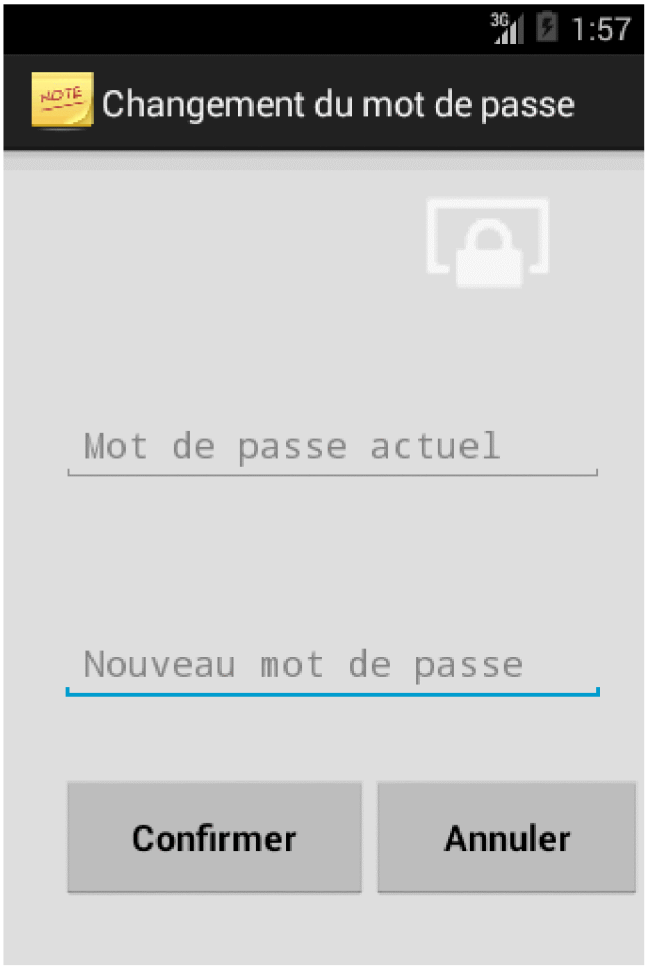


Figure IV.11 : Interface de modification de mot de passe.

En cliquant sur le sous menu "changement de mot de passe", cette interface s'affiche. L'utilisateur peut modifier son mot de passe à tout moment.

Conclusion :

Nous avons présenté dans ce chapitre une vue globale sur le système réalisé. La partie mise en œuvre traduit les besoins fonctionnels et techniques déjà définis par l'implémentation de différentes interfaces.

Conclusion générale

Conclusion générale et perspectives :

L'élaboration de notre travail était dans le but de concevoir une application dédiée aux terminaux mobiles disposant de la plateforme Android. Cette application permet au propriétaire du téléphone la gestion avancée, simple et rapide des tâches souhaitées. Pour ce faire, on a eu recourt à différentes technologies et outils nécessaires pour aboutir à l'objectif de l'application développées.

Notre application ainsi réalisée permet de :

- ✓ Manipuler des notes (Mémos) et les regrouper dans des listes.
- ✓ Modifier le mot de passe.
- ✓ Partager une ou plusieurs notes avec un ou plusieurs utilisateurs.

Ce projet s'est révélé profitable sur plusieurs points, il nous a permis de travailler sur une technologie pour terminaux mobiles et accroître nos connaissances dans ce domaine de développement mobile en abordant plusieurs aspects techniques d'Android. Il nous a donné de plus l'occasion d'acquérir de nouvelles connaissances à propos d'UML, Eclipse, et de maitriser le langage de programmation Java, qui seront certes utiles dans notre future vie professionnelle.

Enfin, on ne peu pas sans doute affirmer que notre travail est complet d'où plusieurs améliorations peuvent être apportées. Mais, nous espérons au moins que nous avons réussis à réaliser une simple application mobile Android fonctionnelle, qui satisfait les besoins des futurs utilisateurs et qui convient à leurs attentes.

Les perspectives :

Comme d'autres applications Android, notre application peut être aisément améliorée, comme par exemple: une note peut être une image ou un son ou une vidéo. En plus on peut penser à améliorer l'architecture web en ajoutant d'autres serveurs pour éviter que l'application ne fonctionne plus si jamais le serveur actuel tombe en panne.

En effet, grâce à son aspect ouvert, Android offre l'opportunité de créer des logiciels mobiles innovants et révolutionnaires en encourageant les développeurs à puiser dans leur imagination et à mobiliser toutes leurs compétences pour un meilleur de cette plateforme.

