République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes
De MASTER PROFESSIONEL

Domaine : **Mathématiques et Informatique**
Filière : **Informatique**

Spécialité : **Ingénierie des Systems d'informations**

Présenté par
Abderrahmani El husseine

Thème

Conception et réalisation d'une application mobile sous Android pour : La Discussion instantanée.

Soutenu publiquement le 26/09/ 2016 devant le jury composé de :

Président : mlle Yasmine YESLI
Encadreur : Mme Farida Dahmani Bouarrab
Examinateur : Mr idir Filali
Examinateur : Mr Ahmed Dib

2015/2016

Remerciement

je tiens tout d'abord à remercier le bon dieu de m’avoir donner santé, courage , volonté et foi pour réaliser ce travail.

Je tiens à exprimer ma profonde gratitude à ma promotrice M^{me}. Farida Dahmani Bouarab pour tout ce qu’elle m’apporté comme aide, connaissances et conseils pour l’accomplissement de ce travail.

Je remercie aussi l’ensemble de la communauté stackoverflow et les forum d’Openfire pour les conseils et leur aide.

Je remercie vivement les membres du jury pour avoir accepté d’évaluer mon travail.

Aux enseignants de département d’informatique De L’UMMTO pour l’effort qu’ils ont déployé afin d’assurer notre Formation, pour leurs compétences, et surtout leur modestie.

J’aimerai aussi remercier ma famille

Mes amis, que tous ceux qui ont contribués de près ou de loin à la réalisation de ce travail.

Dédicace

Je dédie ce travail à Mes chers parents

Ma mère Djahida et à mon père Khaled
Je ne trouverai jamais de mots pour vous exprimer mon profond attachement et ma reconnaissance

Pour l'amour, la tendresse et surtout pour votre présence dans mes moments les plus difficiles

A mes chères sœurs, Et leurs familles

A mes petits neveux, a ma petite niéce

A Sergey Brin et Larry page

A tout mes amis et ceux qui m'ont aidé à la

Réalisation de ce mémoire.

EL Husseine .

SOMMAIRE

Chapitre II : Analyse et conception

CHAPITRE III : Présentations des outils.

CHAPITRE IV : Implémentation.

Introduction Générale

Depuis la création des réseaux télécommunications et leur développement accompagnée par l'apparition des téléphones mobiles et des micros ordinateurs, notre manière de vivre à radicalement changé d'où la création de nouveaux besoins comme les Smartphones appelé aussi téléphone intelligent basé sur un système d'exploitation ouvert tel que (iphone OS, Android, Blackberry OS, Windows phone ou encore symbian).

Il ne fait désormais plus aucun doute que les technologies de l'information et de la communication représentent la révolution la plus importante et la plus innovante qui a marqué la vie de l'humanité en ce siècle passé. En effet, loin d'être un éphémère phénomène de mode, ou une tendance passagère, ces technologies viennent nous apporter de multiples conforts à notre mode de vie, car ils ont révolutionné le travail des individus par leur capacité de traitement d'information, d'une part, et de rapprochement des dimensions espace/temps, d'une autre.

Parmi ces TIC (Technologies de l'information et de la communication), la messagerie instantanée s'est rapidement développée dans les organisations aux cours de ces dix dernières années, par sa facilité d'utilisation, rapidité et son utilité perçue. Désormais, elle représente l'outil de communication le plus utilisé partout au monde . les messages et appels a moindre cout (Gratuits) sont devenu une partie intégrante de notre mode de vie , Skype de Microsoft ou bien viber, what'app ou facebook Messenger , sont tous des applications cliente qui offre ces services gratuitement a travers le monde entier sans restrictions ni frontières .

Dans ce mémoire , je vais réaliser ma propre application de communication a divergence sociale, le client peut alors communiquer avec un contact et passer des appels vidéo et audio en illimité , envoyer des fichiers. Mon choix c'est prononcé sur le system Android ,qui est selon moi un system mobile d'actualité et muni des dernières exigences technologiques .

Par la suite, j'installerais et configurerais le serveur openfire, MySQL, et mon propre serveur de fichiers

ainsi qu'un serveur apprtc pour les deux types d'appels (vidéo/audio) et interconnecté l'ensemble a un réseau local ou WAN (cas d'hébergement du serveur sur internet) .

L'objectif principal de mon application est de mettre en contact des personnes via messages instantané, appel audio et appel vidéo, transfert de fichiers multimédia.

Mon mémoire sera donc scindé en quatre chapitres comme suit :

- Chapitre I : Android et la téléphonie
- Chapitre II : Analyse conception.
- Chapitre III : Présentation des outils utilisé.
- Chapitre IV : Implémentation de l'application

Chapitre I : Android et la téléphonie

1. Introduction :

Les systèmes informatiques sont de plus en plus présents dans notre quotidien, sous forme de téléphones portables, cartes à microprocesseur, ou autres PDA (*assistant numérique personnel*). Mais ils sont tout aussi présents dans l'industrie qui les utilise comme système multimédia et/ou télécommande d'équipements à distance.
Ces systèmes informatiques sont particuliers, ils ne correspondent pas à la vision que l'on peut avoir du conventionnel ordinateur personnel.
L'évolution de ce marché a induit au développement d'un autre qui est celui des systèmes d'exploitation, qui permettent de les gérer. Tous ces développements ont donné naissance à de nouveaux logiciels et matériels qui sont perfectionnés de jour en jour. Tel que la messagerie instantanée.

2. Les systèmes embarqués :

2.1. Définition :

Un système embarqué est un système électronique informatisé autonome ne possédant pas d'entrées/sorties standards, et qui constitue une partie intégrante d'un système plus large. Il possède des ressources d'ordre spatial et énergétique limitées (taille et consommation restreintes). Le terme de « Système Embarqué » désigne aussi bien le matériel que le logiciel utilisé.
En anglais embedded system – « embedded » signifie « enfoui », ce qui traduit le coté non visible autant que tel de l'équipement.
Ils se sont développés pour répondre à des enjeux de productivité, de fiabilité et de sécurité des systèmes.

2.2. Historique :

La plupart des machines qui nous simplifient la vie ont besoin d'un système de régulation ou de contrôle pour fonctionner de manière correcte.
Ces systèmes de contrôle existaient déjà bien avant l'invention des ordinateurs.
Exemple : Pour maintenir une locomotive à vapeur à une vitesse constante, on a besoin d'un système qui régule la quantité de vapeur envoyée dans les pistons. Si la locomotive ralentit (pente montante...) il faut injecter plus de vapeur, si la locomotive accélère, il faut injecter moins de vapeur On aimerait que cette tâche se fasse de façon «automatique», c'est-à dire avec un minimum d'intervention de l'être humain.
Solution : le gouverneur à force centrifuge.

Ces systèmes de contrôle peuvent donc être réalisés de manière forte simple. Avec le développement de la technologie, on a opté pour des systèmes basés sur l'électronique. Parallèlement à ce développement des systèmes de contrôle, les systèmes informatiques se sont développés. Ceux-ci sont vite sortis du cadre des «machines de bureau» ou de «machine à calculer» dans lequel ils avaient initialement été développés.
Il est donc naturel d'utiliser les possibilités de calcul de l'ordinateur comme composant d'un système plus large. Pour remplacer un système de régulation analogique, et pour réaliser un traitement qui serait trop complexe / impossible en analogique...
Un des premiers exemples de système embarqué date du début des années 1960. Il s'agit de

l'ordinateur de bord des vaisseaux spatiaux du programme Apollo, qui a amené N. Armstrong sur la lune. Cet ordinateur contrôlait en temps réel les paramètres de vol et adaptait la trajectoire. Il fonctionnait en mode interactif.

Le premier système embarqué qui a été produit en série est vraisemblablement le D-17 d'Autonetics. Il servait de système de contrôle aux missiles nucléaires américains LGM-30 Minuteman, produit à partir de 1962.

Depuis les systèmes se sont diversifiés, ils ont permis l'explosion du marché des «*consumer electronics*» où tout est devenu numérique (GSM, électroménager, MP3s, etc.). Ils sont également bien présents dans le domaine industriel pour Contrôle de processus de production, etc.
La convergence entre les applications électroniques pour grand public et les ordinateurs est de plus en plus grande : La console de jeu XBox de Microsoft n'est qu'un PC «emballé» sous la forme d'une console. Il est de plus en facile de transformer son PC de bureau en «*media center*» qui remplace la chaîne hi-fi et le lecteur DVD..

2.3. Architecture d'un système embarqué :

L'architecture d'un système embarqué se définie par le schéma suivant:

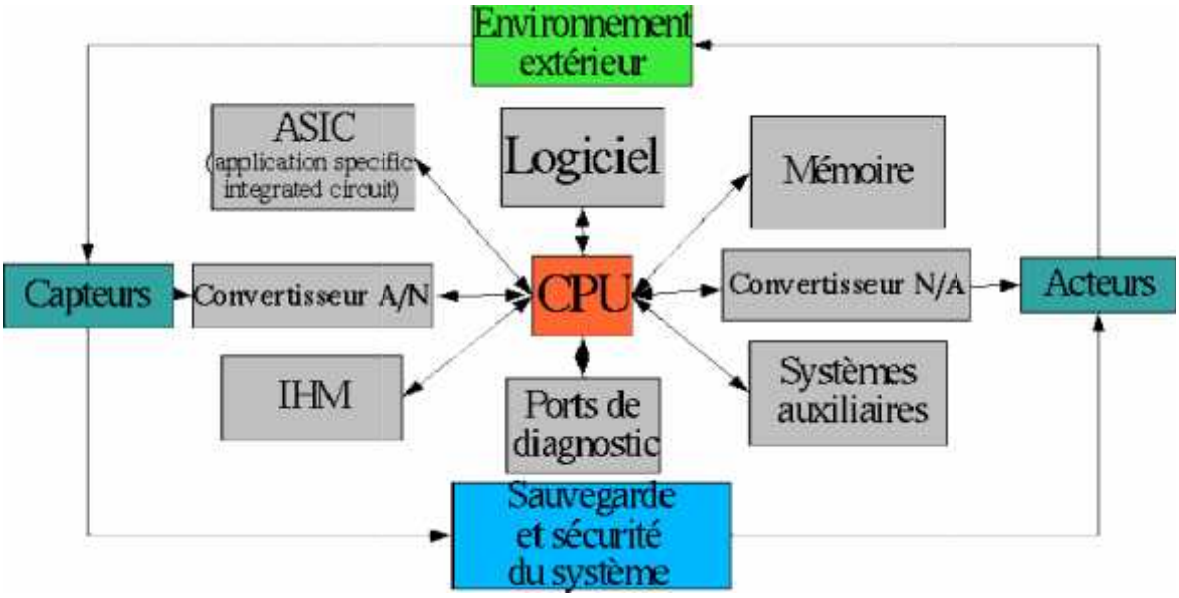


Figure I.1 : Architecture d'un système embarqué.

Cette architecture peut varier selon les systèmes: on peut par exemple, ne pas trouver de systèmes auxiliaires dans de nombreux systèmes embarqués autonomes et indépendants. En revanche, l'architecture de base est la plupart du temps composée d'une unité centrale de traitement (CPU), d'un système d'exploitation qui réside parfois uniquement en un logiciel spécifique (ex: routeur), ou une boucle d'exécution (ex: ABS). De même l'interface IHM n'est pas souvent existante, mais est souvent utile pour reconfigurer le système ou vérifier son comportement.
Le fonctionnement du système ainsi défini se résume comme suit:

- Il reçoit des informations de l'environnement extérieur qu'il converti en signal numérique
- L'unité de traitement composée du CPU, de la mémoire, du logiciel, de l'ASIC et éventuellement de système externe traite l'information
- Le traitement génère éventuellement une sortie qui est envoyée vers la sortie, les systèmes auxiliaire, les ports de monitoring ou l'IHM

2.4. Les caractéristiques des systèmes embarqués :

Les caractéristiques principales d'un système électronique embarqué sont :

- **Autonome.** Une fois enfouis dans l'application il n'est (le plus souvent) plus accessible.
-**Temps réel.** Les temps de réponses de ce système sont aussi importants que l'exactitude des résultats.
-**Réactif.** Il doit réagir à l'arrivée d'informations extérieures non prévues.
-**Interface spécifique.** Ce n'est pas des PC mais possède des architectures similaires. Il n'a pas réellement de clavier standard et l'affichage est limité.
-**Ciblé.** Il exécute des applications logiciels dédiées précises.

2.5. Les types de systèmes embarqués :

Il existe plusieurs types de systèmes embarqués dont :

- **Système transformationnel :**

Activité de calcul, qui lit ses données et ses entrées lors de son démarrage, qui fournit ses sorties, puis meurt.

- **Système interactif :**

Système en interaction quasi permanente avec son environnement, y compris après l'initialisation du système; la réaction du système est déterminée par les événements reçus et par l'état courant (fonction des événements et des réactions passés); le rythme de l'interaction est déterminé par le système et non par l'environnement.

- **Système réactif ou temps réel :**

Système en interaction permanente avec son environnement, y compris après l'initialisation du système; la réaction du système est déterminée par les événements reçus et par l'état courant (fonction des événements et des réactions passées); mais le rythme de l'interaction est déterminé par l'environnement et non par le système.

3. Le système d'exploitation Android :

3.1. Définition :

Android est un système d'exploitation open source, basé sur un noyau linux, adapté aux systèmes embarqués (Smartphone principalement).
Il est développé par un consortium : l'Open Handset Alliance.
Ce dernier regroupe beaucoup de sociétés liées aux nouvelles technologies (Intel, HTC, Motorola, Garmin…etc.), mais le principal contributeur est Google.
Android est à L'heure actuelle le système d'exploitation pour Smartphones et tablettes le plus utilisé.

Dans ce chapitre je vais présenter la plateforme Android avec ces différents composants (équipements utilisant, sont Architecture ainsi que ces fonctionnalités).

3.2. Historique des versions d'Android :

Figure I.2 : Evolution des versions d'Android juin 2016.

Android a débuté avec la sortie de la version 1.0 en septembre 2008, il a connu plusieurs mises à jour depuis sa première version. Ces mises à jour servent généralement à corriger des bugs et à ajouter de nouvelles fonctionnalités. Dans l'ensemble, chaque version est développée sous un nom de code basé sur des desserts.Ces nom de codes suivent une logique alphabétique, en voici quelques unes :

- **Android 1.0 Apple Pie** : (Tarte aux pommes) ou Alpha, version connue uniquement ou presque des développeurs car c'est la version du Sdk[1] distribuée fin 2007 avant la sortie du premier téléphone Android.

[1] Software Developpent Kit, Kit de développement, Permettant la création des applications de type défini Voire plus d'explication dans le chapitre3 page

- **Android1.5Cupcake** : (Petit Gâteau), sortie en avril 2009, dernière révision officielle en mai 2010, corrigea le manque d'API et rendit le système plus utilisable.

- **Android1.6, 2.0 et 2.1 :** ont apporté d'importé d'importantes améliorations respectivement sur les fonctionnalités et sur l'interface graphique du système.

- **Android 2.2 Froyo :** a fortement mis l'accent sur la synergie avec Internet. L'envoi d'applications et de liens instantanés depuis un ordinateur est désormais possible.

- **Android 2.3 (2.3.7) Gingerbread** : (Pain d'épice), sortie le 6 décembre 2010, dernière version dédiée uniquement aux Smartphones. Cette version est parfois utilisée sur de petites tablettes. C'est depuis cette version que le protocole SIP (Session Initialisation Protocole) fut implémenté .

- **Android 3.0 Honeycomb :** est spécialement étudié pour les tablettes tactiles.il rajouter quelques nouveautés comme la prise en charge de la vidéoconférence via Gtalk, la nouvelle interface Gmail ou encore le lecteur de livre électronique Google.

- **Android 4.0 :** (mi 2011) il rajouter encore plus de fonctionnalités aux terminaux. Pour le développement, ces nouvelles versions d'Android, il propose de nouveaux composants permettant de réaliser des applications avec une ergonomie plus adaptée aux tablettes tactiles.

- **Android 3.0 et Android 4.0 :** il apporte plus d'outils aux constructeurs leur permettant de proposer des tablettes tactiles, qui seront capables de rivaliser (surtout au niveau de l'ergonomie) avec Ipad.

- **Android 4.2Jelly Bean** : introduit photo sphère, permettant une prise des photos à 360° type Street View, un système multi-compte sur tablette uniquement.

- **Android 4.4 Kitkat :** Le 3 septembre 2013, Google annonce que la version 4.4, portera le nom de KitKat, alors que plusieurs rumeurs annonçaient le nom de Key Lime Pie (tarte au citron vert). Cette version est sortie le 4 novembre 2013, en même temps que le Nexus 5. La version 4.4.2 est sortie le 9 décembre 2013, la version 4.4.3 le 2 juin 2014, et la version 4.4.4 le 20 juin 2014. Nouveautés : Amélioration du désigne et restrictions liée a la mémoire extensible SD .

- **Android 5 LolliPop :** Annoncée le 15 octobre 2014 et sortie le 3 novembre 2014, *Android 5.0 Lollipop* est une évolution majeure d'Android qui propose de nombreuses modifications et nouveautés, et qui étend sa disponibilité sur de nouveaux supports tels que la télévision, la voiture ou les montres connectées. Cette version se caractérise par la refonte totale de l'interface graphique, implémentation du moteur ART de compilation au lieu de la compilation JIT de dalvik et du projet volta pour la conservation de l'énergie.

- **Android 6 Marshmallow :** C'est fin mai 2015, lors de la Google I/O, que la firme de Mountain View a dévoilé les détails sur Android 6.0. La version grand public a été quant à elle déployée au courant du troisième trimestre 2015 , nouveautés : intégration finale du mode multi fenêtres et l'intégration du « Adoptable storage » pour la mémoire SD .

3 .3 . Différents équipements utilisant Android :

Le succès d'Android est indéniable, et ça les fournisseurs l'ont bien compris, d'où l'apparition sur le marché de plus en plus d'appareils utilisant ce système d'exploitation, car depuis l'introduction du téléphone HTC Dream, on a assisté à une explosion d'appareils adoptant Android comme SE accompagnée d'une croissance exponentielle des parts de marché de ce dernier.

a. Les Smartphones :

Le premier mobile commercialisé sous Android fut le HTC Dream/G1 produit par la firme Taïwanaise HTC, lancé aux Etats-Unis le 22 octobre 2008.

En France, le premier mobile sous Android fut commercialisé par Orange avec le HTC Dream depuis le 12 mars 2009. Le HTC Magic est arrivé debut mai 2009 chez SFR, suivi par le Samsung Galaxy chez Bouygues. Le HTC Hero, bénéficie de Sence, un habillage particulier de l'interface. Son successeur : le HTC Legend est commercialisé sur le marché européen en avril 2010 avec la version 2.1 d'Android et, comme le Hero, il bénéficie de Sence.

En novembre 2009, Motorola a lancé aux Etats-Unis le Droid, le premier téléphone muni de la version 2.0 d'Android. L'appareil qui a trouvé 250 000 acheteurs une semaine après son lancement au Canada début février 2010.

Le 5 janvier 2010 Google annonce le Nexus One, téléphone conçu par la firme de Montain View et sous-traité par HTC. Doté de caractéristiques alors

assez impressionnantes (écran AMOLED de 3,7 ‘’, processeur de 1Ghz, 512 Mo de RAM et Android 2.1), il avait également l'avantage de recevoir ses mises à jour de Google. Il fut en effet le premier Smartphone à bénéficier d'Android 2.2 Froyo en juin 2010. Cependant ses ventes n'ont pas été exceptionnelles.

En décembre 2010, Samsung fabrique le Nexus S pensé par Google sous Android 2.3, au cours du même mois celui-ci fut envoyé dans l'espace à plus de 32 000 mètres d'altitude.

Le 18 octobre 2011, Samsung et Google dévoilent le Samsung Galaxy Nexus, premier Smartphone sous Android 4 « Ice Cream Sandwich ». Celui-ci intègre le déverrouillage par reconnaissance faciale, l'utilisation de boutons virtuels et un système de reconnaissance vocale avancé.

