MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI DE TIZI OUZOU

FACULTE DE GENIE ELECTRIQUE ET DE L'INFORMATIQUE

DEPARTEMENT D'ELECTRONIQUE

Mémoire de Magister

En vue de l'obtention du diplôme de Magister en électronique

Option télédétection

Présenté par : **M^r^ ZERROUKI Fodil**

Intitulé :

Conception et réalisation d'une carte d'acquisition ambulatoire de transmission sans fil et de traitement de signaux biomédicaux

Jury

Mr AMEUR Soltane	Professeur à l'UMMTO	Président
Mr HADDAB Salah	Maître de conférences A à l'UMMTO	Rapporteur
Mr LAGHROUCHE Mourad	Professeur à l'UMMTO	Examinateur
Mr HAMMOUCHE Kamal	Professeur à l'UMMTO	Examinateur
Mme AMIROU Zahia	Maître de Conférences B à l'UMMTO	Examinateur

Remerciements

Remerciements

En préambule à ce mémoire en vue de l’obtention du diplôme de Magister en électronique, effectué au sein du laboratoire LAMPA (Laboratoire d’Analyse et de Modélisation des Phénomènes Aléatoires), je souhaiterai adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu’à la réussite de cette formidable année universitaire.

La première personne que je tiens à remercier est mon encadrant Monsieur HADDAB Salah, pour l’orientation, la confiance et la patience qui ont constitué un apport considérable sans lequel ce travail n’aurait pas pu être mené au bon port. Qu’il trouve dans ce travail un hommage vivant à sa haute personnalité

Mes remerciements s’adressent également au Professeur LAGHROUCHE Mourad pour avoir contribué à ce travail depuis la proposition du thème.

Que le Professeur AMEUR Soltane, qui nous fait l’honneur de présider ce jury, ainsi que les membres de ce jury, le Professeur HAMMOUCHE Kamal et Mme AMIROU Zahia, Maitre de conférences B, veuillent bien accepter mes sincères remerciements.

J'exprime ma gratitude à tous les membres du laboratoire LAMPA qui m’ont toujours soutenu et encouragé pour surmonter toutes les difficultés rencontrées durant ce mémoire.

Je n'oublie pas mes parents pour leur contribution, leur soutien et leur patience.

Enfin, j'adresse mes plus sincères remerciements à tous mes proches et amis, qui m'ont toujours soutenu et encouragé.

Table des matières

Sommaire

Introduction générale

Introduction générale

La télémédecine est une remarquable application des nouvelles technologies de l'information, avec pour but d'améliorer l'accessibilité aux soins de santé en faisant voyager les données plutôt que les patients et l'expertise au lieu des experts, et ce par l'intermédiaire de transferts de données (imagerie médicale, enseignement à distance, données sur des patients) ou par l'action directe du praticien sur le malade.

Ce nouveau mode d'exercice de la médecine, pouvant s'appliquer à chacune des spécialités, met en rapport entre eux, par la voie des nouvelles technologies :

- Soit le patient et un ou plusieurs professionnels de santé ;
- Soit plusieurs professionnels de santé entre eux.

De nos jours, la télémédecine connait un essor sans précédent surtout avec l'évolution du téléphone portable, un composant qui a complètement révolutionné la médecine à distance.

A l'origine, le téléphone portable est censé permettre à son utilisateur de téléphoner relativement n'importe où. Mais depuis quelques années, le marché des téléphones polyvalents – ou Smartphones – a gagné du terrain et s'impose dans nos poches ou sacs à main. Sous Android, iOS ou BlackBerry ils font désormais presque oublier la fonction principale qui leur est destinée : téléphoner.

L'arrivée sur le marché de cette nouvelle catégorie de terminaux puissants et financièrement abordables est le principal vecteur de plusieurs innovations dans divers domaines et notamment dans le domaine médical.

Grace à son succès exponentiel, le Smartphone est vite devenu l'un des maillons indispensables dans une chaine de télémédecine. Cet outil miniature, doté de nombreux capteurs, pouvant se connecter presque à tous les réseaux (internet, GSM, 3G…), est vu désormais comme étant l'outil idéal pour différents types d'applications.

L'un des plus importants domaines préférentiels d'applications qui peut être identifié est le suivi des maladies chroniques telle que la maladie cardiaque, l'Alzheimer, le diabète, etc.

Parmi les maladies chroniques notre choix s'est porté sur la maladie du diabète, qui touche une tranche très importante de la population mondiale et qui est qualifiée par certains de « la maladie du XXI siècle ».

L'ambition de notre projet est d'exploiter l'intelligence de notre téléphone afin de suivre le patient diabétique.

