République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de La Recherche Scientifique

Université Mouloud MAMMERI de Tizi-Ouzou

Faculté de Génie électrique et Informatique

Département Informatique

Projet de fin d'étude pour l'obtention du diplôme de Master

Option : Réseaux, Mobilité et Systèmes Embarqués

Réalisation d'une application Android de navigation par GPS en utilisant des cartes OSM

Réalisé par :

Mr. BELLIL Amine

Mr. KOUMAD Salim

Encadrés par :

Mr. DAOUI Mehammed

Devant le jury composé de :

Mr. LALAM Mustapha	**Président**
Mme AOUDJIT Rachida	**Membre**
Mme BELKADI Malika	**Membre**

Remerciements

D'abord nous remercions le bon dieu de nous avoir donné santé, courage, volonté et foi pour réaliser ce travail.

Nous tenons à exprimer notre profonde gratitude à notre encadreur Mr DAOUI Mehammed, pour tout ce qu'il nous a apporté comme aide, connaissances et conseils pour l'accomplissement de ce travail.

Nous remercions vivement les membres du jury pour avoir accepté d'évaluer notre 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.

Nous aimerions aussi remercier nos familles, nos amis, ainsi que tous ceux qui ont contribués de prêt ou de loin à la réalisation de ce travail.

Résumé

La localisation et le guidage d'un individu sur une carte en offline nécessite de passer par deux principales étapes. Dans un premier temps, un effort est porté sur l'aspect « Représentation des données d'une carte OSM ». Pour cela il est nécessaire de chercher à représenter graphiquement ces cartes qui sont sous format de données structurés. Dans un deuxième temps il sera question de chercher la meilleure solution pour la recherche des itinéraires. Ceci en dégageant une représentation du graphe des routes de la carte sous forme d'une base de données, ensuite en appliquant un algorithme de recherche d'un chemin entre deux points de départ et d'arrivé. Enfin il sera question d'implémenter les deux premières étapes en ajoutant la localisation de l'utilisateur en temps réel grâce à la technologie du GPS.

Mots clés : GPS, Android, Géolocalisation, Open Street Map, Recherche Itinéraire.

Table des matières

Liste des figures :

Acronymes

API	Application Programming Interface
BSSID	Basic Service Set Identification
DoD	US Department of Defense
GPS	Global Positioning System
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
OHA	Open Handset Alliance
OSM	Open Street Map
PDA	Personal Digital Assistant
PNG	Portable Network Graphics
RFID	Radio Frequency Identification
SIG	System d'Information Géographique
SMS	Short Message Service
UMTS	Universal Mobile Telecommunications System
XML	eXtensible Markup Language

Introduction Générale

La nature de l'être humain à toujours chercher à automatiser les tâches quotidiennes, a mené à une ère qui est aujourd'hui dominé par les systèmes embarqués. Ils ont su se rendre indispensable dans la vie de tous les jours et sont présents pratiquement partout autour de nous, pour se réveiller on utilise un réveil, pour appeler ses amis on utilise un téléphone, pour se repérer on utilise un récepteur GPS, pour prendre des photos on utilise un Smartphone, la liste reste longue, ils sont omniprésents et continuerons à l'être encore plus à l'avenir.

Les récent progrès technologiques ont permis la réalisation d'un des plus vieux rêves de l'homme. L'orientation, une des plus anciennes préoccupations de l'être humain, depuis la nuit des temps des techniques diverses on été inventés pour se repérer. Signaux de fumée, positionnement grâce aux astres, invention de la boussole, … Ce besoin de se situer est vite devenu vital. Grâce aux avancées technologiques en terme de géolocalisation notamment avec le lancement des satellites GPS, et à l'avènement des équipements mobiles, le rêve est devenu réalité, la localisation géographique d'un individu est rendu possible avec un simple appareil mobile entre les mains. En effet il suffit d'un Smartphone pour se localiser à n'import quel endroit du monde, mais l'inconvénient majeur est que la plus par des applications existantes nécessitent une connexion internet pour fonctionner, ce qui limite la mobilité de l'utilisateur.

L'objectif de notre travail est de réaliser une application de géolocalisation par GPS fonctionnant dans un environnement dépourvu d'internet.

Notre travail est structuré en quatre chapitres. Le premier chapitre décrit les généralités en rapport avec notre projet. Le deuxième chapitre est consacré à la géolocalisation et les cartes collaboratives. La conception d'une application de navigation GPS est illustré dans le chapitre trois. La réalisation de cette application est enfin décrite dans le chapitre quatre.

CHAPITRE 1
Généralités

I.1) Introduction :

De nos jours on entend par localisation géographique, le positionnement sur un système de référence une personne ou un objet, par exemple un véhicule. Ce système de référence n'est ni plus ni moins qu'un système cartographique : une carte routière, ou un plan de ville.

Actuellement, toute automobile, tout bateau, tout avion, etc... , qui voudrait se repérer géographiquement, utilisera, entre autres le G.P.S. (Système Global de Positionnement) qui est le système le plus répandu à l'heure actuelle.

Les appareils utilisés pour la localisation doivent être muni d'une puce GPS qui permet de récupérer les données satellites et de les traiter afin de positionner l'individu ou l'objet sur une carte donnée.

I.2) Navigation GPS :

Certains appareils sont conçus exclusivement pour la localisation, d'autres sont multifonctions et incluent une puce GPS, ils peuvent aussi être incrustés dans une voiture pour un suivi ou faire parti du tableau de bord d'un avion par exemple.

Figure 1.1 : Dispositif Garmin EDGE 800.

Le dispositif de la figure 1.1 est spécialement conçu pour la navigation GPS, une application y est installée avec toutes les fonctionnalités nécessaires à une navigation sure et confortable, sauvegarde d'itinéraires, utilisation simplifiée grâce à un écran tactile,

disponibilité des plans des villes et des routes... L'inconvénient majeur de ce type d'appareil est le rapport fonctionnalité/prix, en effet ils n'offrent finalement qu'une seule fonctionnalité pour un prix avoisinant celui d'un ordinateur.

Figure 1.1. : TRACEUR GPS K100 POUR VOITURE ET MOTO.

Le traceur de la figure 1.2 est un outil de surveillance et de géo-localisation efficace. Il permet de localiser, ou de suivre en temps réel, un véhicule. Fonctionnant sur plusieurs fréquences GSM (850 MHz, 900 MHz, 1800 MHz, and 1900 MHz), il peut être utilisé partout dans le monde. Il est possible de le paramétrer pour recevoir sur un téléphone portable un SMS et ainsi être prévenu de sa position exacte. En cas de vol du véhicule un simple SMS suffit pour le localiser[1].

Figure 1.2. : Exemple Application utilisant le GPS Android sur Sony Ericsson Xperia.

[1] http://www.allomotors.com/traceurs-gps/216-traceur-gps-k100-pour-voiture-et-moto.html

L'appareil le plus communément utilisé pour une navigation GPS est le Smartphone.

Un Smartphone (téléphone « futé » ou « intelligent ») est un téléphone mobile couplé à un PDA (Personal Digital Assistant). La saisie des données se fait par le biais d'un écran tactile ou d'un clavier. Il permet une meilleure gestion du temps grâce à des fonctionnalités agenda/calendrier mais également de la navigation web, de la consultation de courrier, une connectivité à un client de messagerie instantanée, la navigation GPS, etc. [Ilyas & Ahson,2006]

De plus en plus de personnes possèdent un Smartphone équipé de la fonctionnalité GPS, Ericsson anticipe une explosion des usages dans ce secteur. Le nombre de terminaux intelligents en circulation atteindra 3,3 milliards en 2018[2].

I.3) Applications mobiles de navigation GPS :

Un Smartphone permet d'installer des applications additionnelles sur l'appareil. Les applications de navigation GPS les plus connues de nos jours sont les différentes versions issues de Google Maps, le service gratuit de cartographie en ligne, qui représente la carte du monde avec des images satellite.

Ces versions mobiles utilisent les réseaux des téléphones, notamment 3G (Internet), pour charger les cartes, une fonction de géo-localisation est disponible sur certaines versions. Le succès de Google Maps est en grande partie dû à la mise à disposition du public d'un outil lui permettant de visualiser, par photo satellite, le monde entier de manière précise.

Néamoins En octobre 2011, Google décide de faire payer l'accès à l'API[3] de Google Maps à partir d'un certain nombre de requêtes par jours[4]. Plus un site ou une application est populaire, plus son développeur court le risque de devoir payer pour continuer à afficher une carte Google.

Une des alternatives les plus intéressantes à Google Maps est Open Street Map. Contrairement à Google Maps les cartes OSM sont totalement gratuites et disponibles en téléchargement sous forme de données ou d'images sur le site officiel. Elles sont représentées

[2] http://www.numerama.com/magazine/25329-le-nombre-de-smartphones-dans-le-monde-va-tripler-selon-ericsson.html

[3] **API :** Une API a pour objet de faciliter le travail d'un programmeur en lui fournissant les outils de base nécessaires à tout travail à l'aide d'un langage donné. Elle constitue une interface servant de fondement à un travail de programmation plus poussé.

[4] http://www.google.com/permissions/geoguidelines.html#maps-web

sous forme de plans 2D, et sont disponibles pour les développeurs qui souhaitent les intégrer dans leurs applications sans rien payer en contre partie. Un autre avantage majeur est la possibilité de modifier la carte en ligne, chaque utilisateur peut cartographier une zone et l'ajouter à la base de données, une communauté de bénévoles assure la mise à jour fréquente de la carte, une partie du prochain chapitre explique en détail les cartes OSM.

I.4) Les systèmes d'exploitation pour Smartphone:

Les Smartphones sont caractérisés par une faible puissance de calcul, un faible espace de stockage, une autonomie limitée, … Des contraintes qui font qu'un système traditionnel (windows) n'est pas adapté à ces derniers. Plusieurs systèmes d'exploitation pour mobile ont été conçus.

Cette partie offre un aperçu des systèmes d'exploitation pour mobile qui se partagent le marché, tout particulièrement Android, qui est le système utilisé dans notre travail d'implémentation.

I.4.1) iOS : [Wiley & Sons, 2011]

iOS, anciennement iPhone OS, est le système d'exploitation mobile développé par Apple pour l'iPhone, l'iPod touch, et l'iPad. Il est dérivé de Mac OS X dont il partage les fondations. Le système d'exploitation occupe moins d'un demi-gigaoctet (Go) de la capacité mémoire totale de l'appareil. A fin d'utiliser le kit de développement nécessaire à réaliser des applications et les publier il faut adhérer au programme des développeurs Apple, pour la somme de 99 $ par an.

I.4.2) Windows Phone [1] :

Windows Phone est un système d'exploitation mobile développé par Microsoft pour succéder à Windows Mobile, sa précédente plateforme. Contrairement au système qu'il remplace, Windows Phone est d'abord principalement destiné au grand public plutôt qu'au marché des entreprises. Cependant depuis Windows Phone 8, Microsoft propose des fonctions avancées pour les entreprises en offrant, par exemple, un espace d'applications réservé aux entreprises. Il a été lancé le 21 octobre 2010 en Europe et le 8 novembre 2010 aux États-Unis.

I.4.3) Symbian OS : [Wiley & Sons, 2008]

Symbian OS est un système d'exploitation pour téléphones portables (on parle de « Smartphone ») et PDA conçu par Symbian ltd.

Il est né d'un consortium de différents constructeurs. Il dispose de nombreuses API spécifiques pour la communication mobile voix et données, et utilise des protocoles standard de communication réseau : IPv4/IPv6, WAP, MMS, Bluetooth, GPRS/UMTS, Java, SyncML.

I.4.4) BlackBerry OS : [Wiley & Sons, 2011]

BlackBerry OS est un système d'exploitation propriétaire pour téléphone mobile de la gamme BlackBerry, conçu par la société canadienne Research In Motion (RIM), maintenant BlackBerry.

Il s'agit d'un système multitâche. Le système est surtout connu pour son support natif des courriels à travers le protocole Mobile Information Device Profile (MIDP 1.0), et plus récemment permet une synchronisation complète avec les messageries d'entreprise telles que Microsoft Exchange ou IBM Lotus Domino.

I.4.5) Android: [2]

Android est un système d'exploitation open-source pour Smartphones, PDA et autres terminaux mobiles, conçu par Android, une start-up rachetée par Google en juillet 2005. Il existe d'autres types d'appareils possédant ce système d'exploitation tels que les téléviseurs et les tablettes.

Afin de promouvoir ce nouveau système d'exploitation ouvert, Google a su fédérer autour de lui un consortium d'une trentaine d'entreprises : l'Open Handset Alliance (OHA) créée officiellement le 5 novembre 2007. Toutes ces entreprises interviennent, plus ou moins directement, dans le marché de la téléphonie mobile. Le but de cette alliance est de mettre en place des normes ouvertes dans le domaine de la téléphonie mobile. Les développeurs d'application Android pourront accéder aux fonctionnalités du cœur du téléphone via une API très fournie. [Mohan,2013]

Android aura comme principaux concurrents Apple avec l'iPhone, Microsoft et son Windows Mobile et Nokia avec Symbian mais également des solutions libres telles que LIMO ou OpenMoko.

a) Historique :

Figure 1.3. : Aperçu des versions d'Android.

La première version d'Android apparait en octobre 2008 et n'avait pas de nom officiel. Cette version s'est avérée être la **Béta** du système.

La version 1.5 **Cupcake** corrigea le manque d'API et rendit le système plus utilisable.
Depuis, **Android 1.6, 2.0** et **2.1** ont apporté d'importantes améliorations respectivement sur les fonctionnalités et sur l'interface graphique du système.

Android 2.2 Froyo a fortement mis l'accent sur la synergie avec Internet. L'envoi d'applications et de liens instantanés depuis un ordinateur est désormais possible. Aussi, Google annonce que le navigateur chrome intégré à Android 2.2 est le navigateur mobile le plus rapide au monde grâce à l'intégration du moteur JavaScript V8.

Android 3.0 Honeycomb est spécialement étudié pour les tablettes tactiles. Les premiers modèles ont été annoncés en 2011.
On y apprend quelques nouveautés comme la prise en charge de la vidéoconférence via Gtalk (Google Talk), la nouvelle interface Gmail ou encore le lecteur de livre électronique Google.

Android 4.0 IceCreamSandwich unifie le développement des interfaces Smartphone, tablette, télévision connectée et système embarqué. Avant la version 4.0, le développement d'une application pour une tablette n'était pas compatible à un Smartphone car de tailles d'écrans différentes.

Android 4.2.2 Jelly Bean Basé sur un nouveau kernel Linux 3.1.10, *Jelly Bean* est une mise à jour incrémentale avec le but premier d'améliorer l'interface utilisateur en matière de fonctionnalité et de performance. L'amélioration des performances se fait via le *« Project Butter »*, qui utilise une anticipation tactile, et le passage à l'affichage de 60 images par seconde pour créer une interface utilisateur harmonieuse et fluide.

Après ce bref historique de l'évolution des versions d'Android depuis sa création, voici l'architecture générale du système basé sur un noyau Linux et nous présentons les avantages à utiliser ce dernier.

b) Architecture du système d'exploitation Android :

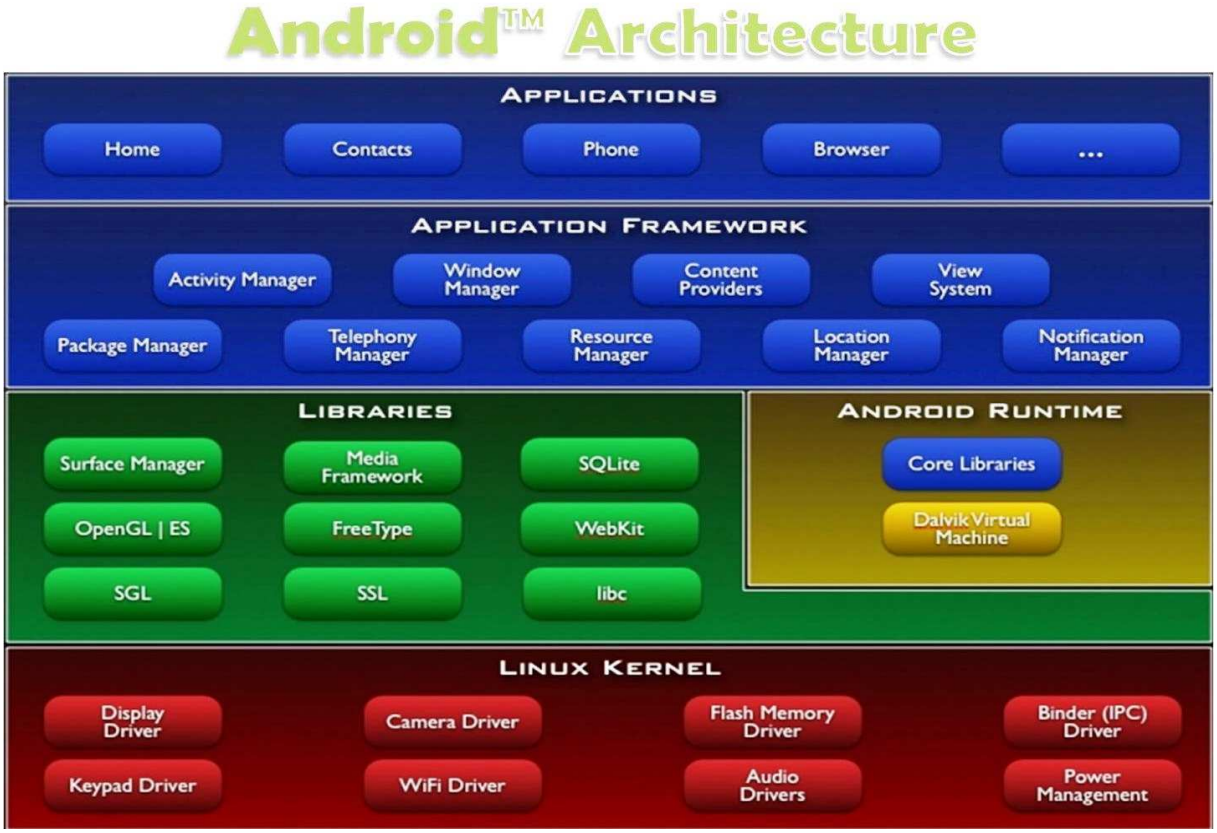


Figure 1.4. : Architecture d'Android.

Android est basé sur un kernel (noyau) linux. Au-dessus de cette couche, on retrouve les librairies C/C++ utilisées par un certain nombre de composants du système Android.
Au-dessus des librairies, on retrouve l'Android Runtime. Cette couche contient les librairies cœurs du Framework ainsi que la machine virtuelle exécutant les applications.

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

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

- **Framework de développement**

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

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

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

- **Android Runtime**

Android inclut un ensemble de librairies de base offrant la plupart des fonctionnalités disponibles dans les bibliothèques de base du langage de programmation Java.

Chaque application Android s'exécute dans son propre processus, avec sa propre instance de la machine virtuelle Dalvik. Dalvik a été écrit pour que le dispositif puisse faire tourner plusieurs machines virtuelles de manière efficace. La machine virtuelle Dalvik exécute des fichiers dans l'exécutable Dalvik (. DEX), un format optimisé pour ne pas encombrer la mémoire.

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

- **Bibliothèques**

Android dispose d'un ensemble de librairies C / C++ utilisées par les différents composants du système. Elles sont offertes aux développeurs à travers le Framework Android.

- **Linux Kernel**

Android est basé sur un kernel linux 2.6 (3.1 sur android 4.2.2) mais ce n'est pas linux. Il ne possède pas de système de fenêtrage natif (X window system).

Un kernel avec différents patches est utilisé pour la gestion de l'alimentation, le partage mémoire, etc. permettant une meilleure gestion de ces caractéristiques pour les appareils mobiles.

L'avantage de ce dernier est son système de gestion mémoire et de processus reconnu pour sa stabilité et ses performances. Le model de sécurité utilisé par linux, basé sur un système de permission, est connu pour être robuste et performant. Il n'a pas changé depuis les années 70.

Parmi les points qui ont poussé l'équipe en charge du noyau à décidé d'utiliser un kernel linux :

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

c) Applications Android :

Les applications Android sont constituées d'un ensemble d'activités qui permettent l'interaction avec l'utilisateur, et d'un ensemble de services qui fonctionnent en arrière plan (cachés) et s'occupent d'effectuer certains traitements précis, comme par exemple la récupération des données GPS en temps réel.

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

Figure 1.5 : Exemple d'une Activité d'une application Android.

