Université Sidi Mohamed Ben Abdellah
Faculté des Sciences et Techniques Fès
Département Génie Electrique

Mémoire de Projet de fin d'étude

Préparé par

Hajar ZOULGAMI

Pour l'obtention du diplôme

Ingénieur d'Etat en

SYSTEMES ELECTRONIQUES & TELECOMMUNICATIONS

Intitulé

Conception et développement d'une application paramétrable et configurable pour la collecte des données géolocalisées

Encadré par :

Pr S. Najah

Mr S. Salim

Soutenu le Mardi 28 Juin 2011, devant le jury composé de :

Pr S. Najah……: Encadrant

Pr Y.Balboul…… : Examinateur

Pr A.Mechaqrane …… : Examinateur

Pr Pr H. El Moussaoui …… : Examinateur

ANNEE UNIVERSITAIRE 2010- 2011

Dédicaces

A mes chers parents,

Aucun mot, aucune dédicace ne saura exprimer mon respect, ma considération et l'amour éternel pour les sacrifices que vous avez fait pour mon instruction et mon bien être.

A ma chère sœur Fadoua,

Dieu seul sait à quel point je suis reconnaissante à ton égard, que Dieu te bénisse ainsi que ta petite famille, vous protège et vous offre tout le bonheur.

A mes deux chers frères Mouhssine et Hamza,

Avec tout mon amour et mon affection, je vous souhaite une vie pleine de bonheur et de réussite.

A toute ma famille,

A tous mes amis,

A tous ceux qui m'aiment,

Du fond du cœur je vous dis Merci infiniment.

Remerciements

Au terme de ce travail, je tiens à remercier toutes les personnes qui ont contribué à la réussite du projet et qui n'ont pas ménagé leurs efforts pour me soutenir à tous les niveaux.

Je remercie en premier lieu mon encadrant M. Said NAJAH pour ses précieux conseils et ses directives notamment dans la rédaction du rapport.

Mes vifs remerciements vont aux cadres d'I2S ingénierie pour leur disponibilité et leur assistance lors de la réalisation de ce projet. Je remercie en particulier M. Said SALIM qui m'a encadré durant toutes les phases du projet.

Je voudrais faire part de ma gratitude à l'ensemble du personnel d'I2S ingénierie pour leur amabilité et la bonne ambiance qu'ils ont fait régner durant la période de mon stage.

Je remercie également les membres du jury qui ont accepté d'évaluer mon travail.

J'exprime enfin ma profonde gratitude à tous ceux qui ont participé de près ou de loin au bon déroulement de ce travail.

Merci à tous

Résumé

Le présent rapport est une synthèse du travail effectué dans le cadre de notre projet de fin d'étude au sein de la société I2S Ingénierie. L'objectif de ce projet est de réaliser une application paramétrable et configurable pour la collecte des données géolocalisées. Cette dernière sera dotée de plusieurs fonctionnalités de base. Cette application sera utilisable sur un terminal mobile fortement communiquant (Communication 3G, WIFI) se localisant par satellites (système GPS). La réalisation de cette application se basera sur la plate-forme Android.

Notre objectif consistait en l'intervention à toutes les phases du projet. En commençant par la rédaction du cahier des spécifications détaillées, puis en procédant à l'analyse, la conception et la réalisation de l'application, et enfin le test et le déploiement de l'application.

Notre projet est articulé autour du processus 2TUP, il a été réalisé en quatre étapes :

La première étape est une étude fonctionnelle du projet. Elle comprend la rédaction du cahier des spécifications détaillées.

La deuxième étape est une étude technique. Elle décrit les architectures logicielle et physique adoptées.

La troisième étape consiste en la réalisation de la phase de conception en se basant sur le Langage UML.

La quatrième consistait à implémenter, tester et déployer l'application dans un terminal mobile.

Mots-clés: Géolocalisation, GPS (Global Positioning System), Android, LBS (Location Based Service).

Liste des abbreviations

GPS : Global Positioning System

2TUP : 2 Track Unified Process

UML : Unified Modeling Language

KML : Keyhole Markup Language

ADT : Android Development Tools

SIG : Système d'Information Géographique

VM : Virtual Machine

RIM : Research In Motion

JVM : Java Virtual Machine

NDK : Native Development Kit

SDK : Software Development Toolkit

DDMS : Dalvik Debug Monitor Server

IDE : Integrated Development Environment

APK : Android Package

AIDL : Android Interface Definition Language

API : Application Programming interface

Liste de figures

Liste des tableaux

Table des matières

Introduction Générale

Auparavant, le processus de collecte et de mise à jour des données géographiques sur le terrain prenait beaucoup de temps et comportait souvent des erreurs à causes des outils traditionnels utilisés. Chose qui a dégradé et ralentis le processus de traitement de ces données , que l'utilisateur, une fois de retour au bureau, était obligé de les insérer manuellement dans la base de données SIG à partir des cartes papier et des notes pris sur le terrain et vu la quantité souvent grande des données à traiter et la difficulté des mises à jour éventuelles, la probabilité d'erreur s'avérait être importante ce qui a poussé les acteurs à chercher d'autres alternatives pour pallier à ce problème .

Et d'opter vers plus de mobilité en faisant appel aux applications métier portables qui ont beaucoup amélioré la qualité du travail et donc la productivité de l'entreprise. C'est dans ce cadre que s'inscrit notre projet proposé par la société I2S Ingénierie qui consiste en la conception et la réalisation d'une application paramétrable et configurable pour la collecte des données géolocalisées, basée sur la plateforme Android.

Pour ce faire, nous avons veillé à ce que notre rapport présente bien une description détaillée des différentes étapes du projet.

En effet, il se compose de trois parties. La première partie présente le contexte général du projet, elle commence par un chapitre de présentation de l'organisme d'accueil. Un deuxième chapitre traite quelques notions sur les techniques de Géolocalisation.

La deuxième partie présente la plateforme Google Android, ses mécanismes ainsi que les outils qui nous permettent de la manipuler.

La troisième et dernière partie, est consacrée à la mise en œuvre. Elle présente à travers le chapitre quatre l'analyse et la conception. Le dernier chapitre de cette partie, à savoir le chapitre cinq, présente les fonctionnalités implémentées dans l'application.

Chapitre 1

Contexte général du projet

Ce chapitre a pour objectif de présenter le contexte général du projet. Il commence tout d'abord, par la présentation de l'organisme d'accueil. Ensuite il traite la problématique et présente les objectifs du projet. Enfin, il décrit le déroulement du projet en présentant le processus de développement adopté ainsi que la planification du projet.

1 Présentation de l'organisme d'accueil

I2S Ingénierie, Présentée par son directeur M.SALIM Said, est une jeune société de services en ingénierie et informatique, qui a pour mission de conseiller et fournir des solutions et des produits informatiques pour le système de gestion des réseaux routiers et sa maintenance, ainsi que la maitrise des logiciels, spécialement, de conception en génie civil, sans oublier son expertise dans le domaine SIG (système d'information géographique).

Les concepts innovants qu'elle propose lui confèrent naturellement une dimension nationale, elle s'adapte aux évolutions du marché et aux exigences de souplesse et de performance économique de ses clients. La société a pour objectif, entre autres, de développer des offres toujours plus pertinentes fondées sur un savoir-faire industrialisé. Ses missions reposent sur le principe d'une double compétence permanente :

- Connaissance du métier de ses clients.
- Un sens du service lié à un haut niveau de technicité.

Etant l'une des sociétés spécialisées dans le système d'information géographique, Elle œuvre dès sa création dans la production et la validation des données géographiques et dans les SIG, depuis la collecte des données sur le terrain et la constitution de référentiels géographiques propres au besoin de ses clients, jusqu'à leur exploitation à travers des outils informatiques dédiés et interfacés avec les systèmes d'information et de gestion du client.

Depuis sa création 2010, ses pôles d'activités sont orientés dans :

- La gestion des infrastructures publiques (cadastre des surfaces et souterrain).
- La gestion des infrastructures privées (gestion du patrimoine, des biens, des personnes et des évènements).
- Les SIG (système d'information géographique).
- Le développement informatique de haut niveau.
- GPS – gestion et localisation de flotte de véhicules.

2 Cahier des charges

Le projet porte sur la réalisation d'une application paramétrable permettant de :

- localiser l'utilisateur et mettre à jour sa position géographique au fur et à mesure de son déplacement sur le terrain.
- tracer /visualiser en temps réel le chemin empreinté par l'utilisateur sur une carte géographique.
- Sauvegarder tous les parcours en vue d'une consultation ou traitement ultérieure.
- Exporter les chemins enregistrés sous le format de fichier KML (Keyhole Markup Language) qui est compatible avec les coordonnées GPS.
- Documenter chaque chemin enregistré par des photos, vidéos, audio ou texte.

3 Conduite du projet

Ce paragraphe décrit la conduite du projet. Il commence par présenter le processus de développement adopté ensuite il décrit le déroulement du projet à travers le paragraphe planification.

3.1 Processus de développement

Le processus de développement adopté est le 2TUP ce qui signifie «2 Track Unified Process». Il s'agit d'un processus unifié (c'est-à-dire construit sur UML, itératif, centré sur l'architecture et conduit par les cas d'utilisation), qui apporte une réponse aux contraintes de changement continuel imposées aux systèmes d'informations des entreprises. Il propose alors un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels.

En effet, l'axiome fondateur du 2TUP a été le constat que toute évolution imposée au système d'information peut se décomposer et se traiter parallèlement, suivant un axe fonctionnel et un axe technique. A l'issue des évolutions du modèle fonctionnel et de l'architecture technique, la réalisation du système consiste à fusionner les résultats de ces deux branches du processus, ce qui nous donne un cycle de développement sous forme de Y.

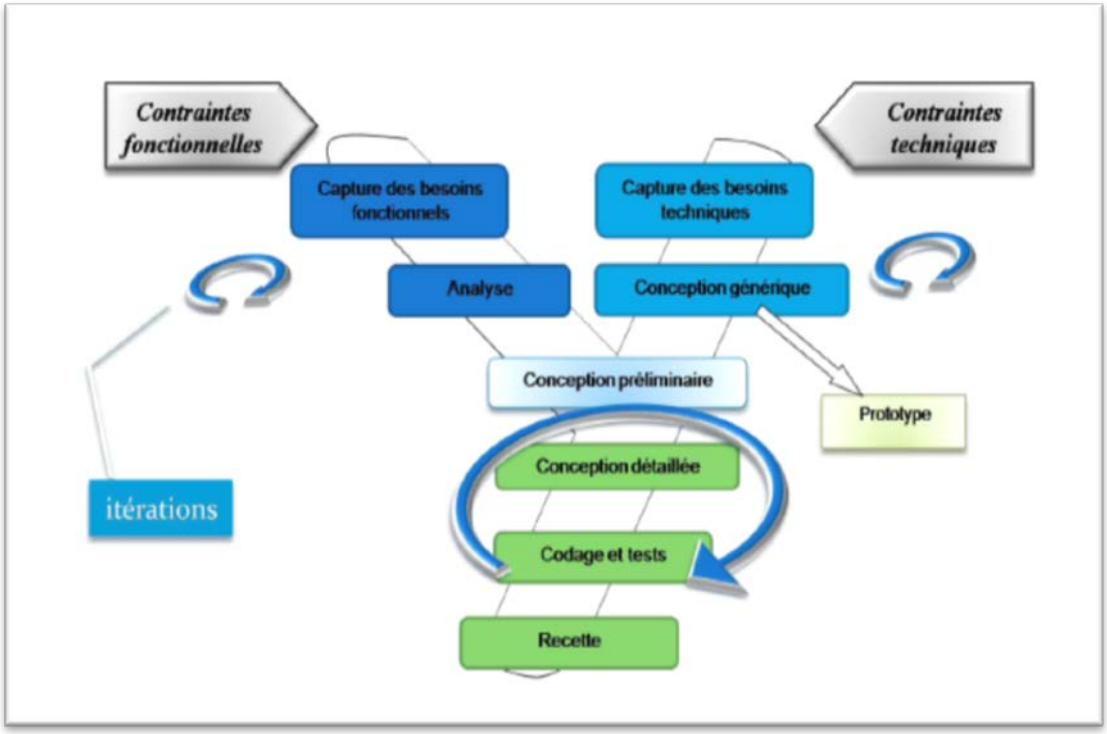

Figure 1 : Processus de développement en Y

- **Branche fonctionnelle (gauche):**

• Capture des besoins fonctionnels, qui produit le modèle des besoins focalisé sur le métier des utilisateurs. Elle qualifie, au plus tôt le risque de produire un système inadapté aux utilisateurs

• L'analyse, qui consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système en terme de métier.

❖ **Branche architecture technique (droite) :**

• La capture des besoins techniques, qui recense toutes les contraintes sur les choix de dimension et la conception du système, les outils et le matériel sélectionné ainsi que la prise en compte des contraintes d'intégration avec l'existant (pré-requis d'architecture technique).

• La conception générique, qui définit ensuite les composants nécessaires à la construction de l'architecture technique. Cette conception est complètement indépendante des aspects fonctionnels. Elle a pour objectif d'uniformiser et de réutiliser les mêmes mécanismes pour tout un système. L'architecture technique construit le squelette du système, son importance est telle qu'il est conseillé de réaliser un prototype.

❖ **Branche conception (milieu) :**

• La conception préliminaire, qui représente une étape délicate, car elle intègre le modèle d'analyse fonctionnelle dans l'architecture technique de manière à tracer la cartographie des composants du système à développer.

• La conception détaillée, qui étudie ensuite comment réaliser chaque composant.

• L'étape de codage, qui produit ses composants et teste au fur et à mesure les unités de code réalisées.

• L'étape de recette, qui consiste enfin à valider les fonctionnalités du système développé.

3.2 Planification

Dans le cadre de la conduite du projet, la réalisation d'un planning à suivre tout au long du notre stage s'impose. Ainsi, après avoir déterminé le processus de développement adopté, l'étape suivante est de planifier les tâches selon ce processus et selon les contraintes du projet. Nous avons établi un planning prévisionnel pour la réalisation du projet, et pour le suivi nous avons utilisé l'outil Gantt Project (figure 2).