Ainsi, dans notre projet, on se propose de concevoir et de réaliser un système ambulatoire, à base de Smartphone, pour l'acquisition, le traitement et la transmission de signaux biomédicaux, qu'on a appliqué au diabète pour un suivi et une surveillance locale et à distance du patient diabétique.

L'idée est d'exploiter les mesures de glycémie interstitielle, très proches des mesures capillaires, prélevées par un capteur inséré en sous-cutané, et de les acheminer, par réseau sans fils Bluetooth, vers le Smartphone. Ce dernier s'occupera du traitement, de l'enregistrement, de l'affichage numérique et graphique ainsi que de la lecture vocale des mesures qui lui parviennent.

Le patient sera donc tout le temps au courant des fluctuations de sa glycémie et pourra ainsi prendre les décisions adéquates au moment opportun. Ces décisions lui éviteront de plonger dans un état critique tel que l'hyperglycémie ou l'hypoglycémie qui pourraient mettre en péril sa vie et même, dans certains cas, celle des autres (ex : une hypoglycémie pendant que le patient conduit sur une autoroute sera de lourdes conséquences).

Notre application sera même en mesure d'alerter le médecin et les proches du patient, si jamais il arrive à franchir les limites de sa normo-glycémie fixées par son médecin et ce par l'émission automatique d'un SMS contenant l'alarme et les coordonnées GPS du patient afin que les télé-surveillants parviennent à connaitre l'état et la position du patient, ce qui permettra d'accélérer énormément une éventuelle intervention.

Notre application ne se contentera pas seulement de surveiller le patient. En effet, grâce à la capacité de stockage importante dont disposent les Smartphones, nous pourrons l'utiliser comme Holter glycémique grâce à l'affichage graphique de toutes les mesure prises sur une longue période qui peut aller jusqu'à 7 jours.

Comme nous pouvons le constater, le sujet abordé dans cette thèse s'inscrit à la croisée des domaines de la télémédecine, des technologies mobiles et de la diabétologie afin d'améliorer la prise en charge des patients diabétiques.

Pour ce faire nous avons subdivisé notre travail en quatre principaux chapitres :

Le premier chapitre sera consacré aux généralités relatives à la télémédecine, au diabète et à la mesure de glycémie avec deux approches différentes.

Le deuxième chapitre portera sur l'étude du Smartphone et en particulier sur la plateforme Android qui représente le cerveau de notre solution.

Dans le troisième chapitre nous étudierons les concepts de base du réseau sans fils Bluetooth ainsi que le système de géolocalisation GPS.

Notre mémoire s'achèvera par un chapitre qui sera consacré à la description et à la mise en application de la solution de télésurveillance.

Chapitre I : Généralités sur la télémédecine et le diabète

1. Introduction

Notre travail étant axé sur l'exploitation de la télémédecine pour le suivi du diabète, nous allons, dans ce qui suit, donner un aperçu sur la télémédecine avec ses différents actes puis nous définirons le diabète et ses différents types. Nous terminerons par la mesure de la glycémie selon deux approches différentes : L'approche médicale pour voir son intérêt dans une surveillance à court terme du diabète et l'approche technologique pour voir les différentes méthodes utilisées pour effectuer la mesure.

2. Aperçu sur la télémédecine

2.1 Définition de La télémédecine

La télémédecine a été définie par l'OMS en 1997 comme « la partie de la médecine qui utilise la transmission par télécommunications d'informations médicales (images, comptes-rendus, enregistrements, etc.), en vue d'obtenir, à distance, un diagnostic, un avis spécialisé, une surveillance continue d'un malade, une décision thérapeutique » (1) . Elle permet, entre autres, d'établir un diagnostic, d'assurer, pour un patient à risque, un suivi à visée préventive ou un suivi post-thérapeutique, de requérir un avis spécialisé, de préparer une décision thérapeutique, de prescrire des produits, de prescrire ou de réaliser des prestations ou des actes, ou d'effectuer une surveillance de l'état des patients.

La télémédecine a déjà une longue histoire puisque, autrefois, des villages africains utilisaient des signaux de fumée pour avertir de rester loin du village ou sévissait une grave maladie (2).

2.2 Les actes de télémédecine

Devant la multitude d'applications que peut trouver la télémédecine dans notre quotidien, l'élaboration d'une liste exhaustive s'avère de plus en plus malaisée. Il est tout de même utile de mentionner les principales qui couvrent le champ de ce vaste domaine de télémédecine à savoir :

- La téléconsultation ;
- La télé expertise ;
- La télésurveillance ;
- La téléassistance.

2.2.1 La téléconsultation

La téléconsultation est l'un des principaux actes faisant partie de la télémédecine. Elle permet, à un patient, d'obtenir un diagnostic médical à distance.

Figure 1: La téléconsultation.

Le patient peut également être assisté d'un professionnel de santé (3).