Figure I .3: Samsung Galaxy S III sous Android 4.1.

b. Les Tablettes :

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

Motorola a présenté au CES 2011 la Xoom, première tablette bénéficiant de la nouvelle version du système mobile de Google, Honeycomb (Android 3.0). Elle y a reçu le prix de la meilleure innovation. S'ensuivent de nombreuses tablettes sous Android Honeycomb proposées par un très large choix de constructeurs, dont Samsung, Acer, ASUS, Toshiba et Sony.

Android a été porté sur d'autres appareils comme la HP TouchPad, le portage a été réalisé débute 2012 avec la version 9 de CyanogenMod, basé sur Ice Cream Sandwich, et fin 2011 avec la version 7 de CycanogenMod basé sur Gingerbread.

Figure I .4 : Tablette HP TouchPas sous Android 4.0 Ice Cream Sandwich.

c. Les télévisions : (smartTV)

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

Figure I .5 TV Bravia.

d. Audio radios:

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

Figure I.6: VW radio car Android.

e. Consoles de jeux vidéo:

DEA commercialise depuis septembre 2012, sous le nom de MyPlay, une console de jeux vidéo portable sous Android. Archos par contre commercialise une console nommée GamePad.

f. Les montres :

Android Wear vous montre des suggestions et des informations utiles avant même de les avoir demandé: messages d'amis, détails de la réunion, des infos de vos applications préférées, et plus encore. Votre affichage de la montre d'Android Wear est toujours en service, de sorte que le temps et les applications sont toujours visibles.
Android Wear peut être connectée à votre téléphone via Bluetooth et Wi- Fi. Cela vous permet de recevoir des notifications, écouter de la musique et de suivre votre course de footing sur votre montre. Même lorsque votre téléphone n'est pas avec vous.

Figure I .7 Smartwatch par Sony sous Android.

3.4.Architecture Android :

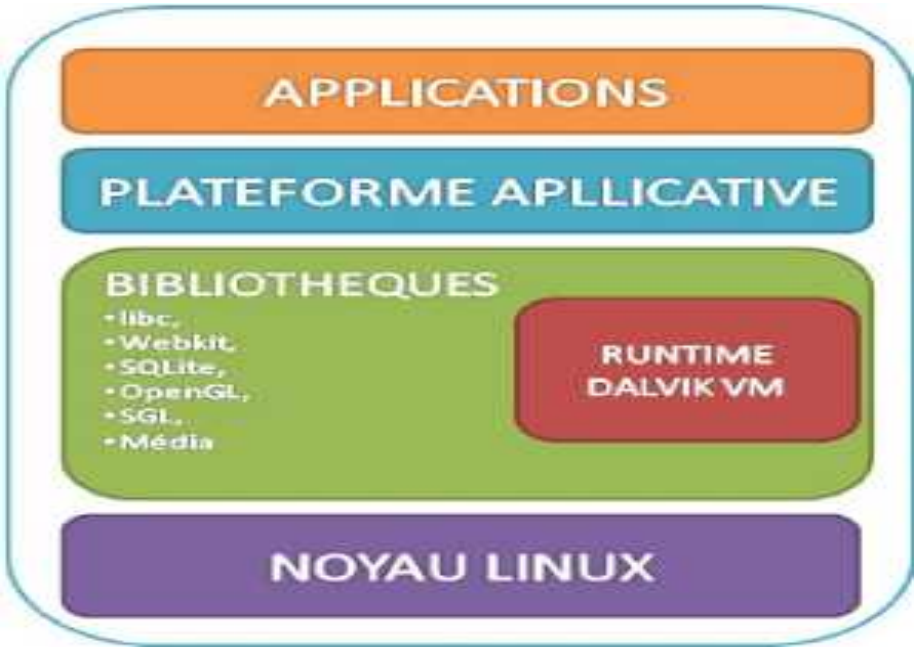


Figure I.8: Architecture d'android.

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

a. Le noyau Linux :

Le système d'exploitation Android se base sur Linux, c'est le noyau (« kernel » en anglais) de Linux qui est utilisé. Le noyau est l'élément du système d'exploitation qui permet de faire le pont entre le matériel et le logiciel. Par exemple les pilotes Wifi permettent de contrôler la puce Wifi.
La version du noyau utilisée avec Android est une version conçue spécialement pour l'environnement mobile, avec une gestion avancée de la batterie et une gestion particulière de la mémoire. C'est cette couche qui fait en sorte qu'Android soit compatible avec tant de supports différents.

b. Les bibliothèques pour Android :

Ces bibliothèques proviennent de beaucoup de projets open-sources, écris en C/C++ pour la plupart, comme SQLite pour les bases de données, WebKit pour la navigation web ou encore OpenGL afin de produire des graphismes en 2D ou en 3D.

c. Le Framework pour les applications :

Un Framework peut se traduire littéralement par un cadre de travail. Il s'agit d'un ensemble de composants qui définissent les fondations ainsi que les grandes lignes directrices de l'organisation d'un code, en d'autre termes en peut parler de son architecture ou de squelette. Un framework prodigue aussi quelques fonctionnalités de base (accès à la base de données par exemple). Cet outil fournit ainsi une démarcation radicale entre plusieurs aspects d'un programme et permet de mieux diviser les taches. En fait ce frameworks là qui sont disponibles quand on programme en Java, et qui s'occupent d'interagir avec la bibliothèque Android.

d. Les applications :

Il s'agit tout simplement d'un ensemble d'applications que l'on peut trouver sur Android, par exemple les fonctionnalités de base inclues un client recevoir/envoyer des emails, un programme pour envoyer/recevoir des SMS, un répertoire, etc.

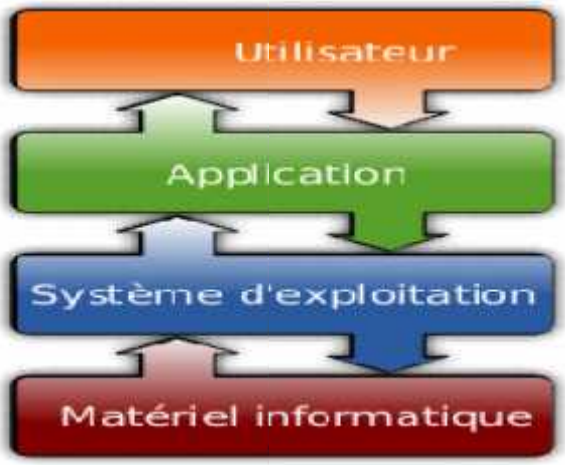


Figure I.9: Interactions entre les couches.

4.Les applications Android :

Android est livré avec un ensemble d'applications de base, dont un client de messagerie, un programme des SMS, un calendrier, un navigateur, le répertoire des contacts. Toutes les applications sont développées en utilisant le langage de programmation Java.

4.1.Les composants principaux d'une application android :

Une application Android est écrite en Java et est compilée à l'aide du SDK Android, elle est composée en 4 composants principaux :

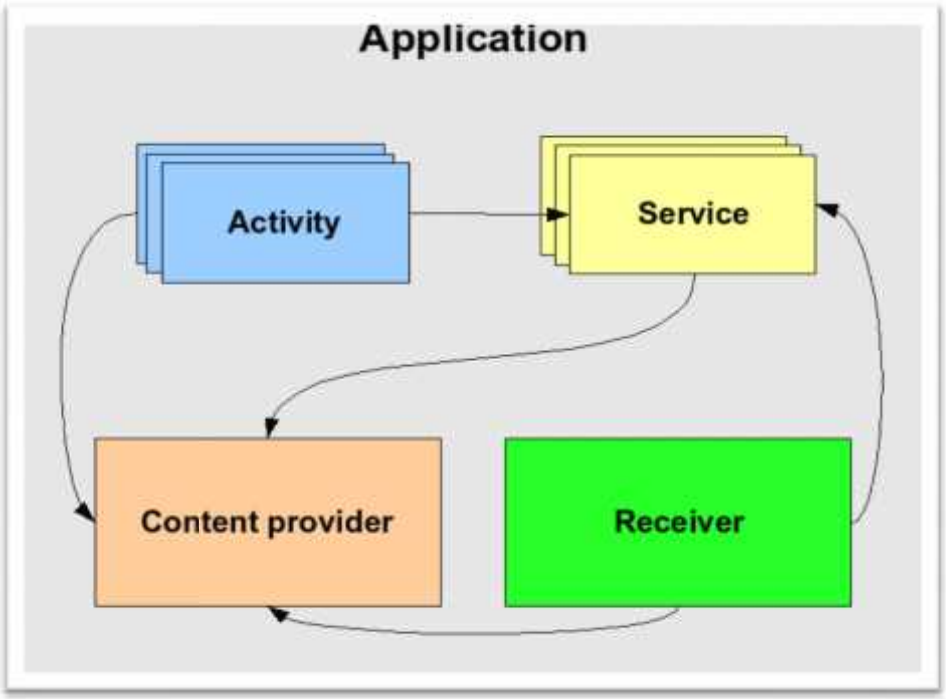


Figure I.10 : Les composants principaux d'une application Android

4.1.1.Composant « Activity »

L'activité est le premier composant essentiel permettant de gérer le cycle de vie d'une application et l'interactivité avec l'utilisateur moyennant une interface graphique. Par extension, on dit que l'activité est la forme disponible sur l'écran.Elle offre à l'utilisateur l'interface (écran) pour interagir avec l'application. Chaque écran de l'application représente une activité.

Une application Android est composée d'une ou de plusieurs activités (un jeu sur plusieurs écrans).

Dans « Java » chaque activité est une classe qui étend par héritage la classe« Activity ».

Lors du démarrage d'une activité, 3 méthodes sont appelées automatiquement : « onCreate », « onStart » et « onResume ». Lorsque l'on quitte l'activité, 3 méthodes sont appelées automatiquement : « onPause », « onStop » et « onDestroy ».

Ces méthodes sont définies dans la classe « Activity » et devront être redéfinies si c'est nécessaire.

L'ensemble du code « Java » est généralement écrit dans la méthode « onCreate ».

La sauvegarde des données importantes quant à elle est effectué dans la méthode « onPause ».

En effet quand une activité n'a plus « le focus », elle est mise en pause d'où l'appel à la méthode « onPause ». Il est donc préférable de sauvegarder les informations critiques avant de se mettre en pause.Quand l'activité « onPause » termine sa période de repos, le système appelle la méthode « onResume ».

❖ Description des états :

- État « **start** » : quand l'activité n'est pas encore référencée en mémoire, elle est dans un état « **start** ».
- État « **Resume** » / « **Running** » : une activité qui a le focus, est dans un état « **Running** » à un instant « **t** ». Cette activité se trouve au sommet de la pile.
- État « **Pause** » : une activité qui n'a pas le focus (donc n'interagit pas avec l'utilisateur) mais reste visible à l'écran, est dans un état « **Pause** ». Elle maintient

Son état et reste attachée au gestionnaire de fenêtres. Cette activité peut quand même être détruite par le système si ce dernier se trouve en manque de mémoire par exemple.

- État « **Stopped** » : une activité qui n'est pas visible à l'écran mais existe quand même en mémoire, est dans un état « **Stopped** ».
- État « **Destroyed** » : quand l'activité n'est plus en mémoire, elle est détruite. Ce phénomène arrive, car le gestionnaire d'activités décide qu'une telle activité n'est plus utilisée.

❖ Cycle de vie d'une activité :

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

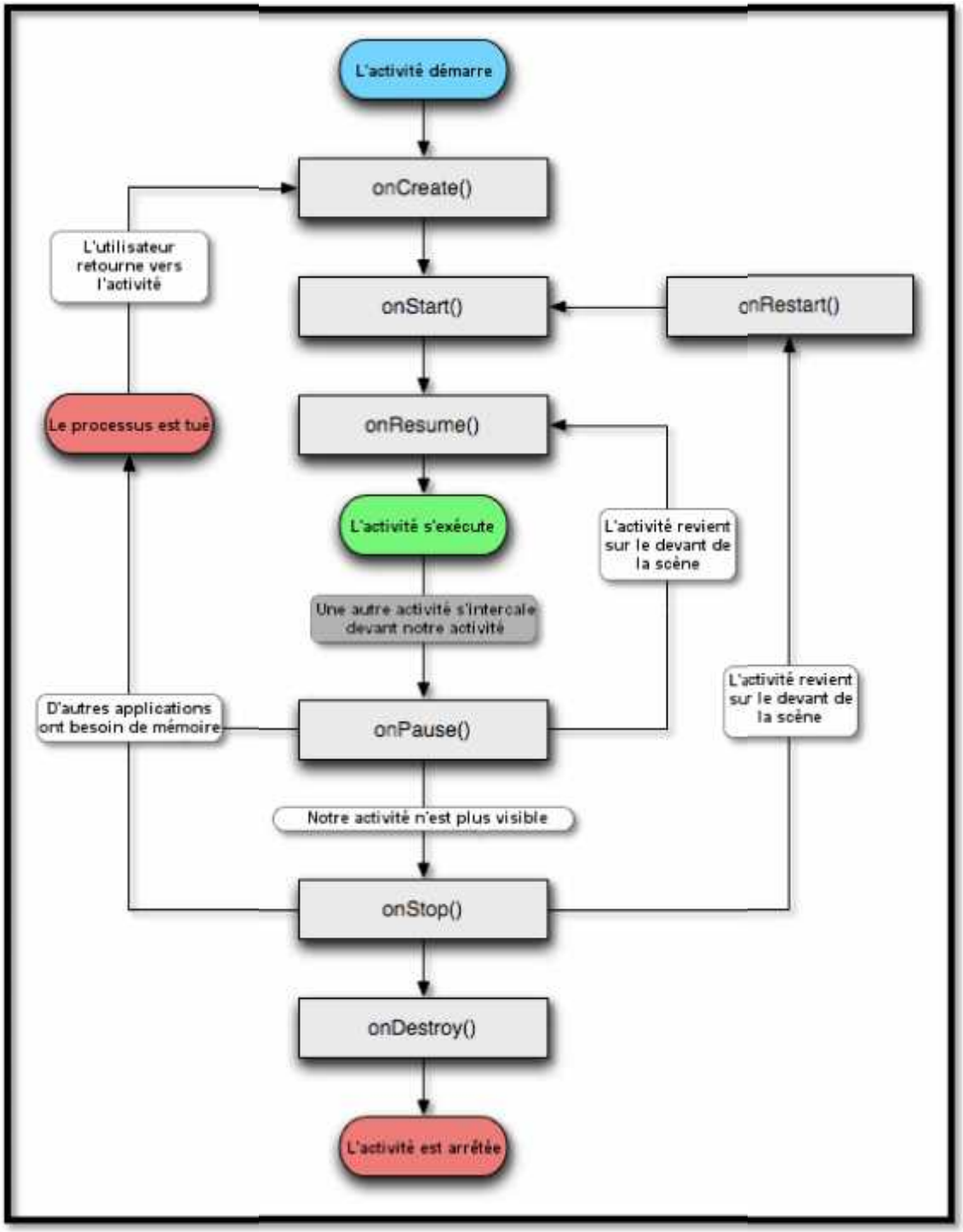


Figure I.11: Cycle de vie d'une activité

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

✓ onCreate() :

- Exécuté quand l'utilisateur clique sur l'icône de l'application pour une première fois
- Utilisé pour l'initialisation :
  - vue XML.
  - des fichiers/données temporaires.

✓ onRestart() :

- Exécuté lors l'activité arrêtée via un stop() redémarre (repasse en premier plan).

✓ onStart() :

- Exécuté après chaque onCreate() ou onRestart().
- Chargement des données sauvegardées durant le dernier arrêt.

✓ onResume() :

- Exécuté après chaque onStart().
- Exécuté a chaque passage en premier plan de l'activité.
- Initialisation :
  - Connexion DB.
  - Mise à jour des données qui auraient pus être modifiées entre temps (avant l'onResume).

✓ onPause() :

- Exécuté avant chaque onStop ().
- Exécuté chaque fois que l'utilisateur passe à une autre activité, ou bien lorsqu'il demande un "finish ()" sur cette activité, ou bien encore si le système à besoin de libérer de la mémoire.
- Libération des ressources :
  - sauvegarde des données qui seront perdues après l'arrêt si elles ne sont pas sauvegardées.
  - connections DB.

✓ onStop () :

- Exécuté avant chaque mise en sommeil.
- Exécuté avant chaque onDe stroy:
  - Libération des ressources;

✓ onDestroy() :

- Exécuté lors du kill / de l'arrêt de l'activité.
- onCreate () devra à nouveau être exécuté pour obtenir à nouveau l'activité.
- libération des ressources.
- fichiers temporaires.

❖ Implémentation d'une activité :

Les activités héritent de la classe Activity. Or, la classe Activity hérite de l'interface Contexte dont le but est de représenter tous les composants d'une application. On les trouve dans le package android.app.Activity.

Un point des plus importants, toute activité de l'application doit être déclarée dans son fichier manifeste.

A la création d'un nouveau projet Android, l'activité crée en premier est celle qui sera définie comme étant l'activité principale de l'application et sera lancée en premier lors de l'exécution de l'application, mais toutes les applications si ce n'est toutes contiennent plusieurs activités, il devient alors nécessaire de naviguer entre elles.

• La navigation entre activités :

La navigation entre les écrans peut se faire de deux façons différentes:
Soit explicitement, soit implicitement. Dans les deux cas, l'ordre de changement d'activité est véhiculé par un objet de type Intent (intention en anglais).

• Le mode explicite :

Les activités s'enchaînent les unes aux autres par invocation directe. C'est-à-dire qu'une activité donnée déclenche l'affichage d'une autre activité en appelant la méthode startActivity avec un Intent mentionnant clairement le nom de l'activité.

Le mode explicite est donc très classique : comme dans la plupart des applications, les écrans à afficher sont invariablement les mêmes d'une exécution à l'autre et identifiés à l'avance. Aucune particularité n'est à noter dans ce mode de fonctionnement si ce n'est qu'Android permet d'afficher des activités n'appartenant pas à l'application d'origine.

• Le mode implicite :

Par contre est une spécificité d'Android extrêmement puissante. Dans ce mode, le nom de l'activité n'est pas précisé nominativement. L'objet Intent qui encapsule la demande de changement ne porte pas l'identification de l'activité mais un descriptif des caractéristiques ou plutôt des capacités de traitement dont l'activité devra être dotées.

Ensuite, une mécanique de résolution se met en marche, Android recherche dans tout le système, et non pas uniquement dans l'application courante, les activités répondant aux exigences exprimées dans l'Intent. À la fin de cet algorithme, l'activité identifiée s'affiche alors. Au cas où il y aurait plusieurs activités en mesure d'assurer le service demandé, leur liste est proposée à l'utilisateur qui devra alors en sélectionner une.

4.1.2.Le composant « intents »

Les intents sont l'une des pierres angulaires de la plateforme Android. Nous pouvons les comparer à des actions ou même à des intentions, ils permettent de dialoguer à travers le système à partir de canaux qui leurs sont dédiés. Quand le mobile reçoit un appel, la plateforme lance un Intent signalant l'arrivée d'un appel, de même pour un SMS.

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

4.1.2.1. Content Provider :

Est un composant d'application qui permet les accès en lecture/écriture aux données privées de cette application sous une restriction. Il utilise les syntaxes, le mécanisme standard pour demander et renvoyer de données.

4.1.2.2. Les services (service)

Un Service est un composant d'application qui peut effectuer des opérations de longue durée dans le fond et ne fournit pas d'interface utilisateur. Un autre élément d'application peut démarrer un service et il continuera à fonctionner en arrière-plan même si l'utilisateur passe à une autre application.

En outre, un composant peut se lier à un service à interagir avec lui et même effectuer communication interprocessus. Par exemple, un service peut gérer les transactions réseau, écouter de la musique, effectuez déposer des entrées/sorties tous issus de l'arrière-plan.

4.1.2.3. Broadcast receiver

Pour pouvoir recevoir des intents, Android nous permet de créer une classe qui implémente BroadcastReceiver. Ces objets sont conçus pour recevoir des intents (intentions) et appliquer des comportements spécifiques à notre code.

L'interface BroadcastReceiver ne possède qu'une seule méthode onReceive() que notre classe devra implémenter.

5. La téléphonie :

5.1 Introduction :

Pour la réalisation d'un système de messagerie instantanée ou de visiophonie (téléphonie comprise) j'aurais besoin d'outils spécifiques plus exactement des serveurs dédiés a ces taches la .
Je citerais donc quelque serveurs pour la visiophonie/téléphonie IP open source :

- Jistsi.
- WebRTC/apprtc(20142016).
- Meeting.
- RED5 Server.
- Asterisk.
- YATE.