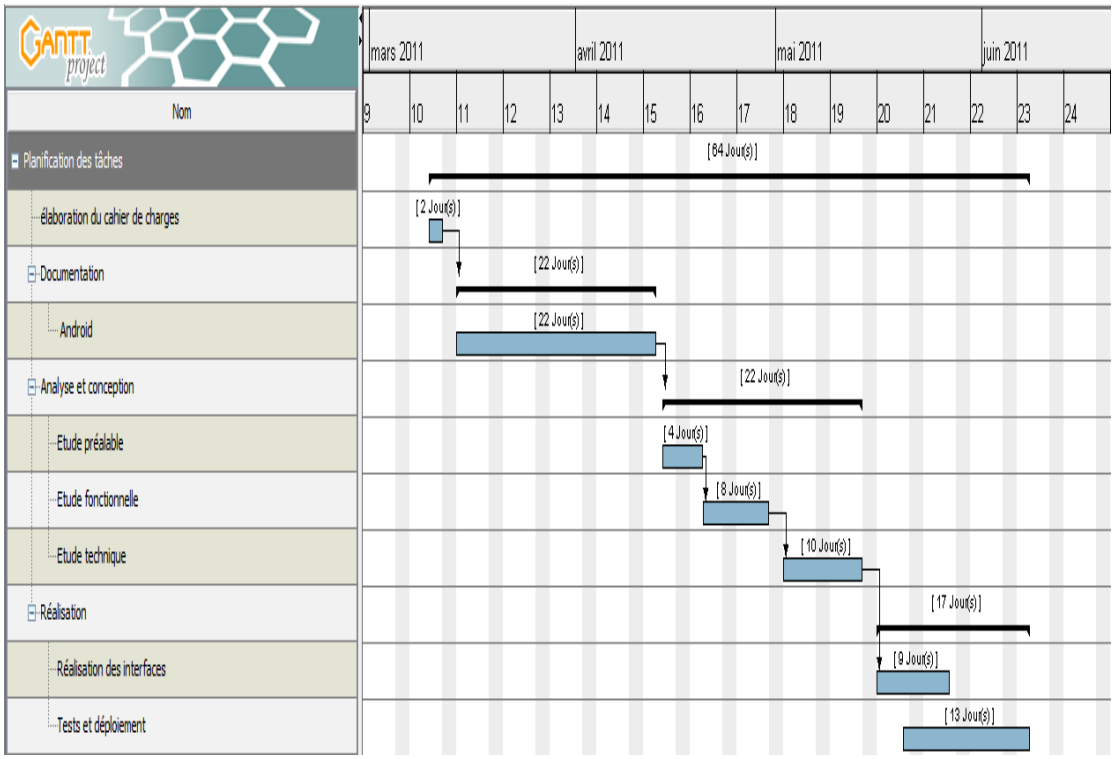

Figure 2 : Planning du projet

Conclusion

Le contexte général du projet étant déterminé, il est nécessaire par la suite d'aller présenter quelques notions sur les différentes techniques de Géolocalisation.

Chapitre 2

Techniques de Géolocalisation

Ce chapitre présentera les différentes techniques mises en jeux dans la localisation d'un terminal. Les techniques de localisation sont de diverses sortes, elles peuvent être divisées en techniques basées sur le réseau, techniques utilisant des dispositifs intelligents aux endroits fixes et enfin, des techniques basées sur le GPS (Figure 3).

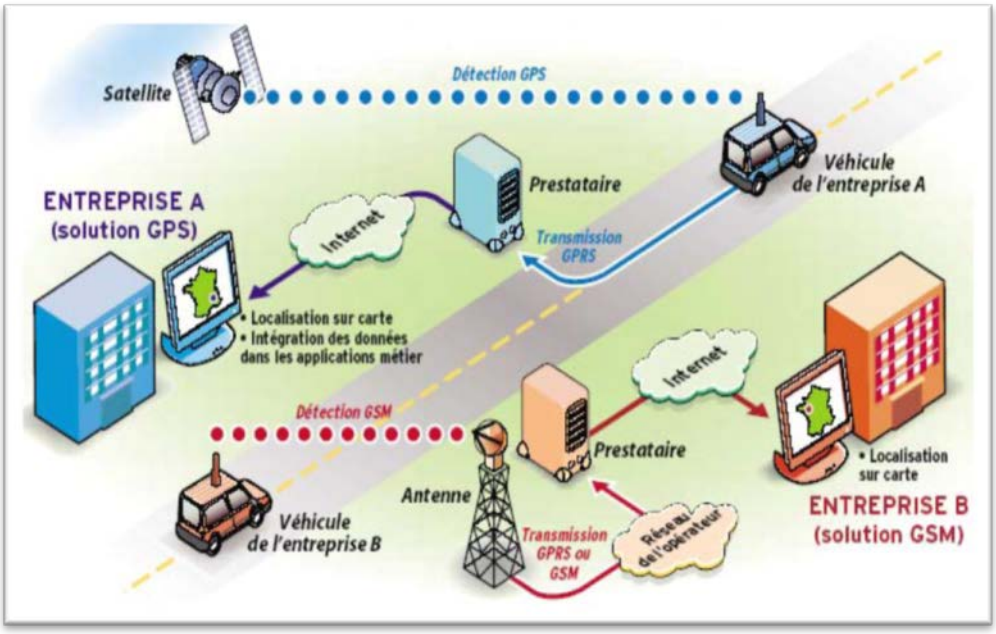

Figure 3 : Les techniques de localisation

1 Géolocalisation par Satellites

Elle consiste à calculer, grâce aux signaux émis par une constellation de satellites prévue à cet effet, la position actuelle sur la face terrestre d'un terminal équipé d'une puce compatible. Cette position est alors traduite en termes de latitude, longitude et parfois altitude et peut alors être représentée physiquement sur une carte. Le réseau satellite de positionnement le plus connu est le GPS (Global Positionning System).

1.1 Géolocalisation par GPS

Le GPS se base sur une constellation de 24 satellites, qui émettent en permanence un signal daté, et un réseau de stations au sol qui surveillent et gèrent ces satellites. La localisation est possible dès lors que quatre satellites sont visibles : il y a en effet quatre inconnues à déterminer, les trois coordonnées spatiales, ainsi que le temps,

puisque le récepteur au sol n'est pas synchronisé avec les satellites. Pour ce faire, les 24 satellites du système sont équi-répartis sur six orbites de façon à garantir qu'au moins quatre satellites soient visibles en permanence et ce, partout sur la Terre. A partir de trois satellites, un tel récepteur est capable d'effectuer une triangulation pour déterminer sa position. Cette position est déterminée instantanément d'où la possibilité de poursuivre des cibles mobiles. Chaque mesure représente le rayon R d'une sphère centrée sur un satellite particulier.

Le récepteur GPS est sur cette sphère. Avec trois mesures, donc trois satellites, la position du récepteur se réduit à l'intersection de deux points dont l'un est très éloignée dans l'espace (Figure 4).

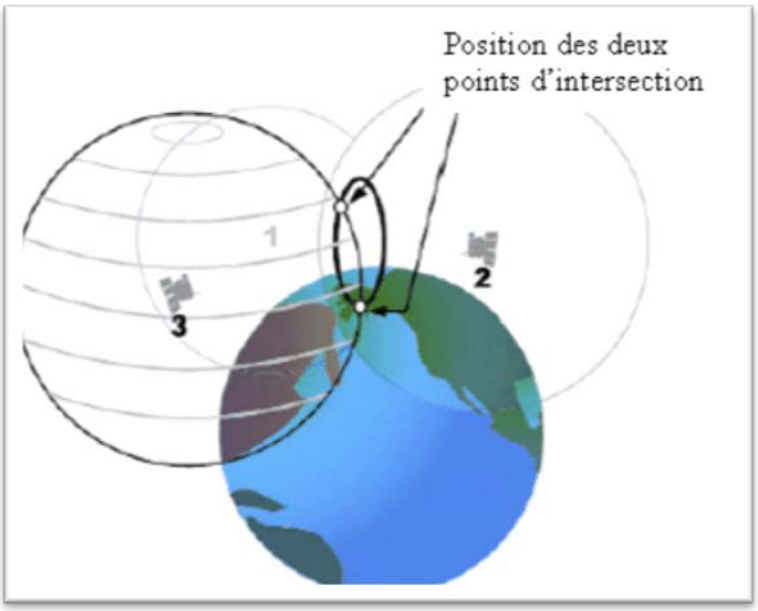

Figure 4 : Localisation du récepteur par trois satellites

Pour information, la précision moyenne du système GPS actuel est :

- Pour usage gratuit: de l'ordre de 20 mètres.
- Pour usage civil : 3-8 m
- Pour usage militaire : 1-3 m
- Pour le GPS différentiel (DGPS) : < 1m
- différentiel avec post-traitement : quelques cm

Autre alternative qui se prépare prochainement d'entrer en concurrence avec le GPS classique c'est le système GPS Européen : Galileo dont le lancement de ses deux premiers satellites est prévu pour le 20 octobre 2011, selon L'Agence spatiale européenne (ESA).

1.2 Géolocalisation par Galileo

L'architecture du futur système de positionnement par satellites européen se base sur une constellation de 30 satellites (27 actifs et 3 de secours) placés sur trois orbites circulaires d'altitude moyenne (24000 Km), des stations au sol, de centres de contrôle et des utilisateurs dotés de récepteurs mobiles.

Le principe de fonctionnement est simple: les satellites de la constellation sont équipés d'une horloge atomique mesurant le temps avec une extrême précision. Ils émettent des signaux personnalisés indiquant leur heure de départ du satellite. Le récepteur au sol, intégré par exemple dans un téléphone portable, possède pour sa part en mémoire les coordonnées précises des orbites de tous les satellites de la constellation. Il peut ainsi en lisant le signal qui arrive reconnaître le satellite émetteur, déterminer le temps mis par le signal pour arriver jusqu'à lui et donc calculer la distance qui le sépare du satellite. Dès qu'un récepteur au sol reçoit les signaux d'au moins quatre satellites simultanément, il peut calculer sa position la plus exacte possible (précision variant de 10 à 1 m).

Galileo diffusera dix signaux:

- six pour les services gratuits.
- deux pour le service commercial
- deux pour le service public réglementé

2 Géolocalisation par GSM

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 si le terminal se trouve en milieu urbain (où la densité d'antennes est supérieure), ou en milieu rural.

Plusieurs techniques existent. Aujourd'hui, la méthode GSM la plus utilisée est celle du Cell ID. Cette méthode consiste à récupérer les identifiants des antennes GSM auxquelles le terminal est connecté. Par la suite, grâce à une base de données faisant le lien entre les identifiants des cellules et les positions géographiques des antennes, le terminal est capable de déterminer sa position et d'émettre une estimation.

Étant donné que les bases de données Cell ID ne sont pas stockées localement dans le terminal, une connexion internet de type GPRS/EDGE ou 3G peut être nécessaire afin d'émettre une requête pour obtenir la correspondance Cell ID / Longitude Latitude.

3 Géolocalisation par WiFi

De la même façon qu'un terminal GSM peut se localiser par la méthode du Cell ID sur un réseau GSM, un terminal WiFi peut utiliser la même méthode en se basant sur les identifiants des bornes WiFi (Adresses MAC) qu'il détecte. Il existe des bases de données recensant une multitude de bornes d'accès WiFi ainsi que leur position géographique.

4 Géolocalisation par adresse IP (sur internet)

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, 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.

5 Géolocalisation par RFID

La technologie RFID peut être utilisée pour la géolocalisation indoor. Pour ce faire, une série de lecteurs RFID équipés de différents types d'antennes sont positionnés de façon à couvrir l'ensemble de la zone souhaitée. La zone est alors découpée en cases dont la surface varie en fonction du nombre de lecteurs déployés et de leur puissance. Lorsqu'une personne équipée d'un tag RFID actif sera dans ces zones là, le système sera capable de calculer sa position en se basant sur le nombre de lecteurs qui détectent le tag et de déduire la position approximative de l'individu en se référant au schéma de découpage établi.

Conclusion

Chacune de ces techniques se caractérise par le degré de précision de ses informations, la consommation d'énergie qu'elle engendre ou encore les conditions qui doivent être réunies pour assurer son fonctionnement (certaines capacités matérielles comme la présence d'une puce GPS, un accès à internet, tag RFID...).

Bien entendu, pour notre cas, notre technique de localisation est basée sur le GPS vu ses avantages qui font de lui un choix approprié comme technique de localisation pour randonneurs.

Chapitre 3

Technologies utilisées

Ce chapitre est consacré à la présentation de la plate-forme Google Android, ses mécanismes ainsi que les outils qui nous permettent de la manipuler.

1 Histoire de naissance d'Android

Android a vu le jour en novembre 2007, c'était le fruit d'un projet ambitieux des équipes Google en partenariat avec l'Open Handset Alliance. Il s'agit d'un système d'exploitation dédié aux téléphones mobiles modernes et indépendant de toute plateforme matérielle spécifique, fondé sur Linux, avec une interface tactile similaire à celle de l'iPhone et intégralement Open Source.

Cette particularité pourrait bien faire que ce système s'impose sur le marché des mobiles à l'instar du PC sur le marché des micro-ordinateurs. Il fallut attendre octobre 2008 pour avoir un téléphone Android entre les mains.

le développement se fait en Java, pas un « sous-Java » comme c'est le cas avec JavaME, mais un environnement complet, reprenant une grosse partie de la librairie du JDK et y ajoutant tout ce qui permet de construire très facilement des applications tactiles, graphiques, communicantes, géolocalisées, etc.

L'architecture d'Android, comme vous allez découvrir par la suite est d'une grande élégance : en permettant la décomposition d'une application en « activités » communicantes, l'ensemble des applications présentes sur un téléphone Android peuvent coopérer, et de nouvelles applications peuvent enrichir celles qui sont déjà présentes. On est très loin du modèle fermé des téléphones qui ont précédé, et même de l'iPhone.