2.2.2 La télé-expertise

La télé expertise a été limitée souvent dans sa définition aux échanges entre spécialistes pour obtenir un deuxième avis. Cette définition peut être élargie à tout acte diagnostic ou thérapeutique qui se réalise en dehors de la présence du patient (en temps différé). L'acte médical de télé-expertise se décrit comme un échange entre deux ou plusieurs médecins, qui arrêtent ensemble un diagnostic à base des données cliniques, radiologiques ou biologiques qui figurent dans le dossier médical d'un patient (4).

2.2.3 La télésurveillance

La télésurveillance est un acte médical qui découle de la transmission et de l'interprétation, par un médecin, d'un indicateur clinique, radiologique ou biologique, recueilli par le patient lui-même ou par un professionnel de santé (5).

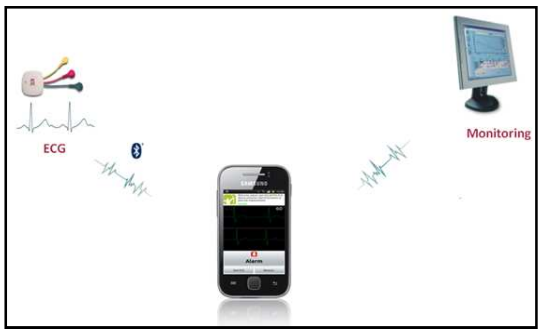


Figure 2 : La télésurveillance cardiaque.

L'interprétation peut, dans certains cas, conduire à la décision d'une intervention auprès du patient.

2.2.4 La téléassistance

La télé assistance peut être un acte médical lorsqu'un médecin assiste, à distance, un autre médecin en train de réaliser un acte médical ou chirurgical. Le médecin peut également assister un autre professionnel de santé qui réalise un acte de soins ou d'imagerie, voire, dans le cadre de l'urgence, assister, à distance, un secouriste ou toute personne portant assistance à une personne en danger, en attendant l'arrivée d'un médecin.

Nous délimitons ainsi le champ de la télémédecine à ces 4 actes. Les autres appellations sont incluses dans ces actes.

2.3 Intérêt de la télémédecine

Comme on peut le constater, la télémédecine n'est pas une nouvelle discipline médicale, mais un nouveau mode d'exercice de la médecine, pouvant s'appliquer à chacune des spécialités, dont l'objectif est, lors d'une prise en charge médicale, de minimiser les problèmes de distance entre différents intervenants en faisant voyager les données médicales plutôt que les patients. Pour faire un résumé des avantages de la télémédecine, nous pouvons citer ce qui suit :

- Développer les soins à domicile ;
- Limiter les déplacements aux personnes âgés ou handicapés ;
- Faciliter l'accès aux soins dans les zones d'accès difficile ;
- Raccourcir les délais d'attente ;
- Apporter un soutien psychologique aux malades, de sorte à ce qu'ils ne se sentent plus seuls face à leur maladie ;
- Faciliter la concertation entre médecins (6).

Pour les spécialités qui doivent se réjouir des bonds en avant réalisés dans cet extraordinaire monde de télémédecine, on pourrait penser aux différentes maladies chroniques qui touchent davantage de personnes, dans un monde qu'on qualifie de vieillissant.

Dans ce projet, et comme on l'a déjà souligné dans l'introduction générale, nous allons nous intéresser à la maladie du diabète, qui, rappelons-le, est l'une des maladies chroniques

les plus répandue dans le monde d'aujourd'hui ; et le patient atteint de cette maladie nécessite vraiment qu'on soit, présent, à son chevet ; une présence au sens large que nous permettent les différents systèmes innovants des télécommunications (la présence physique n'est plus indispensable).

Pour cela, nous avons envisagé de concevoir un système de télésurveillance du patient diabétique, que l'on décrira en détail tout au long de ce mémoire ; un système qui permettra le suivi et la surveillance en continu et en temps réel, surtout que les outils nécessaires à la mise en œuvre de tels systèmes deviennent de plus en plus disponibles : des moyens de télécommunications de plus en plus puissants et sophistiqués (les Smartphones par exemple) et des capteurs de glycémie de différentes technologies et aisément implantables. Mais, avant d'aborder l'aspect technique, un aperçu sur tout ce qui entoure la maladie s'impose.

3. Le diabète

Le diabète est une maladie chronique jusque-ici incurable, causée par une carence ou un défaut d'utilisation de l'insuline, entraînant un excès de sucre dans le sang. Produite par le pancréas, plus précisément par les cellules Beta des îlots Langerhans, l'insuline est une hormone qui permet au glucose (sucre), contenu dans les aliments, d'être utilisé par les cellules du corps humain. Les cellules disposent de toute cette énergie dont elles ont besoin pour fonctionner (7).