5.2 La téléphonie IP/visiophonie :

On utilise le terme de voix sur IP pour désigner une communication vocale et/ou vidéo d'un point fixe a un autre sur le réseau internet, par l'intermédiaire de routeurs . la VoIP (acronyme anglais de « Voice over Internet Protocole ») est un protocole de communication permettant de transférer la voix/image sur internet par les réseaux a large bande passante. Le flux de voix et vidéo est alors décomposée en paquets qui circulent sur la toile mondiale . ce qu'on appel aussi la téléphonie par internet se distingue des réseaux filaires classiques par les lignes de cuivre des opérateurs (FAI).

Depuis les années 80 la VoIP cherche a trouver son marché, mais c'est l'arrivé de larges bandes et donc l'amélioration de la vitesse et de la qualité des transmissions internet qui a permis l'explosion de la voix sur IP .

5.2.1 WebRTC :

Aujourd'hui sur 7 milliards d'habitants, 3,2 milliards sont constamment connectés à internet. L'usage des Smartphones et des tablettes est en croissance considérable, parce qu'à présent, on peut faire quasiment tout ce que l'on veut en utilisant des appareils Smartphones. De façon inévitable, le besoin d'une meilleure communication émerge, et c'est là que le WebRTC intervient.

5.2.2 définition :

Le WebRTC – signifiant Web Real Time Communication (communication web en temps réel) – est simplement un standard ouvert pour intégrer des capacités de communications multimédia en temps réel, directement dans un navigateur web. La structure du standard ouvert s'affranchit des logiciels, plug-ins et téléchargements. L'effort du WebRTC est en cours de standardisation au niveau de l'API par le W3C, et au niveau du protocole par l'IETF.

Figure I.12 : Logo WebRTC

5.2.3. Description générale de la norme :

L'architecture de l'API WebRTC repose sur une construction triangulaire impliquant un serveur et deux pairs. Les deux navigateurs téléchargent depuis un serveur une application JavaScript vers leur contexte local. Le serveur est utilisé comme point de rendez vous afin de coordonner les échanges entre navigateurs jusqu'à ce que la connexion directe entre navigateurs soit établie. L'application téléchargée utilise l'API WebRTC pour communiquer avec le contexte local. Le but est d'avoir une application cliente crossplatforme au travers de l'API WebRTC.

Les flux d'échange entre clients peuvent rencontrer divers serveurs qui se chargeront de modifier, traduire ou gérer le signal au besoin, permettant par exemple la traversée de pare-feux, proxys ou NAT.

Afin d'établir une connexion utilisant le standard WebRTC, les clients A et B doivent être connectés simultanément au service pt permettant de maintenir la connexion ouverte par HTTPS ou socket. Lorsque le client A souhaite établir la connexion avec B, l'API instancie un objet PeerConnection qui, une fois créé, permet d'établir des flux de médias ou de données. Il est aussi nécessaire, pour une vidéoconférence par exemple, que les utilisateurs A et B acceptent le partage de leur caméra et/ou de leur microphone.

Une fois cet objet PeerConnection créé par A, le client envoie au serveur un paquet contenant les informations sur les médias partagés ainsi qu'une empreinte liant la connexion à A. Le serveur va décoder ce paquet et identifier qu'il s'agit d'une communication à destination de B et enverra donc un signal à B. B est notifié du souhait de A d'établir une connexion et accepte ou non sa requête. Si elle est acceptée, le même processus a lieu entre B et A cette fois afin d'établir la connexion bidirectionnelle. Une fois celle-ci établie, les flux de médias ou de données peuvent être ajoutés à la connexion librement.

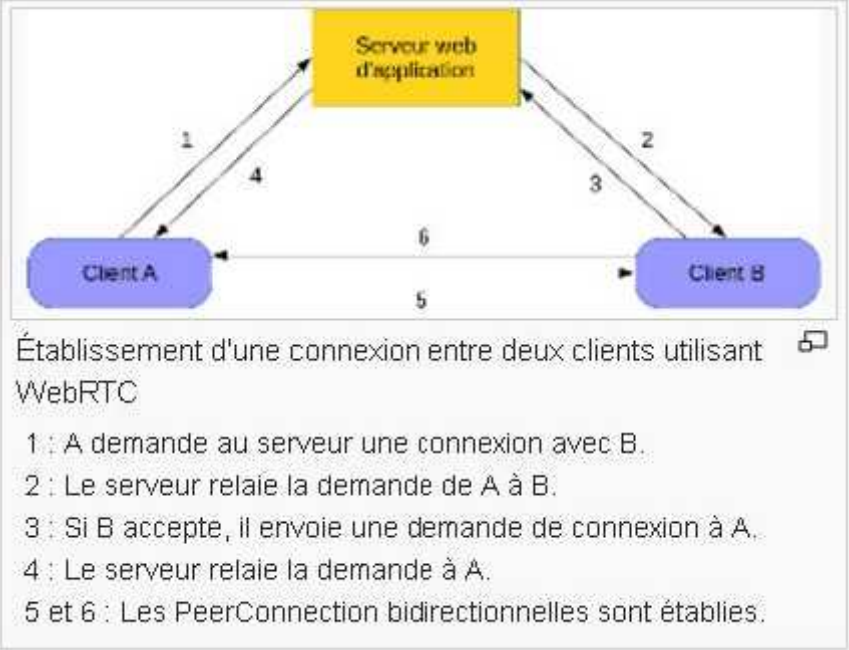


Figure I.13 : Etablissement d'une connexion entre deux clients utilisant WEB RTC

5.2.4. présentation du protocole :

WebRTC est une interface de programmation (API) Javascript développée au sein du W3C et de l'IETF, le but étant de permettre une communication en temps réel, le partage de fichiers (et bientôt le partage d'écran) avec plusieurs utilisateurs de la plateforme en s'affranchissant des plugins propriétaires jusqu'à lors nécessaires et en proposant un service simple d'accès et sans installation complémentaire.

Le WebRTC se décline en trois autres APIs assurant chacune une fonction spécifique :

- **L'API MediaStream** :qui permet au navigateur d'accéder et de manipuler des flux de données audio ou vidéo (comme ceux provenant de la caméra ou du microphone).
- **L'API RTCPeerConnection** :qui assure la communication stable et efficace entre plusieurs utilisateurs (pair à pair) pour l'envoi et la réception de flux de données.
- **L'API RTCDataChannel** :permettant d'envoyer et de recevoir des données génériques à travers un canal de communication pair à pair.

De plus, l'implémentation des deux derniers APIs doit s'accompagner d'un autre mécanisme de fonctionnement essentiel. En effet, avant d'établir une communication pair à pair stable entre des utilisateurs, un mécanisme de signalisation doit être mis en place par les développeurs pour émettre des messages de contrôle afin de coordonner et établir les communications pair à pair. Ce mécanisme n'est pas défini dans l'API WebRTC et il appartient aux développeurs de mettre en place un canal de communication duplex pour l'échange des informations de signalisation. Sur la plateforme Oryba, ce rôle est endossé par le serveur de communication WebSocket.

5.2.4.1 l'API MediaStream :

L'API MediaStream est basée sur la manipulation d'un objet MediaStream représentant un flux de données audio ou vidéo. Typiquement un objet MediaStream est généré par la méthode getUserMedia de l'API prenant en paramètres : - Un objet Constraints spécifiant si le flux généré va être de type audio ou vidéo ou encore des informations sur la résolution, la fréquence d'affichage ou encore le ratio d'aspect du flux. - Une méthode de callback qui sera appelée en cas de succès de génération du flux avec en paramètre l'objet MediaStream représentant le flux généré. - Une méthode de callback qui sera appelée en cas d'échec de génération du flux avec en paramètre un objet représentant l'erreur générée. Un objet MediaStream contient également un ou plusieurs objets MediaStreamTrack représentant les différentes pistes audio ou vidéo du flux. Ainsi, chaque piste est associée à une source d'entrée distincte (Ex : Microphone, Camera avant, Camera arrière, Partage d'écran, etc.).

Enfin, un objet MediaStream peut être converti en une chaîne de localisation (URL) pour définir la source d'un élément audio ou vidéo dans la page web en local ou encore cet objet peut être envoyé grâce à l'API RTCPeerConnection afin de générer un flux de données à travers le réseau.

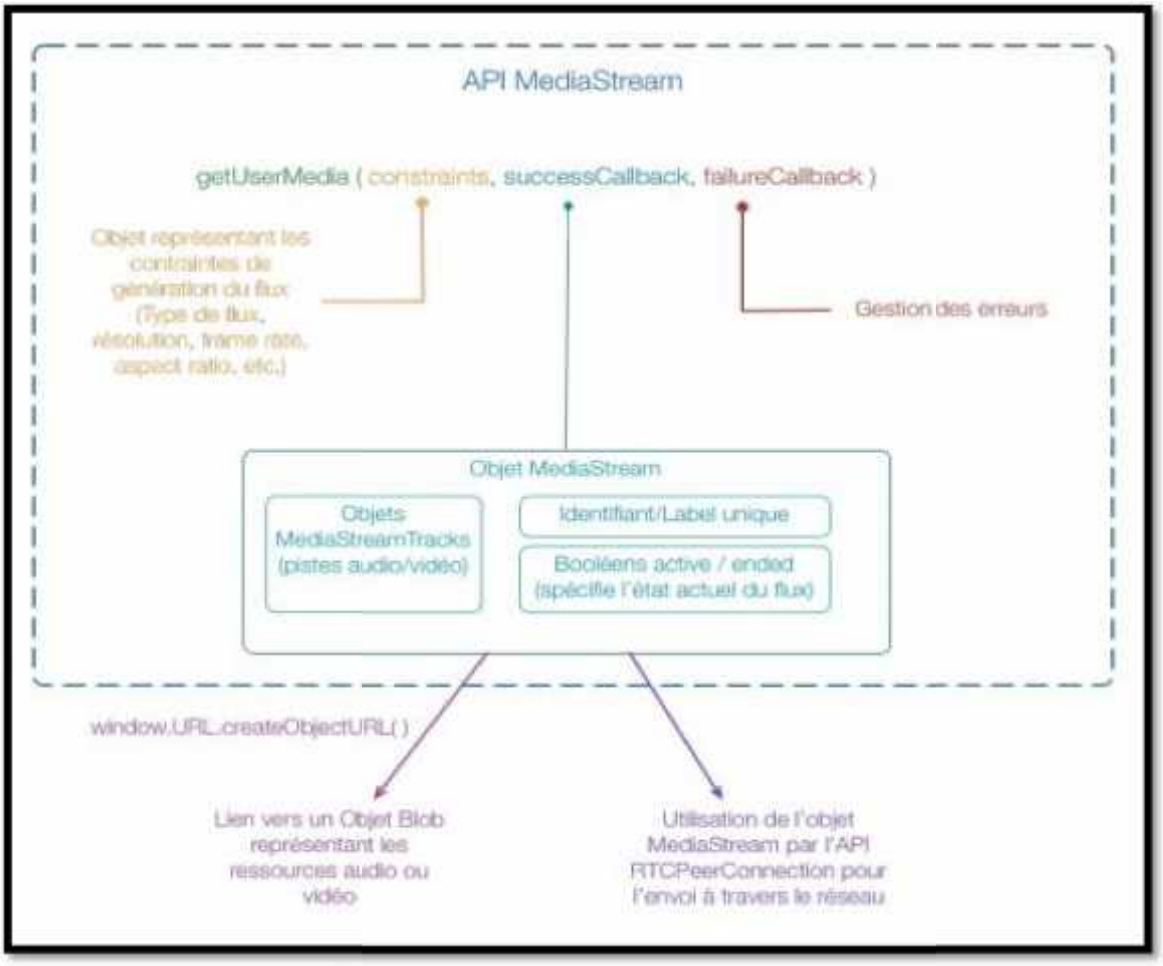


Figure I.14 Fonctionnement globale de l'api MediaStream

- ✓ **Mécanisme de signalisation :**

Avant d'établir une communication pair à pair entre les utilisateurs de la plateforme afin d'échanger des flux de données, certaines informations doivent être communiquées entre les différents clients à travers un canal de communication duplex. Ce mécanisme de signalisation est assuré par le serveur de communication WebSocket de la plateforme et permet aux utilisateurs d'échanger : - Des messages de contrôle de session pour initialiser ou fermer les communications ainsi que pour reporter les erreurs

éventuelles. - Des messages à propos de la configuration du réseau afin de déterminer les adresses IP et ports de chaque client. - Des messages à propos de la configuration des codecs audio/vidéo ou encore des résolutions pris en charge par les navigateurs de chaque client. Ainsi, lorsque le mécanisme de signalisation a été complété avec succès, la connexion pair à pair entre les utilisateurs peut s'établir et fonctionner en autonomie à partir de cet instant pour l'échange de flux de données.

5.2.4.2 API RTCPeerConnection :

L'Api RTCPeerConnection implémente un ensemble d'outils pour permettre la communication pair à pair entre différents utilisateurs de la plateforme et l'échange de flux de données entre ces utilisateurs. Un certain nombre d'étapes faisant intervenir le mécanisme de signalisation et l'Api sont donc nécessaires pour établir une connexion pair à pair et procéder à l'échange de flux. Une demande de connexion pour un appel audio ou vidéo sur la plateforme suit donc le procédé suivant :

- ✓ Pour initialiser un appel, un client envoie une requête d'offre à d'autres clients connectés via le serveur de communication WebSocket.
- ✓ La requête d'offre est accompagnée d'un objet RTCSessionDescription contenant les informations concernant la localisation du demandeur (adresse IP, port) ainsi que des informations sur la configuration du navigateur (codecs audio/vidéo, etc.). Ces informations sont décrites en suivant le protocole SDP (Session Description Protocol) défini par l'IETF en 2006.
- ✓ Lorsque les clients reçoivent l'offre, chacun d'entre eux envoient une réponse (toujours via le serveur WebSocket) au demandeur accompagnée également d'un objet RTCSessionDescription, ce premier processus se termine lorsque chaque paire de clients s'est échangée leur description respective.
- ✓ Pendant le même temps, l'Api RTCPeerConnection génère un ou plusieurs objets RTCIceCandidate grâce notamment au serveur STUN (Simple Traversal of UDP through NATs) mis en place dans notre environnement logiciel dont le rôle est de déterminer l'adresse IP d'un client derrière un routeur NAT. un objet RTCIceCandidate contient des informations sur la localisation d'un client ainsi que la route vers ce client. Lorsqu'un objet de ce type est généré, il va être envoyé aux autres utilisateurs pour qu'ils soient au courant le plus précisément possible de la localisation de chaque client et de la route optimale pour établir une connexion pair à pair avec ceux-ci.

Lorsque ces différentes étapes sont complétées avec succès, une connexion pair à pair peut s'établir entre les différents clients et les flux de données audio et vidéo peuvent être échangés via ce nouveau canal de communication.
Les objets MediaStream alors générés grâce à l'Api MediaStream peuvent alors transiter à travers ce canal. L'envoi d'un objet MediaStream s'effectue grâce à la méthode addStream de l'objet RTCPeerConnection dés lors de sa création avant même de faire une requête d'offre.

La méthode onICECandidateStateChanged est déclenchée lorsque la connexion change de statut, on distingue ainsi quelques statuts principaux :
- **Completed** : La requête d'offre a correctement été envoyée.
- **Connected** : La réponse d'un client à l'offre a bien été envoyée.
- **Disconnected** : La vérification de la bonne connexion entre deux pairs a échoué, il existe donc un problème éventuel sur la connexion.
- **Closed** : La connexion pair à pair entre deux clients a été fermée.

Afin de terminer une connexion pair à pair avec un client et donc pour mettre fin à une communication audio ou vidéo, il faut faire appel à la méthode removeStream de l'Api RTCPeerConnection pour détruire l'objet MediaStream en cours d'utilisation et ensuite de réinitialiser une requête d'offre qui n'aboutira pas car plus aucun flux de données n'est disponible

5.2.4.3 API RTCDataChannel :

L'Api RTCDataChannel permet de créer une connexion pair à pair entre deux utilisateurs afin d'échanger des données génériques. L'implémentation de cette Api permettra de fournir aux utilisateurs de la plateforme une fonction de partage de fichiers simple et efficace. Cette fonctionnalité n'est pas encore disponible au moment où est rédigé ce mémoire mais est prévue dans les prochaines versions de l'application. Le schéma qui suit présente avec précision l'implémentation des différentes APIs de la technologie WebRTC sur la plateforme pour assurer les fonctions d'appels audio et vidéo entre les utilisateurs d'un salon virtuel.

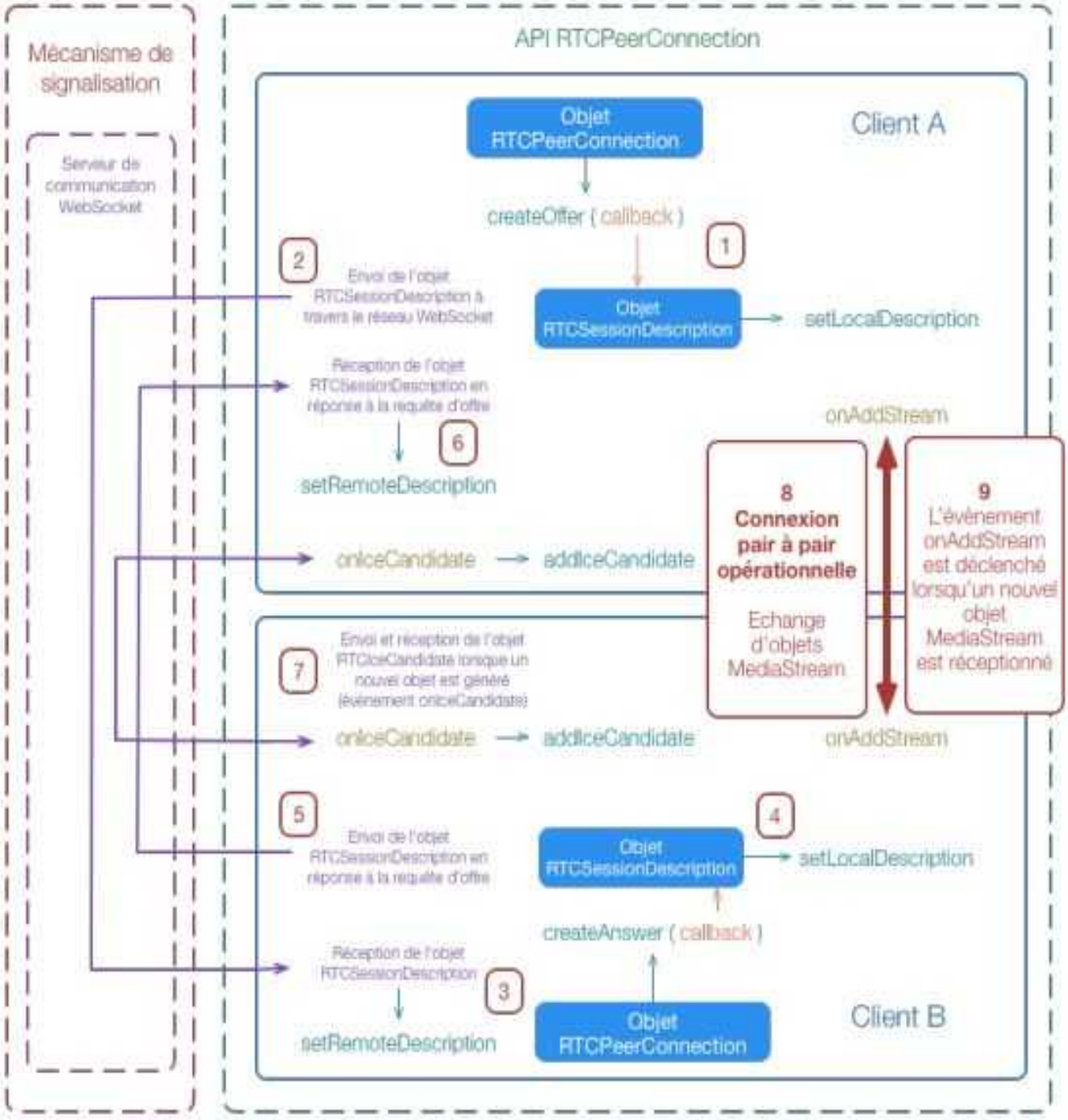


Figure I.15 Fonctionnement global de l'API WebRTC pour la mise en route d'un appel audio ou vidéo

5.2.5 Codecs Standards dans le WebRTC :

Un codec est un dispositif capable de compresser et/ou de décompresser un signal numérique. Ce dispositif peut être un circuit électronique, un circuit intégré ou un logiciel.
Le mot-valise « codec » vient de « codage-décodage » (code-decode en anglais).
D'un côté, les codecs encodent des flux ou des signaux pour la transmission, le stockage ou le chiffrement de données. D'un autre côté, ils décodent ces flux ou signaux pour édition ou restitution.
Ci-dessous un tableau qui défini les codecs qui sont utilisé par le webrtc :

Codecs	Utilisation	Spécification
Opus	De la bande étroite (narrow band) a la bande large d'internet ,codec audio utilisé pour le partage de music et de la voix sur internet	RFC 6716
G.711	modulation par impulsion et codage codec audio (PCM) pour réseau téléphonique commuté public avec une compatibilité compléte avec les systemes voip .	RFC 3551
Telephone events	Transport des informations par les tonalités DTMF	RFC 4733
H.264	Codec video très repandu (MPEG4)	RFC 6184
VP8	Codec video open source	RFC6386

Figure I.16 : Liste des codecs supporté par le WebRTC

5.2.6 Avantages du WebRTC :

Jusqu'à il n'y a pas si longtemps, lorsqu'on voulait proposer un service de visioconférence sur le net, il fallait obligatoirement passer par Flash, Silverlight ou des machins encore plus proprios avec bien évidemment des plugins à installer chez l'internaute pour que ça fonctionne.
Mais cette époque est révolue grâce au WebRTC qui utilise des technos ouvertes implémentées dans les plateformes les plus récentes .
Utiliser du WebRTC sur son site, application Android ou IOS c'est donc participer à un Internet ouvert, se faire plaisir avec le code et surtout ne plus prendre la tête.
De plus, le WebRTC est capable de passer à travers les firewalls / NAT et fonctionne réellement en P2P, évitant la centralisation des connexions (et soulageant les admins sys).