Annexe

Annexe: Manuel d'installation et de mise en marche

Android ADT Bundle :

Android ADT Bundle nous offre tout un « atelier » pour le développement d'application Android, il est composé de :

- ✓ Eclipse + ADT Eclipse plugin,
- ✓ Android SDK Tools,
- ✓ Android Platform-Tools,
- ✓ Android Platform (la dernière version),
- ✓ Android Virtual Device pour l'émulateur.

➢ Téléchargement :

L'ADT Bundle fournit tout ce dont on a besoin pour commencer à développer des applications, y compris une version de l'IDE Eclipse avec haut-ADT (Android Developer Tools) pour rationaliser le développement d'une application Android. Le téléchargement et la configuration d'environnement de développement Android se fait en une seule étape.

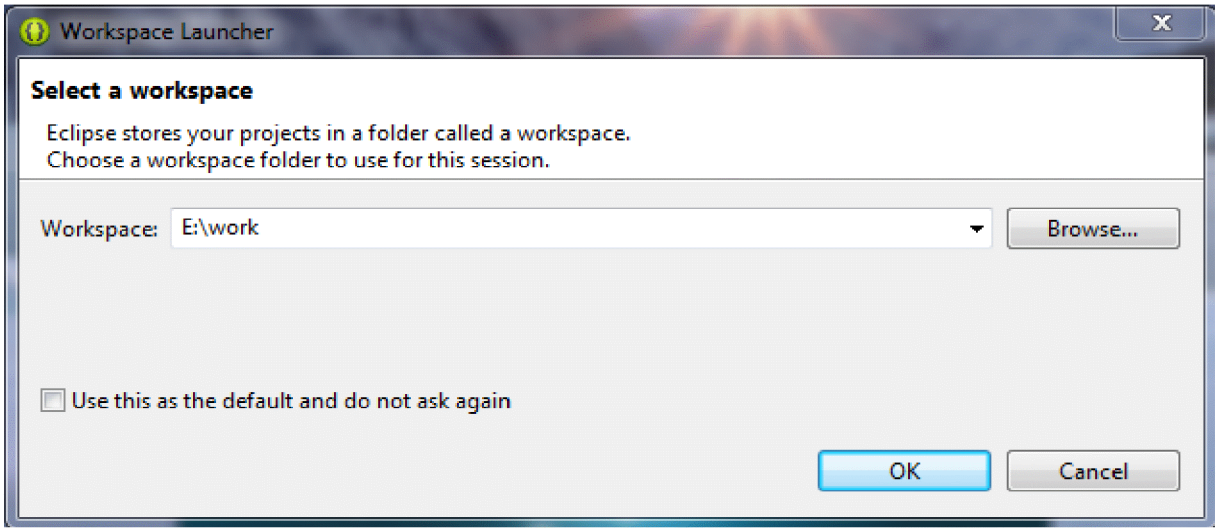
L'Android ADT Bundle est disponible sur le lien suivant :