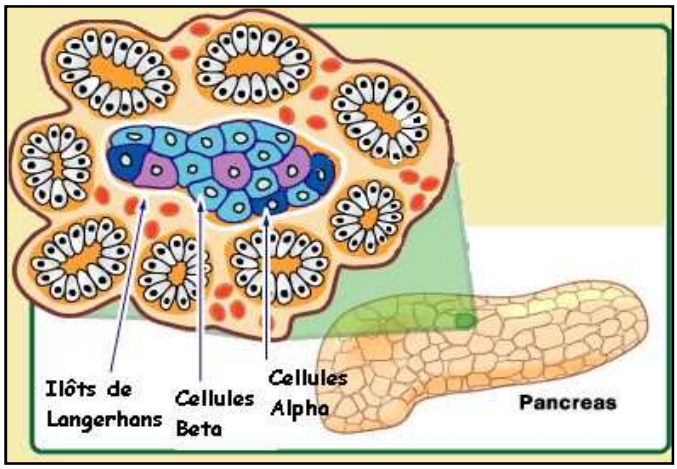


Figure 3 : Le pancréas.

Si l'insuline est insuffisante ou si elle ne remplit pas son rôle adéquatement, comme c'est le cas dans le diabète, le glucose ne peut pas servir de carburant aux cellules. Il s'accumule alors dans le sang et est ensuite déversé dans l'urine. À la longue, l'hyperglycémie provoquée par la présence excessive de glucose dans le sang entraîne certaines complications, notamment au niveau des yeux, des reins, des nerfs, du cœur et des vaisseaux sanguins.

À ce jour, la cause réelle du diabète demeure inconnue. Nous savons toutefois, que certains facteurs peuvent influencer l'apparition du diabète : hérédité, obésité, grossesse, certains virus ou certains médicaments.

3.1 Les symptômes

Les symptômes du diabète ne se présentent pas tous de la même manière ni avec la même intensité. Les principaux symptômes sont :

- Fatigue, somnolence ;
- Augmentation du volume des urines ;
- Soif intense et faim exagérée ;
- Amaigrissement ;
- Vision embrouillée ;
- Cicatrisation lente ;
- Infection des organes génitaux ;
- Picotements aux doigts ou aux pieds ;
- Changement de caractère.

3.2 Les types de diabète

Il existe deux principaux types de diabète : le type 1 et le type 2. Parfois, le diabète se développe aussi pendant la grossesse (8).

3.2.1 Le diabète de type 1

C'est une maladie auto-immune. Le pancréas ne sécrète que très peu, voire pas du tout d'insuline. La maladie se déclare souvent au cours de l'enfance ou de l'adolescence. Cependant, les causes d'apparition de la maladie ne sont toujours pas claires. Certains pensent que le diabète de type 1 est une maladie génétique qui a pour conséquence l'attaque de

certaines cellules du pancréas. D'autres pensent qu'un virus peut être à l'origine de la maladie et inciterait le système immunitaire à attaquer le pancréas. Comme les cellules du pancréas qui sécrètent normalement l'insuline sont détruites, une personne atteinte de diabète de type 1 devra recevoir de l'insuline toute sa vie, grâce à des injections par stylo ou une pompe à insuline (9).

3.2.2 Le diabète de type 2

Le diabète de type 2 se manifeste beaucoup plus tard dans la vie, généralement après l'âge de 40 ans. La très grande majorité des personnes atteintes de diabète présentent ce type de diabète (90 % des cas). Depuis quelques années, on remarque que ce diabète apparaît plus tôt (10).

Une prédisposition génétique, l'obésité et le manque d'activité physique contribuent à l'apparition d'un diabète de type 2. Cependant, certaines études tendent à démontrer qu'une alimentation en trop grande quantité et riche en gras pourrait aussi être un facteur de risque. En apportant des corrections importantes à nos habitudes de vie, il est possible de retarder l'apparition de la maladie et d'en diminuer l'impact.

3.2.3 Le diabète de grossesse (gestationnel)

Il s'agit d'une maladie que les femmes peuvent contracter aux premiers stades de la grossesse. Contrairement aux diabètes de type 1 et type 2, le diabète de grossesse disparaît après la naissance du bébé (11).

3.3 Les complications du diabète

Les complications du diabète sont nombreuses et peuvent être sévères : infarctus, troubles de la vision, cécité, accident vasculaire, amputations, maladies rénales...

Ces complications aggravent le diabète et diminuent l'espérance de vie des personnes atteintes de cette maladie. La majorité des complications, liées au diabète, peuvent être évitées, diminuées ou retardées si le diabète est dépisté et traité précocement et correctement.