Globalement, WebRTC présente de nombreux avantages. Déjà la vidéo est de meilleure qualité qu'avec Flash ou d'autres support open source (possibilité de HD) . Les temps de connexion et la latence sont aussi réduits puisqu'utilisant JavaScript et les Web sockets .
Il ne nécessite pas d'installation ou de plugins , il est rédigé en javascript qui est connu pour être un langage qui s'exécute directement et très rapidement dans la machine , il exploite les d'une manière intelligente les ressources hardware , et crée automatiquement une gestion optimisé de votre solution .
Pour informations le JavaScript s'exécute avec le moteur V8 développé sur chrome , ce dernier le classe en première place dans la rapidité des langages .

6. La messagerie instantané :

La messagerie instantanée (ou IM) est aujourd'hui largement utilisée comme un moyen rapide et léger de communication, aussi bien professionnelle que personnelle. La messagerie instantanée est, par définition, instantanée, ce qui lui donne un avantage sur l'email quand il s'agit de notification rapide

6.1 Fonctionnement de la messagerie instantanée :

La messagerie instantanée est un moyen de communiquer en privé avec d'autres personnes de son choix. Le client se connecte à un serveur qui contient les informations sur tous les utilisateurs inscrits, connectés ou non. Chaque personne possède un pseudonyme qui n'est pas forcément unique, et un identifiant unique dans la base de données du serveur. Cet identifiant peut être un numéro, une adresse e-mail ou autre , dans mon cas a moi sa seras le numéro .
Deux personnes peuvent communiquer en direct si elles sont simultanément connectées au serveur. Sinon, elles peuvent consulter leurs messages dans leur boîte aux lettres au moment où elles se connectent.

6.2 Openfire et la messagerie instantanée :

Openfire est un serveur de tchat ou de collaboration en temps réel utilisant le protocole Jabber/XMPP écrit en Java et distribué sous licence publique générale GNU. L'essentiel de la configuration et de l'administration du serveur se fait à travers une interface Web. Les administrateurs peuvent se connecter de n'importe où et éditer la configuration du serveur, ajouter ou supprimer des utilisateurs, des salons de conversation.
Openfire peut utiliser sa propre base de données ou se connecter sur un SGDB externe comme Microsoft SQL, Oracle ou MySQL.

6.3 Fonctionnalités d’Openfire :

Openfire offre une administration web front-end (interface web), facile a utilisé indépendamment des réglages du navigateur, l’interface offre beaucoup d’informations et propose différentes options a choisir .par exemple, changer la façon dont Openfire traite les messages hors ligne , possibilité de stockage de renvois ou d’ignorer les messages . l’interface permet également un accès plus rapide a des parties pertinentes de la configuration. par exemple, dans la section « information server » , avec une liste des ports utilisés , Openfire a un lien vers la section « paramètres de sécurité » d’où on peut modifier les paramètres de sécurité des ports .

Il permet aussi de créer et gérer des groupes d’utilisateurs personnalisés (amis , famille, collègues ,…) facilement . ces groupes d’utilisateurs peuvent etre partagé avec les nouveau utilisateurs . il permet aussi de partager la présence et le statu des personnes connectés (disponible , occupé, hors-ligne …) . le partage de documents est aussi supporté .

Pour conclure openfire peut etre considéré comme un couteau suisse de la messagerie instantanée ! il dispose d’extensions qu’on appel plug-in qu’on attache au serveur .

6.4 Protocole de la messagerie instantanée sous Openfire :

Pour la réalisation d'un système de messagerie instantanée j'ai besoin d'outils spécifiques plus exactement des serveurs dédiés a ces taches la , je citerais les quelques services open sources :

- 12PLANET.
- erio/IMPS.
- NOVELL.
- IBM lotus sametime 7 server.
- Jabber/Openfire.
- SUN java system instant messaging .

Mon choix s'est prononcé pour openfire , car implémente entièrement le protocole jabber/XMPP pour la messagerie instantanée et la présence .

6.4.1 Le protocole jabber/XMPP :

XMPP (etensible messaging and presence protocol) est un protocole basé sur XML (Extensible Markup Language) et destiné a la messagerie instantanée (IM) ainsi que la détection de présence en ligne, qui a émergé de la communauté Jabber open-source en 1999. Il fonctionne entre ou parmi les serveurs (il peut être le cœur même du serveur ou juste un lien externe de communication avec d'autres serveurs) , et il fonctionne en temps quasi-réel . Le protocole peut éventuellement permettre aux internautes d'envoyer des messages instantanés a n'importe qui d'autre sur internet, indépendamment des différences dans les systèmes d'exploitation et navigateurs.

XMPP est prévu pour supporter des applications de messagerie instantanée avec authentification, contrôle d'accès, une haute sécurité pour la vie privée, le cryptage hop-by-hop , le cryptage de bout a bout , et la compatibilité avec d'autres protocoles .

XMPP est parfois appelé le protocole JABBER, mais c'est un abus de langage technique.

Lorsque la communauté Jabber a soumis le protocole à l'IETF en 2002, elle a choisi d'utiliser le nom « XMPP » au lieu de « Jabber » pour éliminer la possibilité de confusion avec la communauté open-source et la société abber.com commerciale (racheté par Cisco Systems Inc. en 2000).

6.5 Architecture d'openfire/Jabber :

Openfire fonctionne sur une architecture client/serveur et fonctionne sur le modèle de l'e-mail (amélioré pour permettre la messagerie instantanée). De ce fait , toutes les données envoyées au d'un client a un autre passent obligatoirement par le serveur .

Le client établit alors une connexion TCP avec le serveur en utilisant le port 5223 (port sécurisé) . La connexion ainsi établie perdure durant toute la durée de la communication, ce qui évite au client de demander une confirmation au serveur pour chaque nouveau message, selon mon point de vue personnel et suite a l'expérience engendrée par ce projet, il s'agit sans aucun doute d'une session.

La réception des messages se fait :

- Si le client est toujours connecté : Tous les messages destinés au client en question lui sont directement adressé.
- Si le client n'est pas connecté : Le serveur mémorise les messages qui lui sont adressés afin de pouvoir les lui transmettre lors de la prochaine connexion.

C'est cette capacité du serveur a savoir si le client est connecté ou pas , qui permet la livraison des messages en temps réel . c'est donc l'état de présence du client qui permet a la messagerie d'être instantanée .

6.5.1 L'adressage des clients :

Les serveurs Jabber et Openfire , utilisent une adresse appelée « Jabber ID » similaire a une adresse email, afin de permettre la communication entre les entités constituant une architecture Jabber . Un JID est composé de 2 parties :

- Nom d'utilisateur
- Nom du serveur

L'identifiant formé de ces informations a la forme suivante : utilisateur@serveur .

6.5.2 le client (application de communication):

Dans une architecture Jabber, le client dispose de très peu de restrictions . En effet un client Jabber est simple a mettre en œuvre . il se doit d'être capable de :

- D'établir une connexion TCP avec le serveur XMPP.
- D'analyser les messages XML qu'il reçoit afin de pouvoir les interpréter .
- Supporté les types de données de bases de XMPP. (présence par exemple)

6.5.3 Le serveur :

La flexibilité est une composante cruciale du protocole XMPP. Ainsi, sur un serveur Openfire, les administrateurs ont la possibilité d'ajouter des fonctionnalités au serveur d'origine . Dans le cas d'ajout de tels composants, la simplicité du serveur en lui même n'est pas sacrifiée. En revanche une certaine complexité est répercutée sur le déploiement du service .

Un serveur Openfire présente trois fonctionnalités de base :

- Gérer les connexions directes avec les clients.
- Communiquer avec les autres serveurs Jabber(Openfire ou autre serveurs jabber)
- Coordonner les différents composants parfois ajoutés par les administrateurs.
  La modularité du serveur repose sur le fait que chaque fonctionnalité (enregistrement, présence…etc.) est décrite par une portion de code indépendante.

Cette flexibilité et cette modularité sont a l'origine de l'interopérabilité de ce serveur et de sa simplicité de mise en œuvre.

Conclusion :

Pour la réalisation d'un système de téléphonie IP ou de messagerie instantanée , un développeur a le choix entre plusieurs serveurs open source ou propriétaire , qui offrent les fonctionnalités et les outils nécessaires selon le besoin .

Dans ce chapitre , j'ai présenté deux outils webRTC (apprtc) pour la téléphonie et la visiophonie ,et Openfire (pour la messagerie instantanée), leur fonctionnements , les protocoles qu'ils implémentent et leur normes . ces deux outils offre tout ce j'ai besoin pour passer a la conception de mon application , une architecture duplex est donc conçu pour mon application .

Mon choix s'est porté sur asterisk dans un premier temps , mais il y'as un « mais » ! cette technologie ne permet pas la visiophonie et très couteuse , lourde pour un déploiement et nécessite un serveur dédier qui fonctionne sous CentOs tout le temps (Je ne dispose pas de ressources matériels) de plus que la configuration des tables se doit d'être manuelle pour une utilisation sécurisé(périphérique de confiance pour chaque utilisateur) , sinon n'importe qui pourrais entendre ce qu'on dis sous le réseau (nécessitant de plus une adresse unique).

Asterisk devrais changer sa politique et s'aligné aux nouvelles technologies du web, en d'autre le X-platforme et interopérabilité .

Apprtc est projet financé par google , qui ne seras pas abandonnée d'aussi tôt ! pratiquement toutes les solutions de visio conférence (Mediastreaming) sont entrain de changer vers le cap du webRTC tel que Skype ,google duo ou tokbox .

Chapitre II : Analyse et conception

1- Introduction :

Dans cette partie du document, je présente ma contribution dans le domaine de la messagerie instantané avec toutes les ces fonctionnalités a travers une application Android. pour ce faire il me faut d'abord entamer une analyse pour identifier les différents acteurs qui interagissent avec mon application. Puis j'entamerais l'étape de la conception en s'appuyant sur les résultats de la phase d'analyse puis je donnerais la description détaillée du système le classes définies …etc

L'analyse et la conception de toute solution informatique sont d'une grande importance et elles doivent etre traité avec rigueur et précision car elles constituent la base du système a développer .mais avant cela je donnerais un aperçu de l'architecture de ma solution.

J'ai choisis donc la méthode UML pour la conception de mon application .

2- Présentation d'UML :

UML (*Unified Modeling Language*), que l'on peut traduire par « Langage de Modélisation Unifié » , est une notation permettant de modéliser un problème de façon standard, Ce langage est né de la fusion de plusieurs méthodes existantes auparavant , il est devenu désormais la référence en terme de modélisation objet , a un tel point que sa connaissance est souvent nécessaire pour obtenir un poste de développeur objet .

UML est un langage de modélisation graphique et textuel destiné a comprendre et a décrire des besoins , spécifier et documenter des systèmes , esquisser des architectures logicielles , concevoir des solutions et communiquer des points de vue . en effet UML est un langage avec une syntaxe et des règles bien définies qui tentent a réaliser les buts décrits grâce a une représentation graphique formée de diagrammes et d'une modélisation textuelle qui vient enrichir la représentation graphique .

2-1 Modélisation avec UML :

La modélisation consiste a crée une représentation simplifiée d'un problème : le modèle . Grave au modèle, il est possible de représenter simplement un problème , un concept et le simuler . la modélisation comporte deux composantes :

- L'analyse : c'est-à-dire l'étude du problème ;
- La conception : soit la mise au point d'une solution au problème

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

- Une démarche itérative et incrémentale : pour comprendre et représenter un système complexe , pour analyser par étapes , pour favoriser le prototypage et pour réduire et maitrisé l'inconnu.
- Une démarche guidée par les besoins des utilisateurs : Tout est basé sur le besoin de répondre a leur besoin. Chaque étape sera affinée et validée en fonction des besoins des utilisateurs .
- Une démarche centrée sur l'architecture logicielle : c'est la cl de succès du développement, les choix stratégiques définiront la qualité du logiciel .

2-2 La démarche de modélisation avec L'UML :

La démarche que j'ai adopté pour la conception de l'application s'appuie sur quatre étapes :

- Etape1 : Identification des acteurs et des besoins .
- Etape2 : Identification et représentation des cas d'utilisation .
- Etape3 : élaboration des diagrammes de séquences.
- Etape 4 : élaboration des diagrammes de classes.

La figure suivante donne la représentation graphique de la démarche de modélisation.

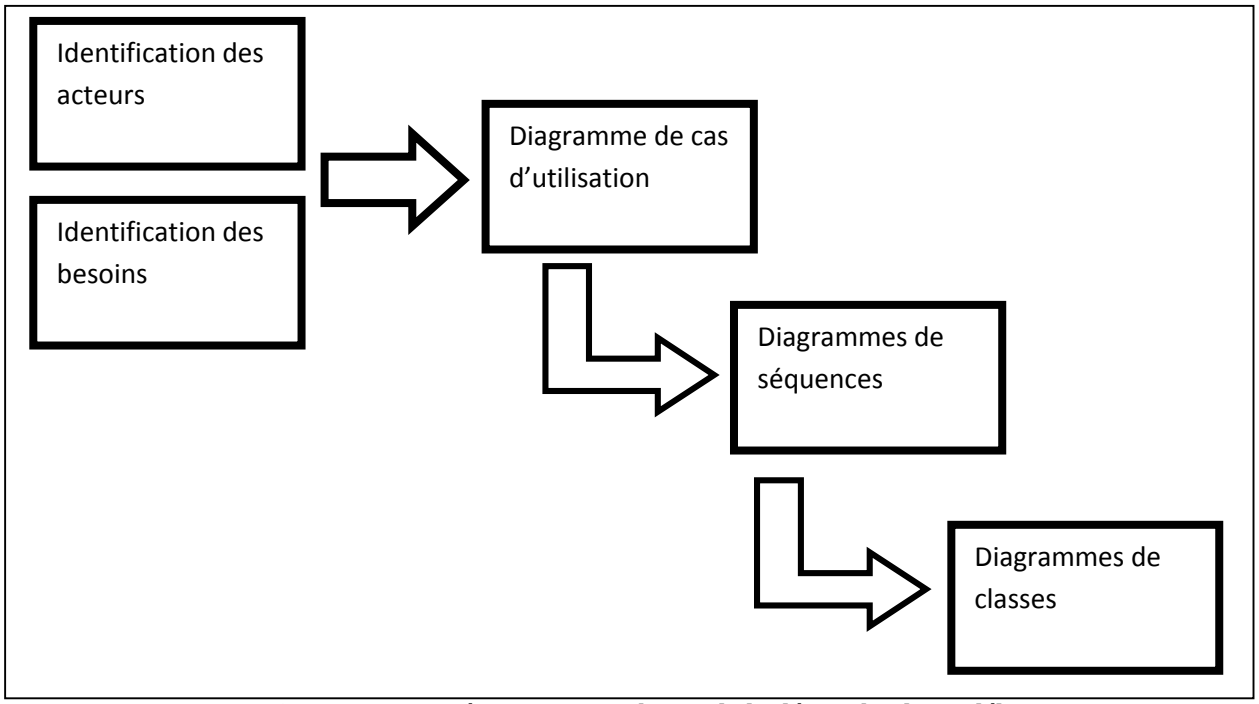


Figure II.1 Représentation graphique de la démarche de modélisation

3- Analyse :

Cette étape commence par l'étude des cas d'utilisation et de leurs scénarios ainsi que les besoins fonctionnels du système (ce que le systéme doit faire en réponse a une requéte d'un utilisateur) , qui aboutira a un ensemble de diagrammes représentant le modèle d'analyse .

III-1 Identification des besoins :

Les Objectifs principaux de mon application , c'est de permettre a l'utilisateur de passer des appels voix et video , la messagerie instantané le transfert de fichier et le partage de la localisation avec un cercle d'amis déjà existant .

La liste des fonctionnalités de mon application :

- récupération de la liste de contacts .
- affichage du profile d'un contact
- chatter avec un contact.
- appeler un contact.
- envoyer des fichiers multimédia a un contact.
- partager sa localisation via l'API MAP.

3-2 Identification des acteurs :

Un acteur est une entité externe pouvant interagir avec le système , le terme acteur ne désigne pas seulement les utilisateurs humains mais également les autres systèmes (matériel et logiciel) externes au système . Un acteur peut avoir le comportement suivant :

- Donner des informations au système.
- Recevoir des informations du système.
- Donner et recevoir des informations du système.

En réponse a l'action d'un acteur le système fournit un service qui correspond a son besoin , pour l'identification des acteurs , il faut se concentrer sur le rôle des différentes entités .

Les acteurs de mon système sont toute personne qui utilise mon application « text'it », qui serons regroupé sous le non utilisateur ou plus commun « User ».

3-3 Spécification des taches :

Une tache est l'ensemble des différentes fonctions qui peuvent être accédée par un acteur bien spécifié.

L'acteur défini précédemment (User) effectue un certain nombre de taches , ces dernières sont résumées dans le tableau ci-dessous :

Acteur	Taches
USER	T1 : Lancer l'application T2 : Inscription T3 :Accéder a la liste de contact T4 : Synchronisation des contacts T5 : Supprimer une conversation T6 : supprimer toutes les conversations T7 : accéder au profile d'un contact T8 : personnalisé les notifications . T9 : personnalisé l'arrière plan du chat T10 : envois d'un fichier audio T11 : envois d'une image/vidéo T12 : chatter avec un contact T13 : envoyer sa localisation T14 : modifier son image T15 : modifier son statu T16 : appelé un contact (audio) T17 : appelé un contact (Vidéo) T18 : Se déconnecté. T19 : Supprimer son compte

Figure II.2 : Spécification des taches .

3-4 Spécifications des scénarios :

Un scénario est une succession particulière d'enchainements qui s'exécute du début a la fin du cas d'utilisation. A chaque fois qu'une instance d'un acteur déclenche un cas d'utilisation, un scénario est crée . Ce scenario suivra un chemin particulier dans le cas d'utilisation.