https://dl.google.com/android/adt/adt-bundle-windows-x86_64-20140702.zip

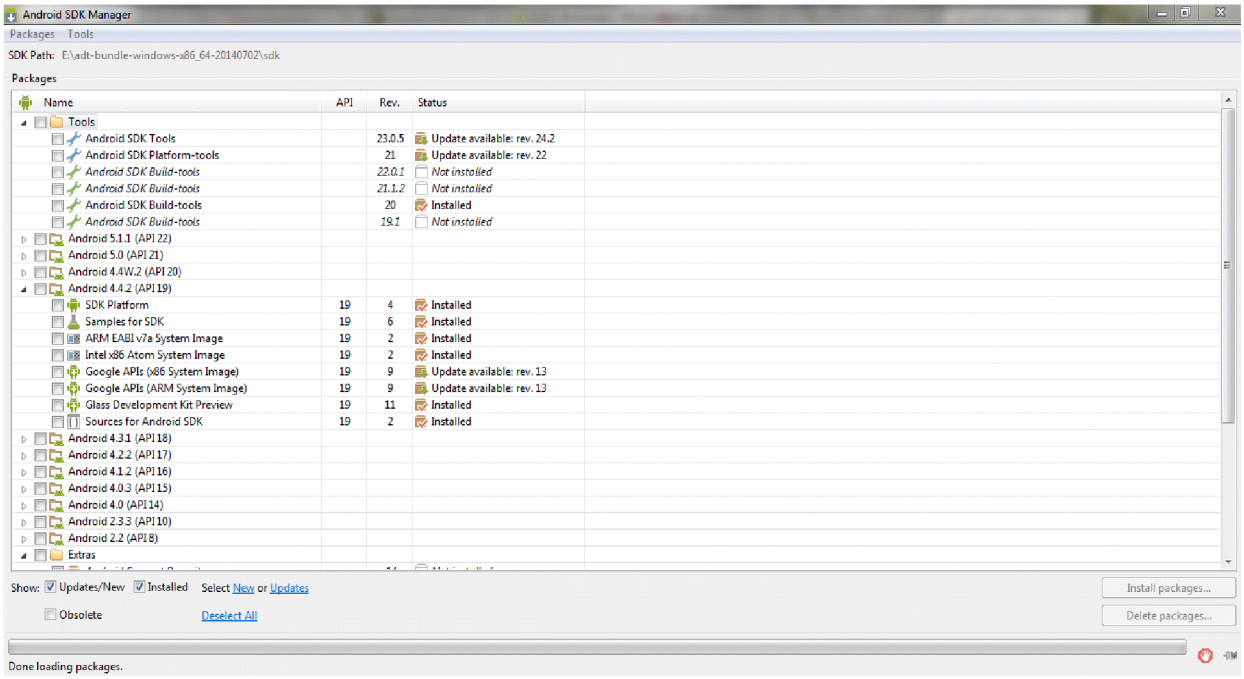
➢ Configuration de l'ADT Bundle :

Apres avoir téléchargé l'ADT, suivrez les étapes suivantes pour le configurer :

- Décompressez le fichier ZIP (appelé : adt-bundle-windows-x86_64-20140702.zip) et enregistrez-le dans un endroit approprié, comme un répertoire "développement" dans votre machine.
- Ouvrez le dossier : adt-bundle- windows-x86_64-20140702.zip / eclipse / et lancez Eclipse, la figure ci-dessous vous apparaîtra pour choisir un répertoire de travail :