Une activité possède un cycle de vie bien précis, qui permet de contrôler aisément l'application de son lancement jusqu'à sa fermeture. L'application passe par plusieurs états, lors du développement on associe un comportement à chaque état, ou bien lors d'un changement d'état.

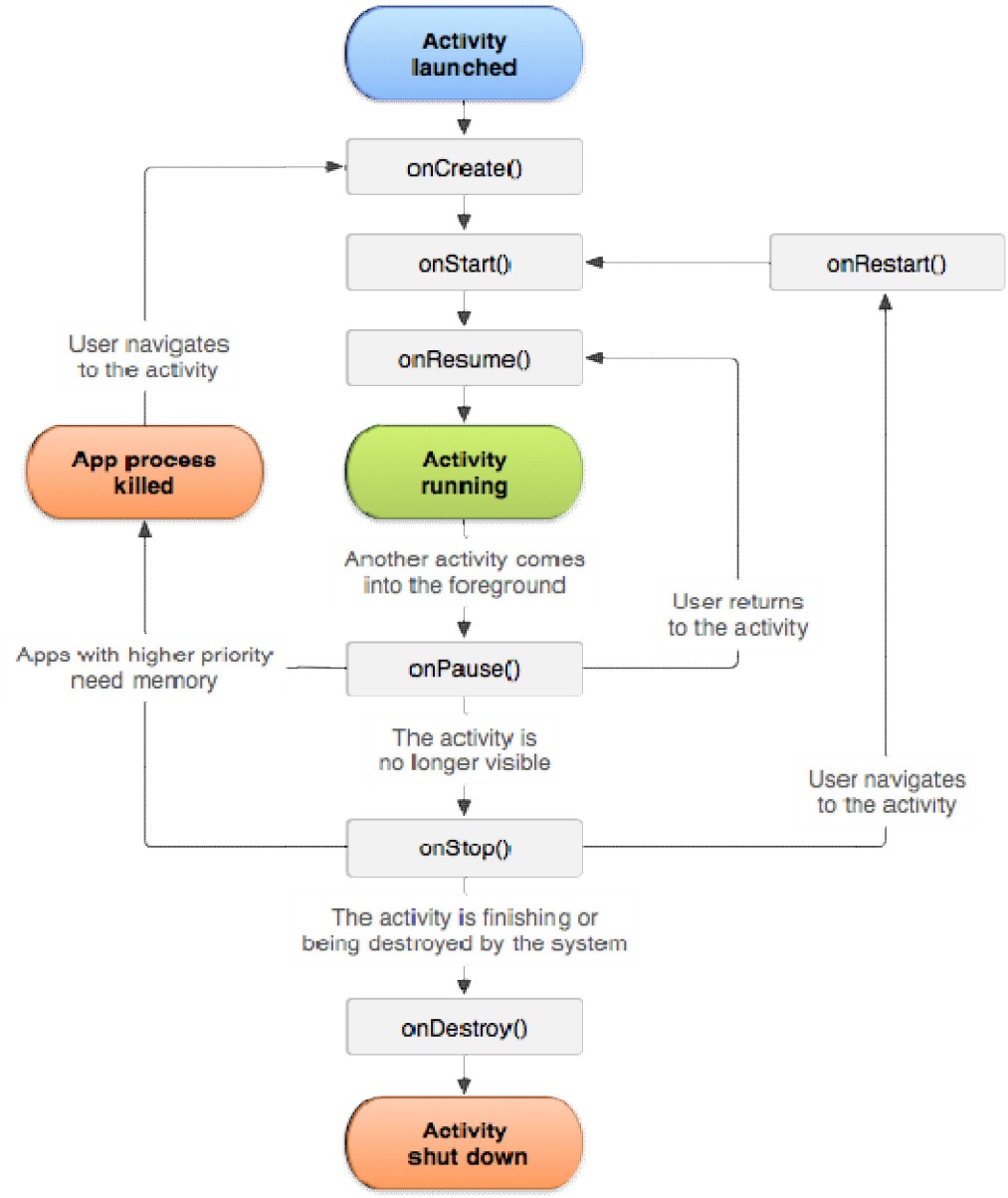


Figure 1.6. : Cycle de vie d'une application Android **[3]**.

Chaque état peut contenir une action précise à effectuer, une description brève des fonctions est présentée ci-dessous.

- onCreate() : Exécuté quand l'utilisateur clique sur l'icône de l'application pour une première fois
- onRestart(): Exécuté lorsque l'activité arrêté (mise en arrière plan) redémarre (repasse en premier plan);
- onStart() : Exécuté après chaque onCreate() ou onRestart();

- onResume() : Exécuté a chaque passage en premier plan de l'activité;
- onPause() : Exécuté avant chaque arrêt de l'activité, ou avant qu'une autre application ne soit lancé (mettant l'activité actuelle en arrière plan);
- onStop() : Exécuté avant chaque mise en veille ou fermeture de l'activité, les ressources sont libérées.
- onDestroy() : Exécuté lors du de l'arrêt complet de l'activité, les ressources sont libérées (fichiers temporaires).

Android offre des avantages non négligeables, que ce soit pour les développeurs, ou pour les utilisateurs qui bénéficient grâce aux dernières versions de nombreuses fonctionnalités et d'un confort d'utilisation de qualité.

Android est :

- **Open source**

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

- **Gratuit (ou presque)**

Android est gratuit, autant pour les utilisateurs que pour les constructeurs. La publication d'applications sur le Google Play[5], coûte la modique somme de 25$. Ces 25$ permettent de publier des applications à l'infini, contrairement à IPhone ou il faut verser une somme de 100$ par an.

[5] Google Play est un site de contenus numériques proposé par Google sur lequel vous pouvez trouver vos titres, vos livres, vos films, vos applications et vos jeux Android préférés.

- **Facile à développer**

Toutes les API mises à disposition facilitent et accélèrent grandement le travail. Ces APIs sont très complètes et très faciles d'accès. De manière un peu caricaturale, on peut dire qu'il est possible d'envoyer un SMS en seulement deux lignes de code.

- **Facile à vendre**

Le Play Store (anciennement Android Market) est une plateforme immense et très visitée ; c'est donc une mine d'opportunités pour quiconque possède une idée originale ou utile.

- **Flexible**

Le système est extrêmement portable, il s'adapte à beaucoup de structures différentes. Les Smartphones, les tablettes, la présence ou l'absence de clavier ou de trackball, différents processeurs... On trouve même des fours à micro-ondes qui fonctionnent à l'aide d'Android.

Non seulement c'est une immense chance d'avoir autant d'opportunités, mais en plus Android est construit de manière à faciliter le développement et la distribution en fonction des composants en présence dans le terminal (si une application nécessite d'utiliser le Bluetooth, seuls les terminaux équipés de Bluetooth pourront la voir sur le Google Play).

- **Ingénieux**

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

En 2013, Android s'arrogeait 75% de part de marché dans la vente de Smartphone. Apple restait en retrait avec 17,3% et Microsoft a décroché la troisième place devant Blackberry[6].

[6] http://www.01net.com/editorial/595261/smartphones-android-loin-devant-apple-et-blackberry-derriere-microsoft/

I.4) Problématique:

Comment assurer la navigation par GPS, à l'aide de cartes OSM, sans avoir recours à internet ?. Telle est la question centrale de notre travail.

La majorité des applications de navigation GPS existantes nécessitent un accès à internet, principalement pour le chargement des cartes et la recherche d'itinéraire. Les données sont stockées dans des serveurs distants et les algorithmes de calculs sont assurés par web service.

Réaliser une application sans connexion à internet nécessite de disposer des données en local qui permettent de charger les cartes et de donner la possibilité d'utiliser une fonction de recherche d'itinéraire.

I.5) Conclusion :

Dans ce premier chapitre nous avons introduit les généralités en rapport avec nos travaux, avec une présentation détaillée du système d'exploitation Android sur lequel est fait le travail d'implémentation. Dans le prochain chapitre il est question d'introduire les notions de base sur la géo-localisation, en présentant les différentes méthodes existantes, une partie est consacrée aux cartes OSM qui sont décrites d'une manière détaillée.

CHAPITRE 2
Géo localisation

II. 1) Introduction :

Afin de bien comprendre notre domaine d'étude nous abordons dans le présent chapitre la géolocalisation ainsi que ces différentes techniques de localisation tout en mettant l'accent sur la technique choisi pour se projet, à savoir la géolocalisation par satellite. On terminera par une présentation des cartes collaboratives où on mettra en évidence l'importance d'utiliser les données OSM dans des projets de géolocalisation ainsi que le format approprié pour une exploitation de ces données sur des Terminaux à ressources limitées[7].

II. 2) Géolocalisation : [4]

La géolocalisation (géoréférencement) est un procédé permettant de positionner un objet ou une personne sur un plan ou une carte à l'aide de ses coordonnées géographiques[8].

Cette opération est réalisée à l'aide d'un terminal capable d'être localisé (grâce à un *système de* positionnement par satellites[9] ou d'autres techniques) et de publier ses coordonnées géographiques. Les positions enregistrées peuvent être stockées au sein du terminal ou être transmises en temps réel vers une plateforme logicielle de géolocalisation. La transmission temps réel nécessite que le terminal soit équipé d'un moyen de télécommunication de type GSM, GPRS, UMTS (Universal Mobile Telecommunications System), radio ou satellite lui permettant d'envoyer les positions à des intervalles réguliers. Ceci permet de visualiser la position du terminal au sein d'une carte à travers une plateforme de géolocalisation le plus souvent accessible depuis internet.

---

[7] ***Ressources limitées :*** *faible* puissance processeur, taille mémoire (disque dure, Ram), etc …

[8] ***Coordonnées géographiques :*** Système de trois coordonnées qui sont le plus souvent : la latitude, la longitude et l'altitude par rapport au niveau de la mer en un point arbitrairement choisi.

Latitude : représentée par une valeur angulaire, expression de la position d'un point sur terre, au nord ou au sud de l'équateur qui est le plan de référence.

Longitude **:** représentée par une valeur angulaire, expression du positionnement est-ouest d'un point sur Terre. La longitude de référence sur Terre est le méridien de Greenwich.

Altitude **:** est l'élévation verticale d'un lieu ou d'un objet par rapport au niveau de la mer.

[9] ***positionnement par satellites :*** Appelé sous le nom complet de **système de positionnement et de datation par satellites** ou sous son sigle anglais **GNSS** *(Global Navigation Satellite System)*, est le nom général des systèmes de navigation satellitaires fournissant une couverture globale de géopositionnement à usage civil.

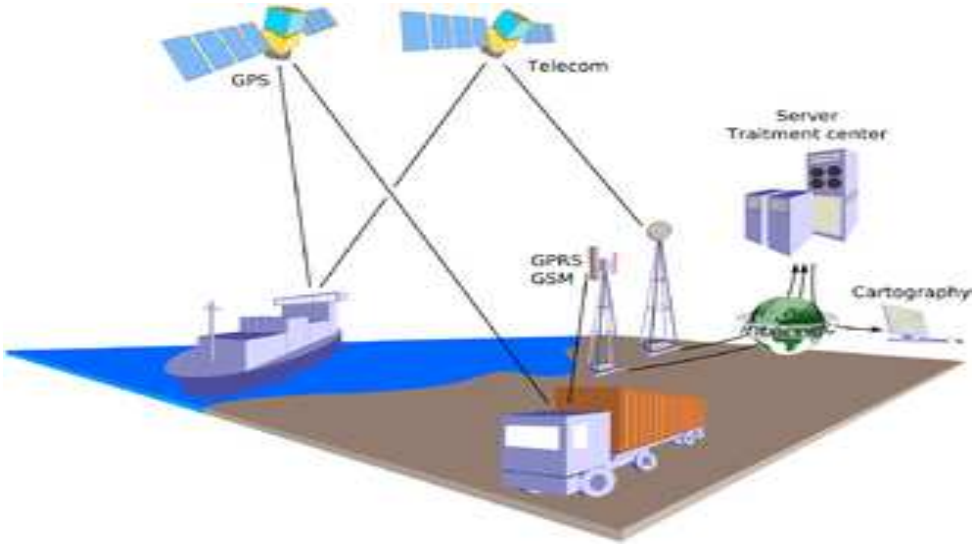


Figure 2.1 Principe de la géolocalisation par GPS.

II. 2.1) Techniques de géolocalisation :

Différents techniques de géolocalisation existent, selon le niveau de précision, le cout, la couverture réseau ou le droit d'accès. Dans ce qui suit nous présenterons quelques unes de ces techniques.

a) Géolocalisation par GSM (Global System for Mobile Communications): [Neviere, 2001]

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

Plusieurs techniques existent :

- **Différence de temps observée ou EOTD** (enhanced-observed timed difference) : le terminal calcule le temps écoulé entre l'émission et la réception de la requête envoyée à l'antenne, il peut alors calculer sa distance par rapport à celle-ci.
- **Temps d'arrivée (time of arrival)**
- **Angle d'arrivée (angle of arrival)**
- **Cell ID (identifiant de cellule)**

Aujourd'hui, la méthode GSM la plus utilisée est celle du **Cell ID** (identification de la cellule radio[10]), c'est la technologie la moins coûteuse car il n'y a pas de matériel à mettre en place. Du moment que le portable est dans une zone couverte par le réseau, il se connecte à une antenne relais GSM. Puis le portable est localisé grâce à une base de données faisant le lien entre les identifiants des cellules et les positions géographiques des antennes. Cette localisation est très rapide (moins de 5 secondes) mais elle est peu précise car elle dépend du nombre d'antennes relais et de leur rapprochement (plus l'antenne est isolée, plus la zone de couverture est vaste et moins la localisation est précise, la précision est de l'ordre de 50 à 500 mètres dans un milieu urbain alors qu'elle peut varier de 5 à 15Km dans les régions isolés).

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

***Remarque** :*

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

b) Géolocalisation par WIFI: [5]

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

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

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

[10]***Cellule radio*** : Permet de réutiliser de nombreuses fois les mêmes fréquences ; il permet aussi à ses utilisateurs en mouvement de changer de cellule (handover) sans coupure des communications en cours.

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

Ces bases peuvent être créées de deux manières :

- à partir des données transmises par les téléphones mobiles eux-mêmes lorsqu'ils demandent à être géolocalisés.
- à partir des données WiFi collectées par des véhicules se déplaçant dans les rues des villes et sur les principaux axes routiers.

c) Géolocalisation par adresse IP (sur internet) : [6]

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

La précision de la géolocalisation par identification de l'adresse IP utilisée est cependant sans commune mesure avec celle permise par un GPS ou par triangulation des téléphones mobiles. On considère généralement que les meilleurs référentiels d'adresses IP sont globalement fiables à l'échelle des villes.

d) Géolocalisation par RFID (Radio Frequency Identification) : [CiscoMag, 2006]

Les technologies de RFID ont pour objectif d'identifier quelque chose à l'aide d'une étiquette passive réagissant à un signal de radio fréquence. Lors de cette sollicitation l'étiquette RFID va fournir de l'information sur l'objet. Elle peut aussi être capable d'enregistrer de l'information. Le RFID va ainsi permettre de déterminer quel objet ou produit traverse un point de passage, où se trouvent les capteurs RFID, mais aussi d'indiquer dans l'étiquette de l'objet qu'il est passé par tel point de passage. L'intérêt est évident, en particulier pour toutes les chaînes logistiques. On parle parfois de RFID passif, pour illustrer

le fait que le l'étiquette RFID ne fait que réagir, mais n'est pas capable par elle-même d'émettre**.**

e) Géolocalisation par GPS :

Le Global Positioning System (GPS) est un système de navigation par satellite qui a été développé par l'US Department of Defense (DoD) au début des années 1970. Initialement, le GPS a été développé comme un système militaire pour répondre aux besoins des militaires américains. Toutefois, il a fini par être autorisé aux civils, et maintenant c'est un système à double usage qui peut être consulté par les utilisateurs civils et militaires [Kleusberg, 1990].

e.1) Présentation :

GPS fournit un positionnement continu et des informations de synchronisation, n'importe où dans le monde dans toutes les conditions météorologiques. Parce qu'il sert un nombre illimité d'utilisateurs ainsi que d'être utilisé pour des raisons de sécurité, le GPS est un système à sens unique (passif) [Phillips, 1996]. L'utilisateur peut seulement recevoir les signaux.

Mis en place par le Département de la Défense des États-Unis. Il est très rapidement apparu que des signaux transmis par les satellites pouvaient être librement reçus et exploités, et qu'ainsi un récepteur pouvait connaître sa position sur la surface de la Terre, avec une précision sans précédent, dès l'instant qu'il était équipé des circuits électroniques et du logiciel nécessaires au traitement des informations reçues. Une personne munie de ce récepteur peut ainsi se localiser et s'orienter sur terre, sur mer, dans l'air ou dans l'espace au voisinage proche de la Terre.

e.2) Historique :

En 1960 l'armée américaine (à la demande du président Richard Nixon) à lancée un projet de recherche sur le GPS dont la réalisation a été confiée a *Ivan A. Getting*[11] qui a conçu le principe d'un groupe de satellites gravitant en orbite et émettant des ondes radio UHF captées par des récepteurs GPS.

En 1978, un premier satellite est lancé. En 1983, suite au crash d'un avion de la compagnie Korean Airlines et la mort de ses 269 passagers, le président Ronald Reagan,

[11] ***Ivan A. Getting (*** 1912-2003), c'est grâce à lui que les GPS sont devenus selon ses propres paroles "*des phares dans le ciel guidant toute l'humanité*".

propose que la technologie GPS soit disponible gratuitement aux civils. En 1995, le déploiement des 24 satellites opérationnels (plus 4 en réserve) est achevé. Le système devient alors opérationnel en permanence sur l'ensemble de la planète, avec une précision limitée à une centaine de mètres pour un usage civil. En 2000, le président Bill Clinton confirme l'intérêt de la technologie à des fins civiles et autorise une diffusion non restreinte des signaux GPS, permettant une précision d'une dizaine de mètres.

e.3) Fonctionnement [3]:

Le GPS comprend au moins 24 satellites tournant à 20 200 km d'altitude. Ces satellites émettent en permanence un ou plusieurs codes pseudo-aléatoires, datés précisément grâce à leur horloge atomique[12], et par un message de navigation. Ce message, transmis à 50 bit/s, inclut en particulier les éphémérides[13] permettant le calcul de la position des satellites, ainsi que des informations sur leur horloge interne. Les codes sont un code C/A (acronyme de « coarse acquisition », en français : « acquisition brute ») de débit 1,023 Mbit/s et de période 1 ms, et un code P (pour « Précis ») de débit 10,23 Mbit/s et de période 1 semaine. Le premier est librement accessible, le second est réservé aux utilisateurs autorisés car il est le plus souvent chiffré : on parle alors de code Y.

Ainsi, un récepteur GPS qui capte les signaux d'au moins quatre satellites équipés de plusieurs horloges atomiques peut, en calculant les temps de propagation de ces signaux entre les satellites et lui, connaître sa distance par rapport à ceux-ci et, par trilatération[14], situer précisément en trois dimensions n'importe quel point placé en visibilité des satellites GPS, avec une précision de 3 à 50 mètres pour le système standard. Dans certains cas, seuls trois satellites peuvent suffire. La localisation en altitude (axe des Z) n'est pas d'emblée correcte alors que la longitude et la latitude (axe des X et des Y) sont encore bonnes. On peut donc se contenter de trois satellites lorsque l'on évolue au-dessus d'une surface « plane » (océan, mer).

[12] **Horloge atomique** est une horloge qui utilise la pérennité et l'immuabilité de la fréquence du rayonnement électromagnétique émis par un électron lors du passage d'un niveau d'énergie à un autre pour assurer l'exactitude et la stabilité du signal oscillant qu'elle produit. Un de leurs principaux usages est le maintien du Temps atomique international (TAI) et la distribution du Temps universel coordonné (UTC) qui sont les échelles de temps de référence.

[13] **Éphémérides** (du grec ἐφημερίς, journal, agenda) sont des tables astronomiques par lesquelles on détermine, pour chaque jour, la valeur d'une grandeur caractéristique d'un objet céleste, notamment les positions des planètes, de leurs satellites, de la Lune, du Soleil, des étoiles, des comètes.

[14] **Trilatération** est une méthode mathématique permettant de déterminer la position relative d'un point en utilisant la géométrie des triangles tout comme la triangulation. Mais contrairement à cette dernière, qui utilise les angles et les distances pour positionner un point, la trilatération utilise les distances entre un minimum de deux points de références.