Remarque : Durant mon étude les scénarios serons symbolisé par Si (i : est le numéro du scénario , noté S1 pour le premier)

Tableau des taches et leurs scénarios :

Utilisateur	Taches	Scénarios
USER	T1 :Lancer l'application	S1 : Cliquer sur l'icône « Text'it » S2 : connexion au serveur
	T2 : Inscription	S3 :Remplire les champs du formulaire S4 :s'enregistré
	T3 : Accéder a la liste des contacts	S5 :cliquer sur l'onglet contacts
	T4 :Synchronisation	S6 :cliquer sur options S7 :cliquer sur Sync contacts
	T5 :supprimer une conversation	S8 :cliquer sur l'onglet chats S9 :long clique sur la conversation voulus , et choisir supprimé
	T6 : supprimer toute les conversations	S10 :long clique sur une conversation , et choisir supprimé toutes les conversations
	T7 :acceder au profile d'un contact	S11 : cliquer sur le contact . S12 :cliquer sur le cercle en haut a droite (photo contact)
	T8 :personalisé les notifications	S13 :cliquer sur options S14 :cliquer sur notifications et sons

	T9 : personnalisé l'arriere plan du chat	S6 :cliquer sur l'onglet options S15 :cliquer sur arrière plan du chat S16 :choisir une image et cliquer sur définir
	T10 : envois clip audio	S17 :accéder a la conversation S18 : maintenir le bouton sous forme de micro S19 :relâché pour envoyer ou glisser pour annulé
	T11 :envois d'un fichier image/video	S17 :accéder a la conversation S20 : cliquer sur le trombone (Pièce jointe) et choisir le type S21 : choisir le fichier et cliquer sur envoyer
	T12 :Chatter avec un contact	S5 :cliquer sur l'onglet contacts S22 : choisir le contact S17 :acceder a la conversation S23 :écrire un message et cliquer sur envoyer .
	T13 : envoyer sa localisation	S17 :accéder a la conversation S24 : cliquer sur le trombone (Pièce jointe) et choisis localisation
	T14 : modifier son image	S6 :cliquer sur l'onglet options S25 : cliquer sur votre avatar S26 : choisir sa source S27 :sélectionner votre image et validé.
	T15 : modifier son statu	S28 : cliquer sur la petite boule en haut S29 :cliquer sur le statu souhaité
	T16 : appelé un contact (audio)	S5 :cliquer sur l'onglet contacts S30 :cliquer sur le bouton sous forme de téléphone a droite du contact
	T17 : appelé un contact (vidéo)	S5 :cliquer sur l'onglet contacts S31 :cliquer sur le bouton sous forme de camera a droite du contact
	T18 : Se déconnecté.	S6 :cliquer sur l'onglet options S32 : Cliquer sur se déconnecté S33 : Confirmer sa déconnexion
	T19 : Supprimer son compte	S6 :cliquer sur l'onglet options S34 : cliquer sur supprimé mon compte S35 : Confirmer l'action .

Figure II.3 : Spécification des scénarios**.**

3-5 Les cas d'utilisation :

3-5-1 Définition :

Un cas d'utilisation permet de mettre en évidence les relations fonctionnelles entre les acteurs et le système étudié .Le format de représentation d'un cas d'utilisation est complètement libre, mais l'UML propose un formalisme et des concepts issus de bonnes pratiques.

L'objectif ^poursuivi par les cas d'utilisation est de permettre de décrire la finalité des interactions du système et de ces utilisateurs.

3-5-2 Spécification des cas d'utilisation

Les figures suivantes présentent des descriptions de quelques cas d'utilisation de mon système

Cas d'utilisation : Inscription

Scénarios : S1,S3,S4

Auteur : User

Description :

1. L'utilisateur lance l'application en cliquant sur son icône.
2. Le système affiche un formulaire d'inscription (cas ou c'est le premier lancement de l'application)
3. L'utilisateur remplit le formulaire.
4. L'utilisateur clique sur le bouton commencer.
   Le système enregistre les informations, lance la synchronisation des contacts ,puis les affiche .

Figure II.4 : Cas d'utilisation « inscription »

Cas d'utilisation : supprimer une conversation

Scénarios : S8, S9

Auteur : User

Description :

1. L'utilisateur clique sur l'onglet CHAT
2. Le système affiche l'ensemble des discutions existantes.
3. L'utilisateur fait une clique longue sur la discussion souhaité.
4. Le système affiche une liste de choix
5. L'utilisateur clique alors sur supprimé.
6. Le système alors affiche une boite de dialogue pour une confirmation
7. L'utilisateur donne la confirmation de son choix
8. Le système supprime l'entité et actualise l'affichage .

Figure II.5 : Cas d'utilisation « supprimer une conversation »

Cas d'utilisation : envois du clip audio

Scenarios: S5, S22, S17, S19

Auteur: User

Description :

1. L'utilisateur clique sur l'onglet contact
2. Le système affiche l'ensemble des discutions existantes.
3. L'utilisateur fait un clique long sur la discussion.
4. Le système affiche la discussion précédente (historique si existe)
5. L'utilisateur clique un clique long sur le bouton micro en bas a droite .
6. Le system comprend qu'il s'agit d'un message vocal (clip audio) et l'enregistre dans sa mémoire temporaire.
7. L'utilisateur relâche son doigt pour envoyer le clip audio .
8. Le système envois le fichier au serveur de fichier pour le télécharger.
9. A la fin du téléchargement (upload) le système envois une requête au serveur XMPP contenant les informations du clip audio (nom et codec) aux clients pour commencer le transfert et la lecture de l'autre partie (User 2)

Figure II.6: Cas d'utilisation « envois d'un clip audio »

Cas d'utilisation : envoyer sa localisation

Scénarios : S5, S22, S17, S19

Auteur : User

Description :

1. L'utilisateur clique sur l'onglet contact
2. Le système affiche l'ensemble des discutions existantes.
3. L'utilisateur fait un clique long sur la discussion.
4. Le système affiche la discussion précédente (historique si existe)
5. L'utilisateur clique sur le bouton en haut de pièces jointes ,choisir localisation .
6. Le système exécute l'api Maps , et récupère les coordonnées de la localisation de la personne et l'intègre dans un lien MAPS .
7. Le système utilise la bibliothèque native (Android) pour envoyer un message image (thumbails) muni des coordonnées GPS de la position actuelle du téléphone.
8. L'utilisateur qui reçoit clique sur l'image, qui exécute le lien maps qui renvois dans une vue MAPS .
9. Le system alors affiche la position du contact dans une vue MAP.

Figure II.7 : cas d'utilisation « envoyer sa localisation »

Cas d'utilisation : appeler un contact (video)

Scénarios : S5, S31

Auteur : User

Description :

1. L'utilisateur clique sur l'onglet contacts
2. Le système affiche l'ensemble des discutions existantes.
3. L'utilisateur fait un click sur le bouton camera à coté des contacts.
4. Le système crée le nom d'une room (Room ID), avec l'identifiant JID des deux contactes et une clé généré pour sécurisé l'accès au a la chambre.
5. Le système crée la chambre au niveau du serveur, envois une requête d'appel au niveau du deuxième client.
6. Le deuxième client se connecte sur dans la même room, l'appel débute.

Figure II.8 : Cas d'utilisation « appeler un contact (video) »

Cas d'utilisation : se déconnecté

Scénarios : S6,S32,S33

Auteur : User

Description :

1. L'utilisateur clique sur le bouton «… » depuis l'interface de lancement et choisis options
2. Le système affiche le panneau d'options.
3. L'utilisateur fait un click sur le bouton se déconnecté.
4. Le système affiche un alerte dialogue, et demande une confirmation.
5. L'utilisateur alors, confirme son choix.
6. Le système alors se déconnecte.

Figure II.9 : Cas d'utilisation « Se déconnecté »

5- Conception :

Le processus de conception de mon application repose sur l'organisation conceptuelle logique et physique des données collectées durant la phase d'analyse . en effet , elle s'appuie essentiellement sur quelques diagrammes du langage de modélisation UML .

En effet après l'identification des différents acteurs ainsi que quelques cas d'utilisation qui sont mis en œuvre par ces acteurs , le diagramme de cas d'utilisation général est dégagé .

Chaque cas d'utilisation se traduit par un pou plusieurs scénarios. Chaque scénario fait l'objet d'une description sous forme graphique a l'aide d'un diagramme de séquence.

Enfin une identification des classes est fournie par la synthèse des diagrammes de séquences ainsi le diagramme de classe sera dégagé .

5-1 Le diagramme de cas d'utilisation :

Lors de la phase d'analyse j'ai pu identifier les acteurs ainsi que les cas d'utilisations associer a ces derniers. Ce qui me donne l'opportunité d'élaboré le diagramme des cas d'utilisation.

Les diagrammes de cas d'utilisation permettent de représenter un ensemble de cas d'utilisation , d'acteurs et leurs relations .

Les diagrammes de cas d'utilisation permettent de représenter un ensemble de cas d'utilisation, d'acteurs et leurs relations.

- **La relation d'inclusion (include) :** elle indique que le cas d'utilisation source contient aussi le comportement décrit dans le cas d'utilisation destinataire .cette relation permet de décomposer des comportements et de définir les comportements partageables entre plusieurs cas d'utilisations
- **La relation d'extension (Extend) :** elle indique que le cas d'utilisation source ajoute son comportement au cas d'utilisation destinataire. l'extension peut être soumise a des conditions.

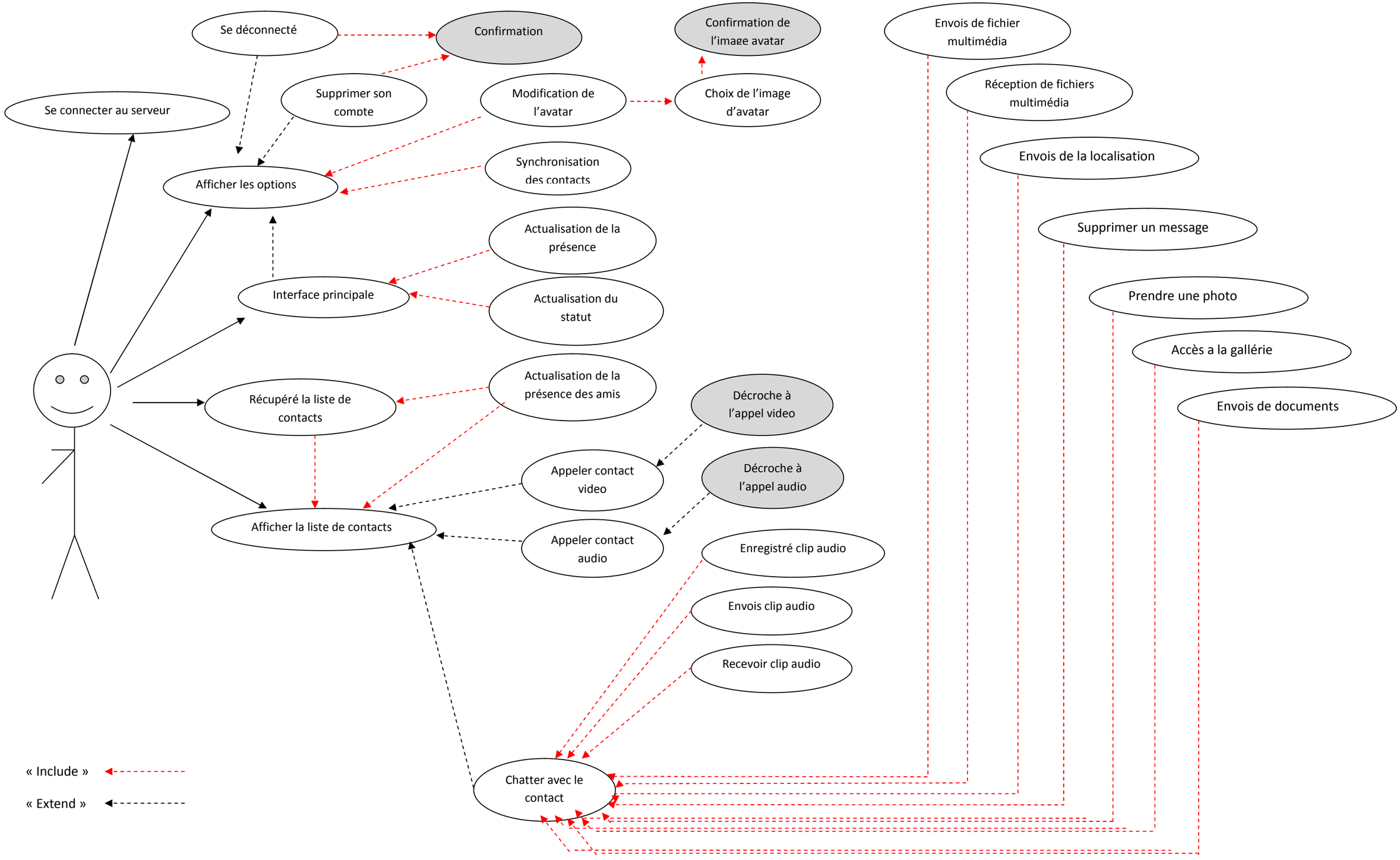


Figure II .10 : Diagramme cas d'utilisations générales.

5-2 Diagrammes de séquences:

Un diagramme de séquences est un diagramme qui représente la séquence de messages entre les objets au cours d'une interaction. Un diagramme de séquences comprends un groupe d'objets, représentés par des lignes de vie, et les messages que ces objets échangent lors de l'interaction. Ils peuvent également représenter les structures de contrôle entre des objets.

Un diagramme de séquences met toujours l'accent sur l'ordre chronologique des messages.

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

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

Les messages : ils sont représentés par des flèches directionnelles. au-dessus de ces flèches figurent un texte nous informons du message envoyer entre les objets.

Ci –dessous je présente quelques diagrammes de séquences :

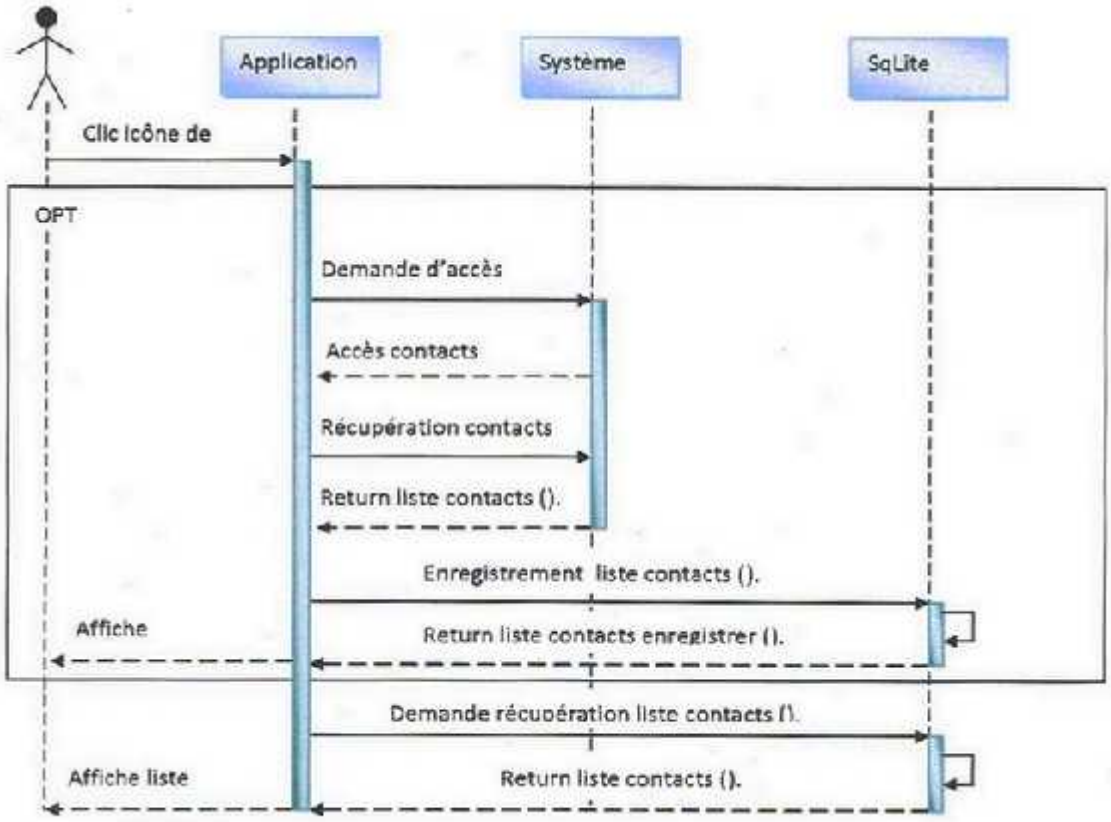


Figure II.11 diagramme de séquences du cas d'utilisation « accéder a la liste de contacts »

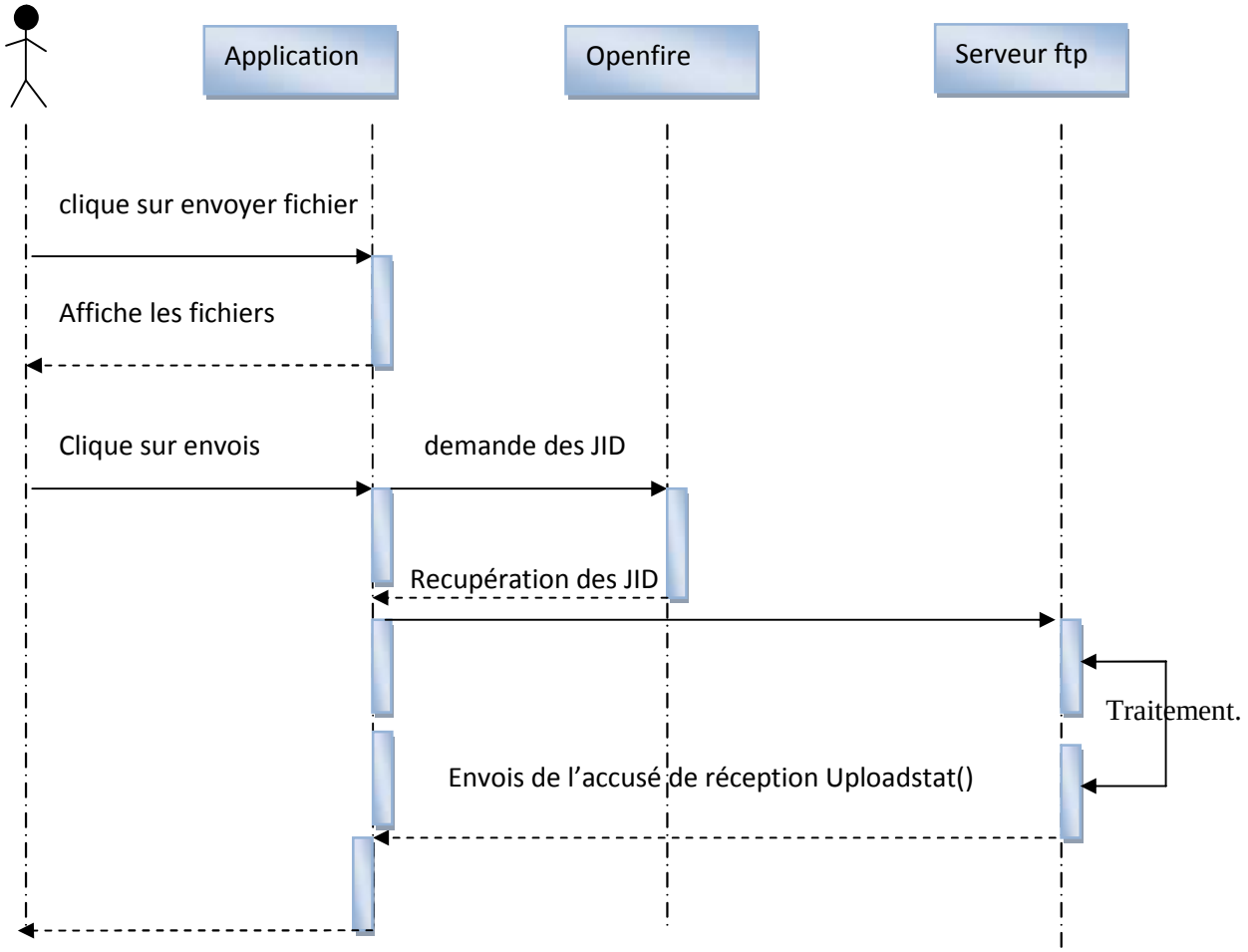


Figure II.12 : diagramme de séquence du cas d'utilisation « envois de fichier »