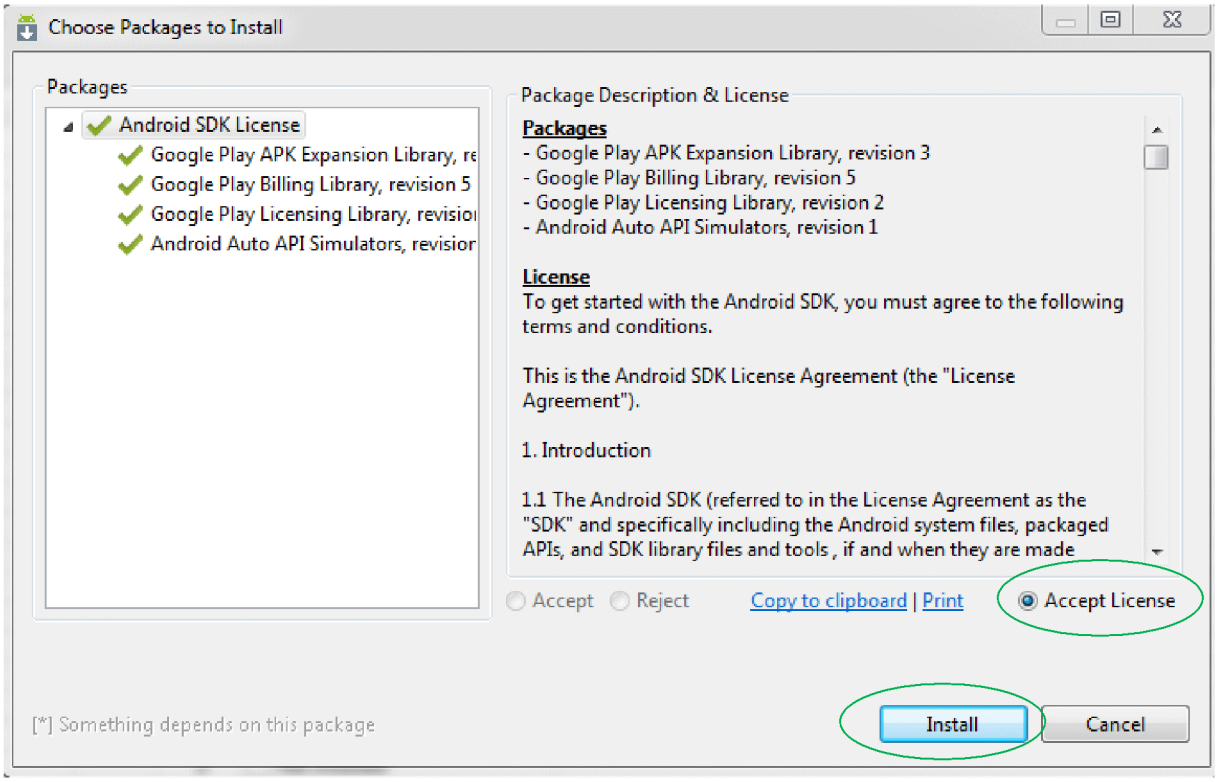


- Indiquer à Eclipse où se trouve le SDK Android : Window->Preferences puis dans la rubrique Android désigner l'emplacement du SDK par le bouton « Browse », terminer par « Apply » puis « Ok ». Maintenant le SDK est prêt pour l'utilisation, vous choisissez cette icône  dans la barre d'outils pour lancer l'Android SDK Manager, la fenêtre ci-dessous sera apparue :



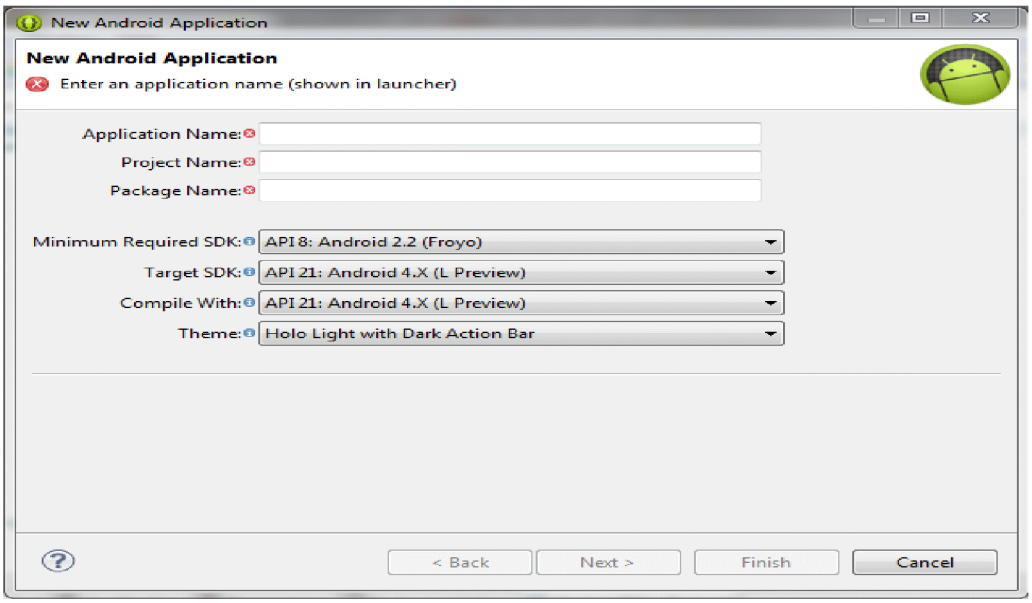
• Vous cochez « Tools » et « Extras » ainsi que la version d'Android que vous souhaitez utiliser, la version « 4.4.2 » dans mon cas. Ensuite vous cliquez sur le bouton « Install <nombre> packages » en bas.

- Vous acceptez la licence « Accept Licence» pour tous les packages sélectionnés puis vous validez par le bouton « Install » tel qu'expliqué à la figure ci-dessous :



➢ Création d'un nouveau projet :

Après le lancement d'Eclipse allez sous : File -> New-> Android Application Project, vous obtenez la figure suivante :



Vous remplissez les champs suivants :

« Application Name » : le vrai nom de notre application.

« **Project Name** »: est le nom de votre projet pour Eclipse.

« **Package Name** » : package où l'application sera sauvegardé.

« **Minimum Required SDK**» : est la version minimale pour laquelle votre application est destinée.