Étant un système développé pour les militaires américains, une disponibilité sélective a été prévue : certaines informations, en particulier celles concernant l'horloge des satellites, peuvent être volontairement dégradées et priver les récepteurs qui ne disposent pas des codes correspondants de la précision maximale. Pendant quelques années, les civils n'avaient ainsi accès qu'à une faible précision (environ 100 m). Le 1er mai 2000, le président Bill Clinton a annoncé qu'il mettait fin à cette dégradation volontaire du service.

II.2.2) Combinaison de différentes techniques :

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

- **La dépendance au réseau GPS :** l'incapacité de l'utiliser en intérieur et le temps de réponse à l'allumage.
- **La dépendance au réseau GSM :** sa couverture géographique, l'accès au réseau GPRS pour exploiter l'information.
- **La dépendance à la présence de bornes d'accès WiFi :** en zone rurale par exemple.

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

II.2.3) Télérelève d'information :

Elle consiste à récupérer à distance une série d'informations issues de capteurs ou de systèmes informatiques, électroniques ou électriques. La géolocalisation est très souvent couplée à des systèmes de télérelève via des boîtiers télématiques[15], ce qui permet de combiner la position géographique d'un terminal ou d'un véhicule à une série d'informations annexes relatives à l'objet géolocalisé. Dans un véhicule par exemple, ces boîtiers peuvent se

[15] **Télématique** est un terme qui recouvre les applications associant les télécommunications et l'informatique.

connecter au chronotachygraphe[16] (pour le transport routier) ou à divers capteurs ou voyants, ce qui permet de relever des informations telles que :

- la vitesse du véhicule
- les kilomètres parcourus
- l'état d'une porte (ouverte/fermée)
- l'état d'une remorque (accrochée/décrochée)
- la température (pour les camions frigorifiques)

II.2.4) Plateformes logicielles de géolocalisation :

❖ Composant :

Les principaux composants d'une plateforme de géolocalisation sont les suivant :

- **Terminal communicant :** C'est le terminal qui reçoit ses coordonnées géographiques (via GPS ou tout autre moyen) et qui les envoie via un réseau de télécommunications à la plateforme ou les utiliser localement si la plateforme est hébergé au niveau local (c'est le terminal lui-même qui s'occupe du traitement de l'information).
- **System informatique capable de recevoir, stocker et traiter l'information :** il s'agit des serveurs qui hébergent l'infrastructure et qui s'occupent de la réception des données (envoyées par les terminaux dans le cas ou la plateforme logiciel est hébergé sur un site distant), leur traitement ainsi que leur mise à disposition des utilisateurs.
- **Module cartographique :** c'est le module qui va permettre d'afficher la position des terminaux sur un fond cartographique adapté. Il prend en charge les calculs de distances, d'itinéraires, détecte l'interaction avec les zones et permet d'avoir accès à des informations terrain (sens interdits, restrictions pour les poids lourds, vitesses autorisées...).

❖ Architecture temps réel :

La donnée généré par un terminal sur le terrain est une information brute qui peut être exploitée et couplée à d'autres données, pour pouvoir l'exploiter elle doit être transmise à une plateforme logicielle qui va la traiter, la présenter graphiquement à l'utilisateur et

[16] **Chronotachygraphe** est un appareil électronique enregistreur de vitesse, de temps de conduite et d'activités (travail, attentes...) installé dans un véhicule de transport routier(8).

l'associer à d'autres données afin d'enrichir les informations relatives à l'état du terminal ou de la flotte de terminaux.

Voici quelques étapes de la chaîne de traitement :

1. Le terminal détermine sa position géographique grâce à une des techniques de géolocalisation (de préférence GPS);
2. Il envoie ces données vers une plateforme logicielle (soit par le réseau GSM/GPRS soit par un réseau satellitaire) ou les stocke au niveau de la plateforme locale (dans le cas où elle est installée au niveau locale);
3. La plateforme logicielle de géolocalisation traite la donnée et positionne le terminal géographiquement sur une carte moyennant la précision offerte par la technique de géolocalisation utilisée. De plus, en combinant plusieurs informations, notamment récupérées via un système de télérelève (trafic routier, autonomie du véhicule, points à visiter etc...), des calculs d'itinéraires ou de tournées peuvent par exemple être générés.
4. Cette carte ainsi que tous les traitements effectués sont mis à disposition de l'utilisateur à travers un portail web hébergé sur un serveur accessible depuis internet, ou à travers une application métier installée sur le terminal.

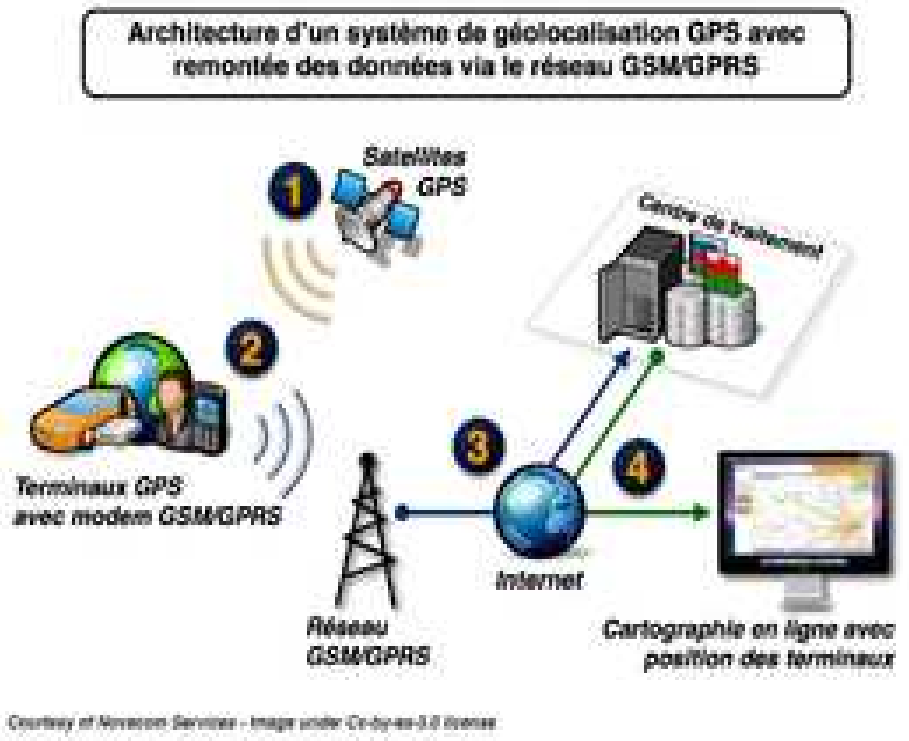


Figure 2.2 Architecture d'un système de géolocalisation par GPS.

❖ Fonctionnalités :

Voici une liste de fonctions typiquement offertes par les plateformes de géolocalisation professionnelles :

- Suivi en temps réel de terminaux.
- Affichage d'un historique de déplacements.
- Création de points d'intérêt.
- Configuration d'alertes automatiques par courriel ou SMS sur des événements.
- Localisation des itinéraires ainsi que le chemin le plus court entre deux points distants.
- Génération de rapports périodiques (temps de conduite, arrêts, vitesses moyennes, zones couvertes...).
- Fonds cartographiques variables (cartes classiques, cartographie photographique, cartes de fonds marins, cartes provenant d'un SIG etc)

II.3) OSM (Open Street Map): [7]

Open Street Map (OSM) est une initiative privée à but non lucratif visant à produire et partager des cartes collaboratives vectorielles en libre-diffusion du monde et en particulier du réseau routier. Partie d'Angleterre, OSM a désormais largement franchi les frontières de ce pays et la qualité des données s'est améliorée de manière spectaculaire à partir de 2008. Grâce à des outils de saisie très évolués (et open source) et à l'effort des bénévoles, les données disponibles sont désormais assez complètes.

II.3.1) Projet et Outils connus pour la cartographie collaborative:

Il y a différents moyens de faire une carte collaborative. Le plus simple est de s'asseoir avec les autres et d'utiliser un stylo et du papier. Bien qu'étant collaborative dans son processus de création, la technique papier-stylo ne permet pas facilement la collaboration au-delà du groupe réduit qui peut se réunir autour de la feuille de papier. Même si la feuille est très grande, il y a une limite physique au nombre de personnes pouvant travailler en même temps sur celle-ci.

OpenStreetMap permet à des groupes de collaborer sur des cartes électroniques, ce qui augmente considérablement ce nombre. Mais ce ne sont pas les seules options. Outre le couple papier-stylo et les outils entourant OpenStreetMap, il existe d'autres façons de créer et partager des cartes collaboratives.

a) **Google Map Maker et Google My Maps:** [8]

Google Map Maker est un moyen de corriger et d'ajouter des informations à Google Maps et Google Earth. Google permet par exemple d'ajouter les routes manquantes ou incorrectes, ou des points d'intérêt. Les modifications apportées par les nouveaux utilisateurs passent par un processus de vérification avant d'être publiées.

Google MyMaps est quand à lui un onglet associé à Google Maps qui permet d'enregistrer des informations dans la carte personnelle d'un utilisateur. Celui-ci peut alors décider rendre sa carte publique, privée, ou seulement partagée avec certaines personnes. Il lui est ainsi possible d'inviter des personnes précises à collaborer sur sa carte ou de rendre sa carte disponible au monde entier pour que chacun puisse y contribuer.

Si l'utilisation de Google Maps est gratuite, les données ne sont pas libres pour autant. Contrairement à OpenStreetMap, Google interdit des usages commerciaux. Il restreint aussi l'usage de ses cartes pour des services qui entreraient en compétition avec Google. Selon les conditions d'utilisation du projet, votre contribution devient ainsi un produit dérivé de Google. La compagnie se réserve en effet les usages commerciaux des données. Google s'appuie sur l'aide de visiteurs pour améliorer sa cartographie, sans pour autant en faire un bien public.

b) **Ushahidi:** [9]

Ushahidi est un projet à but non commercial et un logiciel open-source permettant la collecte d'informations collaboratives et l'affichage de ces informations sur une carte et un calendrier. Ushahidi est un mot Swahili qui veut dire témoin ou témoignage. Son but initial était de créer des cartes des cas de violence déclarés au cours des troubles postélectoraux au Kenya en 2007 -2008. Le site permettait notamment de collecter les témoignages de violence envoyés par email et SMS, en les plaçant sur une carte.

Aujourd'hui, on pourrait le présenter comme une initiative de cartographie sociale, combinaison d'activisme social, de journalisme citoyen et d'information géographique. D'un point de vue logiciel, Ushahidi fournit un mécanisme à un observateur local pour soumettre un rapport via son téléphone mobile ou internet, avec le lieu et l'heure de l'évènement. L'idée d'Ushahidi est de permettre aux utilisateurs de soumettre des « rapports » directement via l'application Ushahidi, à travers les réseaux sociaux, ou même par SMS. Il est possible de rapidement mettre en place une version d'Ushahidi en utilisant CrowdMap ou en téléchargeant

le code source et en l'installant sur un serveur. Le logiciel est ainsi utilisé par plusieurs projets de cartographie collaborative.

c) WikiMapia: [10]

WikiMapia est un projet commercial qui offre « une carte interactive modifiable dont le but est de créer et de maintenir à jour une carte gratuite, complète et multilingue du monde entier.» WikiMapia est surtout utilisée pour référencer des points d'intérêts rassemblés collectivement. Ce projet utilise les vues satellitaires de Google Maps et permet de les annoter avec un système de style "Wiki". Ainsi, WikiMapia fournit un support de carte modifiable par tout le monde, sans qu'il soit nécessaire d'avoir un compte pour consulter ou contribuer au projet.

À la différence d'OpenStreetMap qui a expressément obtenu de Microsoft le droit d'utiliser son imagerie aérienne, WikiMapia propose un travail dérivé de sources propriétaires (Google Earth). Par ailleurs, le travail contributif n'est pas placé sous licence libre, mais sous la licence Creative Commons ***BY-NC-SA***[17] (paternité, non commercial et partage des conditions à l'identique). On doit comprendre que le projet se réserve les usages commerciaux des données.

WikiMapia génère des revenus avec des publicités Google et la plateforme reste la propriété privée de ses fondateurs.

II.3.2) Mise en ouvre d'OpenStreetMap :

Traditionnellement, les données cartographiques sont collectées et tenues à jour par des organisations à but lucratif, qu'il s'agisse d'éditeurs privés ou d'entreprises publiques. Bien souvent, l'utilisation de ces cartes est restreinte, et dans certains cas, quand il n'est pas rentable de créer ou de maintenir des cartes, aucune de ces organisations n'édite de cartes.

OpenStreetMap a été créé pour combler ces manques, c'est une carte collaborative à l'échelle mondiale, ouverte[18], librement modifiable[19] (disponible sous licence ODbL[20]). À la manière

[17] **BY-NC-SA :** l'œuvre peut être librement utilisée, à la condition de l'attribuer à l'auteur en citant son nom, Partage des conditions initiales à l'identique

[18] **Ouverte** (**Opens Source**) s'applique aux logiciels dont la licence respecte des critères précisément établis par l'*Open Source Initiative*, c'est-à-dire la possibilité de libre redistribution, d'accès au code source et aux travaux dérivés.

de Wikipédia, tous les internautes naviguant dans le web peuvent contribuer à la création et à la numérisation de cartes. Des éditeurs permettent de réaliser en ligne des cartes en se basant sur un fond d'image satellitaire. Cependant, ces images satellitaires ne couvrent pas toujours en haute résolution l'ensemble du globe. C'est pourquoi il est possible d'introduire des données provenant de récepteurs GPS. Il suffit de réaliser un itinéraire et de positionner le récepteur GPS en mode enregistrement, puis de le restituer sur le serveur de données d'OpenStreetMap situé au Royaume-Uni et géré par la fondation OpenStreetMap. Cela a pour avantage de les rendre visibles dans les outils d'édition des cartes et facilite la couverture internationale : une personne séjournant dans une autre région ou un autre pays que le sien peut publier les tracés de ses parcours, à charge pour les habitants permanents de les compléter.

Les points d'intérêts (POI, en anglais « *point of interest* »), c'est-à-dire, toutes les mentions utiles (noms, largeur, nature du revêtement, sens uniques, parcs, zones résidentielles et d'activités, barrières, pistes cyclables, boîtes aux lettres, cabines téléphoniques, commerces, fontaines, etc.) sont notés, soit en les écrivant, soit en les photographiant, soit en les décrivant sur un appareil d'enregistrement audio. Tous les modes de locomotion terrestre possibles sont utilisés : à pied, à deux-roues, à rollers, à skis, en véhicule automobile particulier, en bus, en train…

Figure 2.3 Logo d'OpenStreetMap.

[19] **Librement modifiable (Logiciel libre) :** est un logiciel dont l'utilisation, l'étude, la modification et la duplication en vue de sa diffusion sont permises, techniquement et légalement. Ceci afin de garantir certaines libertés induites, dont le contrôle du programme par l'utilisateur et la possibilité de partage entre individus

[20] **ODbL (Open Data base License) :** permet à chacun d'exploiter publiquement, commercialement ou non, des bases de données; à condition néanmoins de maintenir la licence sur la base de données, et éventuellement, sur les modifications qui y sont apportées, et de mentionner expressément l'usage, s'il génère des créations à partir de celles-ci.

II.3.3) Modèle de données d'OpenStreetMap: [7]

Le modèle de données d'OpenStreetMap est une façon simple mais puissante de représenter l'information géographique.

Traditionnellement les données cartographiques sont représentées de trois façons différentes : un **point** est un simple lieu de l'espace défini par ses coordonnées, une **ligne** (*linestring*) est un objet linéaire pouvant représenter, par exemple, une route ou une frontière, et un **polygone** est une surface fermée. Les données attachées à ces objets géographiques sont généralement stockées dans une base de données secondaire, liée à la base géographique. Dans OpenStreetMap, ces trois concepts sont modélisés différemment, en **nœuds** (*nodes*), **lignes** (*ways*) et **relations**, complétés par des **attributs** (*tags*) décrivant chaque objet.

LES ATTRIBUTS

Les **attributs** (*tags*) sont des paires "clé=valeur" qui peuvent être liées à chaque élément géographique d'OpenStreetMap. Ces attributs décrivent l'objet géométrique. La clé et la valeur sont toutes deux des chaînes de caractères, de longueur maximale 255 caractères (en comptant les espaces), avec une seule contrainte : chaque clé doit être unique pour un même élément. Si un objet géographique ne porte aucun attribut, la plupart des moteurs de rendu ne le représenteront pas du tout.

NOEUDS

Un point géographique est appelé un nœud, et il est représenté par sa latitude, sa longitude et autant d'attributs que nécessaire. Par exemple, des nœuds sont utilisés pour représenter des magasins, des arrêts de bus, des bancs, des sommets et des boîtes à lettres. Un nœud sans attribut sera toujours un sous-élément d'un autre élément.

LIGNES

Une liste ordonnée de nœuds est appelée une **ligne** (*way*). Une ligne comporte un maximum de 2000 nœuds, pour s'assurer que les outils et les utilisateurs ne seront pas débordés par des objets énormes qu'il serait trop difficile de manipuler. Les lignes sont utilisées pour représenter des objets linéaires comme les chemins, les routes, les lignes de chemin de fer, les côtes et les frontières. Pour l'instant, les polygones ne constituent pas un type de données particulier : ce sont simplement des lignes fermées, c'est-à-dire dont le

dernier point est confondu avec le premier. Elles sont utilisées pour représenter les bâtiments, les parcs et les zones d'un plan d'occupation des sols.

RELATIONS

Une **relation** est une liste ordonnée de nœuds, de lignes et de relations. Chaque **membre** d'une relation a un **rôle** optionnel qui ajoute une information à ce sous-élément. Comme les attributs, ces rôles sont stockés dans des chaînes de caractère (limitée à 255). Les relations peuvent représenter des itinéraires routiers ou cyclables, des frontières administratives ou des cours d'eau, là encore selon les attributs qui leur sont attachés. Ces relations ne sont pas des catégories et ne doivent pas être utilisées seulement pour regrouper des objets. Pour cela, il existe généralement des attributs dédiés.

IDENTIFIANTS

Un élément dans la base de données OpenStreetMap, qu'il soit un nœud, une ligne ou un polygone, est identifié par un identifiant numérique unique. Cet identifiant n'a pas de signification particulière ; il ne sert qu'à repérer de façon univoque un élément. Une relation et une ligne utilisent ces identifiants pour référencer chacun de ses sous-éléments. Deux lignes se rencontrent uniquement si elles référencent au moins un même identifiant de nœud (et non pas deux nœuds de coordonnées identiques). De même, une ligne fermée représentant une surface référence obligatoirement deux fois le même identifiant de nœud.

FORMATS DE FICHIER D'OPENSTREETMAP

Les fichiers de données OpenStreetMap sont traditionnellement distribués sous un format XML représentant les concepts de nœud, ligne et relation dans un schéma simple et direct. Nativement, ce format XML peut être extrêmement volumineux, de sorte qu'il est souvent fourni sous une forme compressée par un algorithme efficace comme gzip ou bzip2. La plupart des outils conçus pour travailler sur un format XML OSM savent aussi gérer le XML compressé. Pour résoudre les deux problèmes de la taille et du temps d'analyse d'un gros fichier XML, d'autres formats ont étés conçus, utilisant des Protocoles qui permettent de compacter au maximum les données OSM dans un fichier binaire. Le résultat est même plus compact qu'un fichier compressé au format (bz2, map, etc...), et permettent un traitement bien plus rapide que du XML compressé. Il existe des bibliothèques dans les principaux langages pour lire ces formats. Il faut noter que dans le cadre de ce projet nous avons optés pour l'utilisation des cartes au format *.map* qu'on va voir dans la section II.3.4.