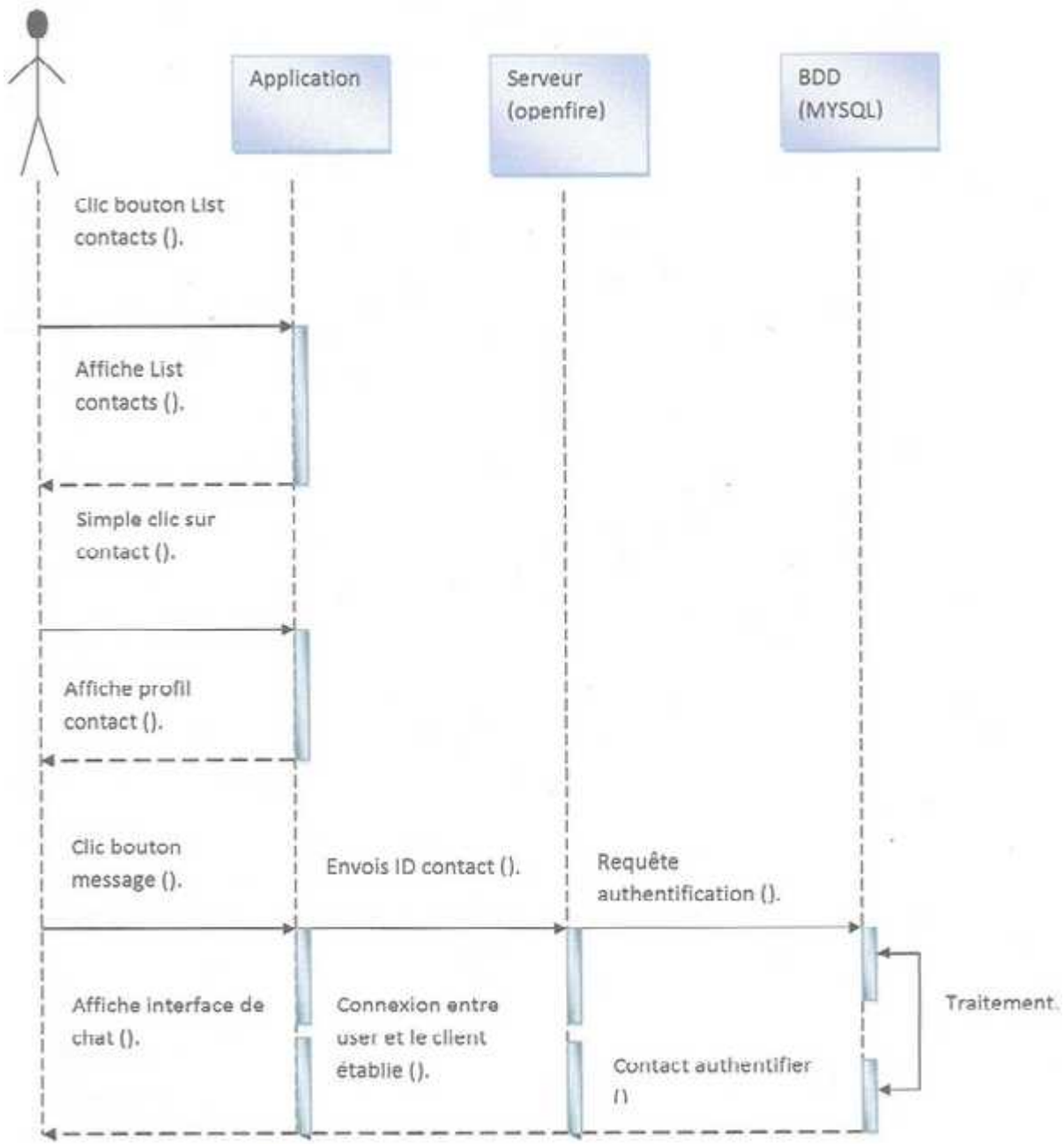


Figure II.13 Diagramme de séquences du cas d'utilisation «chat avec un contact »

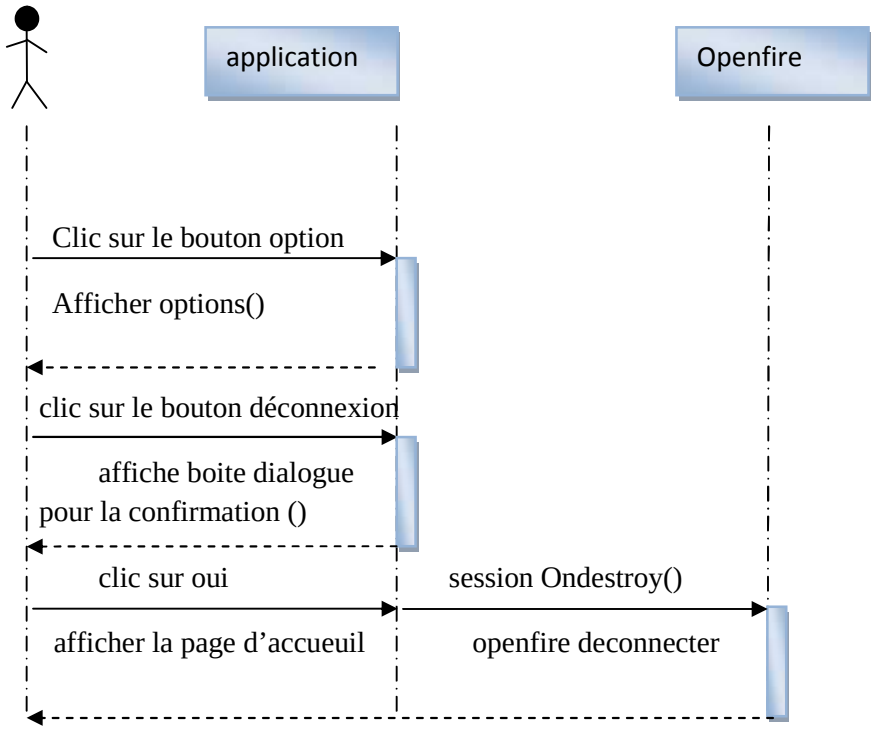
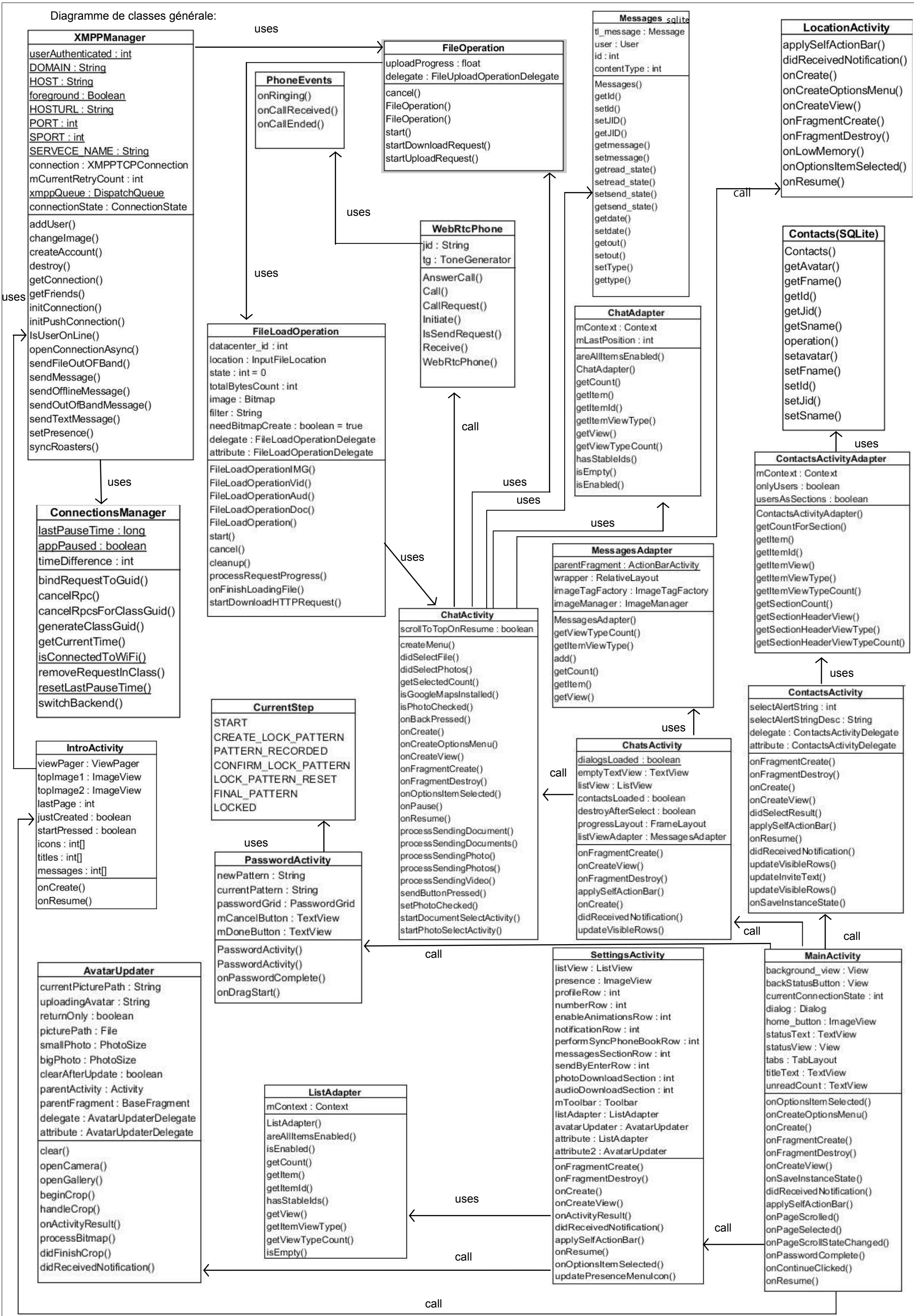


Figure II.14 : Diagramme de séquences du cas d'utilisation « déconnexion »

5-3 Diagramme de classes :

Le diagramme de classe est une partie de la modélisation orienté objet , alors que le diagramme de cas d'utilisation montre un system du point de ue des acteurs , le diagramme de classe en montre la structure interne , il contient principalement des classes reliées par des associations et classe contiens des attributs et des opérations .

Diagramme de classes générale:



5.4 Niveau de données :

- Schéma de la base de données Contacts :
  La base de données contacts n contient plusieurs tables, quelques unes sont représentée ci-dessous

Champ	**Signification**	**Type de donnée**	**Observation**
_userID	ID utilisateur	Int	Clé primaire
JID	Adresse user dans le système	Text	Clé étrangère
Fname	Nom de l'utilisateur	Text	
Sname	Prenom de l'utilisateur	Text	
Avatar	Image du contact	Text	

Figure II.15 : schéma de la base de donnée « contacts »

- Schéma de la base de données message :
  La base de données message ne contient qu'une seule table représentée ci-dessous

Champ	Signification	Type de données	Observation
Mid	Message ID	Int	Clé primaire
JID	Adresse user	Text	
Message	Contenu du message	Text	
Read_status	Message Lu ou non	Int	
Send_status	Message envoyer ou non	Int	
Date	Date du message	Int	
Type	Type du message	int	
TTL	Time to leave	Int	
Data	Contenu du message si fichier	BLOB	

Figure II.16 : schéma de la base de donnée « messages »

- Schéma de la base de données User_setting :
  La base de données user_setting ne contient qu'une seule table représentée ci-dessous

Champ	Signification	Type de données	Observation
JID	Adresse user dans le système	Text	Clé primaire
Name	Nom	Text not nul	
Status	Statu de presence	Int	
Avatar	Image contact	Text	

Figure II.17: schéma de la base de données « User_setting »

6-Architecture de la solution :

Le schéma ci-dessous illustre l'architecture de notre solution avec les serveurs Openfire, serveur de transfert de fichier, MySQL et Apprtc :

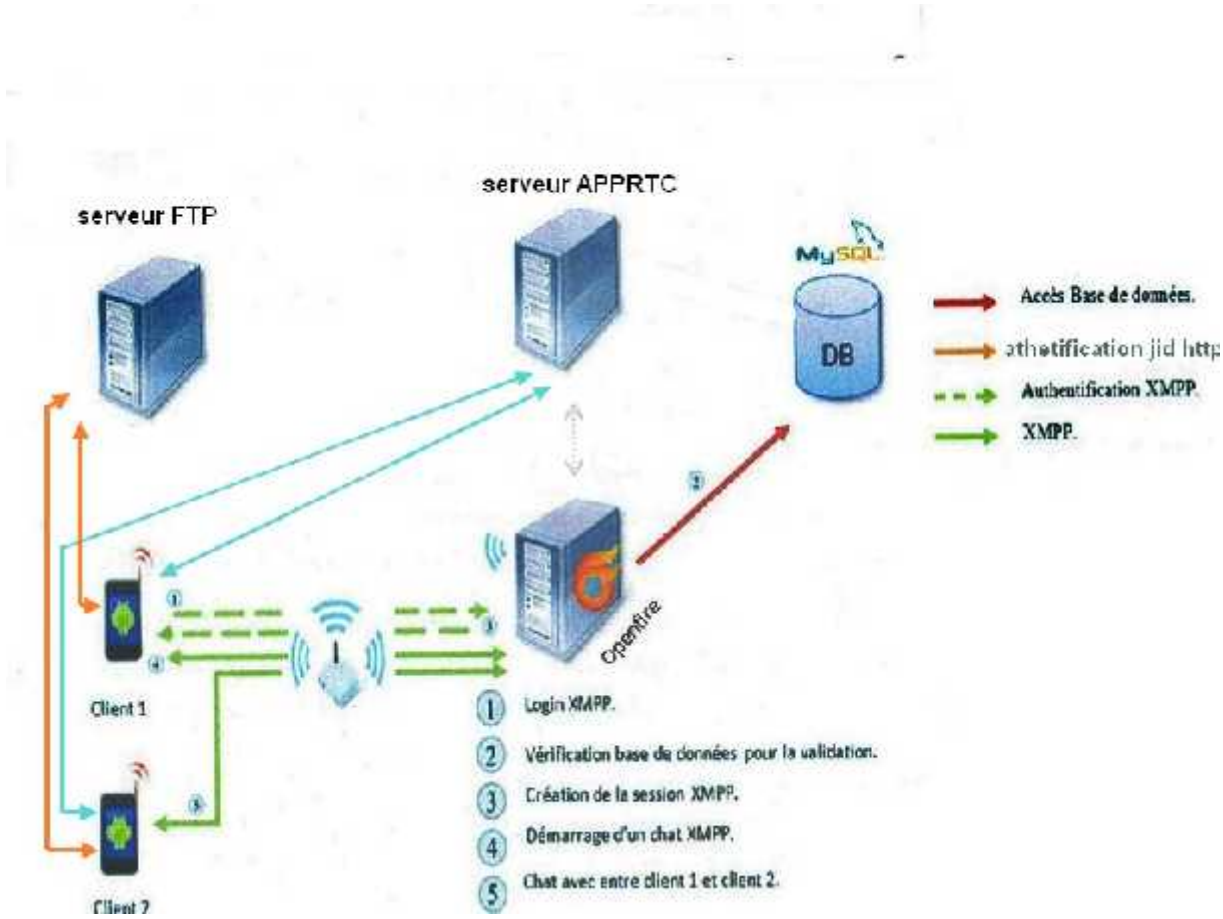


Figure II.18 : Architecture de la solution cas envois de messages

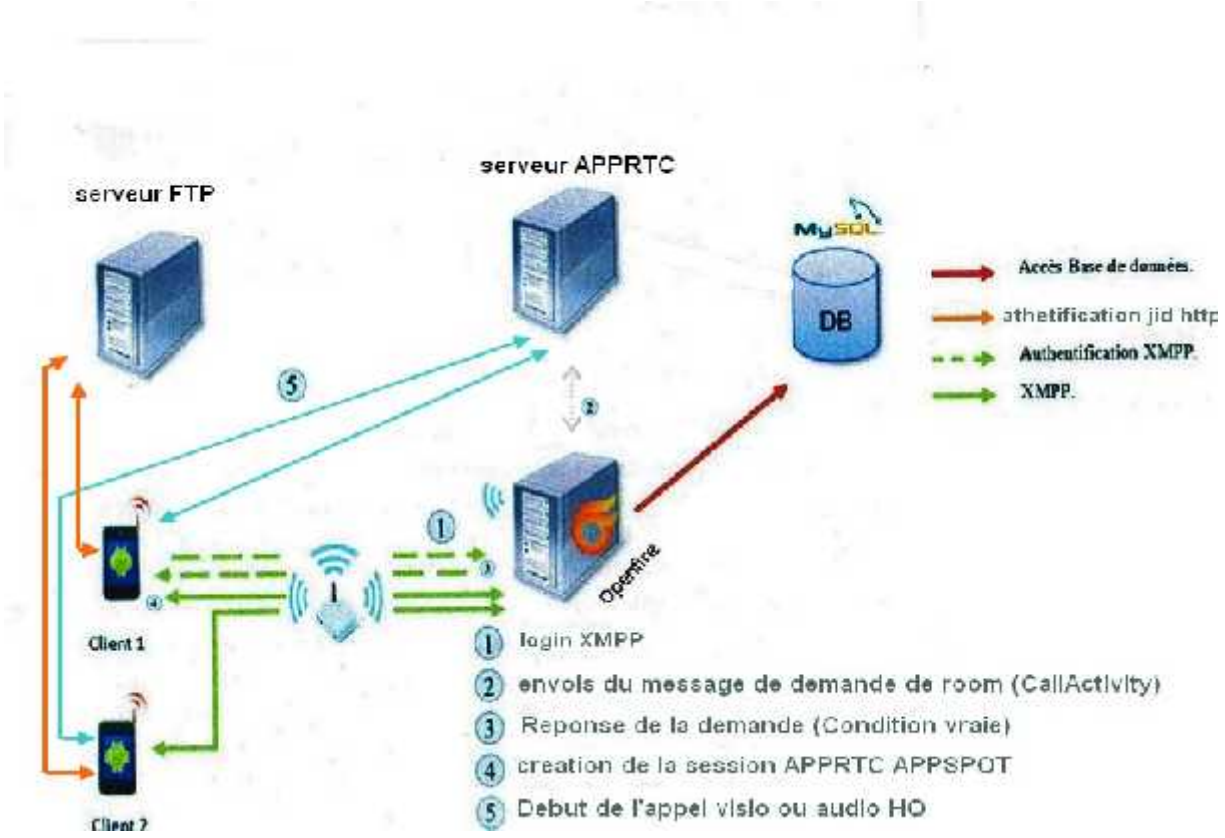


Figure II.18 : Architecture de la solution cas d'appel

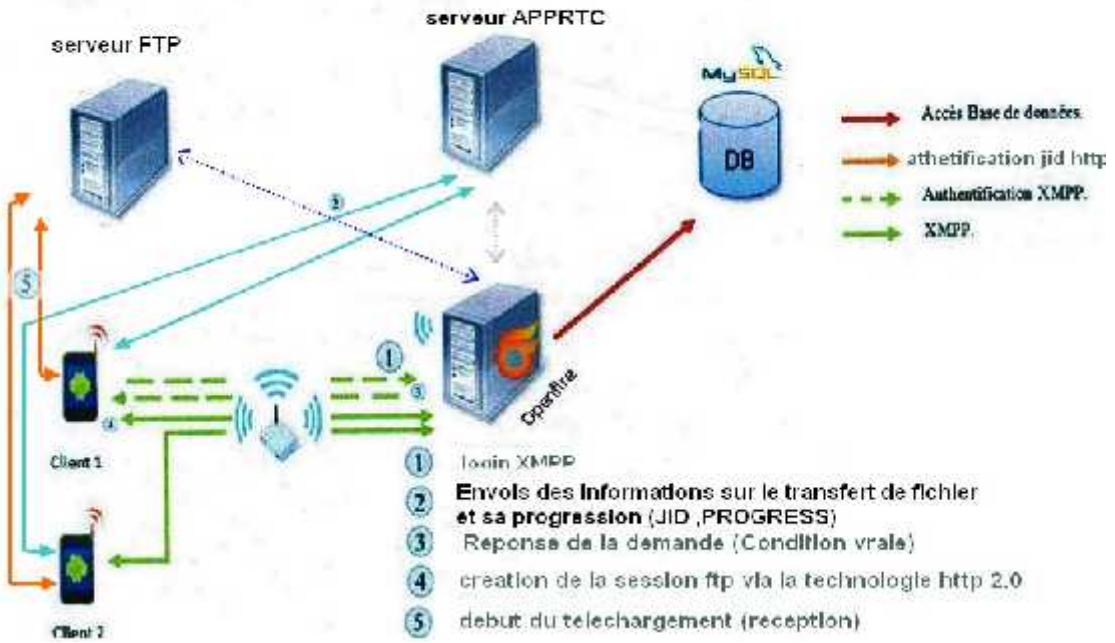


Figure II.19 : Architecture de la solution cas Transfert de fichiers

Les clients (l'application) sont installés sur les appareils des utilisateurs et connectés avec le serveur Openfire. celui-ci est en relation avec le serveur apprtc et le serveur FTP et stock ses données sur le serveur MySQL.