2 Architecture du système Android

Le système Android se compose de 5 couches:

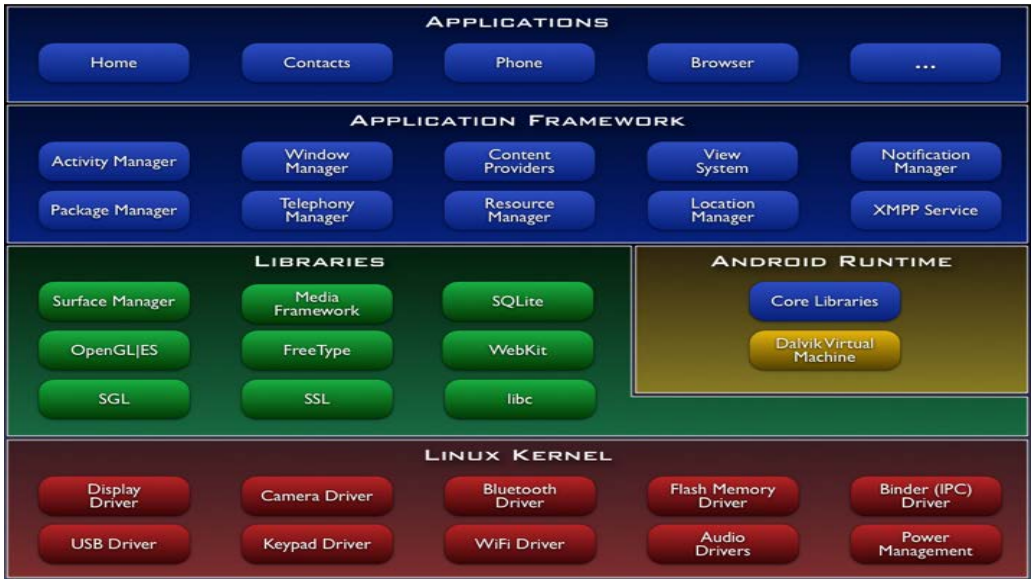

Figure 5 : L'architecture du système Android

2.1 Le noyau Linux

Au-delà du framework de développement, Android est une pile logicielle qui repose sur le couple: Linux/Java. Le noyau Linux employé est le 2.6. Comme n'importe quel OS, ce noyau a la responsabilité de gérer le matériel à l'aide de drivers, la mémoire, les processus ou encore les couches réseaux basses.

2.2 Librairies C/C++

Au-dessus du noyau proprement dit se loge un ensemble de librairies C/C++ constituant les couches bases du système. Parmi celles-ci on peut noter l'incontournable libc (librairie système C standard) dont l'implémentation a été adaptée pour l'occasion à un mode embarqué. Même s'il ne s'agit pas tout à fait de librairies bas niveau, on peut ajouter WebKit qui est un moteur de rendu HTML (navigateur Web) et SQLite qui est un moteur de base de donnée léger et puissant utilisé comme système de stockage interne dans énormément d'applications.

Notre système Android est donc constitué d'un noyau Linux et d'une suite de bibliothèques C/C++ fort utile. Et Pour l'instant l'accès à ces librairies se fait exclusivement par l'API Java d'Android. Il n'est donc pas possible de faire des appels directs aux couches basses et il faudra se contenter des fonctionnalités exposées.

2.3 Le middleware

Au même niveau nous trouvons le middleware, l'environnement d'exécution Android. Cet environnement est constitué en premier lieu d'une machine virtuelle qui a été dénommée Dalvik. En effet, Google a fait le choix de s'écarter des standards Java et de proposer sa propre machine virtuelle. Fondamentalement, la démarche est la même que celle du framework GWT (Google Web Toolkit), Google ne garde de Java que le langage et une partie des API standard et met de côté la plateforme d'exécution (JVM).

La raison d'être de Dalvik est de fournir un environnement optimisé aux mobiles où les ressources CPU et mémoires sont précieuses.
Par exemple, pour de meilleures performances sur un système embarqué, Dalvik est une machine virtuelle dite « registered-based », basée sur les registres, et non pas « stack-based », basées sur les piles, comme la JVM de Sun.

Par ailleurs, sur Android, par défaut, chaque application tourne dans son propre processus Linux, c'est-à-dire sa propre VM. L'un des points forts d'Android étant ses capacités multitâches, il doit être possible de lancer efficacement plusieurs VM Dalvik sur le terminal ; À chaque fois qu'une application est démarrée, une VM se crée, puis le processus meurt à la fermeture du programme.

2.4 Le Framework d'Application

La couche juste en-dessus correspond à l'ensemble des outils que Google mais à disposition des développeurs « Application Framework » .cette interface s'appuie sur les deux couches du même niveau citées auparavant.

En disposant d'une plateforme de développement libre, les développeurs indépendants et les responsables d'Android utilisent les mêmes interfaces afin de développer des applications riches et innovantes.

2.5 La couche Application

Enfin, La couche directement accessible est celle des applications. Lorsque vous achetés un équipement sponsorisé par Google, celui-ci est livré avec des programmes de bases (SMS, Contacts, …). Les applications sont développées en utilisant la technologie Java.

3 Les outils de la plate forme Android

Afin de programmer pour Android, plusieurs outils sont nécessaires, certains sont même fournis par Google.

Comme nous l'avons dit les applications pour Android sont développées en langage Java, qui est une technologie développée par Sun MicroSystems (maintenant Oracle). Présenté pour la première fois en 1995, il s'agit d'un langage orienté objet avec une syntaxe relativement proche du C++. La particularité de ce langage est qu'il est indépendant de la plateforme. Le même code pourra être exécuté sur un OS Windows et un OS Linux. Il s'agit du principe « Compile Once, run everywhere ».

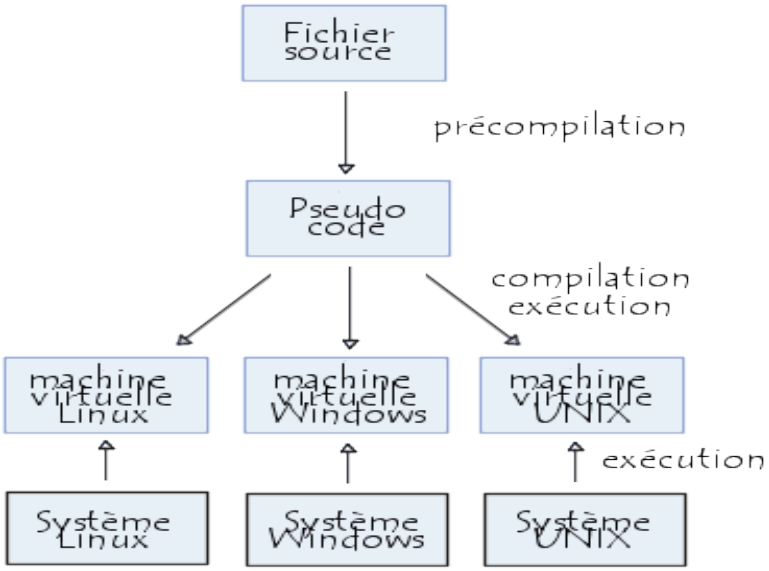

Figure 6 : Le concept de machine virtuelle java

Afin de réaliser cela, la technologie Java s'appuie sur une machine virtuelle (JVM), le code n'est donc pas ciblé pour une machine. En langage Java, le code est compilé pour obtenir du « byte code », il s'agit d'une étape intermédiaire. Ensuite ce code est exécuté par la JVM qui le transformera en code natif pour la machine.

3.1 L'environnement de développement Eclipse

Les applications Android utilisent principalement la plate-forme Java pour s'exécuter. Il existe bien un framework natif nommé NDK (Native Development Kit) permettant de développer en C/C++.

Pour développer et tester des applications en Java, n'importe quel environnement de développement convient. Il est tout à fait possible de ne pas utiliser d'environnement de développement intégré (ou IDE) si on préfère employer un éditeur de texte favori.

Pour le développement des applications Android, l'Open Handset Alliance et Google ont cependant souhaité mettre en avant Eclipse, l'environnement de développement de prédilection au sein de la communauté des développeurs Java. Pour les raisons de ce choix, voici quelques réponses :

- l'Open Handset Alliance tenait à mettre à disposition des développeurs des outils gratuits et ouverts. Eclipse est une plate-forme de développement Java réputée, largement utilisée, libre et gratuite et ne dépendant pas d'un éditeur en particulier ;

- Eclipse est disponible sous Windows, GNU/Linux et Mac OS. Cette portabilité entre les différents systèmes d'exploitation du marché signifie que les développeurs peuvent utiliser cet IDE pour leurs développements sur n'importe quelle machine.

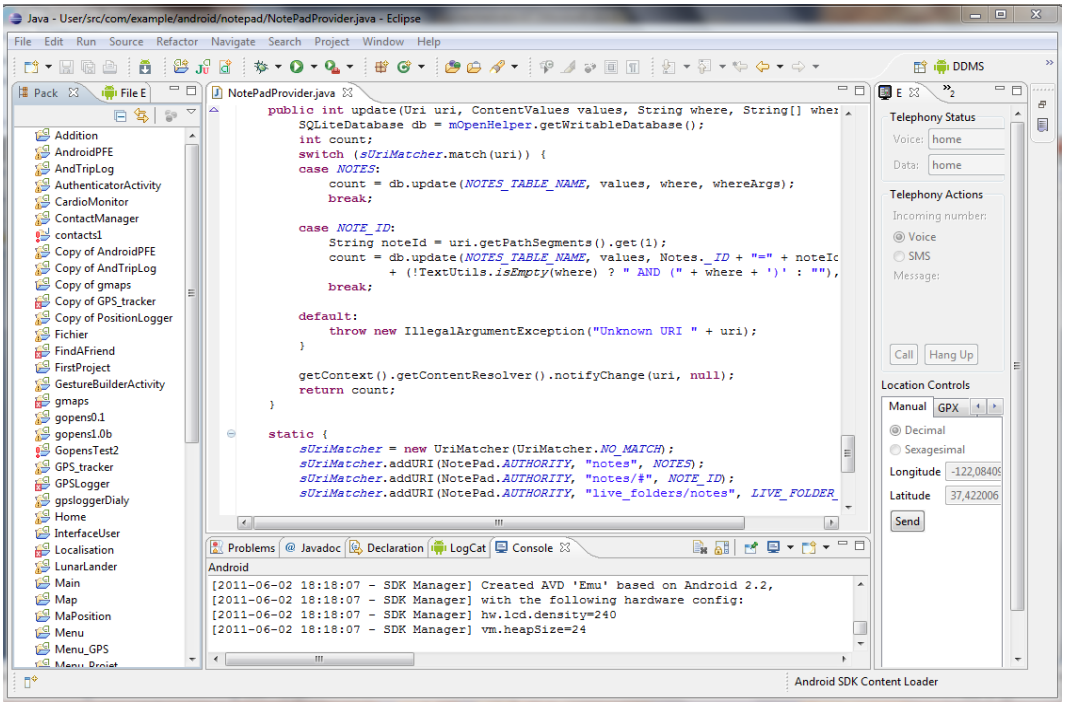

Figure 7 : L'environnement de développement Eclipse

3.2 Le kit de développement Android SDK

Exploiter une nouvelle plate-forme n'est jamais chose aisée. C'est pourquoi Google fournit, en plus du système d'exploitation, un kit de développement (Software Development Toolkit ou SDK). Ce SDK est un ensemble d'outils qui permet aux développeurs et aux entreprises de créer des applications.

Le SDK Android est composé de plusieurs éléments pour aider les développeurs à créer et à maintenir des applications :

- des API (interfaces de programmation) ;
- des exemples de code ;
- de la documentation ;
- des outils parmi lesquels un émulateur permettant de couvrir quasiment toutes les étapes du cycle de développement d'une application.

Le SDK Android est disponible gratuitement sur le site de Google.

Le SDK est livré avec un certain nombre d'outils couvrant différents aspects du cycle de développement d'une application Android. Ces différents outils seront présentés au fur et à mesure.

Le kit de développement propose une boîte à outils complète pour les tâches de compilation, débogage, génération de code AIDL, signature de l'application, etc.

Le répertoire tools du kit de développement contient ainsi un ensemble de programmes dont voici quelques exemples :

- DDMS : est un outil de débogage puissant ;
- Sqlite3 : permet d'accéder aux fichiers de données au format SQLite.

En plus des outils livrés avec le kit de développement, il existe une autre brique logicielle très utile : Android Development Tools Plugin ou ADT. Cet outil s'intègre directement à Eclipse et propose des interfaces et des assistants pour la création et le débogage des applications il permet aussi l'export d'un APK (paquet près à être installé).

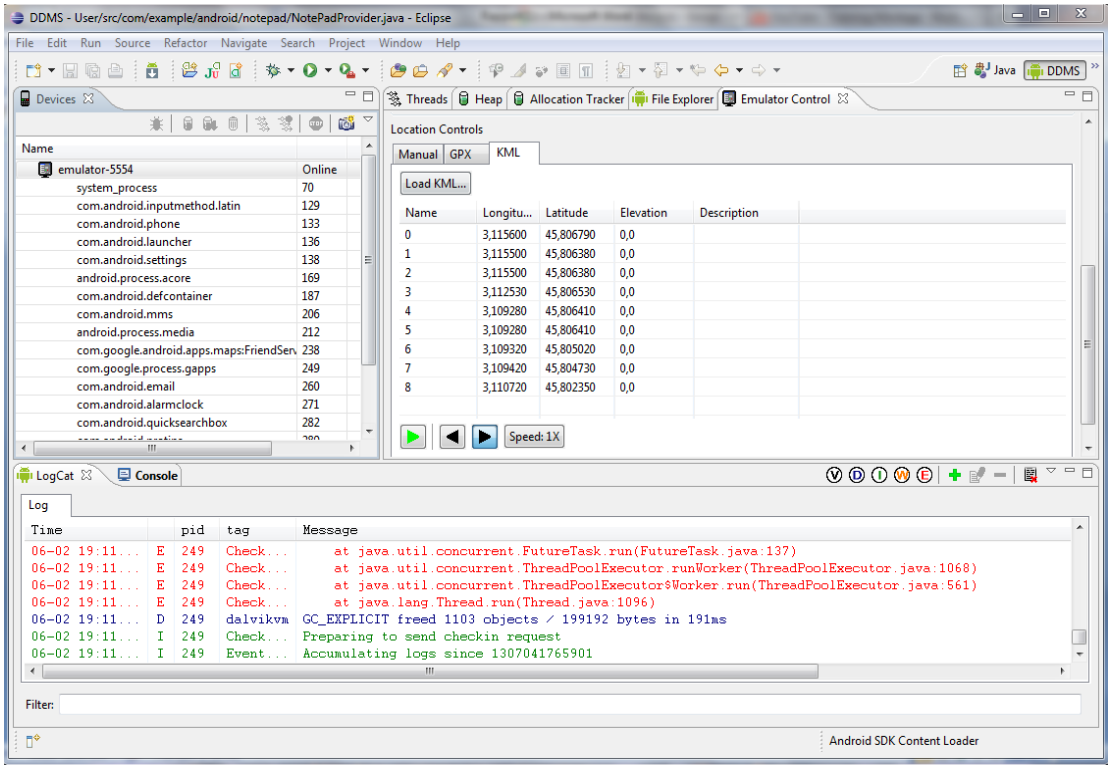

Figure 8 : Le plugin ADT pour développer sous Android

Sur cette capture d'écran, figure 8, nous sommes passés de la vue Java à la vue DDMS pour bien analyser le fonctionnement du plugin ADT, on se rend compte que celui-ci nous permet de gérer entièrement l'émulateur. A gauche nous avons l'état du système avec les processus en cours d'exécution. Dans le volet de droite on aperçoit différent onglets permettant de gérer le tas (pour les allocations dynamiques), les threads en cours d'exécution, un explorateur depuis lequel nous pouvons ajouter ou retirer des fichiers et enfin un onglet qui nous a particulièrement concerné, « Emulator control ».