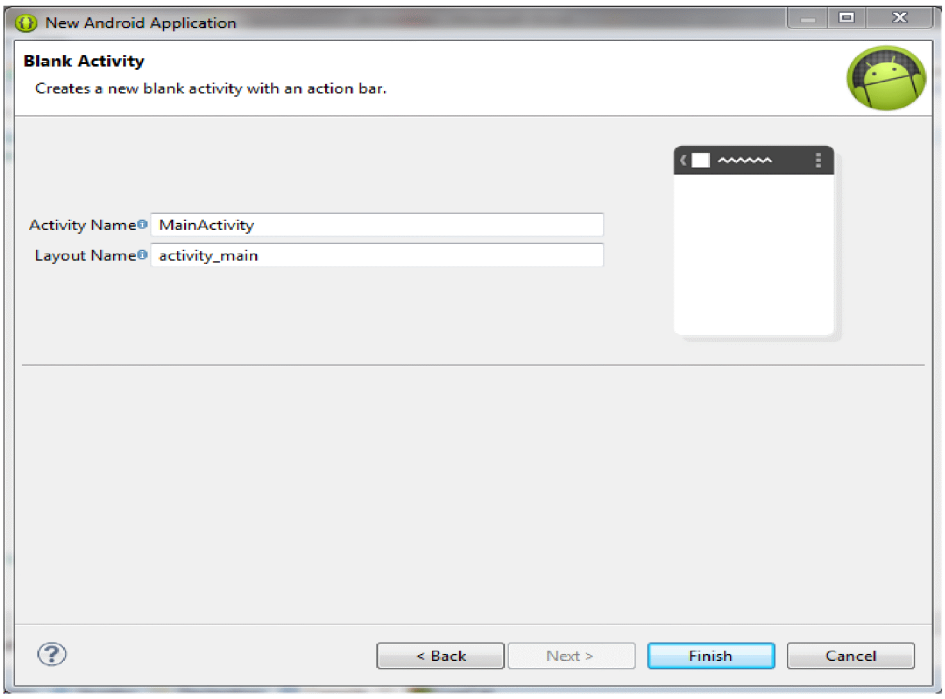
« Target SDK» : est la version sous laquelle votre application fonctionne.

« **Compile With** » : la version du SDK avec laquelle sera compilée l'application..

« **Theme** » : c'est le thème de l'application, puisque aucun des thèmes proposés n'est disponible, le seul choix possible est « None » qui ne signifie aucun thème.

Ensuite vous cliquez successivement trois fois sur « **Next >**» vous aurez trois fenêtres dans lesquelles il est préférable de laisser tous les champs tels qu'ils sont par défaut.

La quatrième fois que vous cliquez sur « **Next >** » vous obtiendrez la fenêtre suivante :



« **Activity Name** » : permet d'indiquer le nom de la classe Java qui contiendra votre activité, ce champ doit donc respecter la syntaxe Java standard.

« **Layout Name** » : renseignera sur le nom du fichier qui contiendra l'interface graphique qui correspondra à cette activité.

➢ Création d'un AVD :

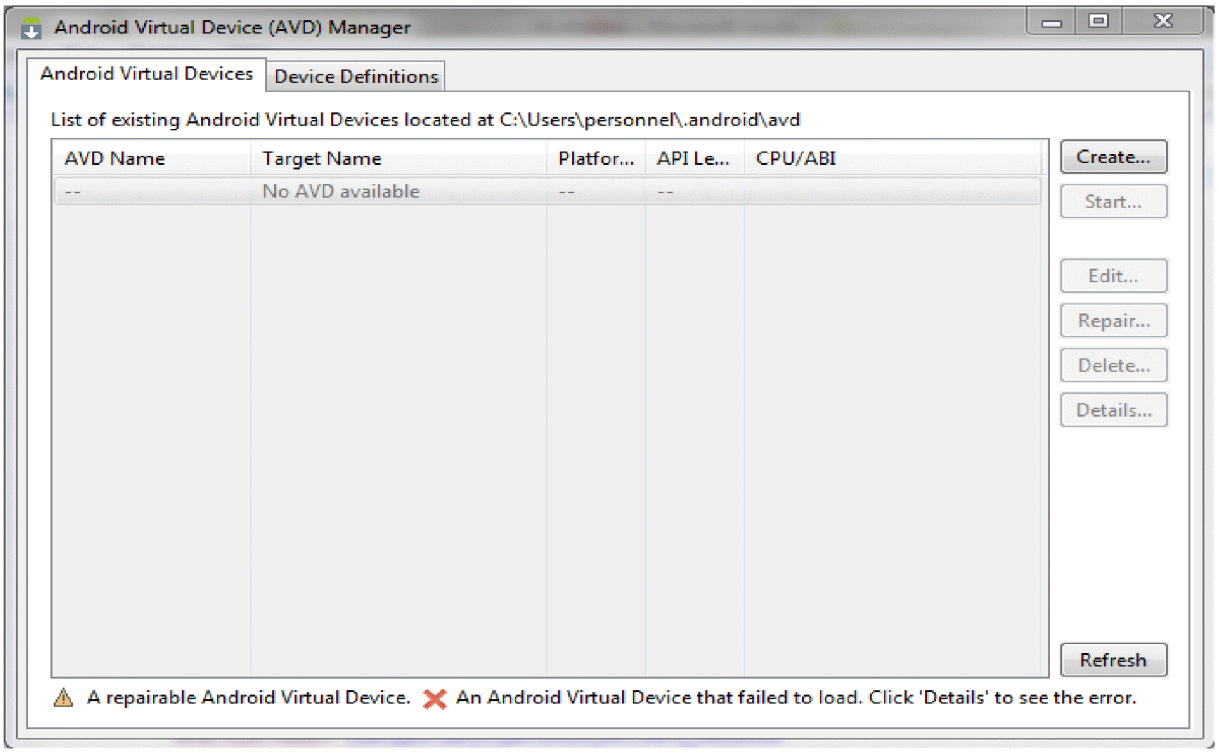
L'exécution d'une application peut se faire sur un véritable terminal Android, ou sur un émulateur Android. Si on veut utiliser l'émulateur, il faudra donc créer un Android Virtual Device (AVD).