Openfire joue le rôle d'une passerelle, ainsi il communique avec le serveur de transfert de fichier et du serveur apprtc, avec des requêtes en JSON, qui contienne les informations nécessaire a leur utilisation.

Conclusion

J'ai consacré ce chapitre a l'étude, l'analyse et la conception de mon application, dans le chapitre suivant j'aborderais les outils utilisé pour la réalisation de mon application nommée « Text'it ».

Let's Do It ! ☺

Chapitre III : Présentation des outils

1. Introduction :

Le développement d'application est en constant e amélioration , le fait qu'il se démocratise de plus en plus fait émergé des outils dédié a cette optique .

Dans ce chapitre je présenterais l'ensemble des outils utilisé pour la réalisation de mon application ,

2. Les outils utilisés :

2.1 Matériels :

- Un PC I5 avec 8go de ram .
- Un Smartphone de marque BLU Advanced 5.0 avec 1GO de ram .
- Un Smartphone de marque Condor p6 Pro avec 1 Go de ram .
- Point d'accès wifi et internet .

2.2 Logiciel :

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

2.2.1 Android Studio :

Android Studio est l' IDE officiel pour Android. Il fournit les outils les plus rapides pour la création d'applications sur chaque type d'appareil Android .

Android Studio permet principalement d'éditer les fichiers Java et les fichiers de configuration d'une application Android. Il propose entre autres des outils pour gérer le développement d'applications multilingues et permet de visualiser la mise en page des écrans sur des écrans de résolutions variées simultanément Avant Android Studio, de 2009 à 2014, Google propose comme environnement de développement officiel une distribution spécifique de l'environnement Eclipse, contenant notamment le SDK d'Android. Android Studio est annoncé le 15 mai 2013 lors du Google I/O et une version "Early Access Preview" est disponible le jour même. Le 8 décembre 2014, Android Studio passe de version bêta à version stable 1.0. L'environnement devient alors conseillé par Google, et Eclipse est délaissé.

Figure III.1 : Logo Android studio.

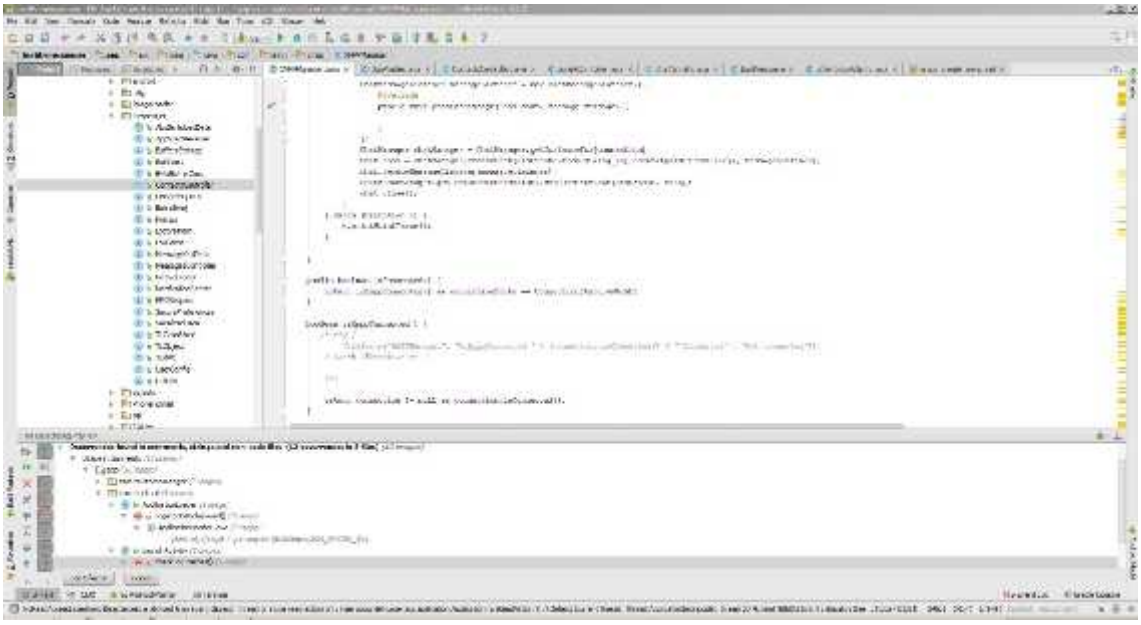


Figure III.2 : Capture d'écran d'Android studio .

2.2.2 Le langage JAVA :

Le langage Java est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy. Il fut présenté officiellement le 23 mai 1995 au SunWorld. La société Sun a été ensuite rachetée en 2009 par la société Oracle qui possède désormais Java.

La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes d'exploitation tels que UNIX, Windows, Mac OS ou Linux, avec peu ou pas du tout de modifications. Pour cela, diverses plateformes associées visent à garantir la portabilité des applications développées en Java.

II.2.3 Le SDK Android :

Le kit de développement (SDK) d'Android est un ensemble complet d'outils de développement. Il inclut un débogueur, des bibliothèques logicielles, un émulateur, de la documentation, des exemples de code et des tutoriaux. Les plateformes de développement prises en charge par ce kit sont les distributions sous Noyau Linux, Mac OS X 10.5.8 ou plus,Windows XP ou version ultérieure. L'IDE officiellement supporté est android studio . Les développeurs peuvent utiliser n'importe quel éditeur de texte pour modifier les fichiers Java et XML, puis utiliser les outils en ligne de commande (Java Development Kit et Gradle Obligatoirement) pour créer, construire et déboguer des applications Android ainsi que contrôler des périphériques Android(pour déclencher un redémarrage, installer un logiciel à distance ou autre).

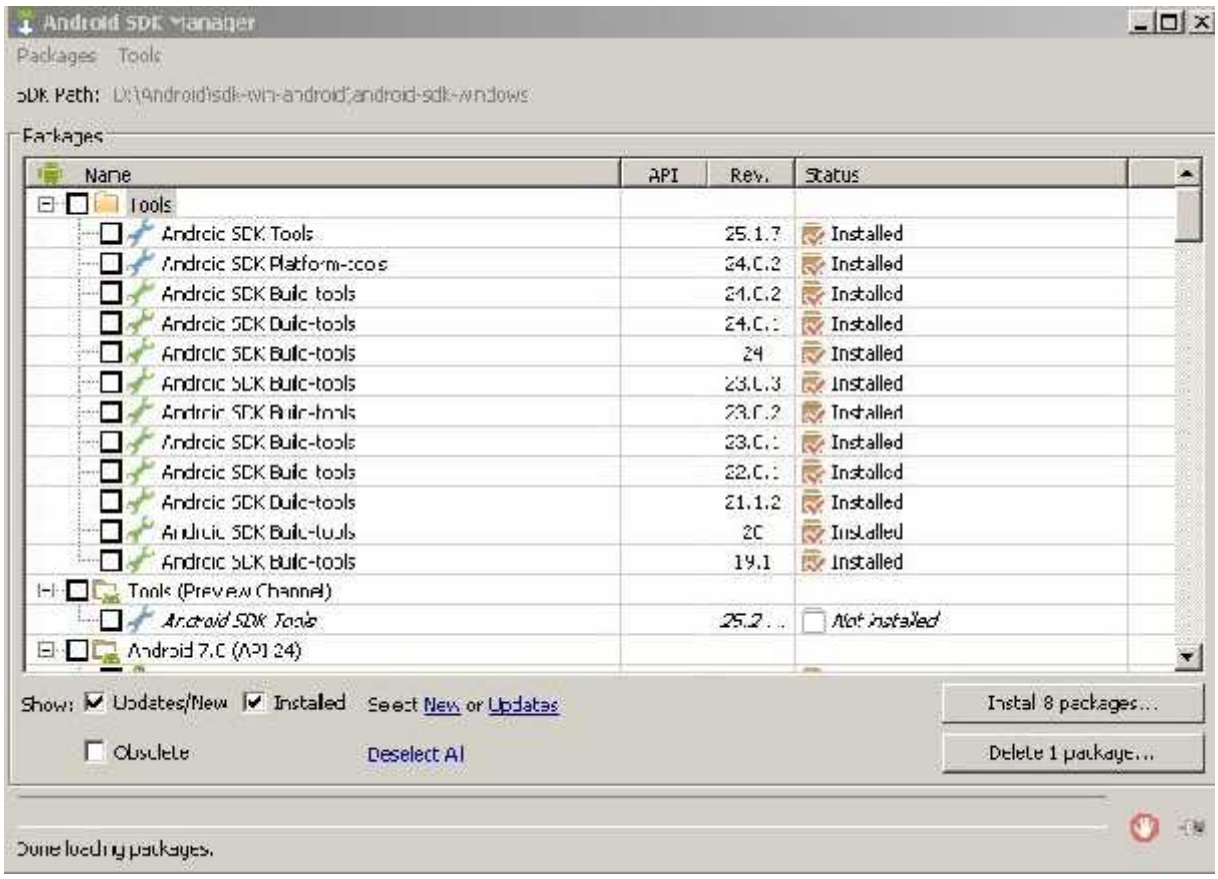


Figure III.3 : SDK Manger , pour la sdk d'android .

2.2.4 Asmack :

Smack est une bibliothèque Open Source XMPP (Jabber) pour client de messagerie instantanée .c'est une pure bibliothèque Java, elle peut être intégré dans vos applications pour créer n'importe quoi ,que ce soit à partir d'un client XMPP complet ou des intégrations XMPP simples telles que l'envoi de messages, de notification et des dispositifs de présence.

2.2.5 Visual Studio :

Visual Studio est un ensemble complet d'outils de développement permettant de générer des applications web ASP.NET, des services web XML, des applications bureautiques des services de Cloud modernes et des applications mobiles. Visual Basic ,Visual C++, Visual C# utilisent tous le même environnement de développement intégré (IDE), qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages. Par ailleurs, ces langages permettent de mieux tirer parti des fonctionnalités du Framework .NET, qui fournit un accès à des technologies clés simplifiant le développement d'applications web ASP et de services web XML grâce à Visual Web Developer.

2.2.5.1 ASP.NET :

ASP.NET est un framework web open source pour créer des applications et des services Web modernes. Avec ASP.NET, vous pouvez rapidement créer des sites web basés sur HTML , CSS et JavaScript , les étendre à des millions d'utilisateurs et facilement ajouter des fonctionnalités plus complexes comme les API Web , les formulaires sur les données ou les communications en temps réel .

Le moteur d'ASP.NET est un filtre branché sur le serveur web Internet Information Services (IIS) par son interface de programmation ISAPI. Le filtre (une bibliothèque de liens dynamiques) est chargé en mémoire à la première utilisation, puis pour chaque demande de page web il lit un modèle de page, puis exécute des instructions qui vont altérer le contenu de la page et renvoie le résultat à IIS, qui le transmet au client.

ASP.NET est distribué avec le framework .NET. Il peut être utilisé avec n'importe quel langage de programmation pour la plateforme .NET. c'est avec l'ASP.NET que j'ai mis en place mon serveur de transfert de fichier.

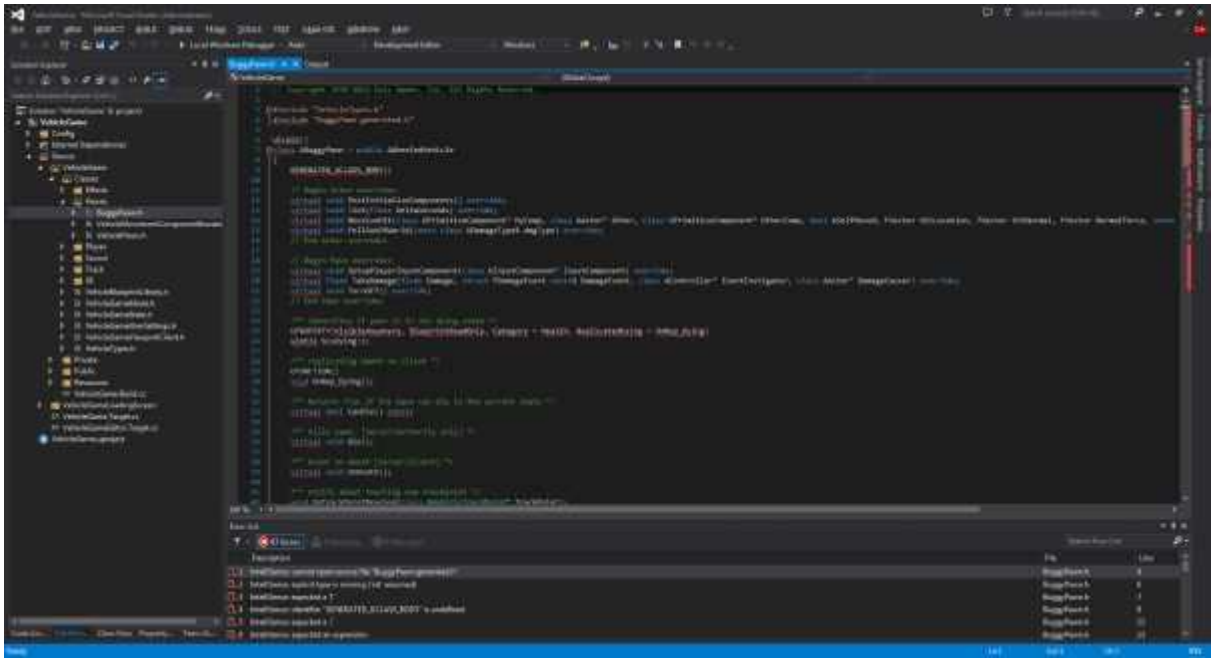


Figure III.4: Capture d'écran Visual Studio 2015

2.2.6 PhpMyadmin SQL server :

phpMyAdmin (PMA) est une application Web de gestion pour les systèmes de gestion de base de données MySQL réalisée en PHP et distribuée sous licence GNU GPL.

MySQL est le système de gestion de base de données SQL Open Source le plus populaire. Il est développé, distribué, et soutenu par Oracle Corporation. Utilisé comme base de donnée dans mon serveur Openfire.

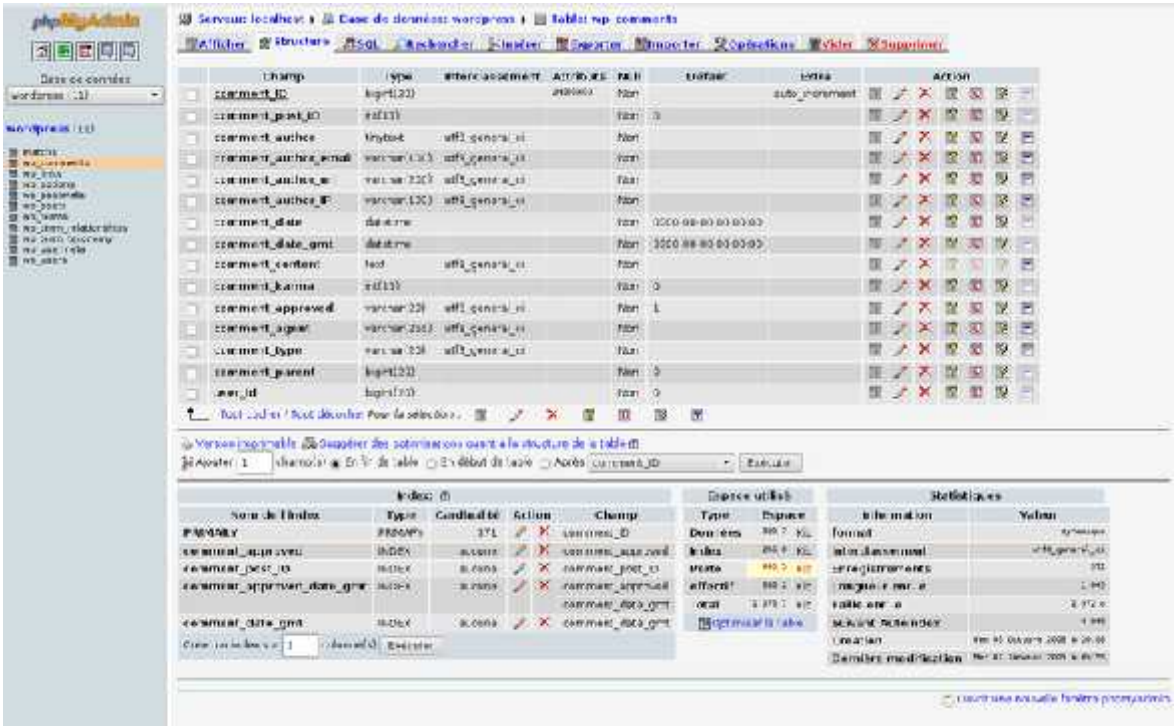


Figure III.5: capture d’écran PhpMyadmin

2.2.7 AppRTC :

Apprtc est une application de démonstration de WebRTC Open Source hébergé sur App Engine (GAE pour google App Engine). Développé par Google et Mozilla, elle permet aux deux navigateurs de «parler » les uns aux autres en utilisant l'API WebRTC.

Lorsque vous allez sur le site, une nouvelle salle de conférence vidéo est automatiquement créé pour vous et vous pouvez partager l'URL fournie avec quelqu'un d'autre et donc vous connecter.

Apprtc donc, seras l’appui de mon application pour permettre les appels video et audio entre les utilisateurs .

Figure III.6 : capture d’écran d’apprtc

2.2.8 L'API GreenDAO :

greenDAO est un ORM exploitant des bases de données SQLite. Il permet de mapper nos objets Java en base de données.

Un mapping objet-relationnel (en anglais object-relational mapping ou ORM) est une technique de programmation informatique qui crée l'illusion d'une base de données orientée objet à partir d'une base de données relationnelle en définissant des correspondances entre cette base de données et les objets du langage utilisé. On pourrait le désigner par « correspondance entre monde objet et monde relationnel ».

Figure III.7 : Schéma du fonctionnement de greenDAO.

J'ai utilisé cet outils pour génerer ma base de données ainsi que toutes mes tables suites a d'infructueuses tentative en écriture manuel . l'ensemble des fichiers généré est ensuite inclus dans un sous package de mon application sous le nom de SQLite.

Remarque : Le pattern DAO (Data Access Object) permet de faire le lien entre la couche métier et la couche persistante, ceci afin de centraliser les mécanismes de mapping entre notre système de stockage et nos objets Java. Il permet aussi de prévenir un changement éventuel de système de stockage de données (de PostgreSQL vers Oracle par exemple). La couche persistante correspond, en fait, à notre système de stockage et la couche métier correspond à nos objets Java, mapper sur notre base. Le pattern DAO consiste à ajouter un ensemble d'objets dont le rôle sera d'aller : lire, écrire, supprimé et modifier dans notre système de stockage. Cet ensemble d'objet s'appelle la couche DAO.

2.2.9 Google Maps API :

L'API Google Maps est l'une des applications de cartographie les plus utilisées au monde. C'est une application de service de géo localisation gratuite en ligne. Il s'agit d'un géo portail lancé il y a quelques années aux Etats-Unis puis à l'Europe.