Depuis cet onglet, on peut simuler des appels ou messages entrants mais mieux encore, simuler la position du téléphone en rentrant (ou utilisant des fichiers) des coordonnés GPS. La partie inférieur affiche le Log du téléphone, il agit comme un véritable filtre d'entrées de journal (erreur, avertissement, information et débogage) pour les applications utilisant l'API de journalisation d'Android, cette vue est la seule et unique méthode afin de savoir comment de comporte notre application.

3.3 L'émulateur

Le manque des dispositifs mobiles embarquant un OS Android, a poussé Google à développer un émulateur afin que tout le monde puisse tester son travail.

Figure 9 : L'interface de l'émulateur

Cet émulateur réagit exactement comme le vrai téléphone, on peut lui créer un espace de stockage, simuler des appels ou messages, etc. D'ailleurs c'est ce même émulateur que les ingénieurs Google en utilisaient avant d'avoir un prototype de téléphone, entre autres l'interface graphique a été réalisée sans posséder un réel appareil. On peut alors estimer que notre application se comportera de la même manière sur l'émulateur et le téléphone.

Conclusion

Après avoir détaillé l'ensemble des outils de développement mis en jeux, on procédera par la suite à la phase de modélisation et conception pour définir la structure générale de notre application.

Chapitre 4

Analyse et Conception

Ce chapitre est consacré à l'analyse et la conception de l'application, il commence par une étude préalable, après une description détaillée des fonctionnalités de l'application et l'identification des différents cas d'utilisation, ensuite il touche le périmètre technique de l'application.

1 Etude préalable

1.1 Recueil des besoins fonctionnels

La solution mobile permet aux agents mobiles de se déplacer sur terrain afin d'effectuer toutes les mises à jour nécessaires grâce à :

- **Affichage et navigation sur la carte :**

Lors de déplacement de l'agent sur terrain, il peut effectuer plusieurs manipulations, tel que le déplacement sur la carte pour se situer .ainsi il peut choisir les fonctions SIG de base « Zoom+, Zoom-», le choix des couches à afficher et la possibilité de centrer un objet donné sur la carte selon le besoin et sans retard ou perte de temps et avec une rapidité d'exécution.

- **Edition des données géographiques et attributaires :**

L'utilisateur peut sélectionner n'importe quel objet et consulter ses informations. Après la sélection de l'objet a édité, un formulaire s'affiche contenant les informations relatives à ce dernier, tout en effectuant les modifications adéquates dans les champs qui peuvent être édités et qui sont indiqués sur ce dernier. Une fois les nouvelles informations sont collectés, l'utilisateur enregistre les modifications effectuées, ou bien les annule et retourne à la carte.

- **Recherche :**

Pour faciliter à l'agent mobile la localisation des objets métiers et son repérage dans la carte, nous allons introduire un module de recherche attributaire et spatial de ces objets.

- **Localisation avec GPS :**

N'oublions pas la fonction GPS, lorsque l'utilisateur lance le GPS, une icône du GPS est affichée sur la carte. L'utilisateur choisit le GPS et un formulaire s'ouvre fournissant le statut actuel du GPS ainsi que ses informations relatives, s'il est en cours d'exécution, d'autre part cette fonction permet à l'agent mobile de se localiser pour collecter les différentes données nécessaires à sa mission.

- **Exportation de données :**

L'agent mobile peut exporter les données sauvegardées sous différents formats, à savoir « fichier KML » et « fichier TXT», et choisir l'emplacement où il veut stocker ces données, soit dans la mémoire du téléphone, soit dans la carte mémoire SD.

1.2 Recueil des besoins opérationnels

En plus de fonctionnalités suscitées dans le paragraphe précédent, nous avons relevé quelques besoins opérationnels que notre solution doit satisfaire :

- **Persistance :**

L'agent mobile collecte des données sur terrain pour mettre à jour la base de données en back-office ce qui implique la mise en œuvre des mécanismes de persistance et de gestion de cycle de vie de ces données.

- **Sécurité :**

Chaque agent mobile dispose d'un accès exclusif à la zone de son intervention. Ceci lui permet de protéger son travail et ses données acquises, mais aussi de retrouver le dernier état de sa ronde.

- **Intégrité de données :**

L'intégrité est le mécanisme qui empêche la mise à jour simultanée d'une même entité par deux utilisateurs différents.

❖ **Gestion de temps de réponse :**

Le traitement d'une demande sur place requiert une réponse instantanée de la part de l'agent.

1.3 Description du contexte du système

Une fois ce premier recueil de besoins effectué, la description de contexte du système peut commencer. Elle consiste en trois activités successives :

- L'identification des acteurs.
- L'identification des flux de communication.
- La réalisation de diagramme de contexte.

❖ **L'identification des acteurs :**

Les principaux acteurs du système sont les personnes ou les systèmes connexes qui interagissent directement avec le système. Dans notre cas, ce sont les acteurs : l'agent desktop et agent mobile.

✓ **Agent mobile :**

Il utilise les fonctionnalités de l'application pour la collecte de données sur terrain.

✓ **L'Agent desktop:**

Il récupère les fichiers contenant les données collectés pour les traiter a part, dans un système d'information géographique desktop.

❖ **Identification des flux de communication :**

Notre solution émet :

- Les données géographiques et attributaires aux agents mobiles.
- le suivi des événements des agents mobiles.
- Les fichiers de données à l'agent desktop.

Notre solution reçoit :

- Les créations, modifications des données des agents mobiles.
- Demande des fichiers contenant les données, par l'agent bureau.

❖ **Diagramme de contexte :**

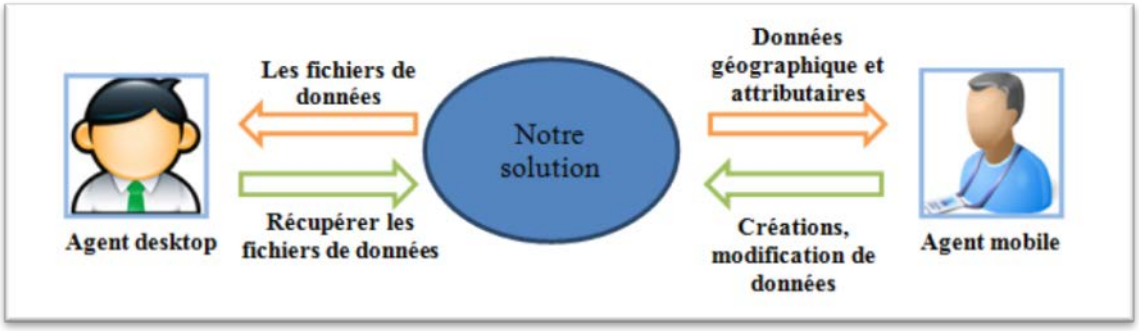

Figure 10 : Diagramme de contexte

2 Périmètre fonctionnel

❖ **Rappel du contexte de projet :**

Notre solution à comme but principal, la collecte de données ainsi que le rassemblement de ces données dans des fichiers de format « KML » ou « TXT », ces fichiers qui vont servir à une autre utilisation dans un système d'information géographique desktop.

2.1 Les Acteurs

Acteur	Type	Description de son rôle
Agent mobile	Humain	Il utilise les fonctionnalités de l'application pour la collecte de données sur terrain
Agent desktop	Humain	il récupère les fichiers générés par l'application pour les traiter les données ailleurs.

Tableau 1 : principaux acteurs du système

2.2 Description des cas d'utilisation

Le diagramme de cas d'utilisation ci-dessous décrit les possibilités d'interaction entre les acteurs et l'application, ainsi que ses différents besoins fonctionnels.

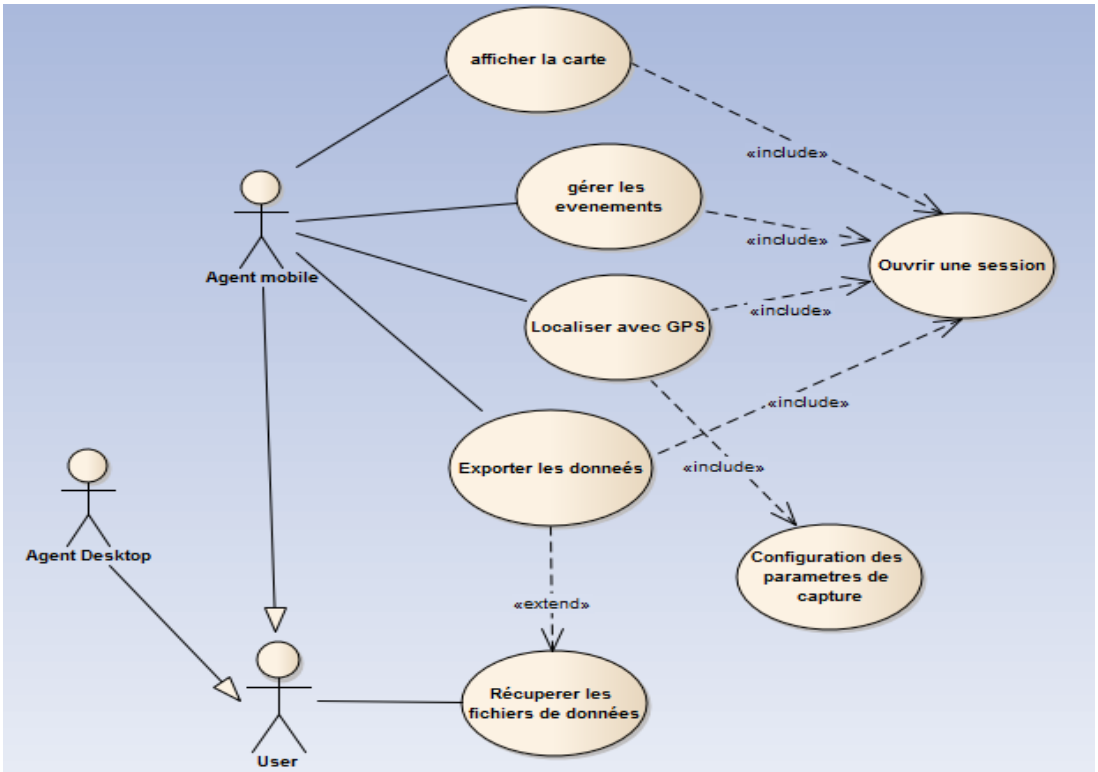

Figure 11 : diagramme de cas d'utilisation général

L'agent mobile est le seul acteur qui a la main sur l'ensemble des fonctionnalités offertes par notre application, après qu'il ouvre une session, il peut configurer la capture des données, afficher la carte, utiliser la fonction GPS pour se localiser.etc.

2.3 Les cas d'utilisation détaillés

2.3.1 Affichage de la carte :

Le but de cette fonctionnalité est de permettre à l'agent de terrain de visualiser les objets d'infrastructure afin de les localiser et les identifier.

- **Description textuelle du cas d'utilisation**

Ce cas permet l'affichage de la carte avec ses différentes couches de données, avec la possibilité de manipuler les outils de navigation et d'identification des objets qui se trouvent sur la carte.

Nom de cas d'utilisation	**Affichage de la carte**
Acteurs	- L'agent mobile
Contexte de déclenchement	- L'agent mobile choisit d'afficher la carte
Pré-conditions	- Ouvrir une session - Connexion internet
Post-conditions	- Connexion internet réussie
Contraintes de performance	- Les supports mobiles sont moins puissants en termes de Mémoire et processeur

Tableau 2 : description de cas d'utilisation « Affichage de carte »

❖ **Description des scénarios de cas d'utilisation :**

Nom de cas d'utilisation	**Affichage de la carte**
Scénario nominal	- L'agent mobile choisit d'ouvrir la carte. - L'application affiche la carte.
Scénario d'exception	Si la connexion internet n'existe pas, l'application affiche un message d'erreur demandant de se connecter à l'internet.

Tableau 3 : description du scenario « afficher carte »

❖ **Diagramme de séquence :**

Ce diagramme détaille les messages échangés entre le système, l'agent mobile et le serveur Google maps. Lorsque l'agent mobile demande à l'application d'afficher la

carte, celle-ci demande la permission d'afficher la carte au serveur Google maps, si la permission est accordée l'application affiche la carte, ceci se fait en mode connecté.