Pour créer un AVD on procède de la façon suivante :

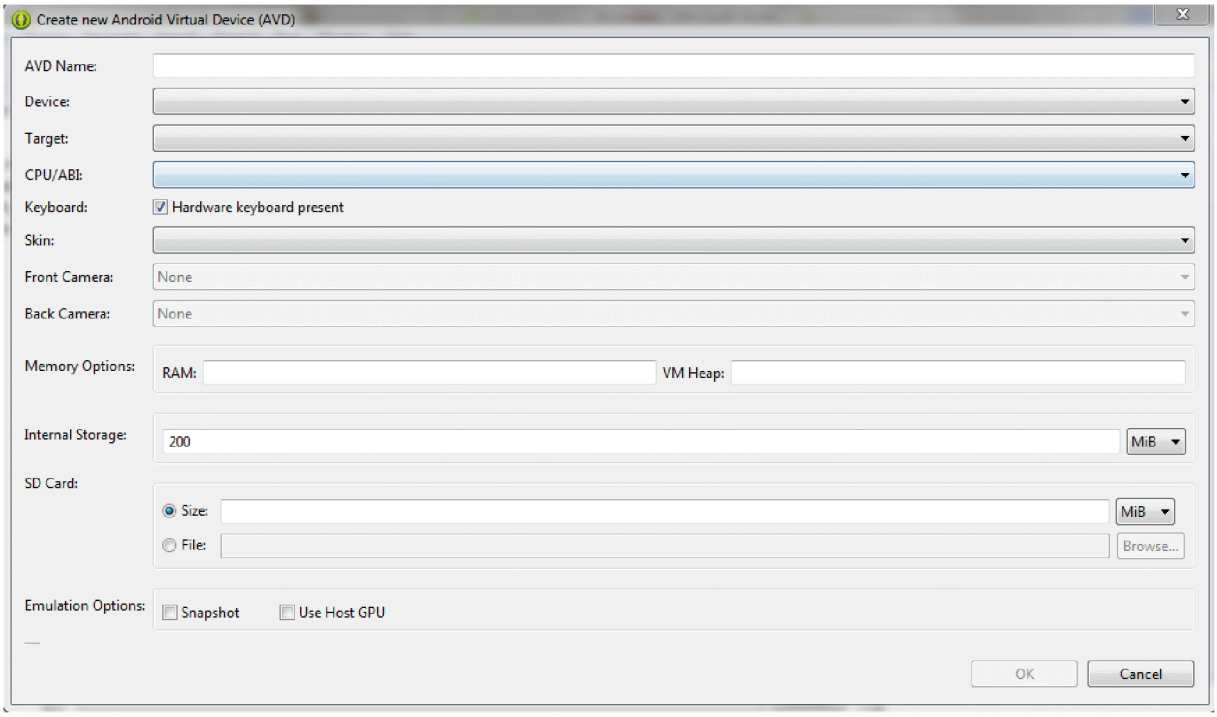
1. Lancer Eclipse ;

2. Aller sous « Window->Android SDK and AVD Manager » et sélectionner « Android Virtual Device Manager », ou bien choisir cette icône  dans la barre d'outils.