Voici un exemple d'une carte OpenStreetMap représentant l'université UMMTO (Bastos) suivi de quelques éléments clé du fichier XML correspondant :

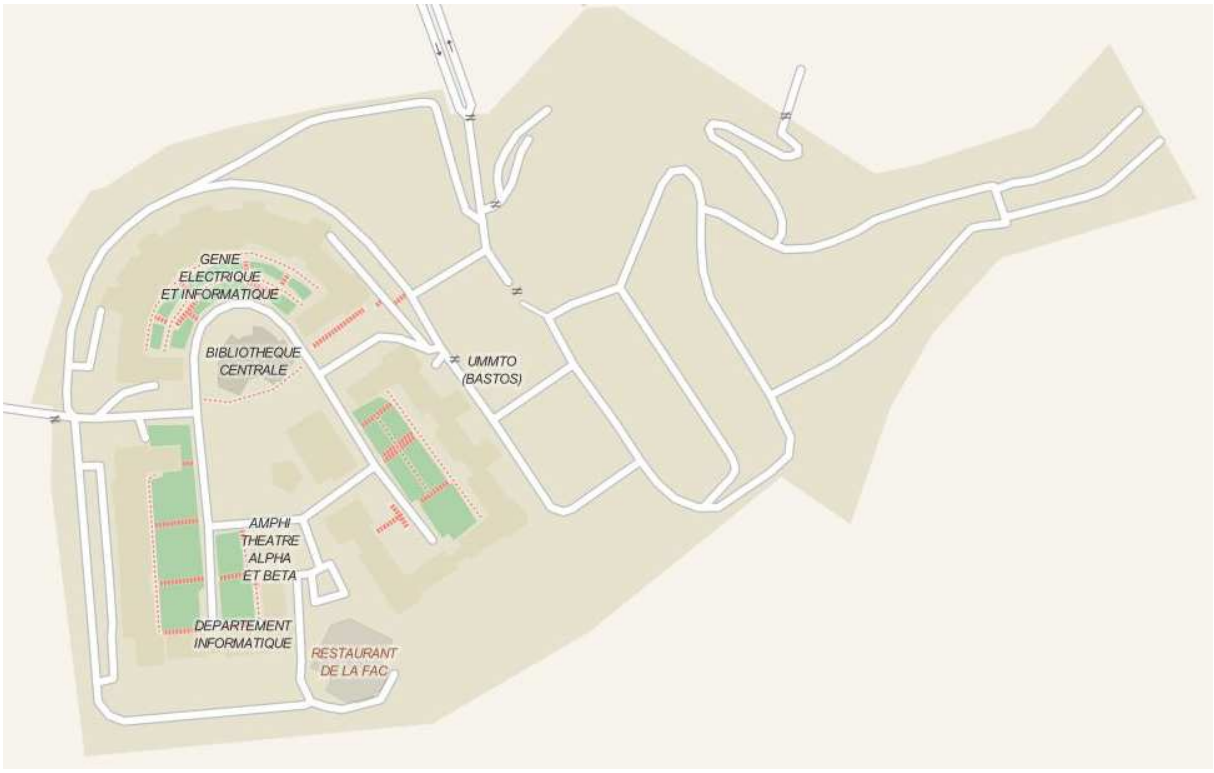


Figure 2.4 Carte OSM représentant l'UMMTO (Bastos).

Entête de fichier XML:

```
-<osm version="0.6" generator="CGImap 0.1.0" copyright="OpenStreetMap and contributors" attribution="http://www.openstreetmap.org/copyright" license="http://opendatacommons.org/licenses
/odbl/1-0/">
   <bounds minlat="36.6949500" minlon="4.0558300" maxlat="36.7004000" maxlon="4.0644500"/>
   <node id="291959051" lat="36.6972615" lon="4.0591553" user="Sandervalya" uid="56658" visible="true" version="3" changeset="12645734" timestamp="2012-08-07T13:50:28Z"/>
```

Déclaration d'un nœud :

```
-<node id="2333321680" lat="36.6996252" lon="4.0577090" user="BELLIL" uid="1452939" visible="true" version="1" changeset="16434792" timestamp="2013-06-05T17:12:25Z">
   <tag k="building" v="school"/>
   <tag k="name" v="UMMTO (bastos) "/>
 </node>
```

Information relative à un way :

```
- <way id="224858528" user="BELLIL" uid="1452939" visible="true" version="1" changeset="16468696" timestamp="2013-06-08T13:19:04Z">
    <nd ref="291959032"/>
    <nd ref="2337076453"/>
    <tag k="highway" v="residential"/>
  </way>
```

Remarque :

L'accès aux données brutes peut se faire soit en travaillant sur des jeux de données statiques (fichiers) soit en interrogeant une interface de programmation (API OpenStreetMap), comme l'API REST utilisée pour modifier les données OSM.

II.3.4) Fichier OpenStreetMap au format .MAP: [11]

Le *mapsforge binary map file format* (fichier .map) est un format qui est utilisé pour la génération dynamique des cartes géographiques sur des téléphones portables ou tout autre terminal à ressource limité utilisant l'OS Android.

Un fichier .map est composé de plusieurs sous fichiers qui stockent les informations relatives à une tuile (partie, carré) de la carte géographique pour chaque niveau de zoom.

STRUCTURE DES FICHIERS *.MAP* :

Pour chaque niveau de zoom un sous fichier (***sub-file***) est créé, chaque sous fichier contient des indexes (***tile index segment***) qui représent des pointeurs vers des segments de données (POI et way) de tuiles (***tile data segment***). La sauvegarde des informations des tuiles dans les ***tile data segment*** ainsi que des pointeurs ***tile index segment*** est comme suit :

Toute carte géographique est divisée sous forme d'un tableau, chaque case de ce tableau (représente une série de tuile pour différent niveaux de zoom) est implicitement référencé par les coordonnées de ces quatre coins qui sont mappées à des numéros de lignes et colonnes puis sauvegardées dans un fichier ***tile header***, en plus des indexes et segments de données.

II.3.5) Outils et Utilité :

Deux principaux types d'outils informatiques sont utilisables : les logiciels de rendu de cartes qui servent à élaborer les couches de la carte mondiale principale et de ses dérivés et les éditeurs de carte qui servent à modifier les couches existantes.

Ces logiciels sont tous sous licence libre et multiplateformes (GNU/Linux, MacOS X et Windows).

Logiciels de rendu de carte :

- *Mapnik.*
- *Osmarender.*

Logiciels d'édition de carte :

- *iD*, éditeur en ligne écrit en HTML5, développé par la société *MapBox.*
- *Potlatch*, éditeur en ligne codé en Flash accessible par un simple onglet à tout utilisateur enregistré sur le site OpenStreetMap.
- *JOSM* (Java OpenStreetMap Editor), application Java indépendante à l'interface proche de celle d'un navigateur Web. Elle permet de gérer plusieurs couches de données (traces GPS converties en XML, photos aériennes, etc.).
- *Maperitive* (logiciel) anciennement *Kosmos* (logiciel).
- *Merkaartor* (logiciel), éditeur de carte multiplateforme basé sur *Qt*.

Ces outils permettent d'utiliser les données d'OpenStreetMap pour :

- Alimenter la carte mondiale et en extraire certaines parties pour son propre usage (du globe complet à la carte locale).
- Créer des cartes interactives ou statiques.
- Créer des cartes pour de nombreux terminaux GPS.
- Alimenter certains SIG (Système d'Information Géographique).

II.4) Conclusion :

Dans le présent chapitre nous avons illustré quelques principes sur les différentes techniques de géolocalisation et les technologies associées, puis nous avons vu une brève introduction à la cartographie en exposant le modèle de données d'OpenStreetMap.

Dans le chapitre suivant nous entamons l’analyse et la conception de notre application.

CHAPITRE 3
CONCEPTION

III.1) Introduction :

Apres avoir définit notre domaine d'étude, nous entamons le processus de développement de notre Application par différentes étapes de conception qui vont nous permettre de mieux situer l'état d'avancement de notre projet.

Exécuter une application dans un environnement hors ligne nécessite de disposer de toutes les données en local. Une application de géo localisation a pour principale fonction de positionner un utilisateur sur une carte représentée graphiquement. La position récupérée et représentée par un marqueur sur une carte. Les principales données utilisées sont donc les cartes. Il s'agit de les utiliser en local (fichier OSM par exemple), de préférence sous un format de volume réduit, étant donné le faible espace de stockage des appareils mobiles.

Autre la localisation de l'utilisateur, il est intéressant de compléter l'application avec une fonction de recherche d'itinéraire, afin de guider l'utilisateur.

Il en résulte que la conception de notre application de navigation par GPS peut être divisée en deux parties :

- **Représentation de la carte sur un Smartphone :**

  Comment pouvoir représenter la carte géographique d'une ville, à partir des données offertes par l'organisation OpenStreetMap, sans avoir recours à internet ?
- **Recherche d'itinéraire :**

  Comment assurer la navigation étant donné la structure des fichiers OSM qui n'est pas directement utilisable par un algorithme de routage ?

III.2) Conception de l'application :

Dans cette partie il est question de présenter les différentes démarches suivies afin de réaliser une application de navigation par GPS, dans un environnement offline.

III.2.1) Représentation de la carte sur mobile :

Pour pouvoir représenter une carte nous avons pensé à plusieurs propositions qui ne sont pas toutes adéquates :

L'une des propositions qui vient à l'esprit est l'utilisation des images .PNG (méthode utilisé par plusieurs applications Smartphone ONLINE) correspondantes à la ville en question, puis d'utiliser un algorithme spécial pour afficher l'image adéquate selon le niveau de zoom et la position gps(coordonnées :latitude, longitude). Cette façon de faire n'est pas réalisable dans le cas de notre projet. La raison principale est la taille des données à utiliser pour représenter une ville (rien que pour la ville de Tizi ouzou la taille du fichier est de 29Méga Octet) sans oublier l'espace mémoire RAM nécessaire au traitement de l'affichage de l'image.

Une autre proposition peut être l'utilisation des données XML brutes (POI, way, relation etc…), accompagné d'un autre fichier XML pour le style. Elle consiste à générer les cartes dynamiquement au niveau local (utilisé souvent par des sites hébergé sur des serveurs à très grande puissance de calcul).

Cette méthode offre l'avantage de styliser les cartes géographiques à notre guise mais elle reste irréalisable vu la grande taille des fichiers XML de données (4 Méga octet) et le temps nécessaire à parcourir ces données pour la génération des cartes.

Enfin, une dernière proposition consiste à utiliser des fichiers compressés au format .MAP qui offre l'avantage de réduire énormément la taille des fichiers XML (31 kilos octets pour le même fichier précédent), accompagné d'un autre petit fichier de style en format XML pour personnaliser les cartes, toutefois la représentation de ce type de fichiers nécessite l'utilisation d'une bibliothèque (Mapsforge) qui sera présenté dans le prochain chapitre.

La deuxième étape du processus de conception est la recherche d'itinéraire entre deux points choisit sur la carte affichée.

III.2.2) Recherche d'itinéraire :

Pour rechercher un itinéraire entre un point départ et un point d'arrivé, nous avons tout d'abord pensé à utiliser directement les données de la carte en format XML ou MAP. Seulement on s'est vite rendu compte que la structure complexe[21] de ces formats ne permet pas d'appliquer facilement un algorithme de routage.

[21] **Structure complexe :** Par complexe on entend le manque d'informations comme la longueur d'une route pour pouvoir y appliquer un algorithme de routage.

Une représentation simplifiée est de générer un graphe des routes à partir du fichier .XML de la ville. En plus de l'avantage de faciliter la recherche d'itinéraire, cette représentation diminue le volume de donnée à traiter, en ne prenant en compte que les nœuds appartenant aux routes.

a) Raisonnement suivi :

Dans cette partie nous faisons part de notre travail de recherche de la meilleure représentation du graphe, en décrivant les différentes solutions que nous avons testé.

Comme présenté dans le chapitre 2, une route sur une carte OSM est représentée par une balise <way> contenant une référence vers tous les nœuds qui constitue cette route. La figure 3.1 représente un graphe composé de routes (différenciés par des couleurs) et de nœuds appartenant à ces routes, les intersections[22] sont mises en évidence avec une couleur bleue, et les nœuds isolés[23] en vert.

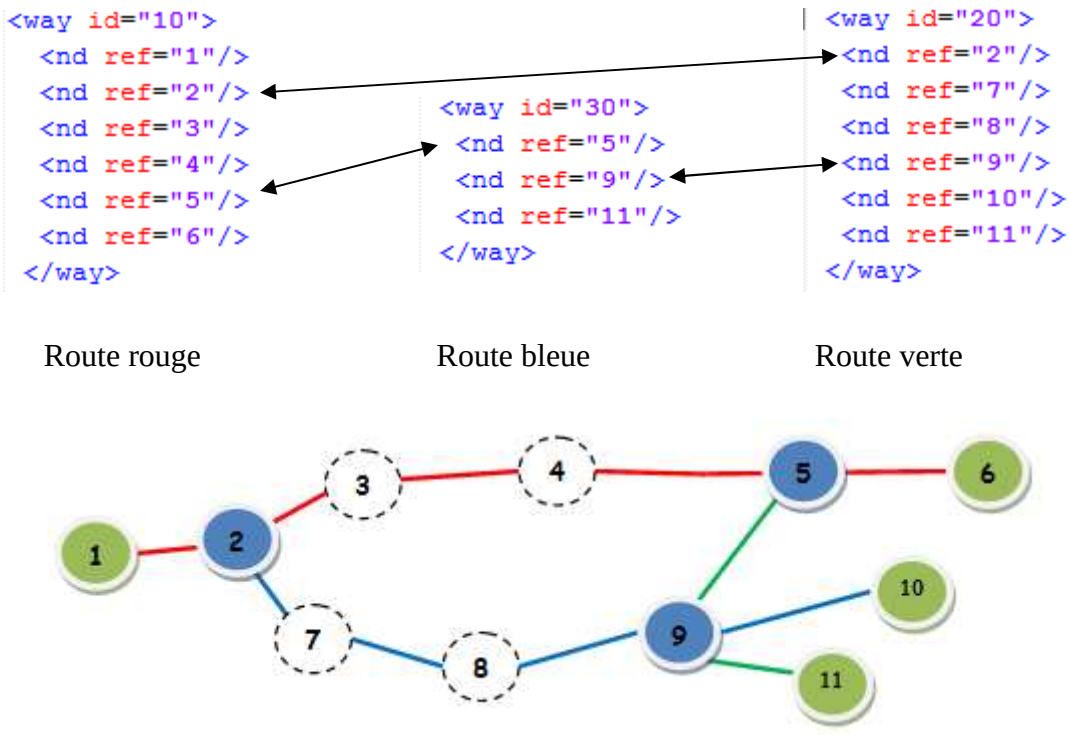


Figure 3.1 Représentation de 3 routes.

[22] **Intersection :** Un nœud est une intersection s'il appartient à au moins deux routes (ways).
[23] Un neoud isolé est le premier ou le dernier nœud d'une route

La première idée de représentation que nous avons eue est celle d'un graphe contenant des nœuds et des arcs entre chaque deux nœud, à partir de la figure 3.1 le graphe en question serait défini ainsi :

G(X,A) avec X ensemble de sommets et A l'ensemble des arcs:

X = {1,2,3,4,5,6,7,8,9,10,11}.

A = {(1,2) ; (2,3) ; (3,4) ; (4,5) ; (5,6) ; (2,7) ; (7,8) ; (8,9) ; (9,5) ; (9,10) ; (9,11)}.

L'inconvénient majeur de cette représentation est le nombre d'arcs et de nœuds à traiter. En effet la recherche d'un itinéraire se fait en parcourant successivement les nœuds du graphe, un calcul détermine à chaque fois le nœud suivant. Une seule route sur une carte OSM peut prendre jusqu'à 2000 nœuds, une carte normale peut contenir des milliers de routes, le nombre de traitements à effectuer deviens très grand et risque de ralentir considérablement la recherche. L'optimisation passe par l'identification des possibles calculs inutiles.En observant bien le graphe, il est évident que les arcs se trouvant entre deux intersections (nœud bleu) peuvent être considérés comme un seul et unique arc[24]. Pour bien illustrer cette idée voici un exemple déroulé sur le graphe de la figure 3.1, avec pour objectif de chercher un chemin allant du nœud 1 au nœud 6 :

Commençons par le nœud 1, le seul nœud suivant est le 2. Une fois arrivé au nœud 2 un choix s'offre alors à nous, soit le nœud 3 ou le nœud 7, choisissons le nœud 3, puis le nœud 4, ce n'est qu'au nœud 5 qu'un choix est possible entre le nœud 6 et le nœud 9. Les seuls cas où il a fallu « réfléchir » sont les nœuds 2 et 5 qui sont des intersections, les transitions par les nœuds 3 et 4 sont inutiles (un seul choix possible), dès qu'on passe du nœud 2 au nœud 3, on continue dans la même direction jusqu'à arriver à une intersection qui mène au moins à deux arcs différents, cas dans lequel un calcul est indispensable. Le seul cas où il est possible de prendre en considération un nœud entre deux intersection successives est que ce nœud soit un point de départ ou un point d'arrivé.

Le gain de temps grâce à cette solution par rapport à la première est indéniable, surtout que certaines routes contiennent des dizaines voir des centaines de nœuds[25].

[24] **Un arc du graphe** aura alors comme extrémités uniquement des intersections ou des nœuds isolés (nœuds en vert dans la figure ci-dessous).

[25] Un way peut avoir jusqu'à 2000 nœuds.

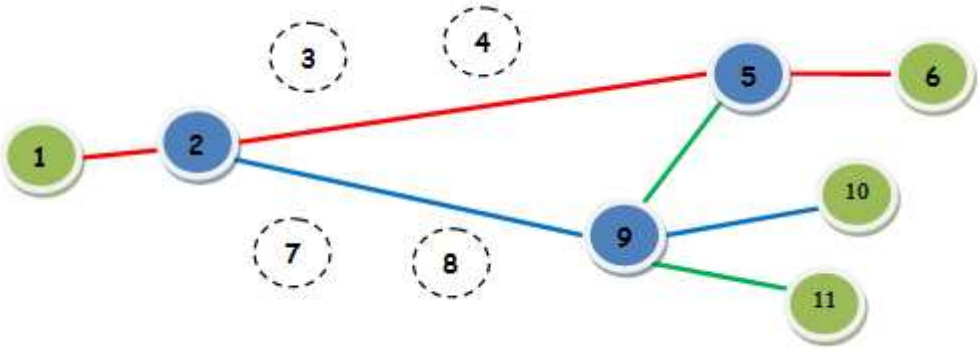


Figure 3.2 Représentation des routes avec intersections.

L'inconvénient avec cette représentation est que certains nœuds assurant la précision de la trajectoire de la route sont ignorés. L'itinéraire, une fois trouvé, est retourné sous forme d'une succession uniquement d'intersections, le tracé de la route (courbure) est alors remplacé par une droite reliant les deux intersections.

Il faut pouvoir récupérer à la fin du calcul les nœuds qui se trouvent entre les deux intersections pour avoir un tracé correct, et garder l'ordre des nœuds, car le tracé doit se faire d'une manière séquentielle en partant d'une des deux intersections, si les nœuds sont présentés dans un désordre on peut avoir un tracé complètement erroné comme sur la figure 3.3.

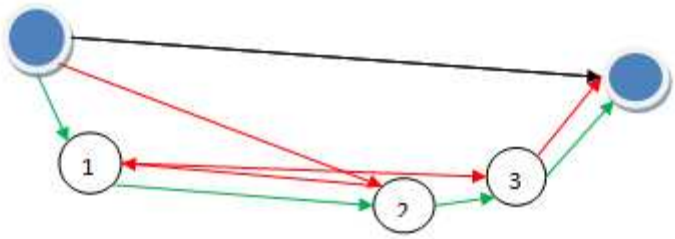


Figure 3.3 Importance de l'ordre des nœuds.

Pour garder une trace des nœuds chaque nœud qui a été ignoré est associé à un arc. De cette façon, lors du tracé d'un arc il suffit de récupérer tout les nœuds associés à ce dernier.

Remarque : Un nœud isolé est pris en compte dans le graphe, car s'il est ignoré les nœuds se trouvant entre lui et la prochaine intersection seront impossibles à retrouver lors du tracé.

Le graphe des routes est stocké dans une base de données, qui facilite bon nombre d'opérations (récupération des intersections, enregistrement des arcs, enregistrement des nœuds et leur association aux arcs, …).

La prochaine partie est consacrée à la présentation de la base de données permettant principalement d'assurer d'une manière optimale la recherche d'itinéraire.

b) Conception de la base de données :

Dans cette partie nous décrivons la manière de concevoir la structure de la base avec un model *Entité-Association* pour optimiser l'implémentation de la base de données.

b.1) Dictionnaire de données :

Attribut	Type	Désignation
Id_Noeud	Entier	Identifiant d'un nœud
Latitude	Double	Latitude d'un nœud
Longitude	Double	Longitude D'un nœud
Rang	Int	Position du nœud dans un arc s'il en fait partie
Nom	String	Le nom d'un nœud s'il en possède un
Amenity	String	Type d'un lieu (cafétéria, restaurant, …etc)
Arc_Id	Entier	Identifaint d'un arc
Longueur	Double	Longueur d'un arc en KM
OneWay	Boolean	Indique si un arc est à double sens ou à sens unique
Way_Id	Entier	Identifiant d'une route, nécessaire lors de la détermination des nœuds d'un arc

b.2) Modèle Etntité Association :

Les entités que nous décrivons sont les arcs et les nœuds, qui modélisent respectivement les tables Arc et Nœud au niveau données. Tandis que l'association est la relation entre ces deux entités.