Qu'il s'agisse du type 1, du type 2 ou du diabète de grossesse, une consultation chez le médecin s'impose. Généralement, la simple mesure de la glycémie à jeun, par prise de sang, suffit pour dépister un diabète.

4 Mesure de la glycémie

4.1 Aspect médical

La glycémie est la concentration plasmatique du glucose. Au cours de la journée, sa valeur varie en fonction des apports et des besoins énergétiques de l'individu. La glycémie est ajustée par l'action d'hormones sécrétées par des cellules du pancréas. Ce système de régulation permet de maintenir le même taux alors même que les cellules des organes ont des besoins différents en fonction de leur activité.

Pour les périodes de mesure de la glycémie, nous avons :

4.1.1 Glycémie à jeun

Pour la glycémie à jeun, la prise de sang a lieu sans apport calorique pendant huit heures au moins.

4.1.2 Glycémie postprandiale

Pour ce qui est de la glycémie, dite postprandiale, la mesure a lieu une heure et demie ou deux heures après le début du repas.

4.1.3 Analyse de la glycémie

La mesure de la glycémie permet de savoir s'il y a une bonne régulation du taux de sucre dans le sang. Cet examen est prescrit lorsque l'on soupçonne une hyperglycémie, qui est symptomatique, entre autre, du diabète. Mais il est aussi prescrit pour détecter une hypoglycémie, c'est-à-dire un taux de sucre insuffisant dans le sang. La mesure de la glycémie est un examen fréquemment prescrit au cours de la grossesse pour détecter si la patiente ne souffrirait pas d'un diabète gestationnel.

4.1.4 Valeurs normales de la glycémie

Le tableau suivant donne les valeurs normales de la glycémie (12) :

Valeurs normales ou cibles chez les personnes non diabétiques	
A jeun	Entre 70 et 99 mg/dL
Apres les repas	Entre 70 et 140 mg/dL
Valeurs normales ou cibles chez les personnes diabétiques	
A jeun	Entre 70 et 130 mg/dL
1 à 2 heures après le début du repas	Au dessous de 180 mg/dl

Tableau 1: Les valeurs normales de la glycémie.

A noter qu'l faut compter (+0,10 g/L) par décennie après 50 ans.

4.2 Aspect technologique

4.2.1 Les méthodes classiques invasives : glycémie capillaire

Le glucomètre est un petit appareil portable qui permet de déterminer la glycémie d'un patient à partir de l'analyse d'une gouttelette de son sang. Le lecteur de glycémie permet, notamment aux patients diabétiques de type 1 (anciennement appelés diabétiques insulinodépendants), de contrôler plusieurs fois par jour leur maladie et d'adapter en conséquence les doses d'insuline qu'ils doivent s'injecter.

Figure 4: La glycémie capillaire.

Le prélèvement de sang s'effectue à l'aide d'un stylo auto-piqueur qui donne l'impulsion nécessaire à une fine lancette (fine aiguille à usage unique), introduite au préalable dans le stylo. Le prélèvement s'effectue, en général, au bout d'un doigt, sur une face latérale de la dernière phalange pour éviter un gène ultérieur à la préhension.

L'analyse de l'échantillon de sang prélevé peut être réalisée par deux types de lecteurs de glycémie.

4.2.1.1 Les lecteurs de glycémie à bandelettes colorimétriques

Le sang déposé sur une bandelette déclenche une réaction d'oxydation du glucose sanguin par la glucose-oxydase, contenue dans la partie réactive de la bandelette. Les électrons libérés par l'oxydation du glucose, sous l'effet de l'enzyme, agissent sur un composé colorimétrique, chose qui entraîne un changement de couleur de cette bandelette proportionnel à la valeur de la glycémie. Une fois la bandelette introduite dans le lecteur, cette couleur est interprétée et l'appareil affiche ainsi le taux de glucose du patient. Ce procédé de mesure est relativement lent en raison des étapes successives du processus d'analyse (13).

4.2.1.2 Les lecteurs de glycémie à électrodes

Le sang est déposé sur une électrode qui est introduite dans le lecteur et qui déclenche une réaction électrochimique (génération de micro-courants interprétés par le lecteur). L'électrode permet une mesure plus rapide et plus précise, et nécessite moins de sang que la bandelette colorimétrique (14).

4.2.2 Quelques méthodes en cours de développement

4.2.2.1 Les méthodes non-invasives

Pendant des décennies, des chercheurs, des scientifiques et des personnes atteintes de diabète ont cherché un moyen non-invasif et, permettant une mesure permanente de la glycémie, réalisant ainsi le rêve d'une « mesure du taux de glucose dans le sang sans le…sang! ». En ce qui suit, un survol de quelques une de ces méthodes.