Elle offre une vue de carte sur quatre plans à savoir un plan classique, un plan en image satellite, un plan mixte et un plan relief de la région.

Pour intégrer ces cartes interactives Google Maps à son propre application et bénéficier des données associées, l'utilisateur doit disposer d'une clé (Google Map API Key) propre à son domaine d'utilisation

2.2.10 Openfire server :

Le serveur de messagerie instantanée , Openfire en version 4.0.1 . définit dans le chapitre I .sont installation et configuration seras expliquer plus tard .

2.2.11 l'API Nexmo :

Créée en 2011, Nexmo propose des solutions permettant aux applications et aux entreprises de téléphoner, transmettre et recevoir des SMS de façon extrêmement simple afin d'améliorer l'expérience clients, et ce, à n'importe quel endroit où ils se trouvent, grâce à une large gamme d'APIs.

3. Conclusion :

Dans ce chapitre j'ai présenté les outils utilisé pour la réalisation de mon application, dans le chapitre suivant je serais en phase d'implémentation et je vous présenterais la concrétisation de la conception.

Chapitre IV : Implémentation de l’application

I. Introduction :

Dans ce chapitre ,je présenterais les étapes d'installation et de configuration des différents outils utilisé ainsi que les composantes applicatives réalisées.

Les différentes implémentations, installations et les configurations ont été réalisé sur une plateforme Windows 7 .

II. Installation Des serveurs :

II.1 Configuration du serveur base de données et la base de données :

Le déploiement du serveur Openfire nécessite une base de données de type MySQL, pour cela j'ai choisis un serveur déjà présent dans mon ordinateur qui est WAMP, qui dispose de PHPMYADMIN serveur MySQL et qui feras largement le travail demandé.

Pour déployer la base de données d'Openfire , on devras dans un premier temps télécharger un serveur MySQL , wamp dans mon cas mais on pourrais utilisé AppServ ou autre serveur .

Je lance wamp avec ces services , puis je me connecte a l'interface graphique de PhpMyadmin et je crée une nouvelle base de données sous le nom « openfire » .

Openfire a toujours inclus dans ces fichiers ressources le schéma de sa base de données , trop compliqué a crée selon eux … d'un coté sa feras gagner beaucoup de temps . le fichier .sql existe dans le répertoire d'installation d'open fire « /ressources/databases/database.sql ». on met un import SQL schéma et on sélectionne le fichier cité au dessus .PhpMyadmin exécute et génèrent toutes les tables et informations nécessaire au bon fonctionnement d'Openfire.

Une fois cette étape franchise, on passe a l'étape de configuration du serveur Openfire .

II.2 Configuration d'Openfire :

II.2.1 Installation d'Openfire :

Openfire est open source, sous Windows il est préférable de télécharger le fichier exécutable et de l'installer , car sinon on disposeras de quelque soucis avec les variable d'environnement Windows

J'ai téléchargé Openfire depuis le site officiel en version 4.0, figurant en annexe . installé sur ma machine , puis on lance le Openfire Server .

II.2.2 Configuration d'Openfire :

Openfire offre plusieurs configurations et options selon les besoins d'utilisation, qui sera sélectionné pendant sa configuration sur son interface Web, une fois Openfire lancé une fenêtre web est lancé , sur le port console 9090.

- Dans la première interface, on sélectionne la langue « Français » dans mon cas.
- Dans l'interface qui suit-on choisit le nom du domaine qui est équivalent au nom du serveur Openfire par défaut qui est le nom de l'ordinateur (HELLBOY dans mon cas)
- La prochaine étape est donc de choisir le type de connexion a la BDD , je sélectionne « Connexion a une base de données standard »

Remarque : Openfire dispose de sa propre BDD embarqué, mais le but étant de maximisé les performances on utilisera un service étranger.

- A l'interface de paramétrage de la base de données on remarque les champs de texte qu'on remplira dans le cas ou ils serons vide par :

  - Pilote de la BDD : « Mysql ».
  - Classe du pilote JDBC : « com.mysql.jdbc.Driver »
  - URL de la BDDD : « JDBC :mysql://HELLBOY :3306/openfire

Remarque importante : il faut que je précise que HELLBOY est le Host-name , equivalent a mon localhost, lien de PhpMyadmin , et 3306 est le port par defaut de tout serveur MySQL et openfire est le nom de la base de données qu'on a configurer plus haut , il faudra bien mettre les bonnes informations , ou alors un crash d'openfire est alors inévitable , la reconfiguration se feras alors depuis le Conf.xml sans interface graphique, sans oublier de vider son cache.

  - Et enfin le nom d'utilisateur et le mot de passe pour le serveur BDD.

- L'interface qui viens après ,permet de choisisr l'utilisateur et le groupe systéme a utiliser avec Openfire .
  - Par defaut (enregistre les utilisateur dans la BDD interne)
  - Serveur LDAP (integrer avec un serveur LDAP comme active directory ou OpenLDAP en utilisant le protocole LDAP, les utilisateurs et le groupe doivent etre presents dans le serveur LDAP et visibles en lecture seule – ReadOnly)
  - ClearspaceIntegration (pas d'enregistrement solide) .

J'ai choisis par défaut car je n'est pas besoin d'un serveur LDAP (annuaire) , car les utilisateurs s'inscrivent directement dans la bdd.

- Dans l'interface suivante ,j'ai choisis le mot de passe et le nom d'utilisateur pour Openfire. Qui est « admin » .
- C'est bon , on arrive a la fin et Openfire nous affiche que l'installation est fini , et nous demande de se connecté a la console d'administration avec admin et le mot de passe choisis précédemment .

II.3. serveur de fichier :

Le serveur d'échange de fichier ici , n'est qu'un serveur http qui gère la réception et l'envois , il enregistre le fichier dans un dossier qu'il crée avec le nom d'utilisateur (JID) de provenance en premier niveau , puis un e arborescence au deuxième nommée par le nom d'utilisateur du récepteur de ce fichier .

Rediger en ASP.NET pour ne pas géné le fonctionnement des autres serveurs , il est autonome il suffira de le lancer depuis Visual Basic 2015.

II.4. l'application apprtc:

Apprtc est rédigé en Javascript et dispose d'un moteur de gestion en python , il n'est pas nécessaire de présenté python car il n'interagis pas avec l'application , il seras déployer depuis Google app engine(GAE) sous linux (Pour la signalisation , Coturn ice server) mais on est libre d'utilisé la version héberger sur internet .

Dans ce cas(hébergement local), une application joueras le rôle d'un serveur qui est assez rare en informatique , le serveur WebSocket (WSS) est pertinent vue qu'il garantie la connectivité pair a pair , a ce jour son déploiement en local est juste réservé aux développeurs sélectionnées par Google elle-même . il ne s'agit pas de vente , mais de privilèges .

Le serveur websocket est en phase de test , et il n'as présenté aucun bug depuis 2014 , une version open source libre du WSS devrais voir le jour d'ici la fin de l'année .il existe d'autres alternatives pour son lancement local , comme déploiement d'un serveur ICE dans sa machine , et implémenté nous même le code pour la communication entre les parties .

L'application est disponible sur https://apprtc.appspot.com/ pour un usage libre.

III. Fonctionnement de l'application :

III.1 Présentation générale de l'application :

L'application que j'ai développée a une activité principale qui est « activitylaunch .java » qui s'exécute au lancement , renvois vers « introAvtivity.java » dans le cas du premier lancement . disposant de 3 fils , sous forme de fragments , elle affiche donc « Update.java » , « chatadapter.java » et « contactadapter.java » qui permet d'afficher les contact .

Voici une partie du manifeste qui demande les permissions :

```
<uses-permission android:name="android.permission.DISABLE_KEYGUARD" />
<uses-permission android:name="com.android.alarm.permission.SET_ALARM" />

<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.RECEIVE_SMS" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.WRITE_CONTACTS" />
<uses-permission android:name="android.permission.MANAGE_ACCOUNTS" />
<uses-permission android:name="android.permission.READ_PROFILE" />
<uses-permission android:name="android.permission.WRITE_SYNC_SETTINGS" />
<uses-permission android:name="android.permission.READ_SYNC_SETTINGS" />
<uses-permission android:name="android.permission.AUTHENTICATE_ACCOUNTS" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<!-- map -->
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
<uses-permission android:name="com.textit.messenger.permission.MAPS_RECEIVE" />
<uses-permission android:name="com.textit.messenger.permission.C2D_MESSAGE" />
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<!-- Needed to route the audio to the bluetooth headset if available -->
<uses-permission android:name="android.permission.BLUETOOTH" />
```

Figure IV.1 : Les permissions .

III.2 Explications de quelque permissions :

La permission est une autorisation d'accès aux ressources de l'application.

- "android.permission.CAMERA" = permet l'utilisation de la camera a l'application.
- **"android.permission.READ_CONTACTS"= permet la lecture des contacts**
- **"android.permission.INTERNET"= permet l'accès au réseau**
- **"android.permission.VIBRATE"= utiliser le vibreur**
- **"android.permission.WAKE_LOCK" = permettre d'éteindre l'écran lors d'un appel vocal.**

III-3 les fonctionnalités de mon application :

- Fonctions 1 « Formulaire d'inscription » :
  Une fois l'application installée, lors du premier démarrage le formulaire ci-dessous apparait. ce formulaire sert a l'inscription de l'utilisateur avec les informations requises sur la base de données centralisées et celle du téléphone lui-même.

Figure IV.2Inscription (entrer le numéro)

Figure IV.3 Inscription (entrer le nom et prénom)

- Fonction 2 : (connexion) :
  Juste après avoir validé son nom et prénom dans le formulaire d'inscription, une requête est adressé au serveur Openfire, une fois l'utilisateur crée et connecté une requête de synchronisation des contactes est envoyer au serveur.
- Fonction 3 :
  Synchronisation des contacts.

- Fonction 4 : Affichage de la liste de contacts :
  Après la requête de synchronisation des contacts , l'application les affiche

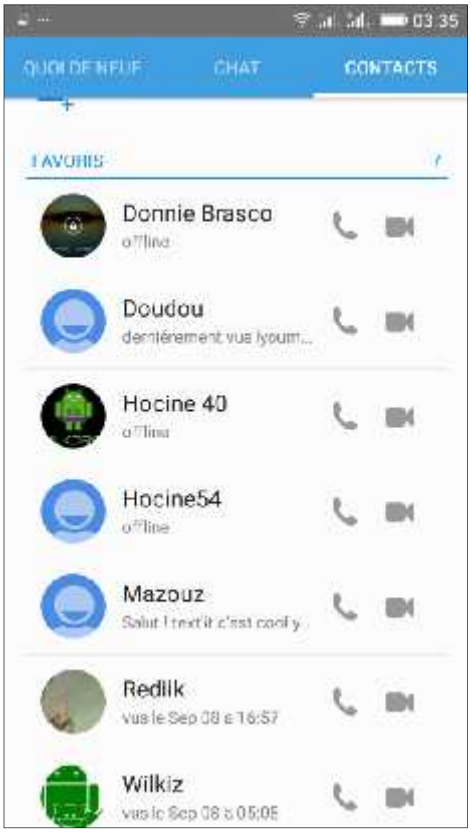


Figure IV.4 : La liste des contacts.

- Fonction 5 : chatté avec un contact
  clic sur le contact souhaité, la fenêtre de conversation s'ouvre. alors on peut envoyer les messages et les émoticônes de la manière la plus intuitive qui existe.
- Fonction6 : envois d'un clip audio
  de plus de pouvoir envoyer des messages, mon application dispose d'une fonctionnalité native d'Android c'est de permettre l'enregistré dans un premier temps un message vocal et de l'envoyer par la suite a notre correspondant qui pourras le lire depuis sa fenêtre de discussion.
- Fonction7 : envois de fichiers multimédia
  permet l'échange de photos, vidéos , documents et autre entre deux parties
- Fonction8 : Partage de localisation
  l'utilisateur lui-même peut choisir d'envoyer sa localisation a son correspondant avec juste deux clics, l'application renvois alors a une interface Google Maps

Figure IV.5 : chat et envois de fichiers multimédia

Figure IV.6 : Partage de localisation

- Fonction 9 : Appel audio
  la fonctionnalité primaire de mon application réside dans le fait qu'elle est capable de faire passer des appels entre les deux parties correspondante a deux utilisateurs.
- Fonction 10 : Appel vidéo
  la deuxième fonctionnalité primaire est de passer des appels, mais cette fois ci ce serons des appels vidéo, l'utilisateur donc est capable de passer des appels avec son correspondant et profité pleinement d'une qualité de son et d'image très bonne.

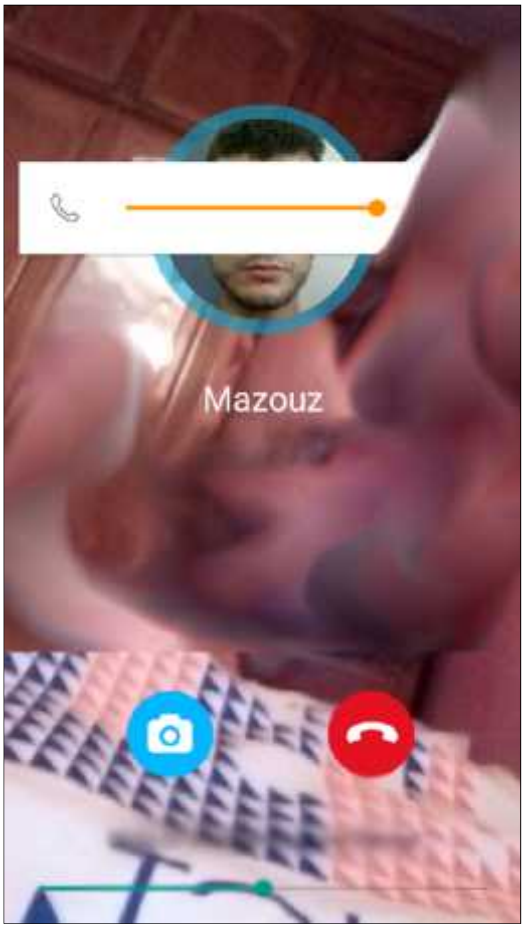


Figure IV.6 :appel video (Sonnerie)

Figure IV.7 : appel vidéo établie .

- Fonction 11 : supprimer un message
  Suppression d'un message unique ou multiple depuis la discussion.
- Fonction 12 : Supprimer les discussions
  permet de supprimé une discussion précise , ou bien l'ensemble des discussions existantes .

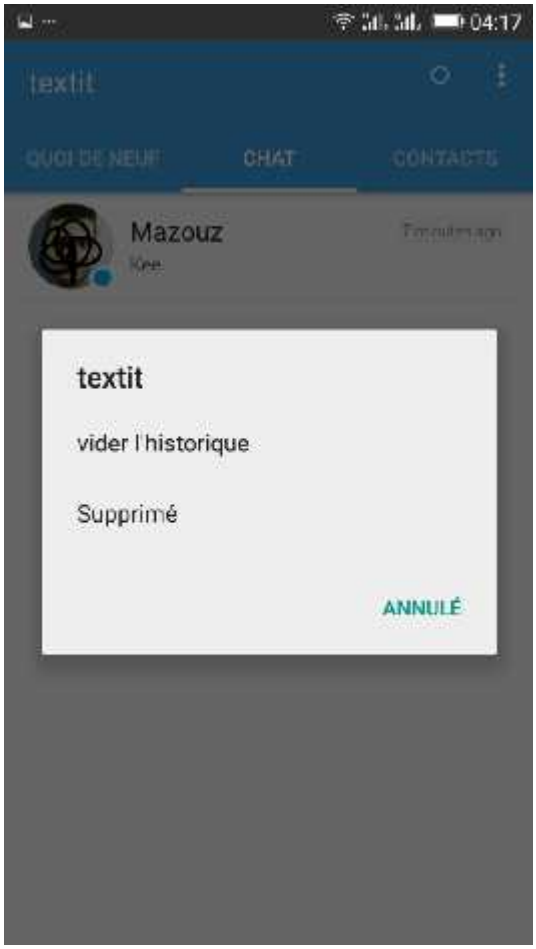


Figure IV.8 : Suppression des messages

- Fonctionnalité 13 : verrouiller l'application avec un code
  mon application respecte la vie privé des gens, plus que les gens eux même elle intègre alors, un verrouillage par schéma palette qui devrais satisfaire le besoin de confidentialité. Elle est accessible par le bais de l'Action Bar, en cliquant sur le nom de l'application.

Figure IV.9 : Verrouillage de l'application par « pattern lock »

- Fonctionnalité 13 : se déconnecté
  se déconnecté du compte .

- Fonctionnalité 14 : Supprimé son compte
  Suppression du compte depuis le serveur, une boite de confirmation apparait une fois cette déconnexion est confirmée le compte n'existe plus dans l'application.

Figure IV.10 Déconnexion.

Conclusion :

J'ai présenté dans ce chapitre les configurations des serveurs utilisé, ainsi une vue globale de « Text'it » La partie mise en œuvre traduit les besoins fonctionnels et techniques déjà définies par l'implémentation des différentes interfaces.

Conclusion générale et perspectives

Ce projet m'as donné la chance de m'intégrer dans le domaine du développement qui m'as toujours passionné et m'as fait confronté a des problèmes qui m'ont fait gagné de l'expérience dans l'optique de l'informatique instantanée, des réseaux, d'internet et technologie web, des systèmes communicant et aux domaines des télécommunications , une meilleure maitrise du langage Java & C , mais surtout il m'as fait gagné en confiance car avant tout c'étais un challenge que j'ai relevé pour moi même , il m'as aussi fait comprendre le fonctionnement des applications comme viber , Skype ou Dolby axon , et m'as fait réalisé a quel point ses applications sont complexes a réalisé nécessitant toute une équipe de développeurs , j'aurais souhaité que ce projet aurais développé aussi mon relationnelle dans une équipe .

Enfin, mon travail ainsi présenté n'est qu'un prototype sur le quel j'espère continuer a concrétisé mes idées ,et l'amélioré en quelque chose de plus beau , de plus pratique mais surtout de quelque chose d'origine algérienne .

Je souhaiterais ajouter ces fonctionnalités dans un avenir proche :

- Historique d'appel.
- Ajouter un Module de rédaction et de partage documentaire.
- Implémenté les API de google+ ,Twitter ou Link in .
- Streaming de flux (Partager la music sans envoyer le fichier, Joindre l'utile a l'agréable)
- Suite au fait que les statu sont disponibles, je souhaiterais lui implémenté un système de réseau social.
- Et bien sur tout cela n'as pas de sens si mon application ne s'étendras pas sur internet !

http://www.xmpp.com

http://www.wikipédia.fr/webrtc

http://www.developper.google.com/Webrtc

http://www.webrtcooking.org

http://www.webrtcbook.com

Salvatore Loreto & Simon Pietro Romano « Real-time communication with Webrtc » ,editions O'REILLY , 2014.

Golam SARWAR ,thèse " Protocoles de transport pour la diffusion video temps-reél sur lien sans fil" Université de Toulouse , 9 juillet 2014.

Peter saint-Andre ,Kevin Smith & Remko tronçon "XMPP, the definitive guide" edition O'REILLY ,2009

Michaels both & Lucian di loco "Android inside the web technologies" Pocket ,2014.

https://www.igniterealtime.org/

Alain Menu , android How to edition 2.1 , septembre 2013

http://www.igniterealtime.org/downloads/

http://download.igniterealtime.org/smack/docs/latest/documentation/extensions/

Laurent Debrauwer et Fien Van Der Heyde , « UML 2 Modélisation des objets » , ENI 2006.

https://ubuntuforums.org/archive/index.php/t-2097926.html

http://stackoverflow.com/questions/tagged/xmpp

http://stackoverflow.com/questions/tagged/webrtc

https://www.visualstudio.com/docs/vs/overview

http://www.xmpp.org/extensions/xep-0080.html

https://xmpp.org/registrar/namespaces.html

http://www.igniterealtime.org/project/Asmack

http://www.ietf.org/rfc/

https://files.phpmyadmin.net/phpMyAdmin/4.6.4/phpMyAdmin-4.6.4-all-languages.zip

Gabriel Svennerberg , Beginning Google Maps API 3 ,edition Apress, 2010.

Raj Amal W. Learning Android Google Maps ,edition packt ,2016.