Le modèle *Entité-Association* de notre domaine d'étude est décrit par la figure (3.4).

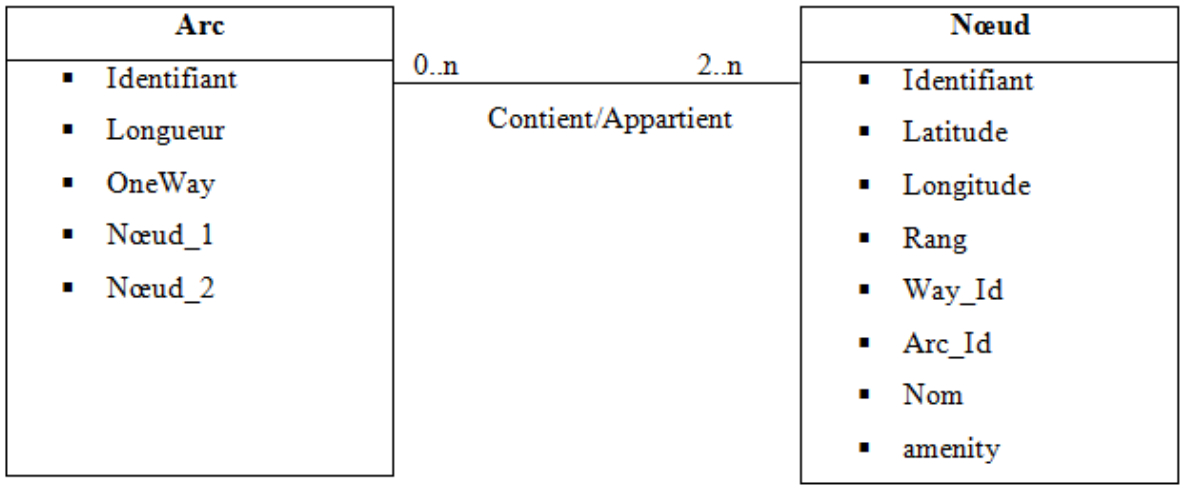


Figure 3.4 modèle Entité-Association décrivant la base de données.

Description textuelle :

- TABLE Noeud :

Cette table contient tous les nœuds de la carte, y compris ceux qui n'appartiennent pas aux routes, tout simplement car durant la recherche d'un itinéraire, il faut pouvoir rechercher certains endroits précis que l'utilisateur souhaite rejoindre, cafétéria, restaurant, cinéma, hôpital …

- Id_Noeud : Identifiant unique pour chaque nœud
- Longitude et Latitude : Coordonnées GPS du nœud afin de le situer sur la carte.
- Way_ID : Identifiant de la route à laquelle appartient le nœud, ce champs prend la valeur de -1 si le nœud ne fait partie d'aucune route.
- Arc_ID : Identifiant de l'arc auquel appartient le nœud, un arc est délimité par deux intersections ou nœuds isolés, c'est ce champ qui nous permet de garder une trace des nœuds ignorés lors du calcul d'itinéraire. Ce champ prend la valeur de -1 si le nœud ne fait partie d'aucune route, il prend la valeur 0 si ce nœud est une extrémité d'un arc.
- Rang : Donne l'ordre d'apparition du nœud sur l'arc, comme expliqué précédemment ce champ est primordial pour un tracé correct.

- Nom : contient le nom d'un endroit précis ou d'une route, … Ce champ est utile lors d'une recherche d'un lieu précis.
- Amenity : contient le type d'un lieu, nécessaire lors de la recherche d'un endroit public (cafétéria, restaurant, …etc)

- TABLE ARC :
  - ID_Arc : Identifiant unique pour chaque Arc.
  - Neoud_1 : Identifiant du nœud de la première extrémité de l'arc.
  - Neoud_2 : Identifiant du nœud de la deuxième extrémité de l'arc.
  - Longueur : La longueur du chemin réel entre les deux intersections en Kilomètres, ce n'est pas la distance entre les deux extrémités de l'arc.
  - OneWay : C'est une valeur booléenne indiquant si l'arc est en sens unique, ce champ est important car il permet lors de la recherche d'un itinéraire de ne pas prendre un arc dans le mauvais sens.
- ASSOCIATION :
  - Appartient : un nœud peut appartenir à 0 ou N arcs. Ces cardinalité sont décrites par 0..N.
  - Contient : un arc contient au minimum 2 nœud (départ, arrivé) mais peut se voir associé jusqu'à N nœuds. Ces cardinalité sont décrites par 2..N.

Une fois l'idée de la représentation des données du graphe fixée, le schéma de la base de données décrit, il nous a fallu concevoir un programme spécialement pour la génération de la base à partir des données OSM. La conception de ce programme est détaillée dans la partie suivante.

c) Conception d'un générateur de base de données : OSM2GRAPHE

Nous allons présenter le programme de bureau que nous avons implémenté, afin de générer la base de données. C'est un programme écrit en langage JAVA qui prend en entrée un fichier OSM et qui génère un fichier de base de données. Il inclut un module de test qui permet de valider les données générées, en représentant à partir de la base de données le graphe des routes, et de tester l'algorithme de recherche du plus court chemin entre deux points. Le schéma général de l'application est présenté dans la figure (3.5).

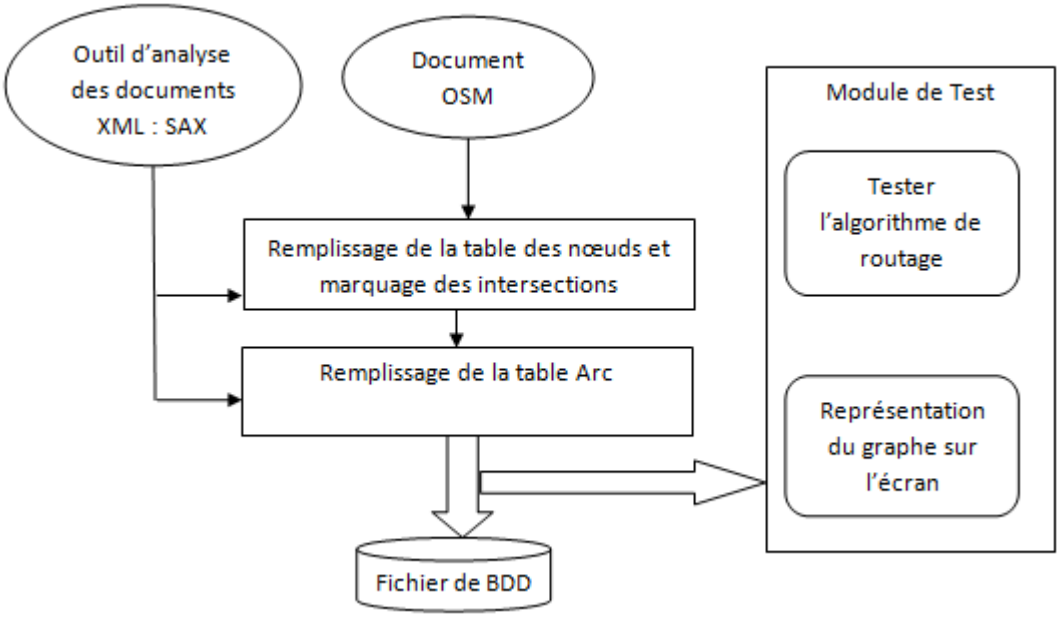


Figure 3. 5 Schéma général de l'application tell qu'elle à été implémentée.

Voici une description des différents composants de notre programme :

- **Fichier OSM :** Carte d'une ville donnée sous format d'un fichier OSM.
- **Parseur SAX[26] :** Analyseur de fichiers XML.
- **Remplissage de la table des nœuds :**

Remarque : *Dans un fichier OSM, les premières balises sont les nœuds, suivi des ways.*

Dans cette étape un parcours complet du fichier est effectué[27] en utilisant l'outil SAX. La tâche la plus importante de cette partie est de détecter les nœuds qui sont des extrémités

[26] Voir Annexes : ii. Analyseur documents XML SAX
[27] Le parcours se fait au fur et à mesure de haut en bas, un traitement est effectué à chaque passage dans une balise.

d'un arc (intersections ou nœud isolé), l'identifiant des routes qui est associé à chaque nœud (way_id) permet de repérer les intersections (un nœud qui appartient à plus d'une route).

Algorithme :

```
Parcourir le fichier {
Si Balise == NŒUD alors {
        Récupérer identifiant, longitude, latitude
        Enregistrer nœud dans la table NŒUD
        }
Sinon Si Balise == WAY alors {
        Récupérer identifiants des nœuds constituant ce WAY.
        Pour chaque nœud appartenant à ce way faire
                Si nœud a déjà un WAY_ID alors
                        Mettre à 0 ARC_ID du nœud car c'est une intersection et donc une
                                extrémité d'un arc
                Sinon
                        Mettre à jour le champ WAY_ID de chaque nœud
                finsi
        fait
}finsi
finsi
}//Fin du parcours du fichier
```

Remarque : Dans cette étape, les nœuds du graphe routier défini précédemment sont insérés, ils sont différenciables des autres nœuds grâce au champ arc_id qui est égale à 0.

La prochaine étape utilisera la table Nœud afin de créer les arcs du graphe routier.

- **Remplissage de la table Arc :**

Cette étape consiste à découper chaque route (way) en plusieurs arcs, le début et la fin de chaque route est considérée comme une extrémité (nœud isolé) d'un arc.

Algorithme :

```
Id_arc = 1 //Variable indiquant l'id des arcs
Rang = 1 //Variable indiquant l'ordre des nœuds associés à un arc
Parcourir le fichier {
Si Balise == WAY alors {
        Sens_way = récupérer le sens du way (sens_unique ou double sens)
        Nœud 1 = le premier nœud du way qui sera la première extrémité de l'arc
        Nœud_précédent = Nœud 1
        Mettre à jour champ Arc_ID de Nœud 1 (arc_id = 0), pour indiquer que
                Nœud_1 est une  extrémité d'un arc

        Longueur = 0. //Initialiser le compteur de longueur d'un arc
        Tant qu'il y'a encore des nœuds dans le way Faire {
                Longueur = Longueur + calculerDistance(Nœud_précédent, Nœud_actuel)

                Si Neoud_actuel est une intersection Alors{
                        Enregistrer Arc (Noeud1, Nœud_actuel) à la base de données avec
                        id_arc, la longueur ainsi que le sens (sens_way)
                        //Préparer la création du prochain arc
                        Id_arc  = id_arc +1 ; //incrémenter l'identifiant de l'arc
                        Nœud 1 = Nœud actuel // Le prochain arc commence à partir du dernier
                                nœud de l'arc actuel
                        Longueur = 0 // réinitialiser la longueur
                        Rang = 1 ;//réinitialiser le rang
                }
                Sinon {
                        Mettre à jour Arc_ID et le rang de nœud
                        Rang = rang +1 //Incrémenter le rang
                }//Finsi
        Nœud précédent = Nœud actuel
        Nœud actuel = Prochain_noeud
        }//Fait
        //Le dernier nœud est une extrémité de l'arc, s'il n'a pas été enregistré alors le faire ici
        Si dernier Nœud n'est pas une intersection alors
                Enregistrer Arc (Noeud1, dernier noeud) à la base de données avec la
                                        longueur ainsi que sens_unique
        finsi
}//fin si
}//Fin du parcours
```

Remarque : La longueur d'un arc est égale à la longueur du chemin reliant les deux extrémités en passant par les nœuds intermédiaires.

Ces deux étapes nous garantissent une représentation optimale de toute la carte, avec comme avantage principal la possibilité d'appliquer facilement un algorithme de routage.

- **Module de test :**

Le module de test est très utile pour valider le bon fonctionnement de l'opération de génération de la base de données. Ce module est lancé à la fin du processus de création de la base, une représentation graphique du graphe routier est affiché, et donne la possibilité de choisir deux points afin de tester l'algorithme de routage. Une fois l'algorithme validé il peut être utilisé directement dans l'application Android, étant donné qu'il est écrit en JAVA.

Un aperçu d'OSM2GRAPH est présenté dans le chapitre 4. L'algorithme de routage utilisé est détaillé dans la partie suivante,

d) Algorithme de routage :

Le processus de la recherche d'itinéraire n'est autre que l'exécution d'un algorithme de recherche d'un chemin le plus court entre deux points donnés d'un graphe. Dans notre application nous nous sommes inspirés de l'algorithme de Dijkstra[28] qui est parmi les algorithmes les plus utilisés, il a l'avantage d'être facile à implémenter et de toujours donner le chemin le plus court possible. L'algorithme utilisé avec les données du graphe routier est présenté.

Entrées : Point de départ, point d'arrivée

Sorties : - Chemin reliant le point de départ et d'arrivé, constitué de points géographique.
- longueur de chemin.

[28] Voir Annexes : i Algorithme de Dijkstra

Algorithme :

- **Initialisation :**

Connexion_BDD

noeud_départ = Rechercher dans la base de données le point le plus proche du point de départ.

noeud_arrivé = Rechercher dans la base de données le point le plus proche du point d'arrivé.

Créer une liste de nœuds qui contient les nœuds des arcs (arc_id = 0)

Initialiser les poids des nœuds :

- Nœud_départ.poids = 0,
- Tout les nœuds de la liste = infini

- **Mise à jour des poids des noeuds**

Répéter

- Nœud_minimal = Parcourir la liste et retourner le nœud ayant le poids minimal.
- Retirer nœud_minimal de la liste des nœuds // Indiquer que le chemin le plus court vers ce nœud est connu
- Sélectionner tous les nœuds voisins de nœud_minimal, sauf ceux qui ne respectent pas le sens unique (si l'arc (nœud → nœud_minimal) est en sens unique alors ne pas prendre nœud).
- Pour chaque extrémité récupérée faire :
  - Récupérer la longueur de l'arc
  - Si (poids nœud > poids nœud_min + longueur arc) alors
    - Poids nœud = poids nœud_min + longueur arc
    - Nœud.précédent = nœud_min

Jusqu'à ce que (noeud_arrivé = nœud_minimal) ou (liste des nœuds = vide)

- **Reconstruire le chemin : on remonte le chemin du point d'arrivé au point de départ**

chemin = liste vide

noeud = nœud_arrivé

Tant qu'il y'a encore un nœud précédent faire

- Insérer nœud dans chemin
- nœud = nœud.précédent

Fermer_BDD

Retourner (inverse(chemin), nœud_arrive.poids) ;

III.3) Architecture générale de l'application :

En plus des méthodes de représentation d'une carte et de recherche d'itinéraire, l'intégration d'un module GPS qui localise l'individu donne une architecture complète qui permet de réaliser une application de navigation GPS, la figure 3.6 illustre une telle architecture.

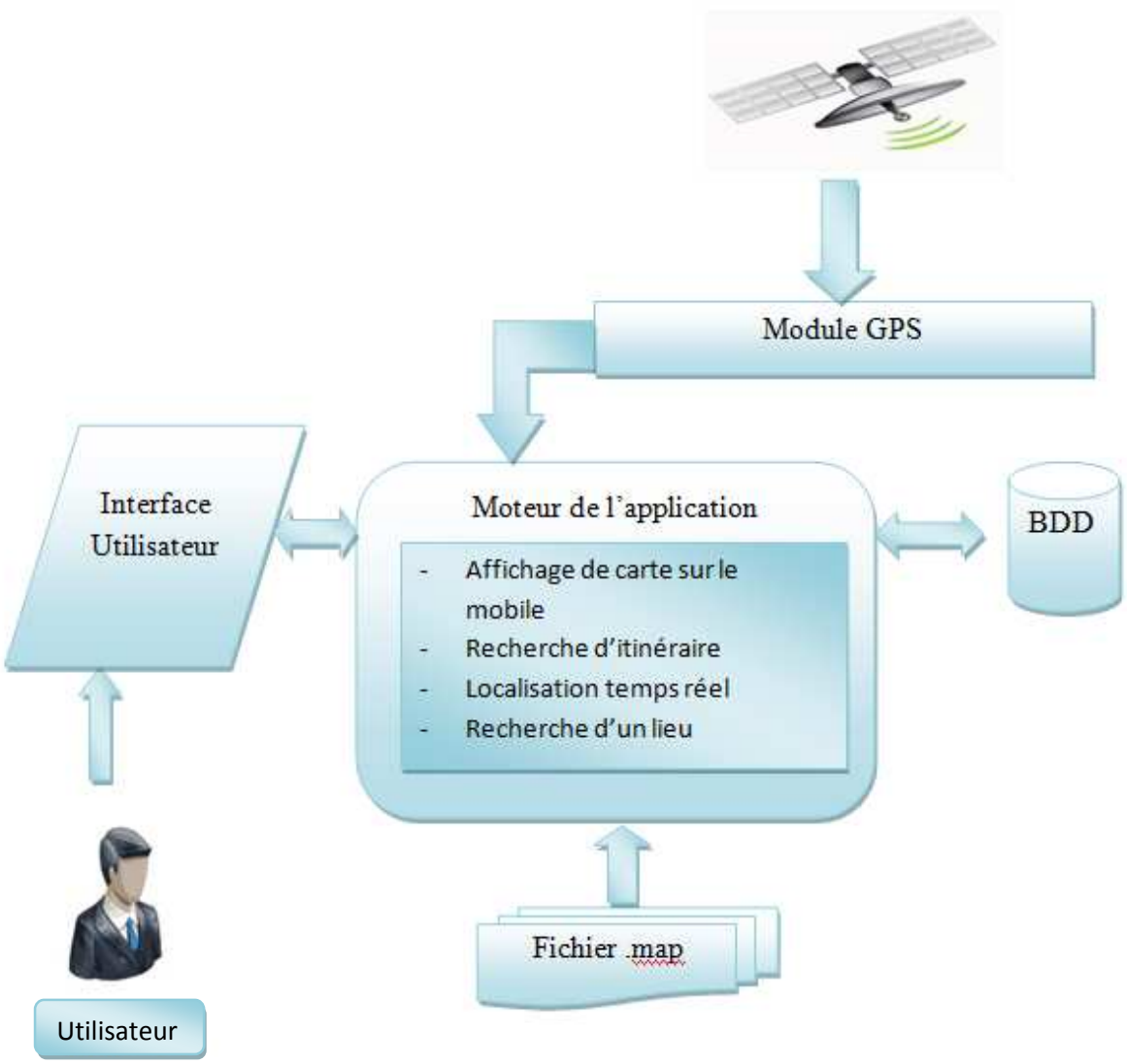


Figure 3.6 Architecture Générale de l'application.

L'architecture générale étant dégagé, passons à la conception des principales classes constituant le moteur de notre application.

III.4) classes principales de l'application :

Alors que l'architecture de la figure 3.6 montre les interactions entre les composants de l'application. Le diagramme de la figure 3.7 décrit les différentes classes constituant son moteur.