4.2.2.1.1 Capteur utilisant iontophorèse inverse

L'iontophorèse (ou ionophorèse) est un examen médical utilisant un faible courant électrique afin de faire passer certaines molécules, à visée thérapeutique, à travers la peau, ou afin d'en extraire certaines dans un but diagnostique, on parle alors d'iontophorèse inverse chose qu'on retrouve dans la montre-bracelet lecteur de glycémie Glucowatch.

Figure 5: La GlucoWatch.

Les molécules de glucose, du liquide interstitiel sous-cutané, atteignent le détecteur par iontophorèse (un très faible courant électrique provoque un mouvement des ions Na qui entraînent les molécules de glucose vers la cathode), où s'effectue une oxydation. Le signal électrique produit par cette réaction est proportionnel à la concentration du glucose (15).

4.2.2.1.2 Capteur optique

Ce genre de mesure est basé sur les propriétés de la molécule de glucose à absorber et à interagir avec une partie de la lumière au niveau cutané, dégageant une énergie thermique (des radiations électromagnétiques) qui peut être mesurée et convertie en concentration de glucose. La figure suivante montre un exemple de capteur s'appuyant sur ce principe :

Figure 6: Le capteur C8 de MediSensors.

La technique consiste à envoyer une source lumineuse, monochromatique, à travers la peau et à détecter la lumière émise. Les couleurs produites par la diffusion Raman sont très spécifiques, en fonction de la structure chimique des molécules contenues dans l'échantillon. Les différentes formes et tailles, ainsi que les différents atomes et types de liaison chimique des molécules produiront un spectre Raman spécifique, une empreinte Raman, qui peut être utilisée pour lire et mesurer le glucose de manière non-invasive (16).

4.2.2.2 Les méthodes peu-invasives

4.2.2.2.1 Capteur utilisant la fluorescence

La fluorescence est la propriété de certains corps d'émettre des radiations lumineuses sous l'effet d'autres radiations.

Pour l'exemple, on peut citer le capteur EyeSense, utilisant la fluorescence pour mesurer la concentration du glucose dans l'humeur aqueuse de l'œil, grâce à un petit implant dans

chaque œil au niveau du tissu sous-conjonctif, mais qui doit être posé et calibré initialement par un ophtalmologiste (17).

4.2.2.2.2 Le capteur interstitiel

La figure suivante montre le schéma de principe d'un capteur de glycémie interstitiel :

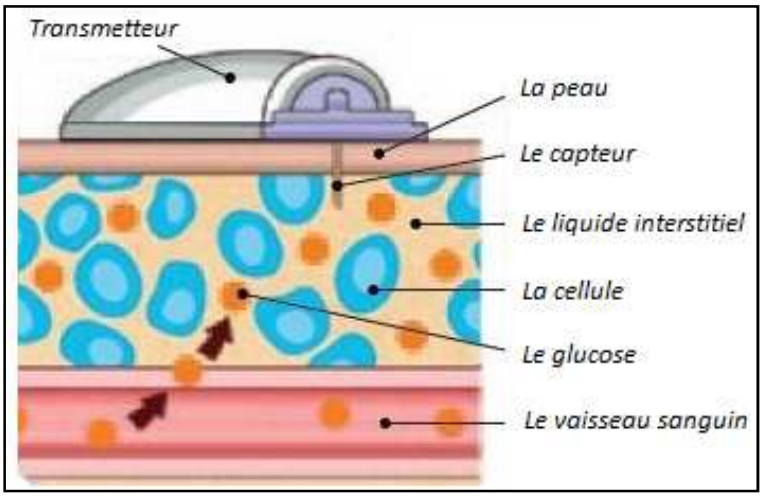


Figure 7: La glycémie interstitielle.

Ce capteur permet de mesurer la glycémie en continu et en temps réel. Le capteur est une aiguille, implantée dans le tissu sous-cutané grâce à un applicateur amovible, sur laquelle est fixée de la glucose-oxydase. La lecture du glucose en continu est, en fait, le taux de glucose contenu dans le liquide qui se trouve dans les cellules (liquide interstitiel). Lorsque vous mangez, les glucides sont transformés en glucose (sucre). Ce glucose passe de notre appareil digestif au sang. Ensuite, le sang l'achemine dans tout le corps. Enfin le glucose passe du sang au liquide interstitiel pour aller nourrir les cellules et les muscles. C'est donc le taux de sucre contenu dans le liquide interstitiel que mesure ce genre de capteurs.

La mesure du courant électrique, qui est proportionnel à la glycémie, généré suite à l'oxydation du glucose nous donne la glycémie interstitielle.

Il l'est à signaler, qu'il existe une légère différence entre le taux de glucose mesuré dans le liquide interstitiel et le taux de glucose mesuré dans le sang, par conséquent les valeurs obtenues doivent être confirmées par une glycémie capillaire avant toute décision thérapeutique.