3. La fenêtre ci-dessous sera apparue :



4. Cliquer sur le bouton ″Create″ pour faire apparaitre cette fenêtre :

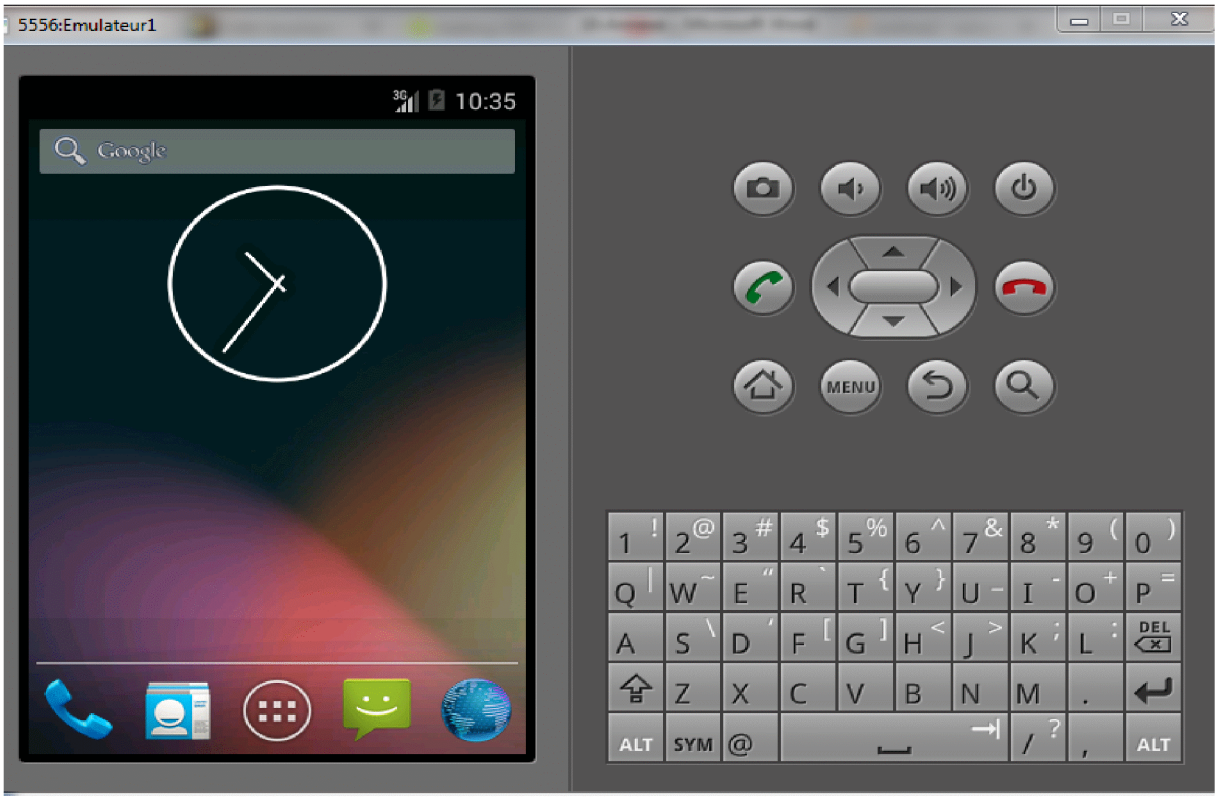


5. Donner un nom pour l'AVD et choisir l'appareil « Device » et la version cible « Target ».

6. Cliquer sur « OK » et l'AVD sera créé.

7. Sélectionner la ligne de l'AVD crée parmi la liste des émulateurs disponibles dans la fenêtre Android Virtual Device Manager et cliquer sur « Start »

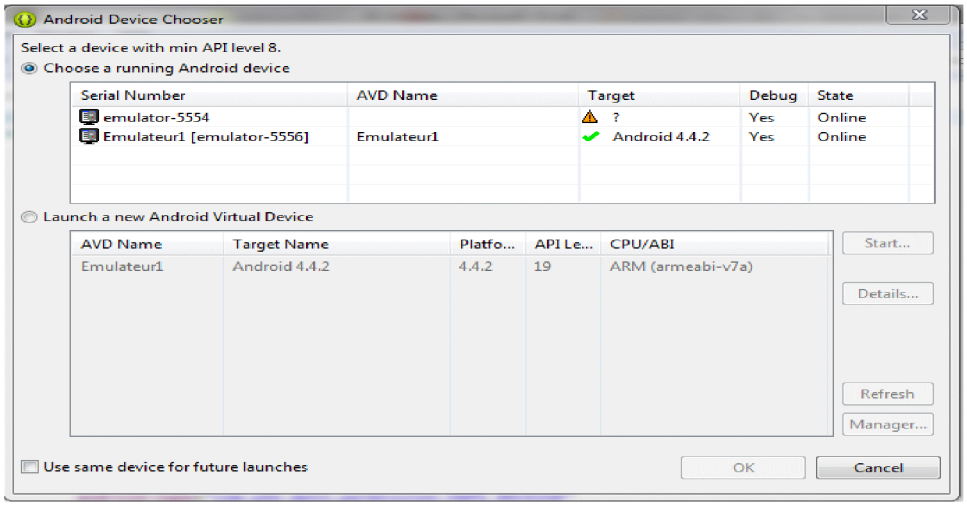
8. L'AVD se lancera comme le montre la figure suivante :