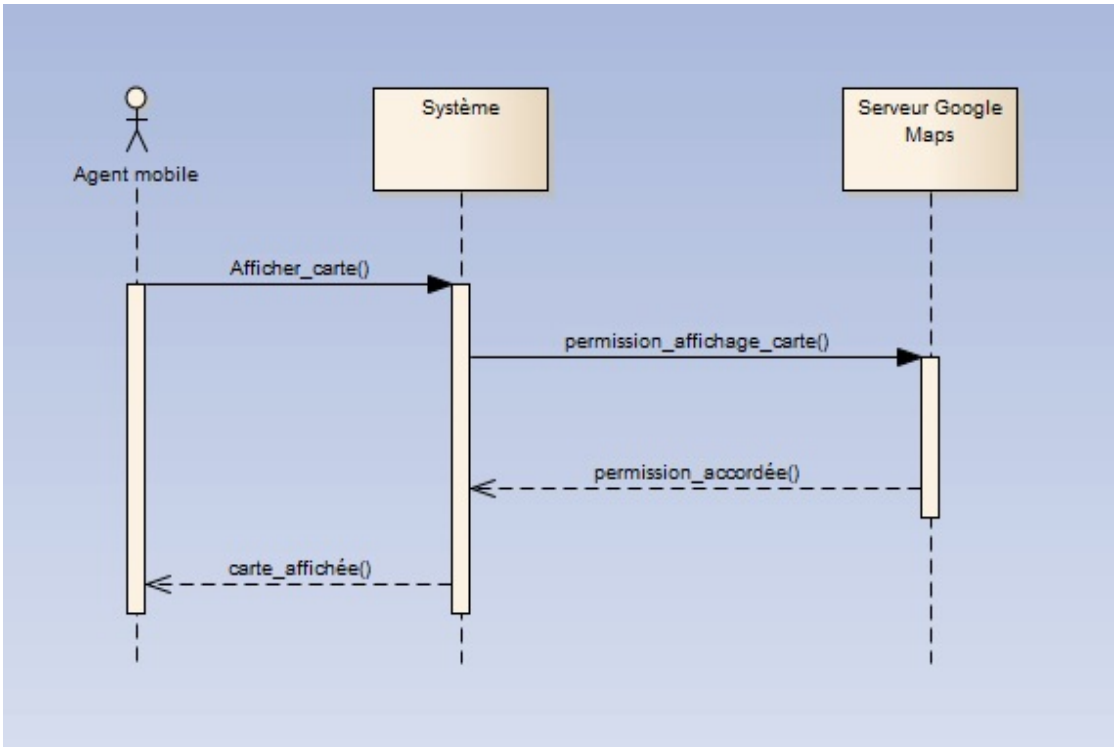

Figure 12 : Diagramme de séquence « afficher la carte »

2.3.2 Localisation par GPS

Le but est de permettre à l'agent de terrain de localiser les objets métiers sur le terrain grâce au GPS lors de sa mission.

❖ **Description textuelle du cas d'utilisation :**

Ce cas d'utilisation permet la manipulation du GPS en termes de, création d'une connexion GPS, consultation des informations GPS et affichage de la position GPS sur la carte.

Cas d'utilisation	**Localisation par GPS**
Acteurs	L'agent mobile.
Contexte de déclenchement	L'agent mobile demande au système la fonction GPS.
Pré-conditions	L'agent mobile doit ouvrir sa session et configurer les paramètres de capture. Le GPS doit être lancé (active).
Post-conditions	Le nombre de satellites fixes du GPS doit dépasser 4.
Cas d'exception	connexion GPS Statut GPS
Autres contraintes non fonctionnelles.	Protocoles de communication entre le GPS et le support mobile

Tableau 4 : description du cas d'utilisation « Localisation par GPS »

❖ **Description des scénarios du cas d'utilisation :**

Nom du cas d'utilisation	**Localisation par GPS**
Scénario nominal	Localisation des objets métier (Latitude et Longitude) ✓ L'agent mobile demande au système sa position actuelle. ✓ Le système vérifie si la connexion GPS existe. ✓ Si la connexion n'existe pas alors il faut exécuter l'exception(1) ✓ Le système vérifie le statut GPS ✓ Si le statut GPS ne permet pas la localisation, alors il faut exécuter l'exception(2) ✓ afficher la carte. ✓ Le système affiche la position actuelle sur la carte ✓ Le système affiche les coordonnées.

	✓ L'agent enregistre (éventuellement) la position GPS sur la carte.
Scénario alternatif	Affichage du statut GPS ✓ L'agent mobile demande au système d'afficher le statut GPS ✓ Le système affiche le statut GPS
Scénario d'exception	[Exception 1 : GPS désactivé] Le système affiche un message demandant d'activer le GPS. [Exception 2 : Statut GPS] Le système affiche un message si le statut GPS ne permet pas la localisation.

Tableau 5 : Description du scénario « localisation par GPS »

❖ **Diagrammes de séquence :**

- Affichage de statut de GPS :

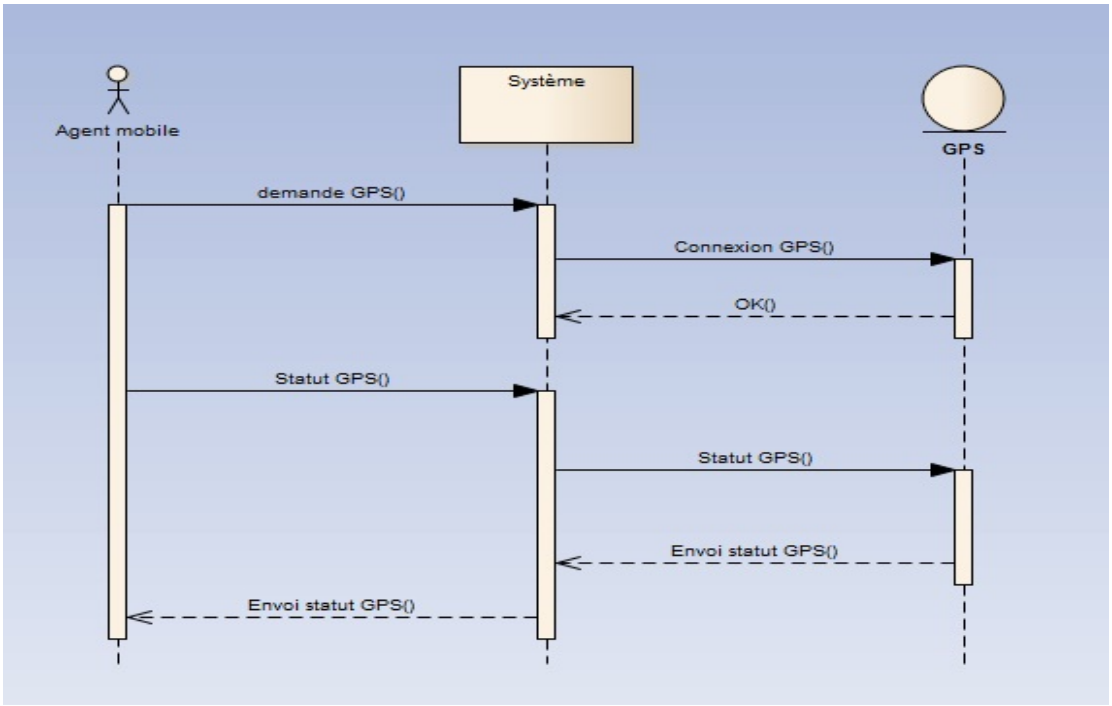

Figure 13 : Diagramme de séquence de l'affichage de statut de GPS

- Affichage de la position GPS :

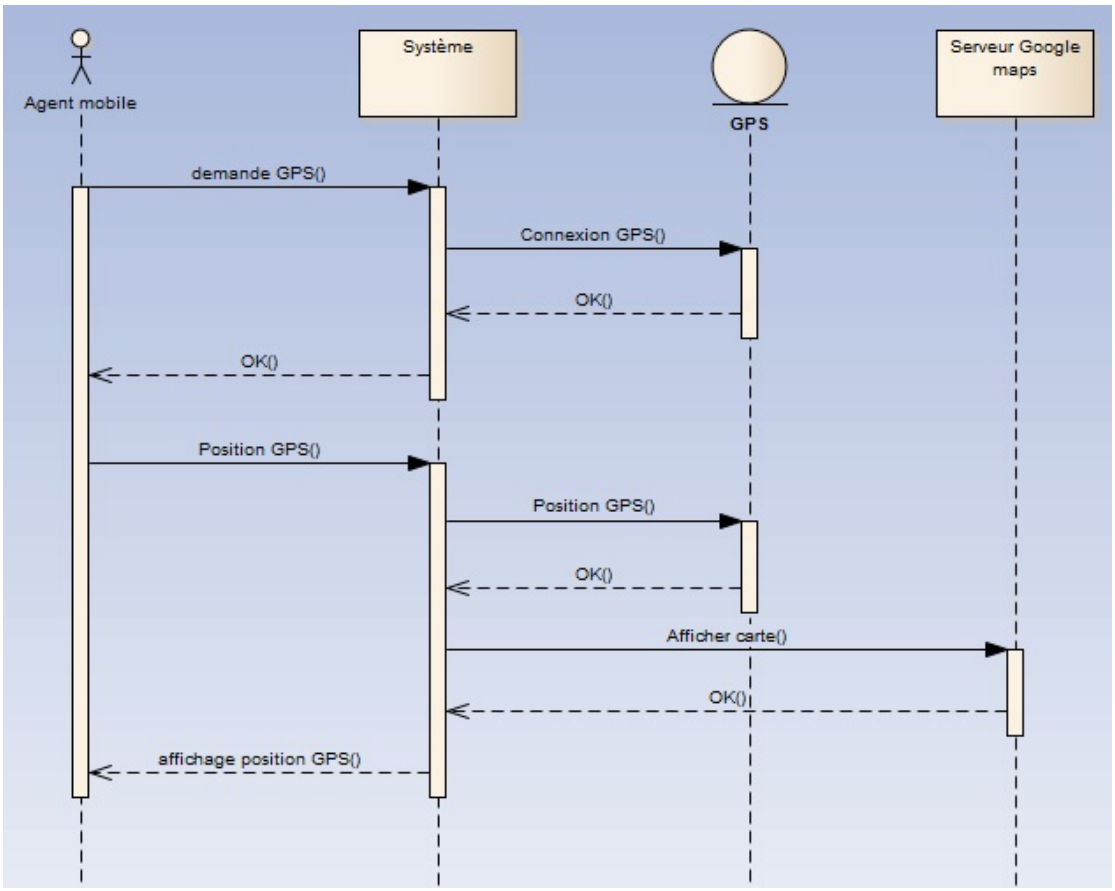

Figure 14 : Diagramme de séquence d'affichage de position GPS

2.3.3 Exportation des données

But : Permettre à l'agent mobile d'exporter les données collectées dans des fichiers de différents formats.

❖ **Description textuelle du cas d'utilisation :**

Ce cas d'utilisation permet d'exporter les données collectées par l'agent mobiles sous format « KML » ou « fichier texte » pour permettre à l'agent desktop de les récupérer afin de mettre à jour la base de données du système d'information géographique desktop.

Nom du cas d'utilisation	**Exportation des données**
Acteurs	Agent mobile
Pré-Conditions	-L'utilisateur doit ouvrir la session. -Les données doivent être collectées.
Post-Conditions	-Données collectées.
Contraintes de capacité de stockage	Le volume de données ne doit pas dépasser la capacité de stockage du support mobile

Tableau 6 : Description du cas d'utilisation « exportation de données »

- **Description des scénarios du cas d'utilisation :**

Nom du cas d'utilisation	**Exportation de données**
Scénario nominal	-L'utilisateur demande au système d'exporter les données. -L'utilisateur choisit sous quel format il veut exporter les données. -L'utilisateur choisit l'emplacement où il veut sauvegarder les données. -L'utilisateur est notifié de la fin d'exportation.
Scénario d'exception	-Si les données n'existent pas, le système affiche un message d'erreur indiquant qu'il n'y a pas de données collectées.

Tableau 7 : Description du scénario « exportation de données »

❖ **Diagramme de séquence :**

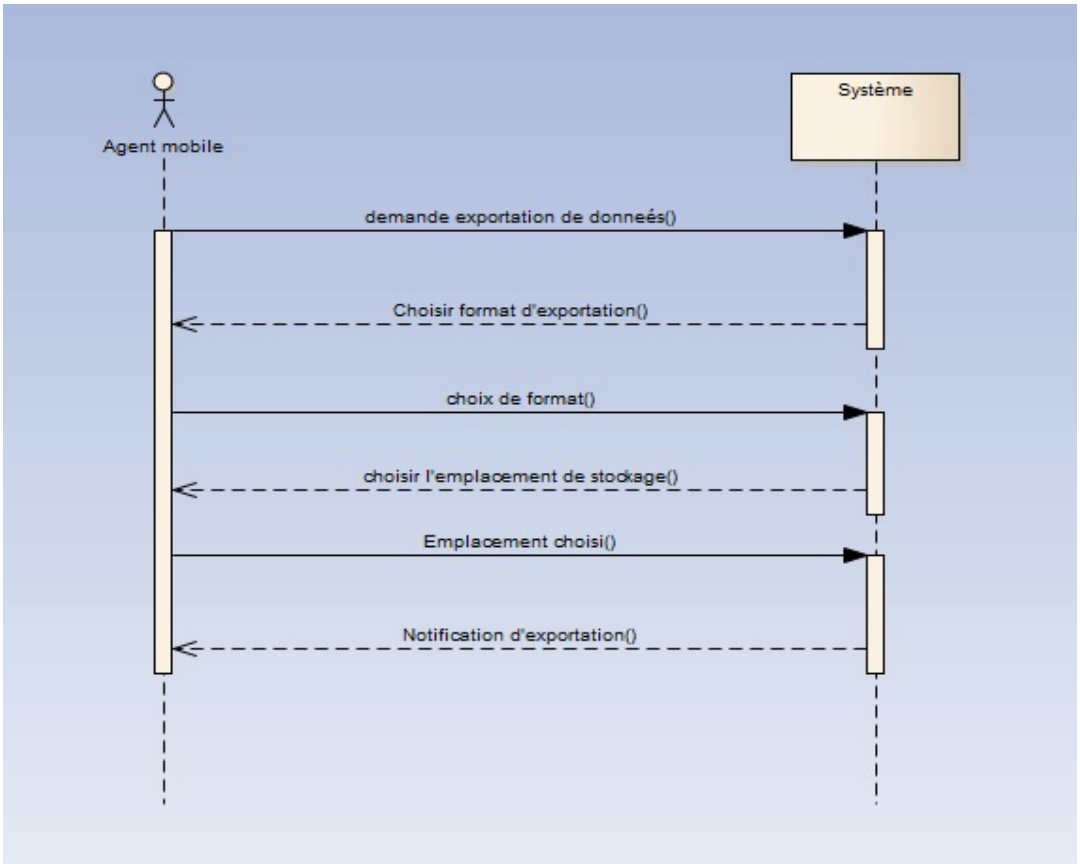

Figure 15 : Diagramme de séquence d'exportation de données

2.3.4 Récupération des données

But : permettre à l'agent desktop de récupérer les fichiers contenant les données collectées par l'agent mobile pour qu'elles soient traitées ensuite dans le système d'information géographique.

❖ **Description textuelle du cas d'utilisation :**

Ce cas d'utilisation permet à l'agent desktop de récupérer les fichiers de données exporté par l'agent mobile et sauvegardé dans un emplacement précis dans le support mobile.