Ce qu'on vient de voir n'est qu'un survol de quelques une des technologies utilisées dans ce monde de la mesure de glycémie, qui reste très captivant, passionnant et surtout prometteur. Toutefois, malgré que la liste ne soit pas exhaustive, les méthodes qu'on vient de

voir ouvrent des perspectives vers un domaine de recherche en pleine expansion qui est le domaine du pancréas artificiel.

5. Le pancréas artificiel : un rêve devenu presque réalité

Ce système repose sur la mesure en continu de la concentration en glucose couplée à l'administration continue, mais variable, d'insuline en fonction des besoins. Les études dans ce domaine concernent donc essentiellement les patients diabétiques de type 1, puisque ces derniers ne sécrètent plus d'insuline, ce qui représente un modèle plus pur.

Figure 8: Exemple de pancréas artificiel.

Sur le plan technique, ce système repose sur :

- La mesure continue du glucose sous-cutané ;
- L'utilisation automatique d'un algorithme ;
- L'adaptation de la perfusion continue d'insuline à l'aide de pompe.
- Ce pancréas artificiel devrait permettre également d'avertir le patient à l'aide d'alarmes (18).

Cette avancée prometteuse doit nous réjouir mais, comme tout progrès, il convient de rester prudent et critique car elle fait toujours partie du domaine de la recherche. En effet, il nous semble important de préciser qu'à l'heure actuelle, cette technique n'est pas disponible de manière courante pour les patients et est actuellement toujours en cours d'évaluation.

5. Conclusion

Dans cette première partie, nous avons essayé, dans un premier temps, de faire un petit tour d'horizon de l'une des plus remarquables applications des nouvelles technologies, en l'occurrence la télémédecine, qui n'arrive toujours pas à être déployée tel qu'il se doit. Puis nous avons abordé la maladie du diabète pour laquelle sera appliquée notre solution de télésurveillance en exploitant les Smartphones. Nous avons également définit certaines méthodes de mesure de la glycémie.

Chapitre II : Développements d’applications sous Android

1. Introduction

Le Smartphone est un compagnon de tous les instants et un véritable ordinateur de poche pour gérer le quotidien, s'informer, se divertir. Il comporte aussi un ensemble de capteurs embarqués ou connectés. Le Smartphone a donc une place à part dans la galaxie des appareils. Pourtant, les utilisateurs savent très peu de choses sur ce qui se passe à l'intérieur de ces boîtes noires. Ainsi, dans ce chapitre nous allons essayer de voir les concepts de base des Smartphone en nous intéressant plus précisément à la plateforme de Google Inc. Android.

2. 1 Présentation du Smartphone

«Smartphone » est un terme d'origine anglo-saxonne, signifiant littéralement "téléphone intelligent". Il désigne un téléphone portable, multifonctions à mi-chemin entre le téléphone portable et l'assistant personnel. Selon le principe d'un ordinateur, il peut exécuter divers applications grâce à un système d'exploitation spécialement conçu pour mobiles, et donc, en particulier, fournir des fonctionnalités en plus de celles des téléphones mobiles classiques comme : l'agenda, la télévision, le calendrier, la navigation sur le Web, la consultation et l'envoi de courrier électronique, la géolocalisation, le dictaphone/magnétophone, la calculatrice, la boussole, l'accéléromètre, le gyroscope, la messagerie vocale , la cartographie numérique, etc.

Il est possible de personnaliser son Smartphone en y installant des applications additionnelles telle que des jeux ou des utilitaires via un magasin d'applications en ligne différent pour chaque système d'exploitation comme Google Play sur Android ou, encore, l'App Store sur iOS.

2.2 Les principaux systèmes d'exploitations pour Smartphone

La figure suivante donne la part du marché des principaux systèmes d'exploitation mobiles selon une étude de Gartner Inc. (19) :

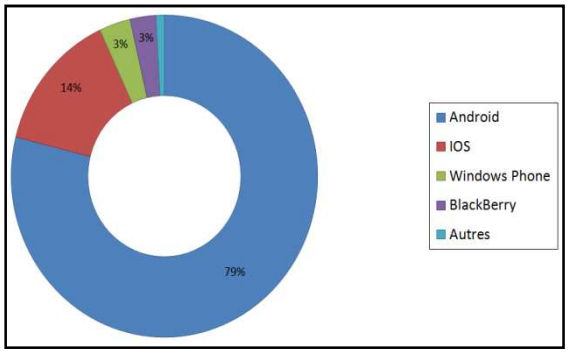


Figure 9: Le marché mondial d'OS mobiles (2eme trimestre 2013).