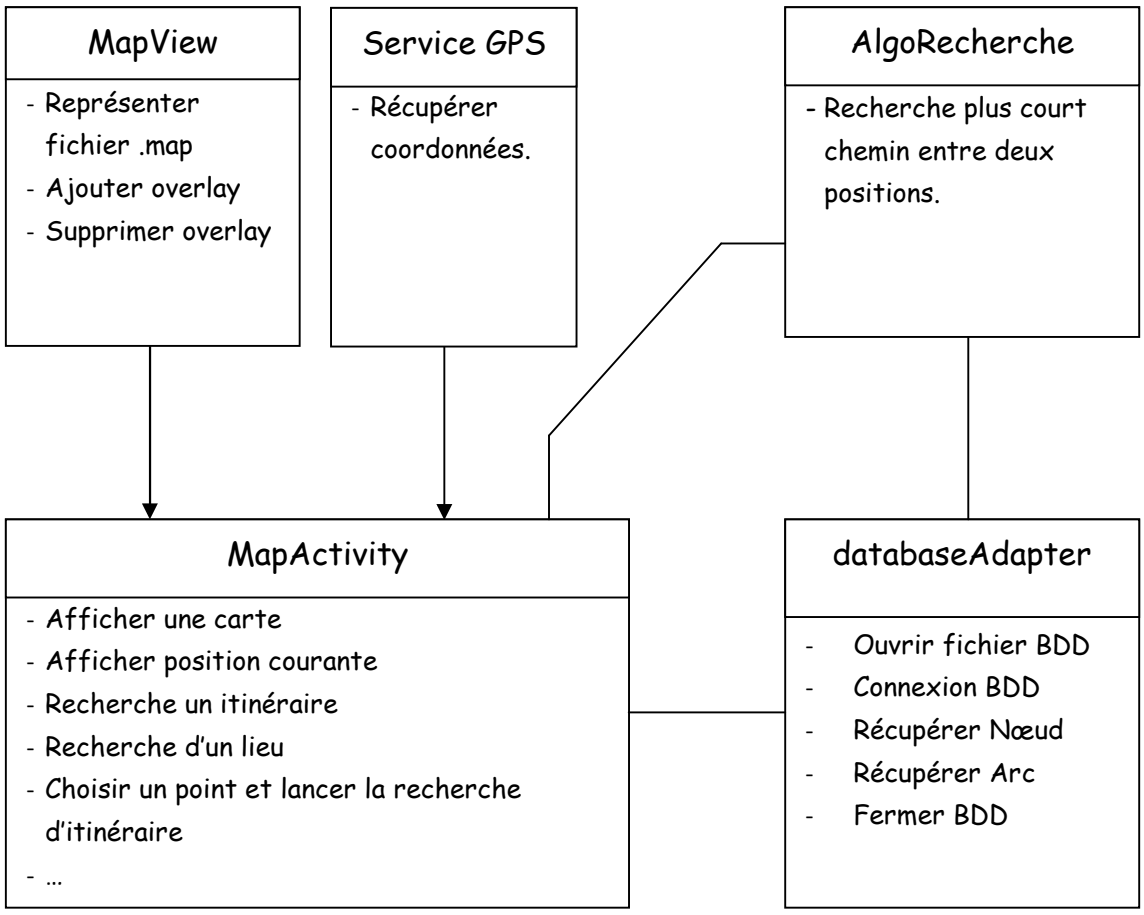


Figure 3.7 Principales classes de l'application

Voici une brève description des rôles de chaque classe :

MapView : classe qui se charge de la création de la carte à partir d'un fichier .MAP et qui offre les méthodes nécessaires à sa manipulation (tracer un chemin par exemple). Cette classe est utilisée par la classe MapActivity, pour afficher la carte ainsi que les différents résultats voulus par l'utilisateur (une route entre deux points, la position actuel, un lieu précis, …etc).

Service GPS : service de localisation, elle s'occupe principalement de fournir en continu les coordonnés GPS (latitude, longitude) à la classe MapActivity.

AlgoRecherche : classe qui implémente l'algorithme de recherche du meilleur chemin entre deux points.

databaseAdapter : la classe databaseAdapter représente une connexion avec la base de données. Les propriétés de cette classe définissent les méthodes permettant d'ouvrir ou de fermer une connexion, d'exécuter des commandes, de contrôler les transactions. Elle permet entre autre de faciliter à MapActivity et AlgoRecherche l'accès aux données de la base.

MapActivity : représente l'activité principale, elle utilise les autres classes décrites au dessus afin de réaliser les taches du moteur de l'application.

III.5) Conclusion :

Dans ce chapitre, nous avons vu dans un premier temps, comment représenter un plan d'une ville sur un Smartphone à partir des données OSM. Puis nous avons conçu une méthode qui permet de chercher un itinéraire entre deux points d'une carte. En plus de la localisation, ces deux solutions offrent l'environnement adéquat pour réaliser notre application.

Dans le prochain chapitre nous présentons la réalisation de l'application, son architecture détaillée, ainsi que les outils et bibliothèques utilisés.

CHAPITRE 4
Réalisation de l'application

IV.1) Introduction

Après avoir présenté la partie conception, nous entamons, dans ce chapitre, la réalisation du projet.

Cette partie met en évidence les différents choix techniques (choix du langage, IDE, bibliothèque logicielle, outils matériels, …) qui entrent en jeux pour réaliser l'implémentation des solutions proposées. Afin de valider notre application, nous présentons des exemples en situation réelle.

IV.2) Environnement et outils de développement :

IV.2.1) Partie matérielle :

Elle est constituée d'un ensemble de ressources destinées à la mise en marche de notre application. Ainsi, nous citons :

- Un Pc (fixe ou portable) ayant Windows comme système d'exploitation.
- Un téléphone intelligent (Smartphone) : Pour notre projet, nous avons utilisé un Sony Erikson Xperia avec un processeur 1 GHZ et une RAM de 1Go de RAM, et portant le système Android (4.0.1) comme système d'exploitation.

Figure 4.1 Sony Erikson Xperia.

IV.2.2) Prérequis Logiciels :

➢ **Langage de programmation Java** : [12]

Java est un langage de programmation et une plate-forme informatique créée par Sun Microsystems en 1995. Il s'agit de la technologie sous-jacente qui permet l'exécution de programmes dernier cri, notamment des utilitaires, des jeux et des applications professionnelles. Java est utilisé sur plus de 850 millions d'ordinateurs de bureau et un milliard de périphériques dans le monde, dont des périphériques mobiles et des systèmes de diffusion télévisuelle.

➢ **IDE Eclipse :**

Eclipse est un environnement de développement intégré libre extensible, universel et polyvalent, permettant de créer des projets de développement mettant en œuvre n'importe quel langage de programmation.

Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions. La spécificité d'Eclipse IDE vient du fait de son architecture totalement développée autour de la notion de plugin : toutes les fonctionnalités de cet atelier logiciel sont développées en tant que plug-in.

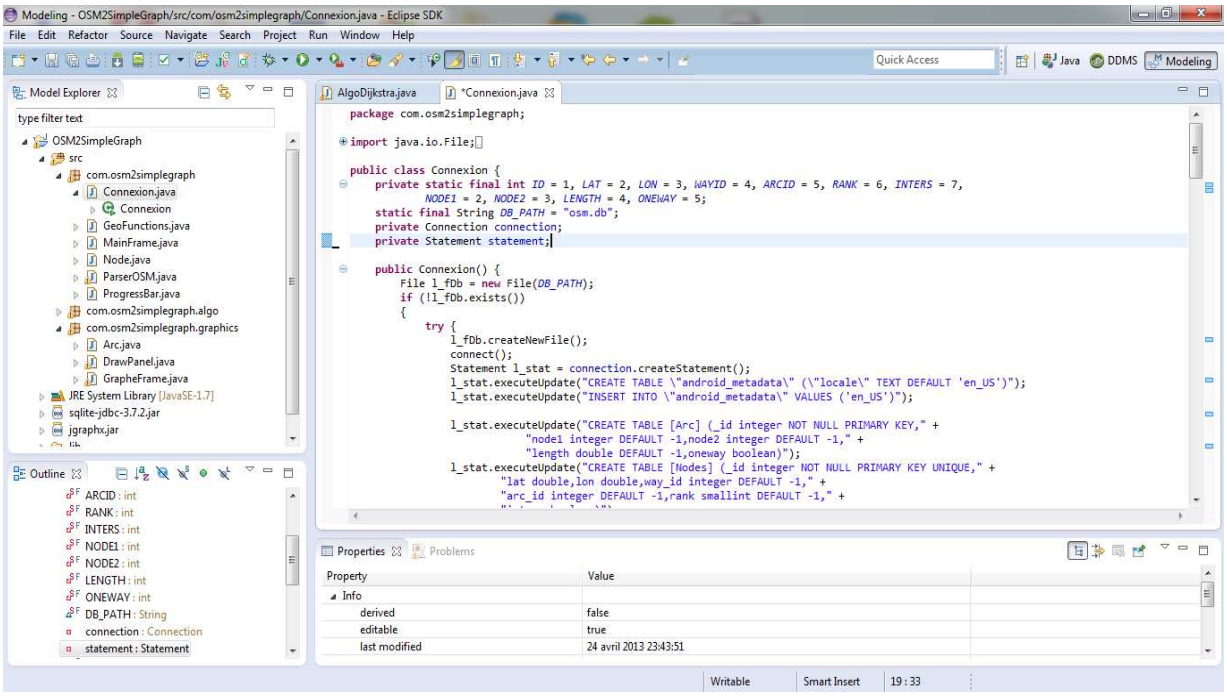


Figure 4.2 Interface d'Eclipse.

Dans le cadre de notre projet, nous avons utilisé la version Eclipse Juno (4.2.1).

IV.2.3) Prise en main de l'environnement Android :

La première étape de notre travail avec l'environnement Android a été d'appréhender le SDK, l'architecture et le développement d'une application ainsi que son déploiement sur un terminal embarquant Android.

IV.2.3.1) Présentation du SDK :

Google à mis en place un grand nombre d'outils pour aider les développeurs Android.
La première chose à visiter est le portail des développeurs Android, mis en place par Google. [13]

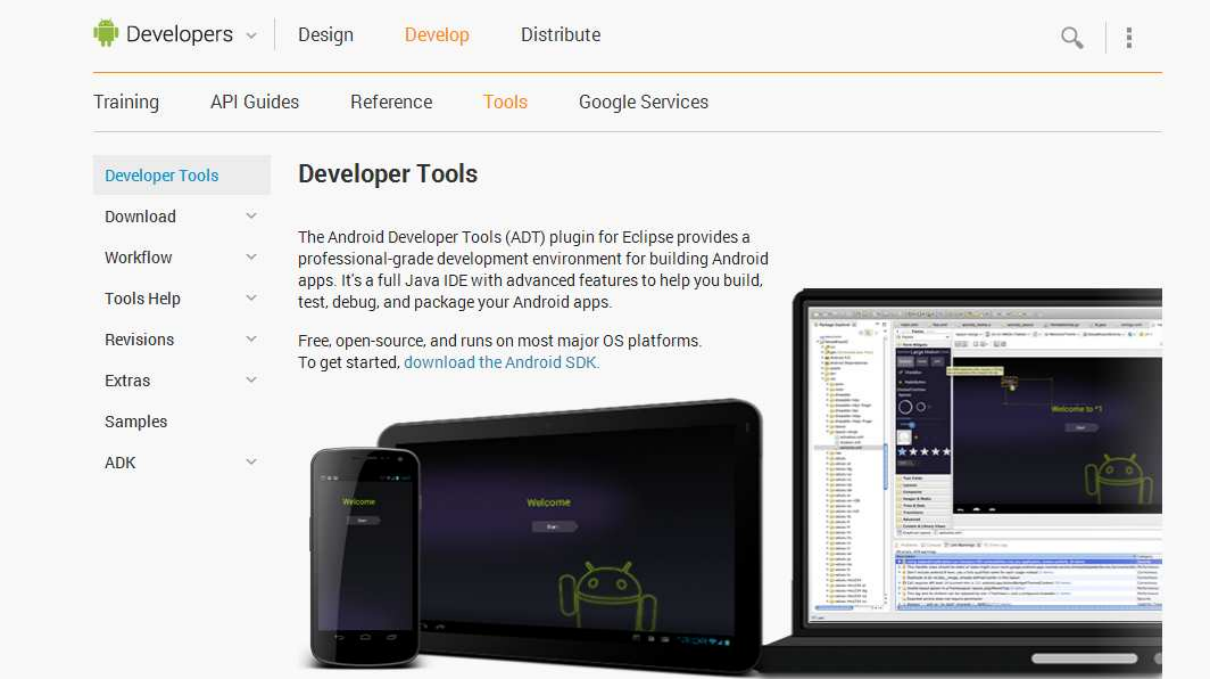


Figure 4.3 Portail des développeurs Android.

Très complet, ce site présente Android, explique comment installer et utiliser les différents outils (SDK, NDK etc.), propose un ensemble de tutoriels et articles concernant le développement d'applications Android, expose la référence de l'API Android ainsi que les actualités liées à Android. Le tout est très bien fait et permet de rapidement être confortable vis-à-vis du développement sur Android.

IV.2.3.2) 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 Eclipse combiné au plugin d'outils de développement d'Android (ADT).

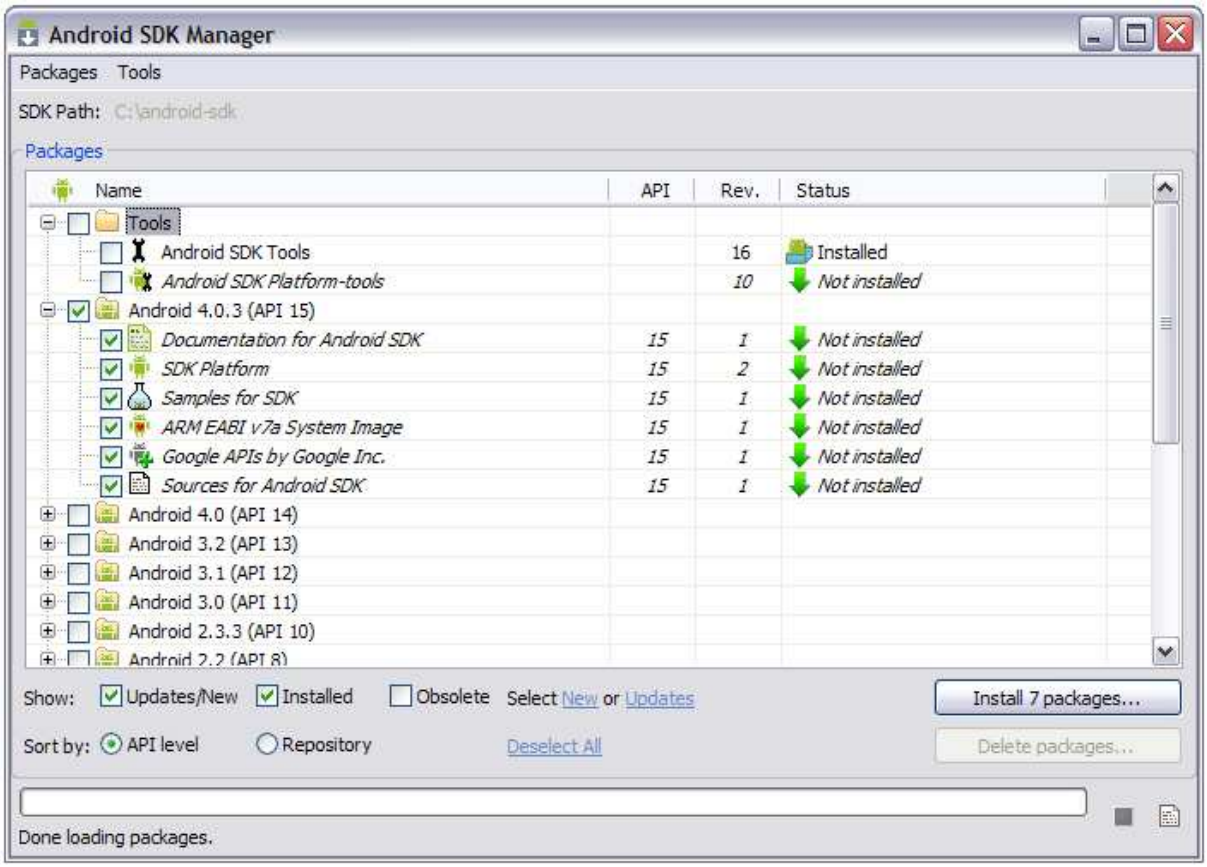


Figure 4.4 Interface d'installation du SDK Android.

IV.2.3.3) ADT pour Eclipse : [14]

Eclipse est l'Environnement de Développement Intégré (ou IDE) le plus largement utilisé pour la programmation Java; très performant, il est de plus gratuit et open source.

Google a donc tout naturellement conçu un plugin[29] pour Eclipse.

Android Development Tools (ADT) est un plugin pour l'IDE Eclipse, qui offre un environnement puissant, afin de créer des applications Android.

ADT étend les capacités d'Eclipse pour permettre de rapidement mettre en place de nouveaux projets Android, créez une interface utilisateur de l'application, déboguer les applications en utilisant les outils SDK Android, et même exporter des .Apk signé (ou non signée) afin de distribuer l'application.

IV.2.3.4) Emulateur :

Nous l'avons évoqué plus haut, le SDK propose un émulateur Android. Il permet de lancer sur la machine du développeur un terminal virtuel représentant à l'écran un téléphone embarquant Android.

C'est bien évidemment un outil indispensable pour le développement mobile. A chaque version d'Android est associée une version de l'émulateur, permettant au développeur de voir exactement à quoi ressemblera son application sur un matériel réel.

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

Figure 4.5 Interface de l'émulateur Android.

[29] Un plugin est un module qui complète un logiciel hôte pour lui apporter de nouvelles fonctionnalités

IV.2.4) API OpenStreet Map : [15]

OpenStreetMap fournit une API en ligne qui permet de télécharger des données d'une zone précise de la planète au format XML (OSM).

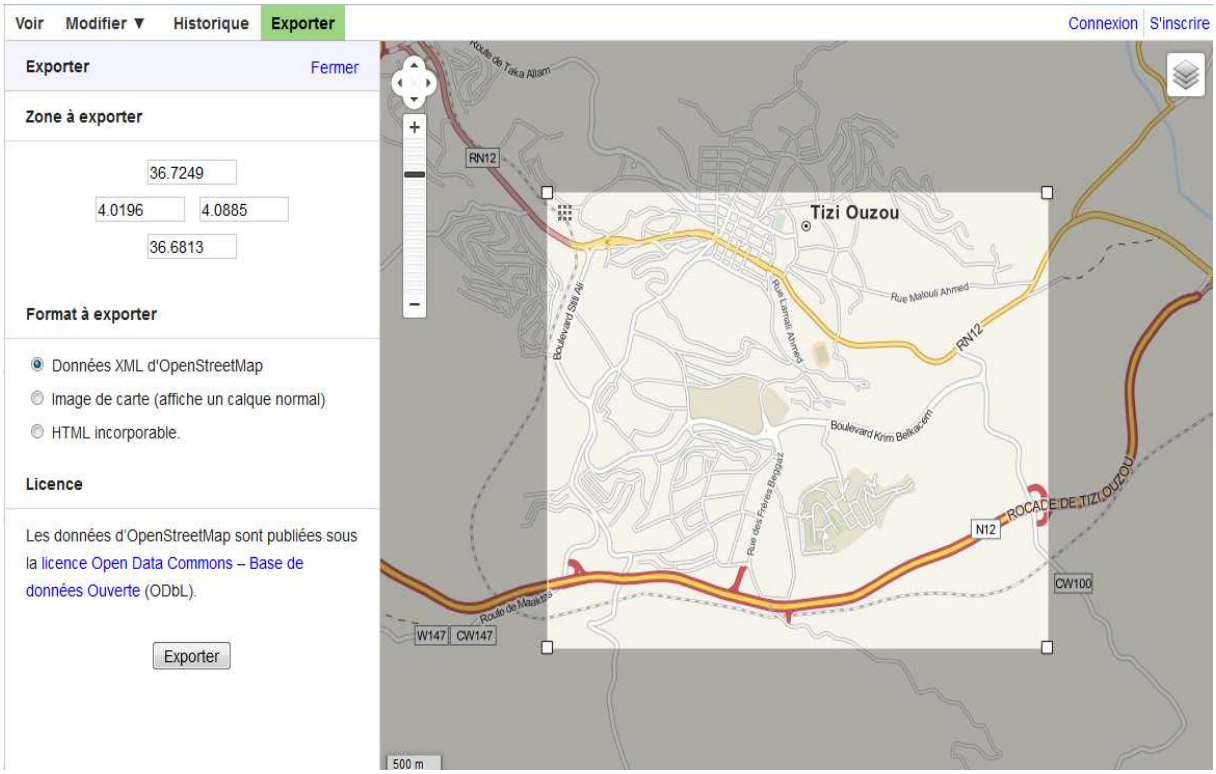


Figure 4.6 Interface de l'api OpenStreetMap en ligne.

Il suffit de sélectionner une zone délimité par quatre mesures, dans notre cas:

Latitude_min = 36.6813, Longitude_min = 4.0196,

Latitude_max = 36.7249, Longitude_max = 4.0885.

Puis de cliquer sur le bouton exporter pour obtenir un fichier .osm de la zone sélectionné.

IV.2.5) SQLite :

SQLite est une bibliothèque écrite en C qui propose un moteur de base de données relationnelle accessible par le langage SQL.

Figure 4.7 Sigle SQLite.

Contrairement aux serveurs de bases de données traditionnels, comme MySQL ou PostgreSQL, sa particularité est de ne pas reproduire le schéma habituel client-serveur mais d'être directement intégrée aux programmes. L'intégralité de la base de données (déclarations, tables,index et données) est stockée dans un fichier indépendant de la plateforme.

D. Richard Hipp, le créateur de SQLite, a choisi de mettre cette bibliothèque ainsi que son code source dans le domaine public, ce qui permet son utilisation sans restriction aussi bien dans les projets open source que dans les projets propriétaires.