Nom du cas d'utilisation	**Récupération des données.**
Acteur	Agent desktop
Contexte de déclenchement	Récupérer les fichiers de données.
Pré-conditions	-Les données sont déjà exportées par l'agent mobile. -L'emplacement des données existe.
Post-conditions	

Tableau 8 : description du cas d'utilisation « récupération des données »

- **Description des scénarios du cas d'utilisation :**

Nom du cas d'utilisation	**Récupération des données**
Scénario nominal	-L'utilisateur ouvre l'emplacement du fichier a récupérer. -L'utilisateur récupère le fichier

Tableau 9 : description des scénarios du cas « récupération des données »

❖ **Diagramme de séquence:**

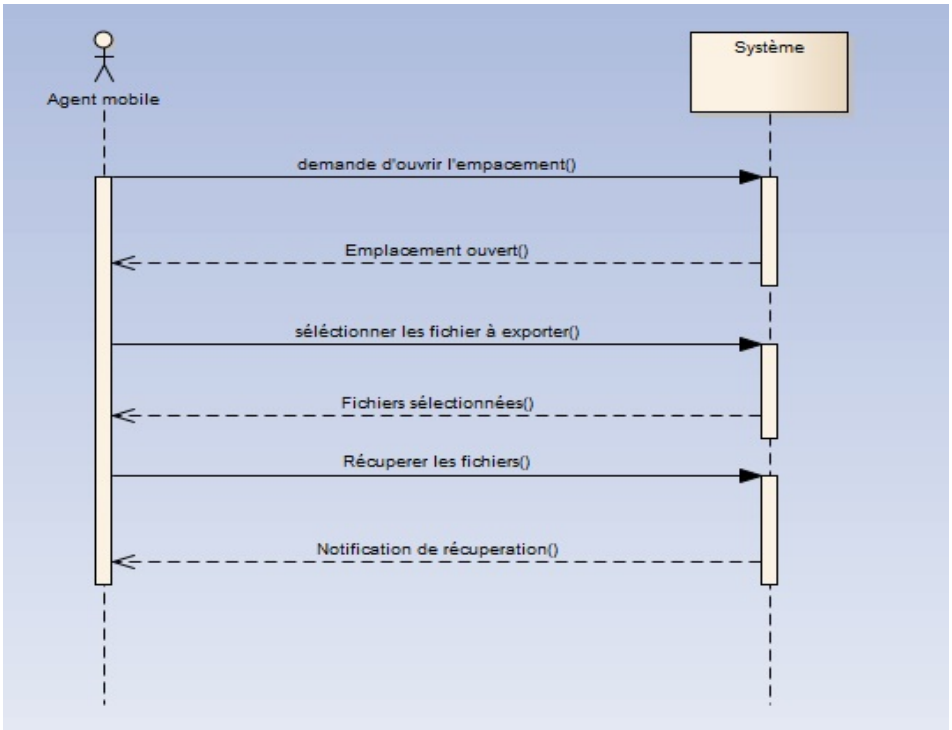

Figure 16 : Diagramme de séquence de récupération de données.

3 Périmètre technique

Notre analyse nous a permis de définir exactement le contexte de notre application, et la nature des informations manipulées. Maintenant il nous faut savoir comment organiser notre développement. Pour cela il nous fallait définir les composants techniques à mettre en place.

Dans le monde du génie logiciel, le but est de rendre les applications maintenables, pour cela une excellente habitude est de séparer le programme en «blocs ». Ainsi chaque blocs ou composant peut être interchangeable. Cependant cette méthode à tendance à rapidement alourdir les applications. Dans le domaine des systèmes embarqués le but est d'avoir des applications réactives, donc les règles de développement sont différentes, on évite la multiplication des classes afin d'accroitre les performances. De manière personnelle nous avons préféré dans un premier temps de garder un découpage logique de notre programme. De manière schématique notre développement peut être représenté ainsi :

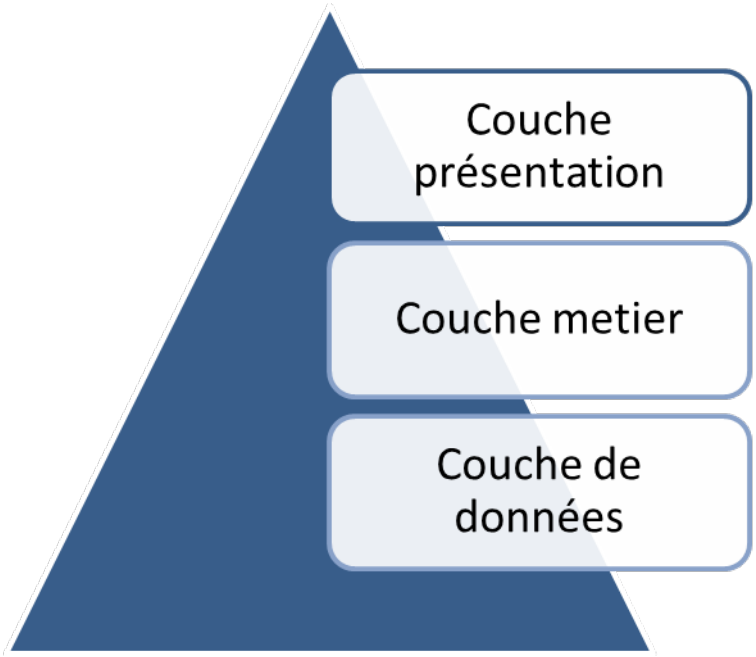

Figure 17 : Découpage de l'application

- **Couche présentation :**

Cette couche contient tous les composants graphiques du module composant l'interface homme-machine (fenêtres, contrôle utilisateur…) avec le code propre à l'affichage de leur représentation et de leur contenu.

- **Couche métier :**

Cette couche contient tous les composants métier. Ces composants métier prennent en charge la gestion du cycle de vie des objets métier géré par le module.

- **Couche de données :**

Cette couche est responsable du stockage physique des données. Elle permet de stocker les objets de la couche métier dans une base de données relationnelle.

Le diagramme de classes correspondant à notre application est schématisé ci-dessous :

3.1 Diagramme de classes :

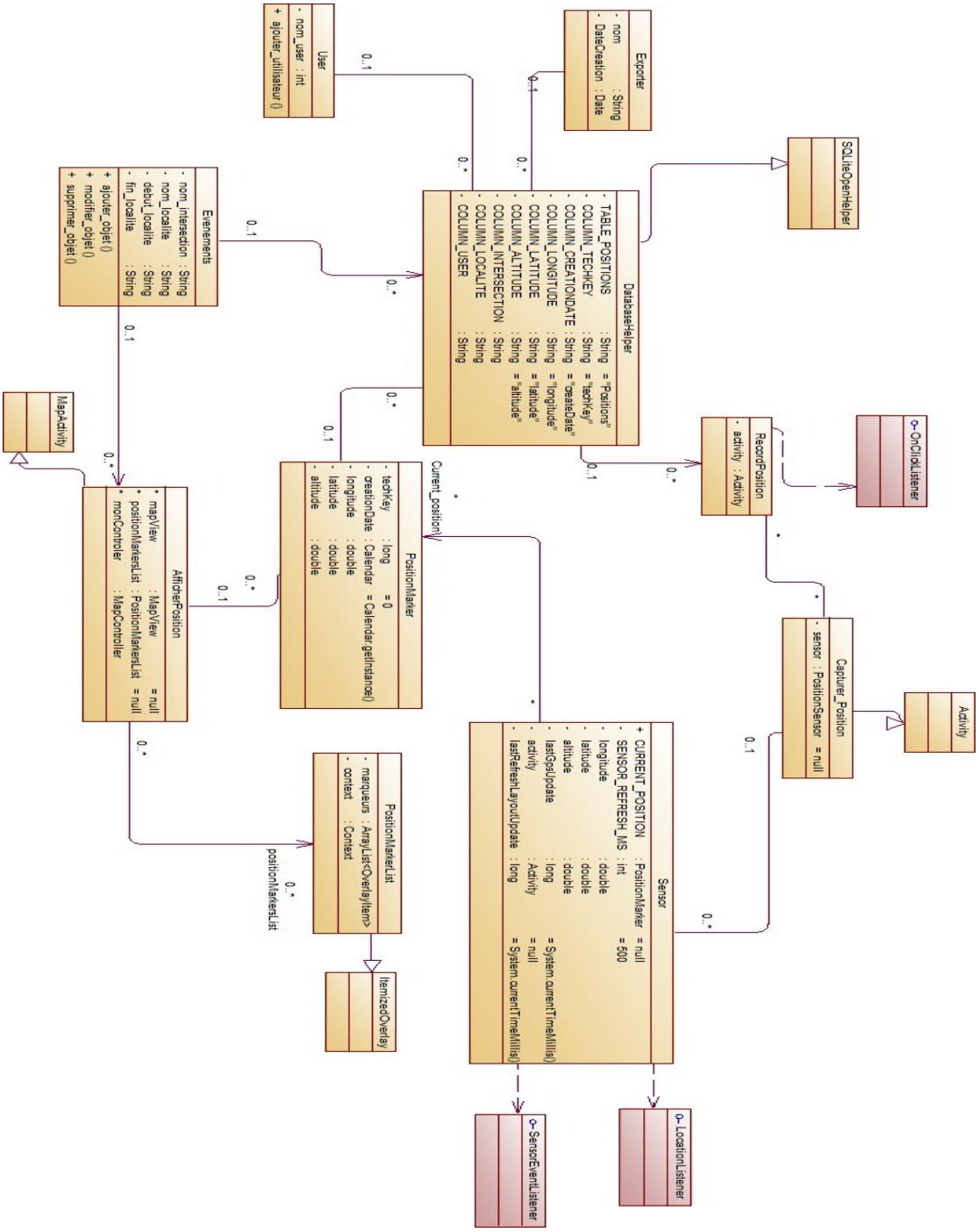

Figure 18 : Diagramme de classe

3.2 Les principaux composants techniques manipulés

En faite, notre diagramme de classe manipule plusieurs composants techniques, dont seuls les composants principaux seront détaillés par la suite.

3.2.1 Activités :

Deux activités principales sont mises en jeux dans l'application

- ✓ **Capturer_position :**

Cette activité est l'activité principale, elle permet à l'utilisateur de visualiser sa position courante (longitude, latitude) et de l'enregistrer.

Elle :

- étend la classe android.app.Activity.
- met en place la sonde « multiposition » Sensor.

- ✓ **Afficher_position :**

Il s'agit d'une activité de type « MapActivity », elle permet l'affichage des positions enregistrées sur une carte Google.

Elle :

- étend la classe com.google.android.maps.MapActivity.
- utilise un layout dédié à l'affichage de la carte (mapview.xml).
- configure l'affichage de la carte.
- récupère à partir de la base de données les positions enregistrées et les affiche sur la carte

3.2.2 Layouts :

- ✓ **Main.xml :**

Layout de l'activité principal, c'est ce layout qui permet la saisie/l'enregistrement de la position courante.

- ✓ **Mapview.xml :**

Layout utilisé pour afficher la carte.

3.2.3 Autres classes :

- ✓ **PositionMarker :**

Cette classe représente une position, il contient de nombreux attributs (longitude, latitude …). Il est utilisé entre autre pour la sauvegarde et la restitution des positions en DB.

- ✓ **PositionMarkerList :**

Contient la liste des marqueurs/points à afficher sur la carte.

3.2.4 Listener/Sensor utilisés :

- ✓ **Sensor :**

Sonde GPS/Accéléromètre/boussole.

3.2.5 Classes outils :

- ✓ **DatabaseHelper :**

Classe utilisée pour interagir avec la base de données.

Elle :

- Étend la classe android.database.sqlite.SQLiteOpenHelper.
- Permet la sauvegarde et la restitution des positions.
- Initialise la base de données si nécessaire (création de la table des positions).
- Utilise la classe android.database.sqlite.SQLiteDatabase pour transférer des ordres vers la base de données.

Conclusion

Après avoir détaillé le côté fonctionnel et technique de notre application

Il est temps de voir, de façon concrète, les fonctionnalités implémentées dedans.

Chapitre 5

Réalisation

Ce dernier chapitre présente les fonctionnalités de l'application qui ont été développées et testées. Il décrit les différentes interfaces qui implémentent ces fonctionnalités.

1 Création et affichage d'une carte

Après le lancement de l'application une carte affichant la position actuelle de l'utilisateur apparait grâce à l'utilisation de la puce GPS intégré à la tablette, si cette carte est configurée de telle façon qu'elle soit capable d'utiliser l'API Google Maps, alors dans ce cas une étape supplémentaire sera nécessaire : l'obtention d'une clé de licence.

Pour cela, la première étape consiste à se rendre sur le site proposé par Google:

http://code.google. com/android/maps-api-signup.html

Il faut également un compte Google pour compléter l'opération. Enfin, pour obtenir cette clé, l'interface du site demande une empreinte MD5 d'un certificat.

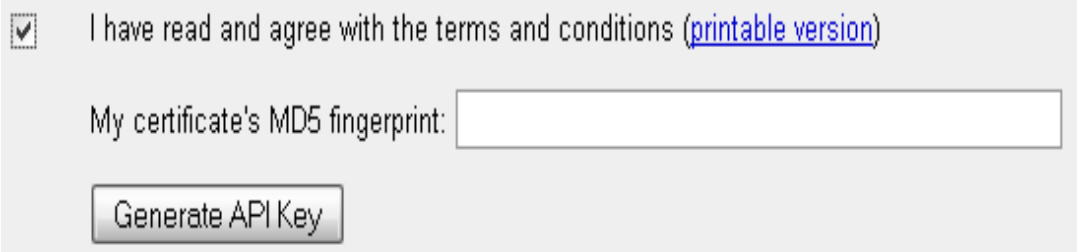

Figure 19 : Formulaire de génération d'une clé Google Maps

Android n'autorise l'installation que des applications signées d'où l'utilisation de cette clé. Avant d'installer une application via l'émulateur, Eclipse signe l'application en utilisant un certificat de débogage qui est fourni avec le SDK d'Android.

Eclipse crée le fichier debug.keystore lors de la compilation du projet, ce fichier étant stocké dans le répertoire Android dont le chemin varie en fonction du système d'exploitation. Dans notre cas, le chemin est « C:\Documents and Settings\Administrateur\.android\debug.keystore».