- **Exécution d'un projet Android:**

Le lancement de l'application est obtenu en faisant un clic droit sur le nom de l'application dans la fenêtre ″Package Explorer″ (à gauche), puis en choisissant dans le menu contextuel ″Run as″ puis ″Android Application″. Le lancement se fera sur le terminal physique s'il y en a un connecté ou sur le terminal virtuel sinon.

Si plusieurs terminaux sont disponibles une fenêtre vous permettra d'indiquer celui que vous voulez utilisez (voir ci-dessous) :



Le terminal virtuel apparait à l'écran et s'initialise (c'est plutôt long) puis votre application y démarre. L'onglet ″Console″ en bas de la fenêtre d'Eclipse permet de suivre l'évolution de ce lancement de l'application.

Une fois lancé il est inutile de fermer ce terminal. Si vous modifiez votre application vous pouvez en exécuter la nouvelle version sur ce même terminal en refaisant ″Run as″ puis ″Android Application″.

Références Bibliographiques :

[1] : Nouha Khiari « Rapport de stage sur le projet ″Locate my car″-google map android », Ecole nationale des sciences de l'informatique Tunisie, 2010.

[2] : http://www.smsenvoi.com/blog/levolution-marche-sms-marketing-grace-smartphones/

[3] : Telli Fatiha et Benjeddou Khadija « Application mobile : une télécommande Bluetooth d'une souris optique », mémoire de fin d'étude, soutenu le 29/9/2013.

[4] : http://www.cours-informatique-gratuit.fr/facile/multimedia/6.les-smartphones

[5] : http://www.ekito.fr/people/application-mobile-web-ou-natif/

[6] : Livre blanc : « Développement d'applications mobiles avec LongRang ».

[7] : http://www.pouledesign.com/design/differencier-native-application-mobiles-web applications

[8] : http://www.silicon.fr/forte-progression-des-applications-mobiles-de-productivite-en-2014-105227.html

[9] : http://www.clubic.com/os-mobile/android/actualite-615662-google-samsung-accordent-harmoniser-android-apps-pre-installees.html

[10] : http://www.phonandroid.com/cupcake-lollipop-pourquoi-google-choisi-noms-desserts-android.html

[11] : http://www-igm.univ-mlv.fr/~dr/XPOSE2008/android/archi_comp.html

[12] : http://www-igm.univ-mlv.fr/~dr/XPOSE2008/android/index.html#cara

[13] : PETISME Daniel et VINCENT Anthony : « Développement d'une application de localisation pour la plateforme Android », Institut Supérieur d'Informatique de Modélisation et de leurs Applications.

[14] : http://openclassrooms.com/courses/creez-des-applications-pour-android/votre-premiere-application-1

[15] : http://www.android-mt.com/news/meilleurs-outils-prise-notes-android-24095

[16] : Alain Le Guennec, « génie logiciel et méthode formelle avec UML spécification et généralisation de testes » juin 2001.

[17] : Grady Booch, James Rambough et Ivar Jacobso « Le guide de l'utilisation UML », Edition Eyrolles, 2001.

[18] : Jim Conallen, « Concevoir des application web avec l'UML » Edition Eyrolles, Octobre 2000.

[19] : ″Analyse et Conception Orientées Objet avec UML et réalisation en C++″ Guide étudiant, Edition : Sun 1999, pdf.

[20] : Grady Booch et al, ″Le Guide de l'utilisateur UML″, Edition : Eyrolles, 2003.

[20] : http://www.gamergen.com/tutoriels/tutoriel-installez-configurez-votre-sdk-116759-1

[21] : http://www.wampserver.com

[22] : http://php.net/manual/fr/intro-whatis.php

[23] : https://fahmirahman.wordpress.com/category/android/