SQLite est le moteur de base de données le plus distribué au monde, elle est également très populaire sur les systèmes embarqués, notamment sur la plupart des smartphones modernes : l'iPhone ainsi que les systèmes d'exploitation mobiles Symbian et Android l'utilisent comme base de données embarquée. Au total, on peut dénombrer plus d'un milliard de copies connues et déclarées de la bibliothèque. [16]

IV.2.6) Librairie MapsForge: [17]

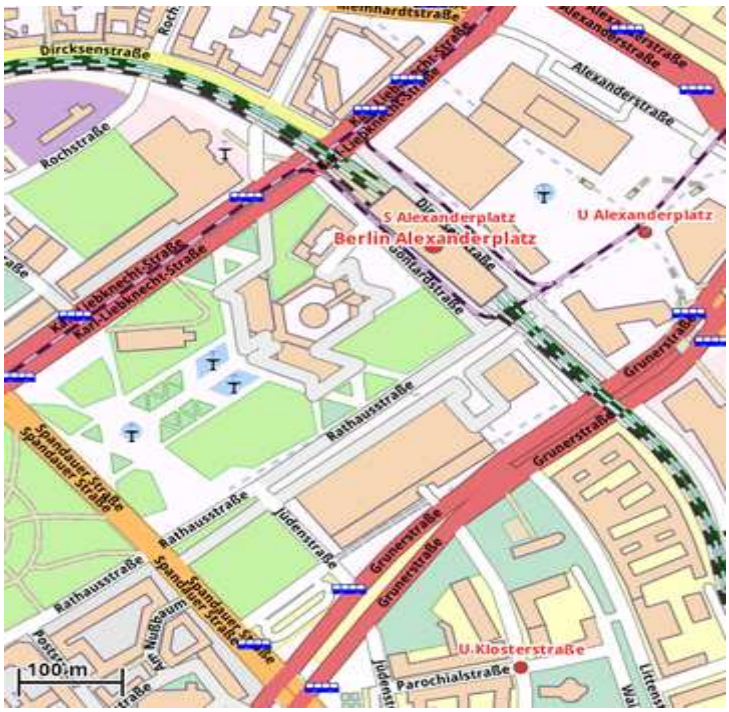


Figure 4.8 Aperçu du rendu d'une carte faite avec MapsForge

mapsforge fournit une bibliothèque de cartographie hors ligne gratuite pour Android. Avec une API simple d'utilisation il est possible de construire une application à base de carte avec moins de 30 lignes de code. Les superpositions (POI et polygones arbitraires) sont également supportées.

L'objectif global du projet mapsforge est de fournir une boîte à outils libre et ouverte qui permet de créer facilement de nouvelles applications fondées sur OpenStreetMap. Les outils fournis et les API comprennent des solutions de représentation de cartes pouvant être stockées sans connexion à internet, la planification des itinéraires et la navigation, POI et plus encore.

Le projet mapsforge a été lancé en 2008 à l'Institut informatique de la l'université de Berlin. Actuellement, environ 10 étudiants et le personnel scientifique sont impliqués dans le projet.

Voici quelques fonctionnalités de mapsforge utilisées dans notre application :

- Représentation de la carte de la ville de Tizi Ouzou.
- Tracé des itinéraires.
- Marquer une position.

IV.2.7) OSM2Graphe : Générateur de base de données :

Dans la partie conception on a présenté le système du programme qui permet de générer la base de données, ici nous montrons brièvement comment utiliser ce programme.

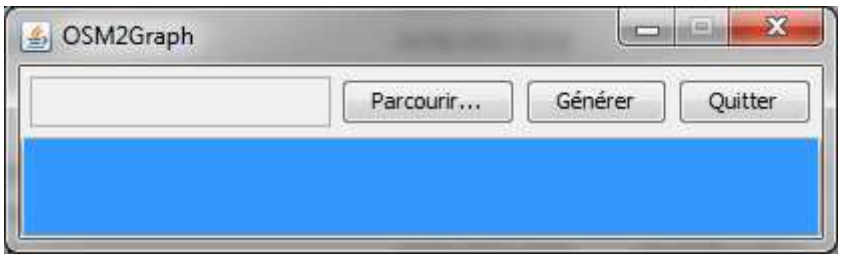


Figure 4. 9 Interface d'OSM2Graph au lancement.

Le bouton parcourir permet de sélectionner un fichier .OSM.

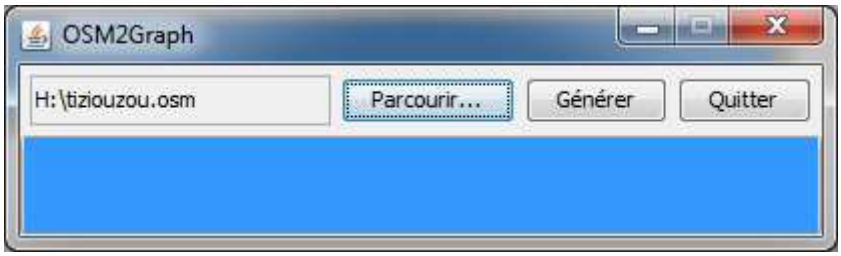


Figure 4.10 OSM2Graph, choix d'un fichier.

Une barre de chargement nous indique l'état d'avancement de l'algorithme de remplissage.

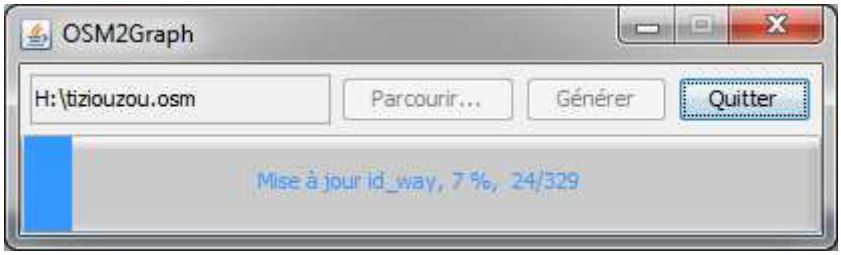


Figure 4. 11 OSM2Graph, Barre de chargement.

Un fichier de base de donnée nommé « osm.db » est créé dans le même répertoire d'exécution du programme, il est désormais possible de l'utiliser dans notre application.

A la fin du remplissage une nouvelle fenêtre s'ouvre et nous invite à dessiner le graphe des routes obtenu, ceci afin de vérifier si le processus c'est bien déroulé. La figure 4.11 montre le résultat final du programme.

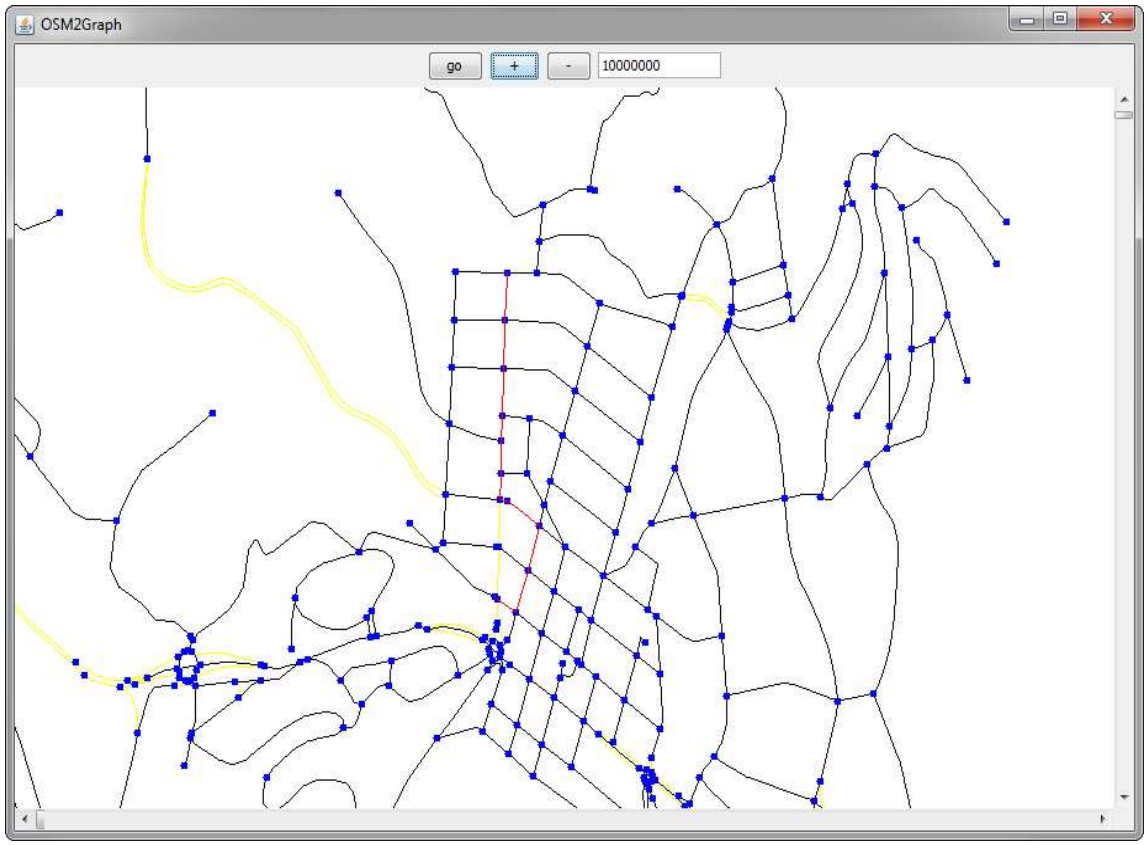


Figure 4.12 OSM2Graph Représentation du graphe.

On voit bien que le tracé est correct, les intersections sont bien représentées, les arcs à sens unique sont en jaune, et nous avons même rajouté la possibilité de tester l'algorithme de routage. Le tracé rouge est un itinéraire qui a été réalisé suite au clic sur deux points de la carte. L'algorithme de routage à contourné un sens unique, en effet au lieu de machinalement prendre la direction vers le haut, il a détecté que ce chemin était un sens unique et par conséquent il a cherché une autre solution.

IV.3) Génération des fichiers .MAP :

Le format utilisé par MapsForge pour représenter les cartes est le format .MAP présenté dans le chapitre 2. Dans ce qui suit nous expliquons comment obtenir un fichier .Map à partir d'un fichier .OSM et comment l'utiliser dans l'application.

IV.3.) Osmosis : [18]

Osmosis est une application java en ligne de commande pour la manipulation des données OSM. L'application se compose d'une série d'outils qui peuvent être chaînés pour effectuer une opération de grande envergure. Par exemple, il comporte des éléments de lecture et d'écriture de la base de données ou d'un fichier, des composants pour calculer et appliquer un ensembles de modifications aux sources de données, des composants pour le tri des données, etc. Il a été rédigé de façon à ce qu'il soit facile d'ajouter de nouvelles fonctionnalités sans réécriture des tâches courantes telles que la manipulation de fichiers ou de bases de données.

Voici quelques exemples des ces fonctionnalité :

- Générer une copie de la planète à partir de la base de données.
- Charger une copie de la planète dans la base de données.
- Créer un groupe de changement à partir de l'historique des tables de la base de données.
- Appliquer un groupe de changement à une base locale.
- Comparer deux fichiers planète et produire un groupe de changement.
- Extraire des données dans un "Bounding box"[30] ou dans un polygone.

L'un des avantages d'Osmosis est qu'il est extensible à l'aide de plugins. MapsForge fournit un plugin qui permet de générer les fichiers .MAP. Voici les étapes d'installation de Osmosis et du plugin.

IV.3.2) Configuration d'Osmosis :

- Télécharger la dernière version d'Osmosis en format .zip à l'adresse suivante :
  http://bretth.dev.openstreetmap.org/osmosis-build/osmosis-latest.zip
- Décompresser l'archive dans un dossier.
- Télécharger le plugin fourni par mapsforge à l'adresse suivante :
  https://mapsforge.googlecode.com/files/mapsforge-map-writer-0.3.0-jar-with-dependencies.jar
- Placer le fichier 'mapsforge-map-writer-0.3.0-jar-with-dependencies.jar' dans le dossier {chemin_vers_Osmosis}\lib\default\.
- Créer un fichier 'osmosis-plugins.conf' à l'aide d'un éditeur texte, et y mettre la ligne suivante :
  org.mapsforge.map.writer.osmosis.MapFileWriterPluginLoader
- Placer le fichier créé dans le dossier {chemin_vers_Osmosis}\config\

[30] Bouning Box : c'est une boite ou carré qui délimite une zone à extraire à partir d'une carte.

IV.3.3) Utilisation d'Osmosis pour générer le fichier .map :

Avant de lancer l'exécution d'osmosis il est nécessaire de placer le fichier .OSM dans le dossier où se trouve l'exécutable, à savoir {chemin_Osmosis}\bin\.

L'exécution de l'application se fait en ligne de commande, sur windows :

- Accéder à l'interpréteur de commande, aller au dossier {chemin_Osmosis}\bin\
- Exécuter la ligne suivante :

> osmosis --rx file=fichier.osm --mw file=fichier.map bbox lat_min, lon_min, lat_max, lat_min

lat_min, lon_min, lat_max, lat_min représentent les limites de la carte lors de son téléchargement via l'api OpenStreetMap, il est possible de récupérer ces valeurs en ouvrant le fichier.osm avec un éditeur.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <osm version="0.6" generator="CGImap 0.1.0" copyright="OpenStreetMap and contributors" at
3   <bounds minlat="36.6924000" minlon="4.0145000" maxlat="36.7438000" maxlon="4.0970000"/>
4   <node id="291045639" lat="36.7400589" lon="4.0152611" user="Sandervalya" uid="56658" vis
5   <node id="291933044" lat="36.7383903" lon="4.0170445" user="Sandervalya" uid="56658" vis
```

Figure 4.13 Récupérer limites d'une carte OSM.

Une fois l'exécution du programme terminée, le fichier.map est créé dans le même dossier. Il suffit d'utiliser la librairie de MapsForge dans un programme Android.

IV.3.4) Création d'une carte sur une application Android:

Voici un exemple d'une activité qui génère une carte à partir d'un fichier .MAP.

```
import android.os.Bundle;
import java.io.File;
import org.mapsforge.android.maps.MapActivity;
import org.mapsforge.android.maps.MapView;