La clé générée pour utiliser l'API Google Maps est basée sur ce certificat de débogage. Pour en obtenir l'empreinte (et en se basant sur un système XP logué en administrateur), il faut utiliser la commande suivante :

keytool -list –keystore

"C:\Documents and Settings\Administrateur\.android\debug.keystore"

Puis taper **entrée** pour avoir le résultat illustré dans la figure suivante :

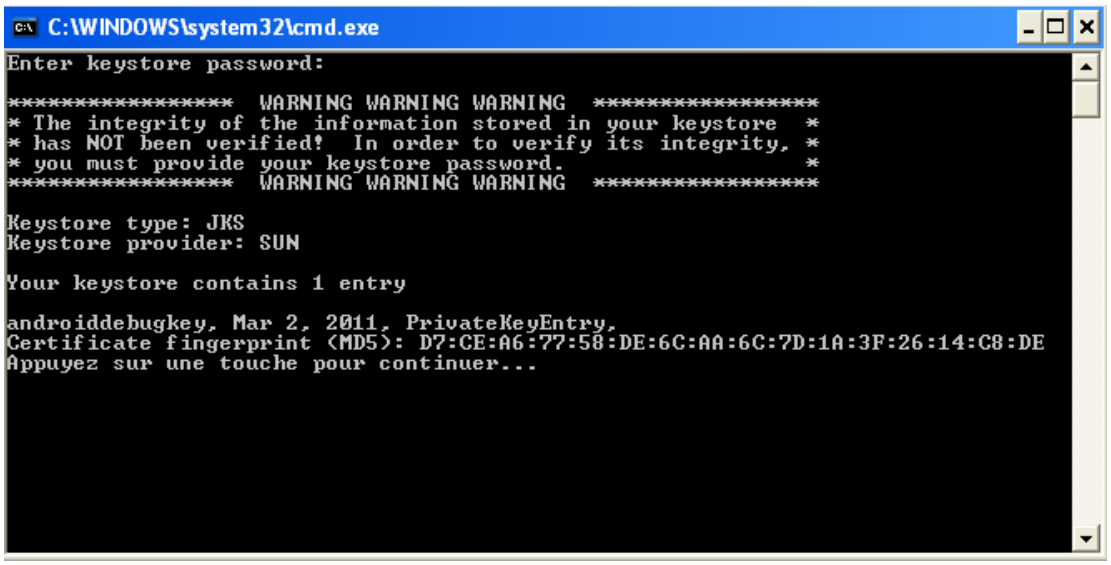

Figure 20 : l'empreinte du certificat de débogage MD5

Une fois l'empreinte saisie dans le formulaire et envoyée au site, la clé est générée. Dans notre cas la clé que nous avons obtenue est :

Figure 21 : la génération de la clé Google Map

Ensuite on ajoute la clé obtenue dans le fichier xml contenu dans le répertoire layout,

Dont voici un extrait :

<com.google.android.maps.MapView

android:layout_width="fill_parent"

android:layout_height="fill_parent"

android:apiKey="0N0InPb7nwvDrzPcX2D8Gr3VtWy34XESDYpbYoQ"

/>

Aussi bien il ne faut pas oublier d'ajouter dans le fichier de configuration **manifest.xml** de notre projet, la permission:

<uses-library android:name="com.google.android.maps" />

Ainsi, on pourra exploiter efficacement les outils cartographiques de Google Map.

Si tout se passe bien la carte doit être affichée comme dans la figure suivante :

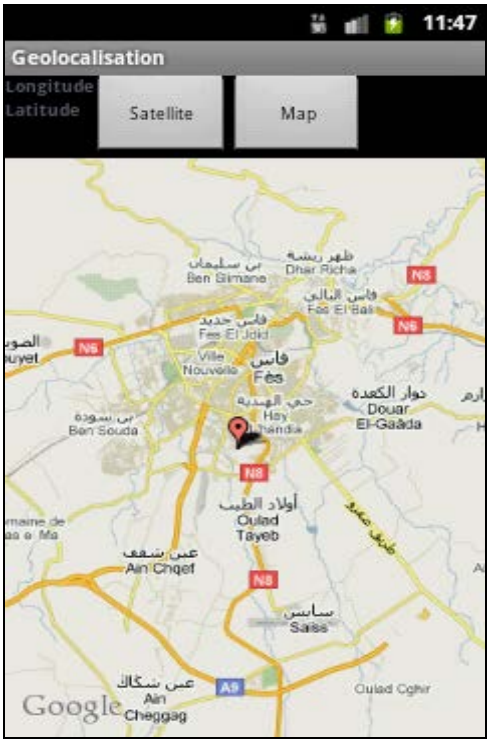

Figure 22 : affichage de la carte

C'est une carte fournit par google maps, qui affiche la position avec quelques informations de bases comme le nom de la place, bien sur cette opération n'est valable qu'à l'existence d'une connexion internet sinon la position sera indiquée uniquement par un point dans une interface blanche (un fond blanc).

Nous pouvons également afficher notre position en passant à la vue satellite sans aucun souci (figure 22).

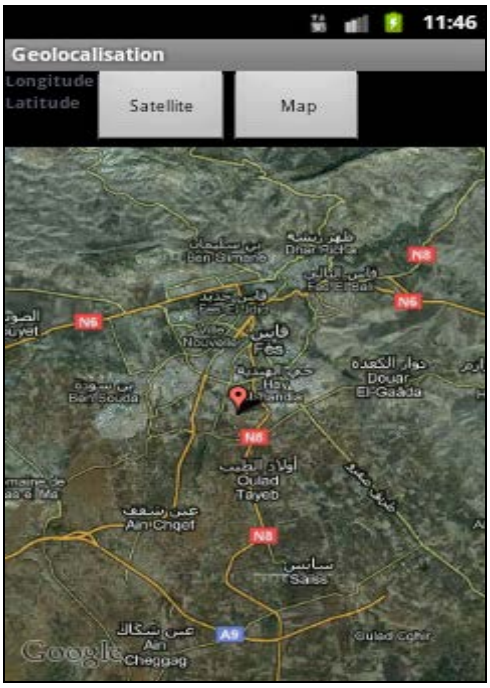

Figure 23 : La carte (vue satellite)

2 Les interfaces de l'application

Lors de son déplacement, l'utilisateur peut mettre à jour sa position toujours en utilisant la touche « menu » de l'émulateur, trois options apparaissent la première pour fournir la nouvelle position qui sera caractérisée par un marqueur, la deuxième pour ajouter un nouvel utilisateur et la troisième pour retourner à l'écran précédente (figure 24).

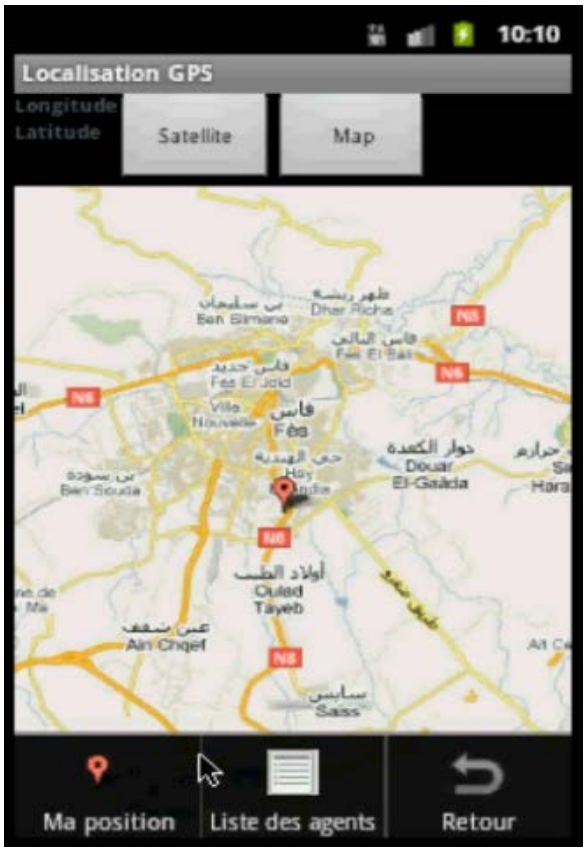

Figure 24 : Mise à jour de la position

L'utilisateur peut profiter aussi des fonctionnalités intéressantes comme le déplacement et le Zoom, ces options sont nécessaires car elles permettent une visualisation facile et professionnelle de la carte comme celle de Google Maps ou encore Google Earth.

Android de son côté nous offre plusieurs caractéristiques mais cela reste dépendant de nos exigences :

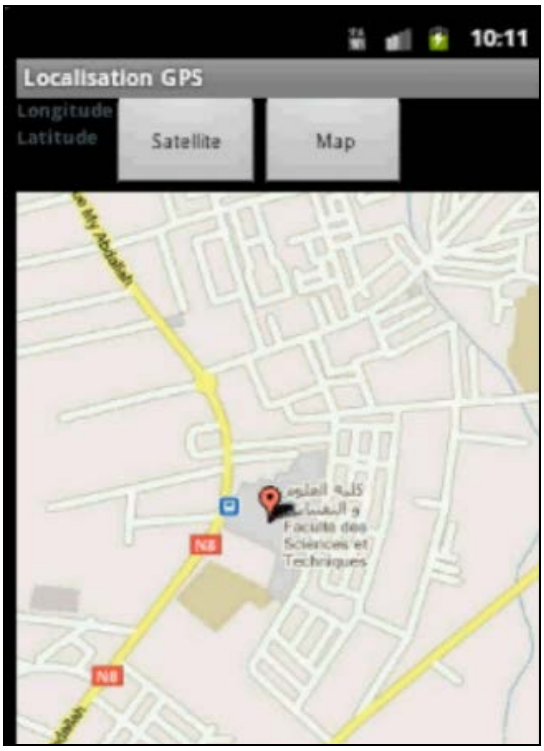

Figure 25 : Fonctionnalité Zoom

3 Ajout d'un nouvel agent mobile

La page d'accueil invite l'utilisateur à créer un pseudo-compte, le but de celui-ci est purement organisationnel, et donc l'utilisateur n'a qu'à entrer son nom ainsi que l'objet de sa mission. S'il veut annuler la création de son nouveau projet, il doit cliquer sur le bouton « menu », comme ça une case menue s'affiche en proposant de changer l'utilisateur courant.

Ceci est illustré via la figure 26 :

Figure 26 : Création d'un nouvel utilisateur

Dès qu'il soit créé, l'utilisateur s'ajoute à la liste des anciens agents mobiles et les informations seront sauvegardées instantanément dans la base de données.

L'agent peut consulter la liste complète de tous les utilisateurs de l'application en cliquant sur « menu » puis sur « Liste des agents ».

La figure ci-dessous montre l'exemple d'une telle liste :

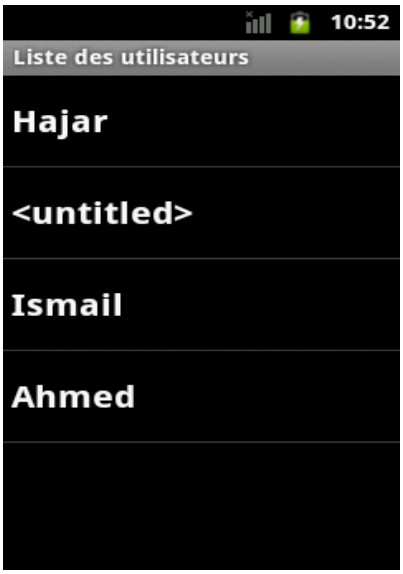

Figure 27 : Lise des utilisateurs

Si l'utilisateur confirme de vouloir procéder à l'édition de son compte, il en aura accès et cela en sélectionnant son nom puis en cliquant sur « menu » puis « Editer ».

Pendant cette opération l'utilisateur a la possibilité d'envisager quelques tâches, toujours via le bouton « menu ». Parmi ces tâches on trouve la fonctionnalité primordiale : l'enregistrement d'un parcours.

La figure suivante illustre ce qu'on vient de dire :

Figure 28 : Edition d'un utilisateur

4 Enregistrement du parcours

Si l'utilisateur veut enregistrer son parcours personnel, il doit alors cliquer sur le petit écran qui mène automatiquement vers une nouvelle interface.la figure suivante montre le résultat obtenu.

Figure 29 : interface d'enregistrement du parcours

Dans cette interface on remarque bien qu'elle contient des champs à afficher, ainsi que quelques options menu que nous allons en parler par la suite. A tout moment l'utilisateur peut modifier quelques options concernant la distance minimale et le temps minimum d'enregistrement entre deux positions données et cela grâce à la fenêtre « Options ».

Figure 30 : Options d'enregistrement

En cliquant sur le bouton « Start » L'interface collecte automatiquement les coordonnées de l'utilisateur en question qui sont envoyées par les satellites qui couvrent sa position, et reçus grâce à la puce GPS du terminal mobile.

Pour simuler une telle situation dans notre émulateur, Android nous fournit l'outil « Emulator control », qui nous permet de simuler la position d'un terminal mobile soit en envoyant la latitude et longitude correspondante à un endroit donné (Figure 32), ou en utilisant des fichiers KML ou GPX qui contiennent plusieurs points géographiques. (Figure 31).