public class HelloMapView extends MapActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        MapView mapView = new MapView(this);
        mapView.setClickable(true);
        mapView.setBuiltInZoomControls(true);
        mapView.setMapFile(new File("/sdcard/path/to/mapfile.map"));
        setContentView(mapView);
    }
}
```

IV.4) Réalisation de l'application :

IV.4.1) Architecture détaillée de l'application :

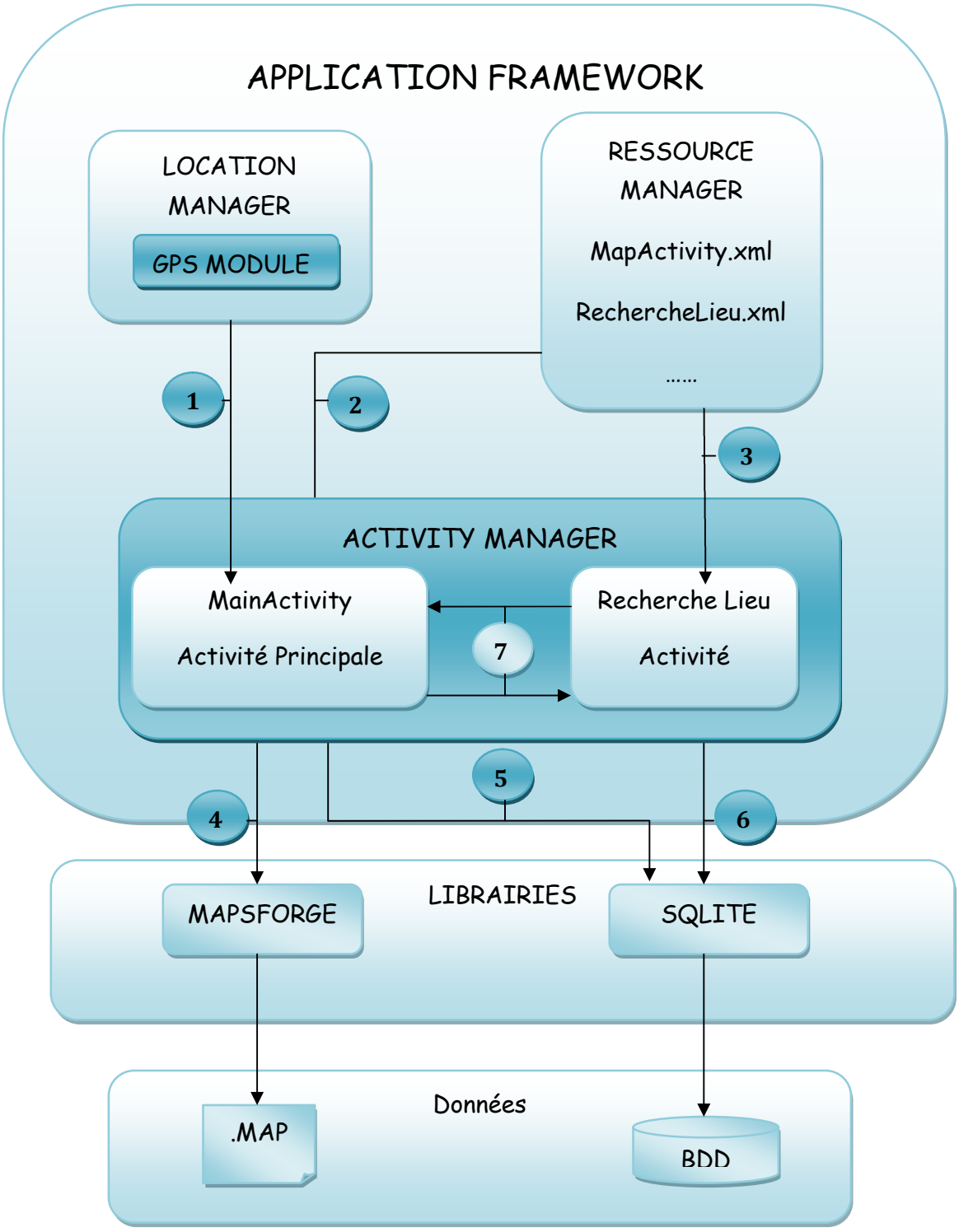


Figure 4. 14 Architecture détaillée de l'application.

IV.4.1.1) Description d'App Framework :

IV.4.1.1.1) Location Manager: [Android Location Manager]

Service qui gère les technologies de géolocalisation, comme le GPS ou le GSM cellulaire pour déterminer la position courante de l'appareil, ce service s'abstrait de la technologie spécifique de localisation et laisse l'utilisateur fixer les besoins minimaux (précision, cout, …) plutôt que de choisir une technologie particulière. Ce qui signifie que l'application utilisant ce service fonctionne, quelle que soit la technologie utilisé par le téléphone.

IV.4.1.1.2) Ressource Manager :

S'occupe de tout ce qui n'est pas du code (image, audio, vidéo…), c'est lui par exemples qui permet de configurer une application de telle sort qu'elle soit accessible sous différent langages (français, arabe, anglais...).

IV.4.1.1.3) Activity Manager : C'est lui qui gère le cycle de vie des applications et assure une partie du multitâches.

IV.4.1.2) Déscription des interactions :

- **1. Location manager → MapActivity :**

  Envoi de coordonnées GPS en temps réel à MapActivity qui spécifie le délai d'attente entre deux réceptions.

- **2. Ressource Manager → Recherche Lieu :**

  Fournit la vue de l'activité RechercheLieu qui n'est autre que le fichier RechercheLieu.xml.

- **3. Ressource Manager → MapActivity :**

  Fourni la vue (description de l'interface graphique) de l'activité MapActivity qui n'est autre que le fichier MapActivity.xml.

- **4. MapActivity → MapsForge :**

  MapActivity utilise MapsForge qui se charge de traiter le fichier .map pour créer une vue de la carte.

- **5. MapActivity → SQLite :**

  MapActivity interagit avec la base de données via la librairie SQLite qui fournit les outils nécessaires à l'exécution des requêtes.

- **6. RechercheLieu → SQLite :**

  RechercheLieu interagit avec la base de données via la librairie SQLite qui fournit les outils nécessaires à l'exécution des requêtes.

- **7. MapActivity ←→ RechercheLieu :**

  C'est un cas d'utilisation de l'application, à partir de l'activité principale il est possible de lancer une recherche textuelle pour un lieu, une liste permet de choisir un lieu précis selon le texte saisi. Une fois le lieu sélectionné RechercheLieu retourne les coordonnées GPS afin de l'afficher sur la carte.

IV.4.2) Présentation de l'application :

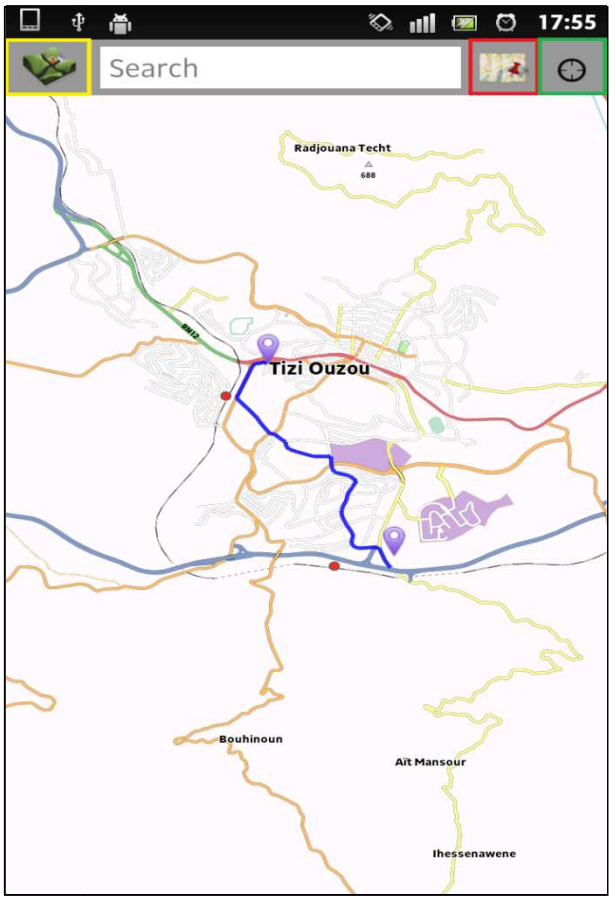


Figure 4. 15 interface graphique de l'application.

L'interface est décomposée en deux parties :

- Partie navigation : Représenté par la carte de la ville de TiziOuzou
- Barre d'outils : La barre d'outils est en haut de l'écran, elle est composée de trois principaux boutons et d'une barre de recherche :

➢ **Localisation (Cadre vert):** Permet de trouver la localisation de l'utilisateur, une position est affiché dès que le service GPS envoi les premières coordonnées. Le service continue de fonctionner tant que ce bouton est activé.

➢ **Routage (Cadre rouge) :** Invite l'utilisateur à sélectionner deux points sur la carte et lance l'algorithme de routage qui se charge de trouver le plus court chemin reliant ces deux points, comme sur la figure ci-dessus.

➢ **Rechercher un type de lieu (Cadre Jaune) :** Lance une boite de dialogue qui permet de choisir parmi plusieurs types de lieux communs (café, hopital, parc, banque …) et de spécifier une distance maximale qui les sépare de la position de l'utilisateur. Une fois le choix effectué les lieux respectant la distance sont marqués.

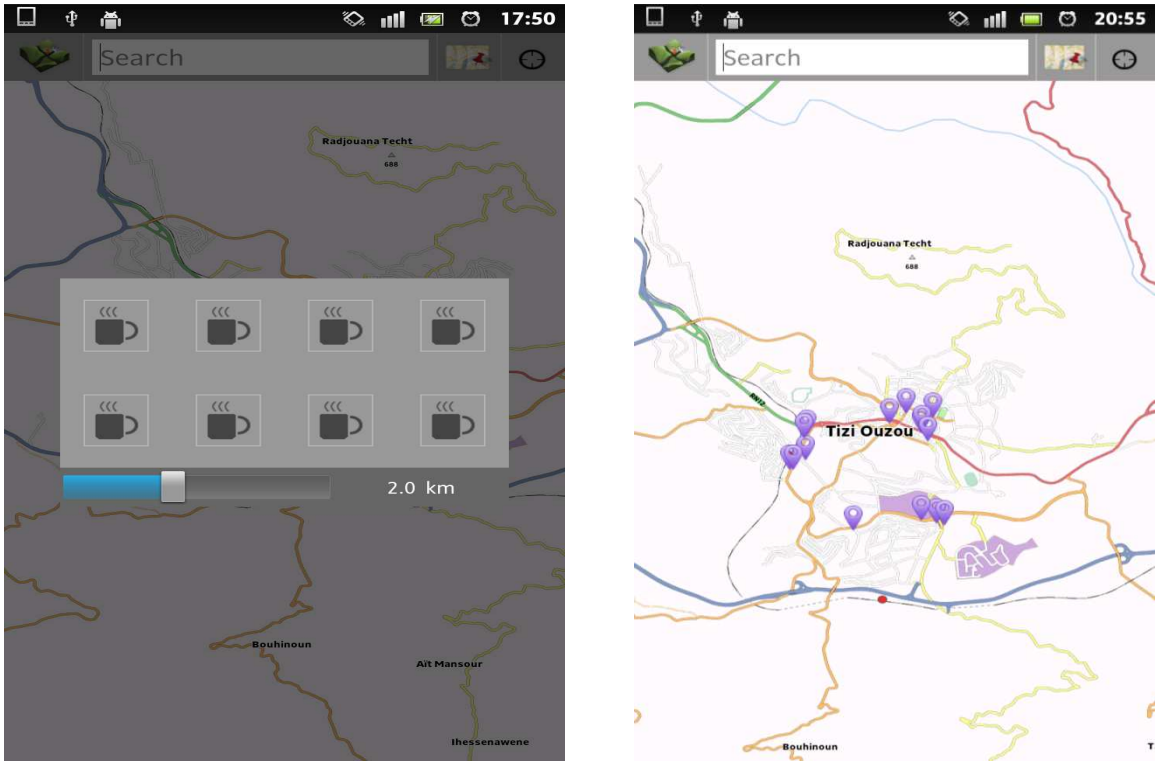


Figure 4. 16 interface graphique de l'application.

- **Barre de recherche :** Permet de chercher un lieu d'une manière textuelle. Après la saisie du texte une liste des lieux possibles apparait, il suffit alors de choisir l'un d'eux.

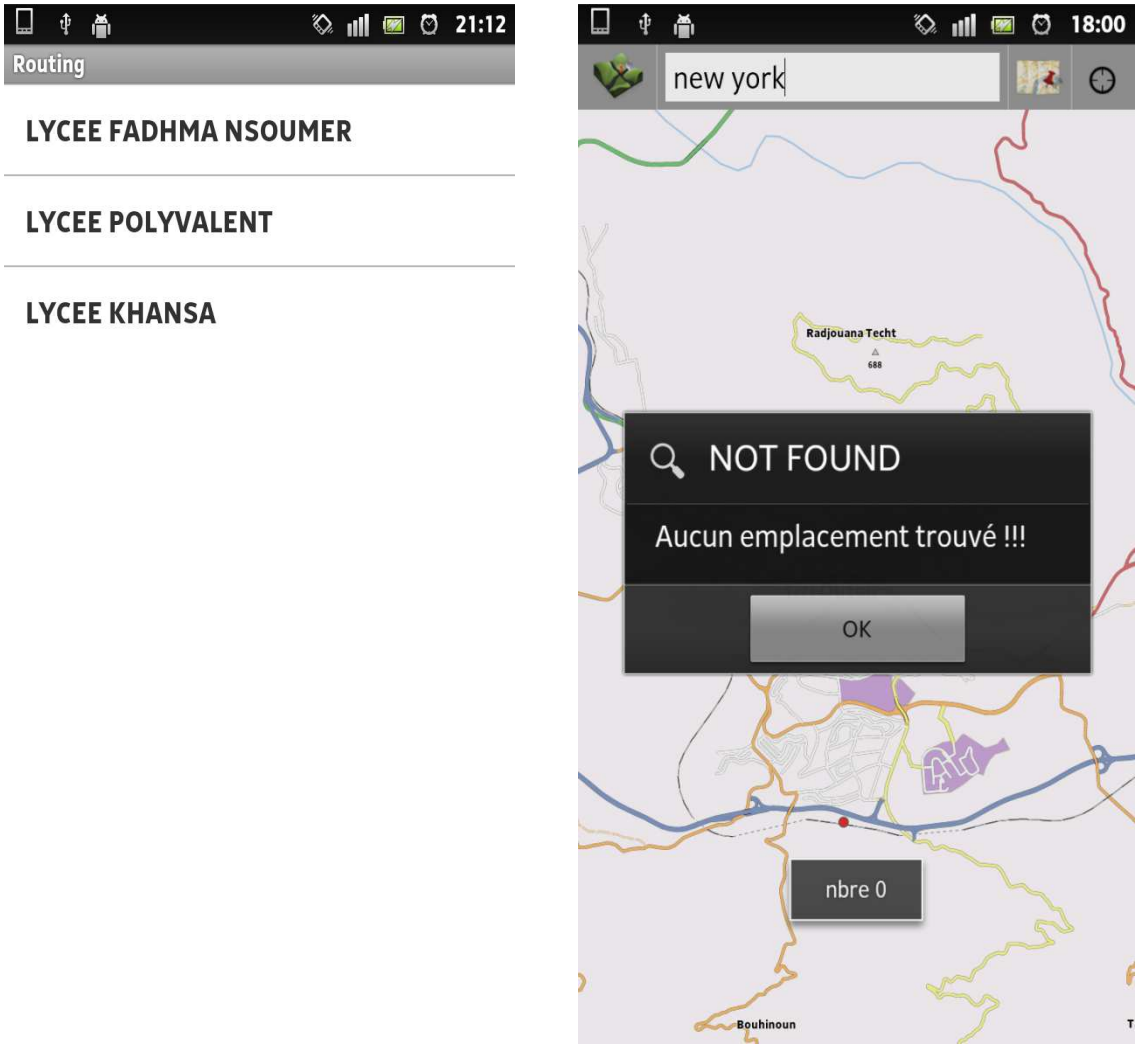


Figure 4. 17 Recherche textuelle d'un lieu.

Réaliser une navigation par GPS :

Afin de naviguer avec notre application, il faut lancer respectivement le service de localisation et la fonction de routage, une fois la position de l'utilisateur affiché sur la carte il suffit de sélectionner un point à rallier pour qu'un tracé le guide.

IV.5) Conclusion :

Dans ce chapitre nous avons présenté, l'architecture générale de notre application qui décrit principalement les interactions entre les principaux composants qu'offre le système Android, suivi des outils et bibliothèques utilisés. A la fin du chapitre un bref aperçu de l'application a été présenté.

Conclusion Générale :

De nos jours concevoir une application de géolocalisation connecté à internet n'est pas très compliqué. Disposant d'un accès aux données cartographique disponible sur la toile, ce type de logiciels est largement utilisé dans des pays fortement couvert par le réseau internet, notamment la 3G. Ayant comme principal inconvénient leur dépendance à internet, ces applications ont vite montré leurs limites dans des environnements dépourvus d'internet.

Ce projet visait à développer une application de géolocalisation par GPS sans avoir recours à internet. Ce qui revient à trouver un moyen de stocker des données géographiques en local et pouvoir ainsi effectuer des opérations de représentation et de routage en offline.

Les perspectives envisagées pour faire évoluer ce projet sont nombreuses dont les plus importantes sont:

- Tenir compte pour le plus court chemin des caractéristiques des routes comme type, trafic, vitesse autorisé…
- améliorer la qualité des cartes générées.
- Résoudre le problème de navigation, cas des routes parallèle très proche.
- Ajouter le guidage par voix.

Il faut dire que ce projet a été riche par les connaissances qu'il nous a permis d'acquérir, d'une part, par la diversité des technologies auxquels nous nous sommes intéressées et d'autre part par le fait qu'il nous a permis de travailler au sein d'une équipe de recherche ambitieuse.

Références Bibliographiques

[CiscoMag, 2006], CiscoMag, Géolocalisation WiFi & RFID décembre 2006.

[Ilyas & Ahson, 2006] Mohammad Ilyas, Syed A. Ahson, Smartphones, 2006.

[Kleusberg, 1990] Kleusberg, A., Comparing GPS and GLONASS, GPS World, Vol. 1, No. 6, November/December 1990, pp. 52-54.

[Mohan, 2013], Operating Systems I. Chandra Mohan 2013.

[Neviere, 2001], Philippe Neviere, Etude sur la co-localisation d'un réseau UMTS avec les réseaux GSM et DCS existants, 2001.

[Phillips, 1996] Phillips, B., GPS Field Applications in Forestry Consulting, Global Positioning System in Forestry Workshop, Kelowna, British Columbia,Canada, November 25-28, 1996.

[Wiley & Sons, 2011], Next Generation Mobile Communications Ecosystem: Technology Management for Mobile Communications John Wiley & Sons, 25 févr. 2011.

[Wiley & Sons, 2008], Developing Software for Symbian OS 2nd Edition: A Beginner's Guide to Creating Symbian OS v9 Smartphone Applications John Wiley & Sons, 28 feb. 2008.

Webliographie

[1] Définition Windows Os
http://mobiledevices.about.com/od/glossary/g/Definition-Of-The-Windows-Mobile-Os.htm

[2] Site Officiel Android
http://android.com

[3] Activité Android
http://developer.android.com/reference/android/app/Activity.html

[4] Géolocalisation
http://blog.nordnet.com/loeil-sur-le-web/decryptages-loeil-sur-le-web/la-geolocalisation.html

[5] Géolocalisation par wifi
http://www.cnil.fr/linstitution/actualite/article/article/la-geolocalisationa-partir-des-points-dacces-wi-fi/

[6] Géolocalisation par adresse IP
http://www.definitions-webmarketing.com/Definition-Geolocalisation-adresse-IP

[7] Open Street Map
http://fr.flossmanuals.net/_booki/openstreetmap/openstreetmap.pdf
[8] Google Maps Marker
https://support.google.com/mapmaker

[9] Ushahidi
http://www.ushahidi.com/

[10] WikiMapia
http://wikimapia.org/

[11] Format de fichier .map
http://code.google.com/p/mapsforge/wiki/SpecificationBinaryMapFile

[12] Site Officiel Java
http://www.java.com/fr/

[13] Portail des développeurs Android [En ligne].
http://developer.android.com/

[14] ADT Plugin android
http://developer.android.com/tools/sdk/eclipse-adt.html

[15] API Open Street Map
http://www.openstreetmap.org/

[16] Site Officiel SQLite
http://www.sqlite.org

[17] MapsForge site officiel
http://code.google.com/p/mapsforge/

[18] Page officielle d'Osmosis
http://wiki.openstreetmap.org/wiki/FR:Osmosis

Annexes

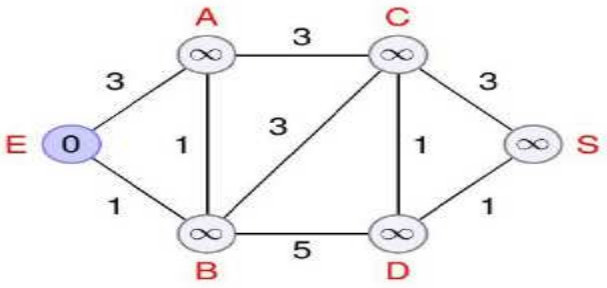
i. Algorithme Dijkstra:

Pour pouvoir trouver le chemin le plus court entre deux nœuds d'un graphe plusieurs algorithmes de recherche peuvent être appliqué. Dans le cadre de se projet nous avons choisis l'algorithme de dijkstra dont le fonctionnement est présenté ci dessous.

1: Initialisation du graphe :
Les valeurs des poids de tous les nœuds sont initialisées à l'infini sauf le point de départ, la distance initiale est : 0.
2: T := ensemble de tous les nœuds non visités.
3: **Tant que** T n'est pas vide **faire**
Choisir un noeud i dans T tel que poids(i) est minimum
Retirer i de T et l'ajouter à S l'ensemble des nœuds visités
Pour chaque nœud successeur j de i, avec j dans T, **faire**
Si poids(j) > poids(i) + distance(i,j) **alors**
poids(j) = poids(i) + distance(i,j)
précédent(j) = i

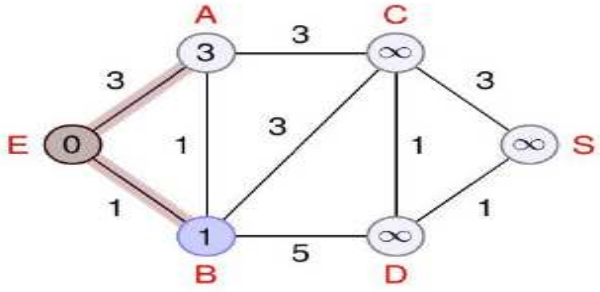
Remarque : L'algorithme peut être arrêté une fois le nœud de destination atteint.

Voici un exemple qui illustre bien le concept de cet algorithme.ici objectif est de trouver le plus court chemin allant de E à C.



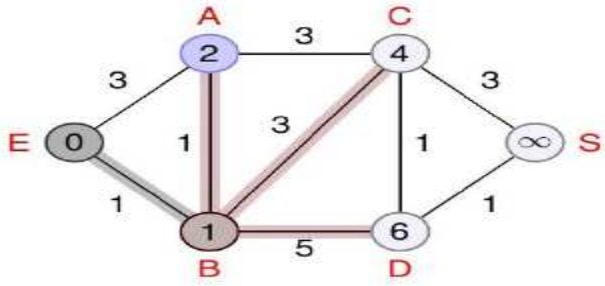
E	A	B	C	D	S
0	∞	∞	∞	∞	∞
•					
•					
•					
•					
•					

Initialement la valeur de tous les nœuds est fixé à '∞', sauf le nœud de départ est mis à '0'.



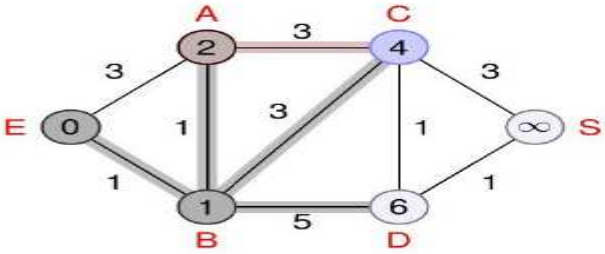
E	A	B	C	D	S
0	∞	∞	∞	∞	∞
•	3_E	1_E	∞	∞	∞
•					
•					
•					
•					

Après avoir calculé la distance entre le nœud E et les nœuds A et B, les cases A et B de tableau sont renseigner. Comme la distance la plus courte est celle qui sépare E de B, le nœud B est marqué.



E	A	B	C	D	S
0	∞	∞	∞	∞	∞
•	3_E	1_E	∞	∞	∞
•	2_B	•	4_B	6_B	∞
•	•	•			
•	•	•			
•	•	•			

Dans cette étape le même principe de l'étape précédente est utilisé sauf qu'ici la valeur du nœud initial (B) est '1', donc la distance le séparant des autres nœuds est augmenté de '1'



E	A	B	C	D	S
0	∞	∞	∞	∞	∞
•	3_E	1_E	∞	∞	∞
•	2_B	•	4_B	6_B	∞
•	•	•	4_B	6_B	∞
•	•	•	•		
•	•	•	•		

Le même processus est suivi pour les nœuds C, D et S. Arriver au nœud S. tous les nœuds sont visités et le calcule de chemin le plus court est terminé (E-> B->A->C-> D->S).

ii. Analyseur documents XML SAX :

SAX signifie Simple API XML, c'est une API très simple qui permet de lire des flux (fichiers) XML. À chaque fois qu'il rencontre un élément particulier, il appelle une méthode correspondante. Par exemple à chaque nouvelle balise XML qu'il rencontre il va appeler la méthode "startElement".

Pour l'utiliser il faut définir une classe qui va implémenter des méthodes particulières, voici les plus importantes d'entre elles:

startElement	appelé lorsque le parser rencontre une nouvelle balise XML.	les balises d'ouverture <maBalise>
endElement	appelé lorsque le parser rencontre une balise fermante XML	les balises de fermeture </maBalise>
startDocument	appelé lorsque le parser débute la lecture du document XML	
endDocument	appelé lorsque le parser termine la lecture du document XML	
characters	appelé lorsque le parser rencontre des caractères à l'intérieur d'un élément	pour les charactères "mes caractères" dans <maBalise>mes caractères</maBalise>

Pour utiliser le parser il faut définir son propre handler, c'est à dire une classe qui va implémenter les méthodes.