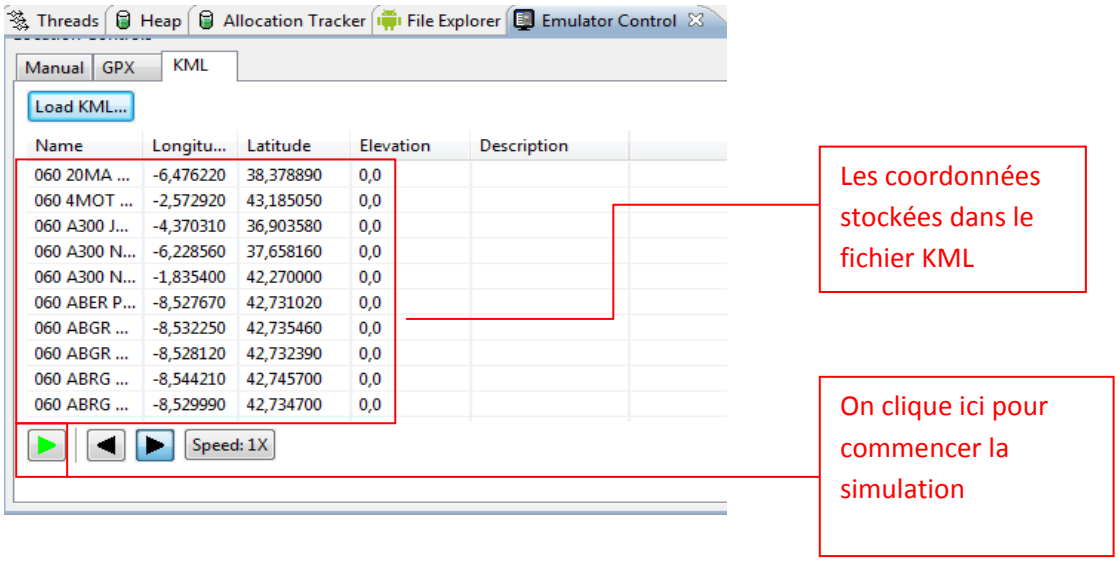

Figure 31 : simulation des coordonnées GPS à partir d'un fichier KML

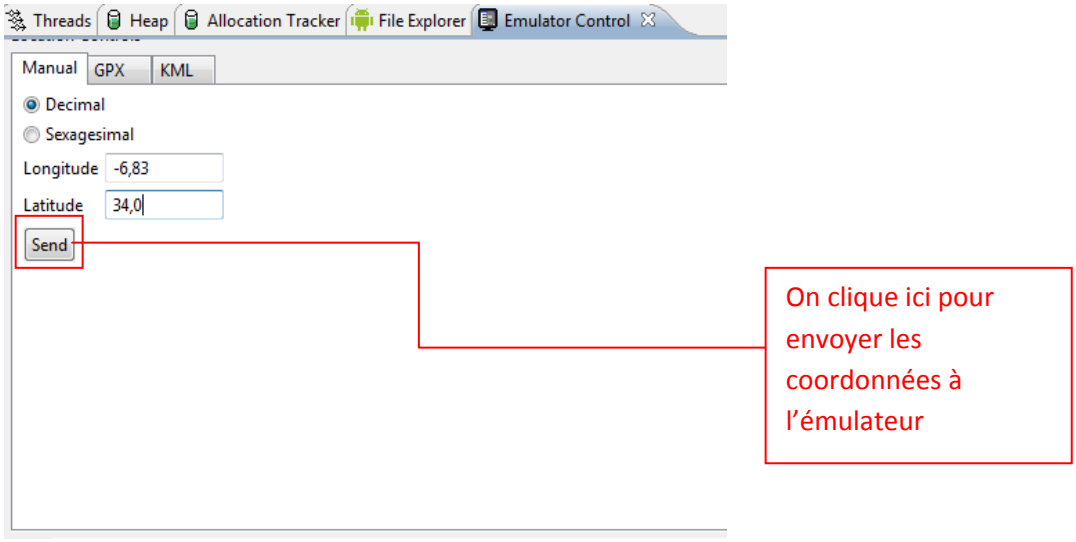

Figure 32 : Simulation manuelle des coordonnées GPS

Les deux figures ci-dessus illustrent les deux alternatives permettant un contrôle de location idéal :

- Manuel via le bouton « Send » qui transmet les coordonnées déjà saisies directement vers l'émulateur.

- A partir d'un fichier KML déjà crée et qui est chargé via le bouton « Load KML », puis on clique sur le bouton à flèche vert pour lancer la simulation automatiquement sur l'émulateur.

Le résultat de la situation en question est représenté par la figure ci-dessous. Dans un premier temps notre application attend l'arrivée des coordonnées, ensuite lorsqu'on exécute l'une des deux méthodes on se trouve dans un état semblable à celle de l'interface à droite.

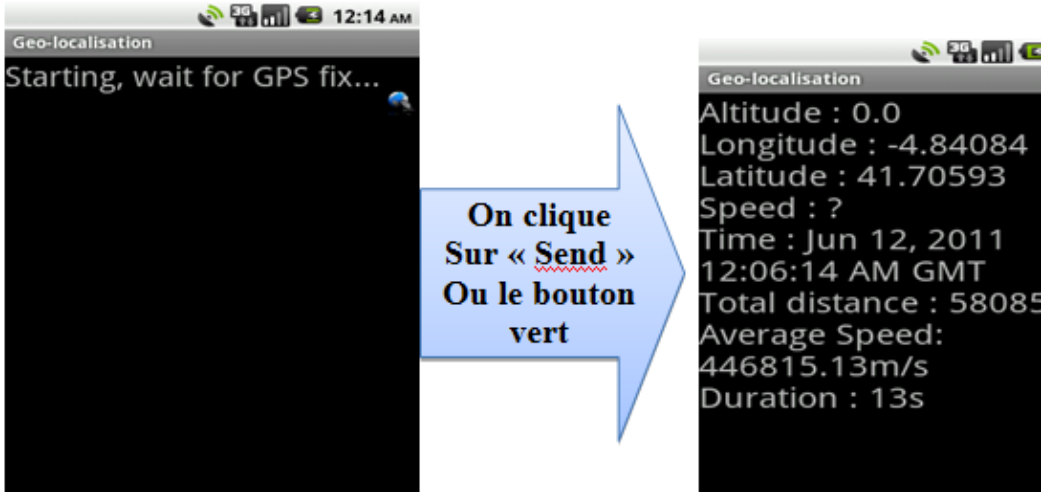

Figure 33 : Capture des coordonnées

5 Liste des enregistrements

Comme on voit dans la figure à droite, les coordonnées sont bel et bien envoyées et récupérées par notre application, ce qui signifie que nous avons réussi à entamer l'enregistrement de notre parcours. Les informations qui seront stockées directement dans la base de données sont l'altitude, la longitude, la latitude, la vitesse, la date, la distance totale parcourue et la durée.

Lorsque l'utilisateur désire arrêter sa sauvegarde il est invité à cliquer sur le bouton menu, alors des options s'affichent de tel sorte que le bouton « Stop » sera activé et les autres boutons « Start », « Tout Supprimer » et « Exporter » seront désactivés. Voir la figure ci-dessous :

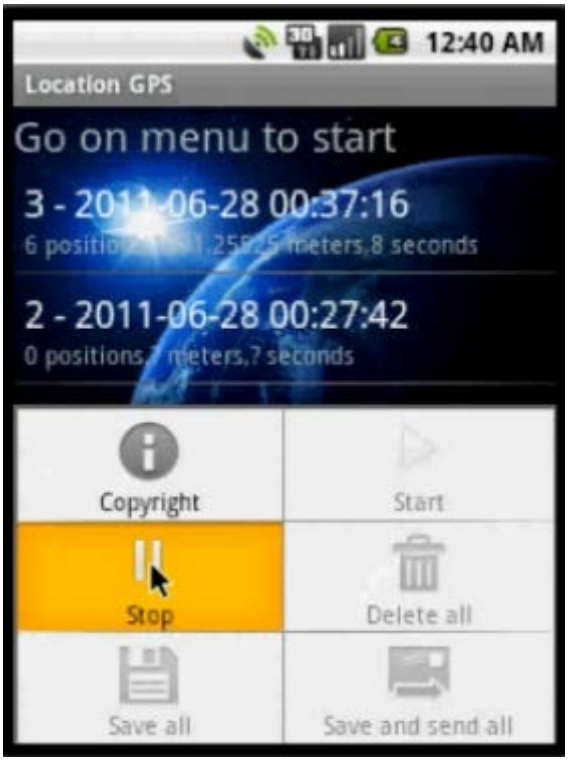

Figure 34 : l'Arrêt de l'enregistrement

Ensuite, le dernier enregistrement s'ajoute automatiquement aux anciens enregistrements, la liste de ces derniers sera affichée sous forme d'une « list view » Tout en indiquant les informations qui sont stockées ainsi que le numéro de position, parmi ces informations on trouve la date, les positions, la distance, et la durée.

Figure 35 : Liste des enregistrements

6 Exportation des fichiers:

Une fois l'enregistrement terminé, l'utilisateur a la possibilité d'exporter son enregistrement en tant que fichier texte, ou en tant que fichier KML, et ceci via le bouton menu en choisissant l'option « Exporter ». Cette option est disponible dans cette application car on a l'intérêt de temps à autre de traiter nos enregistrements d'une façon indépendante c'est-à-dire les utiliser dans un cadre « hors l'application ». Voici la figure correspondante :

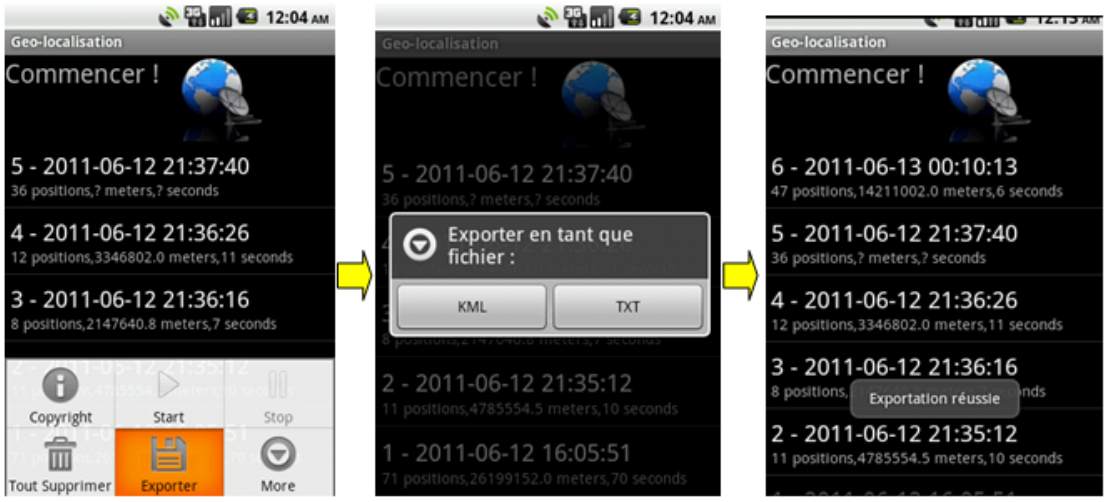

Figure 36 : Exportation d'un enregistrement

Voici un exemple de deux fichiers générés après cette opération, un de type fichier texte et l'autre de type fichier KML :

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.2">
  <Document>
    <name>Paths</name>
    <description>Start date : 19-4-2011 15:11:10
                 End date : 19-4-2011 15:5:46
                 Minimum altitude : 0
                 Maximum altitude : 88
    </description>
    <Style id="Line">
      <LineStyle>
        <color>7fff0000</color>
        <width>2</width>
      </LineStyle>
      <PolyStyle>
        <color>7f00ff00</color>
      </PolyStyle>
    </Style>
    <Placemark>
      <name>Absolute Extruded</name>
      <description></description>
      <styleUrl>#Line</styleUrl>
      <LineString>
        <extrude>1</extrude>
        <tessellate>1</tessellate>
        <altitudeMode>absolute</altitudeMode>
        <coordinates>
          -6.88675,33.9383,0  -6.88675,33.9383,0  -6.8868,33.9376,88 -6.8868,33.9376,88   -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88
          -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88
          -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88  -6.8868,33.9376,88
          -6.8868,33.9376,88
        </coordinates>
      </LineString>
    </Placemark>
```

Figure 37 : Exemple de fichier texte exporté

Après la phase de l'enregistrement, l'utilisateur peut visualiser le parcours qu'il vient de sauvegarder sur une carte Google Map par exemple. Pour ce faire il suffit de se rendre au site de Google Maps pour créer sa propre carte à partir du fichier KML exporté. La figure suivante montre clairement un trajet que nous avons déjà enregistré à l'aide de la tablette « Samsung Galaxy » dans l'entourage de la société.

Figure 38 : Trajet visualisé à travers le site de Google maps

Conclusion et perspectives

Notre projet de fin d'études consiste à mettre en place une solution mobile pour la collecte des données. L'agent mobile peut à travers cette solution, collecter les données grâce à un GPS, naviguer sur la carte en toute liberté avec des outils simples à utiliser. Il peut également consulter, modifier, supprimer ou créer des nouveaux objets métiers, de plus, il peut sauvegarder les données collectées dans des fichiers de formats permettant d'afficher l'information géographique sans développement côté client.

Durant les trois mois de notre projet de fin d'étude, nous avons pu aboutir aux objectifs fixés au début de ce projet. Nous sommes donc arrivés à réaliser une application, Cependant, nous avons eu quelques dérives en termes de délai par rapport au planning prévu.

La réalisation de ce projet a été une excellente mise en pratique des connaissances acquises tout au long de notre cursus. Ce stage nous a permis de les approfondir grâce à sa richesse technique et aux nombreuses notions utilisées. Il nous a également offert la possibilité d'appliquer les différentes étapes de Gestion de Projet, aussi la possibilité de découvrir la plate-forme Android, nous avons compris la logique de ce système.

Le fait de mener à bien un tout nouveau projet depuis le début, analyser les besoins, élaborer le cahier des charges, concevoir la solution, proposer une architecture logicielle, choisir les outils et technologies susceptibles de convenir à sa réalisation, et développer enfin les différentes fonctionnalités proposées, sans s'appuyer sur un existant, nous a appris à forger une schématique d'analyse fonctionnel et technique, de conception, et de réalisation, très intéressante.

Le travail réalisé peut être amélioré, et parmi les perspectives d'améliorations possibles, l'ajout des autres fonctionnalités capables de faciliter plus en plus le travail sur terrain, le transfert de données collectés au serveur SIG desktop pour mettre à jour la base de données géographique.

Bibliographie et Webographie :

- *Damien Guignard, Julien Chable, Emmanuel Robles, Programmation Android de la conception au déploiement avec les SDK Google Android 2, avril 2010, Edition Eyrolles.*
- *J.F. DiMarzio, Android- A Programmer's Guide, 2008, Edition The McGraw-Hill Companies.*
- Florent Garin, *Développer des applications mobiles pour les Google Phones, 2009, Edition Dunod.*
- Daniel PETISME, Anthony VINCENT, Rapport intitulé *développement d'une application de localisation pour la plate-forme Android* ,2010. Disponible sur : http://isihiking.googlecode.com/svn-history/r27/trunk/Rapport.docx.
- *Ismail HAMMOUDI, Ahmed BOUTOR, Mémoire de projet de fin d'étude, 2011.*
- Site officiel d'Android : http://developer.android.com/index.html.
- Encyclopédie Libre : http://fr.wikipedia.org.