Pour le marché des Smartphones deux géants, Android de *Google Inc* et iOS d'Apple dominent toujours le marché et occupent respectivement la première et deuxième place du podium. A ces deux, viennent s'ajouter d'autres systèmes d'exploitation qui partagent la part restante du marché. Parmi ces systèmes, on retrouve Windows phone de Microsoft, BlackBerry de RIM (Research in Motion), Symbian de Nokia, Bada de Samsung et Java ME de SUN Microsystems (rachetée plus tard par Oracle Corporation).

3. Etude de la plateforme Android

Android est un système d'exploitation pour Smartphones, tablettes tactiles, PDA et terminaux mobiles. C'est un système open source, utilisant le noyau Linux, conçu par Android, une startup rachetée par Google, et annoncé officiellement le 5 novembre 2007. D'autres types d'appareils possédant ce système d'exploitation existent, par exemple des téléviseurs, des montres, des autoradios et même des voitures (20).

Android est gratuit, pour les constructeurs d'appareils souhaitant l'utiliser, et partiellement open-source. Concrètement, ce système d'exploitation s'appuie sur un noyau Linux optimisé pour un usage mobile, et sur une machine virtuelle Java assez fortement modifiée, nommée Dalvik JVM. Celle-ci n'exécute pas les (.class) habituels, mais des fichiers portant l'extension (.dex), compilés différemment et optimisés par le SDK Android. Le développement se fait donc en Java, mais sur cette JVM spécifique à Android. Il n'est, du fait, pas possible d'utiliser n'importe quelle librairie Java dans une application Android.

Il est possible d'utiliser divers IDE Java pour développer et compiler une application Android, mais Google fournit un plugin Eclipse, dont il serait dommage de se passer.

3.1 Les différentes versions

Dans l'ensemble, les différentes versions d'Android ont toutes des noms de desserts (en anglais), qui sont sculptés et affichés devant le siège social de Google (Mountain View) et ça, depuis la sortie de la version 1.5 et suivent une logique alphabétique (de A vers Z).

Figure 10: Les différentes versions de l'OS Android.

Au départ, il existait deux variantes de la plateforme Android. Une de ces variantes est dédiée aux petits écrans, principalement les téléphones mobiles (toutes les versions en dessous de 3.0), et l'autre variante est dédiée pour les tablettes : Honeycomb ou Android 3.0. En octobre 2011, la fusion de ces deux variantes, pour avoir une plateforme plus versatile et uniforme, a donné naissance à la version Android 4, connue sous le nom de "Ice Cream Sandwich". C'est la première version qui combine "Gingerbread" et "Honeycomb" pour une plateforme, à la fois, pour les tablettes et les téléphones (21).

3.2 Architecture de la plateforme Android

L'architecture de la plateforme Android se décline, selon une démarche bottom up, en quatre principaux niveaux que sont :

- le noyau linux
- les librairies et l'environnement d'exécution,
- le module de développement d'applications
- les différentes applications (20).

Figure 11: L'architecture de la plateforme Android.

3.2.1 Premier niveau : noyau Linux

Android s'appuie sur un noyau (Kernel en anglais) Linux 2.6. Pour être précis, le noyau est l'élément du système d'exploitation qui permet de faire le pont entre la partie matérielle et la partie logicielle. D'ailleurs, si vous regardez attentivement le schéma, vous remarquerez que cette couche est la seule qui gère le matériel. La version du noyau utilisée avec Android est une version conçue spécialement pour l'environnement mobile, avec une gestion avancée de la batterie et une gestion particulière de la mémoire.

3.2.2 Deuxième niveau : librairies et environnement d'exécution

3.2.2.1 Les librairies

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

Malgré que le développement d'application se fasse en langage JAVA, Android comprend très bien le C et le C++.

3.2.2.2 L'environnement d'exécution

Le runtime Android est basé sur le concept de machine virtuelle, utilisée en Java. Étant donné les limitations des dispositifs (peu de mémoire et la vitesse du processeur), il n'a pas été possible d'utiliser une machine virtuelle Java standard. Google a pris la décision de créer une nouvelle machine virtuelle Dalvik, afin de mieux répondre à ces limitations. Dans cette partie, qui est l'environnement d'exécution, on retrouve essentiellement :

3.2.2.2.1 Core Libraries

Les Core libraries fournissent le langage Java, disponible pour les applications. Le langage Java fournit avec Android reprend en grande partie l'API JSE 1.5. Il y a des choses qui ont été mises de coté, car cela n'avait pas de sens pour Android (comme les imprimantes, swing, etc.) et d'autres APIs spécifiques requises pour Android ont été rajoutées.

3.2.2.2.2 Dalvik

Les applications Java, développées pour Android, doivent être compilées au format Dalvik exécutable (.dex) avec l'outil (dx). Cet outil compile les (.java) en (.class) et ensuite il convertit ces (.class) en (.dex).

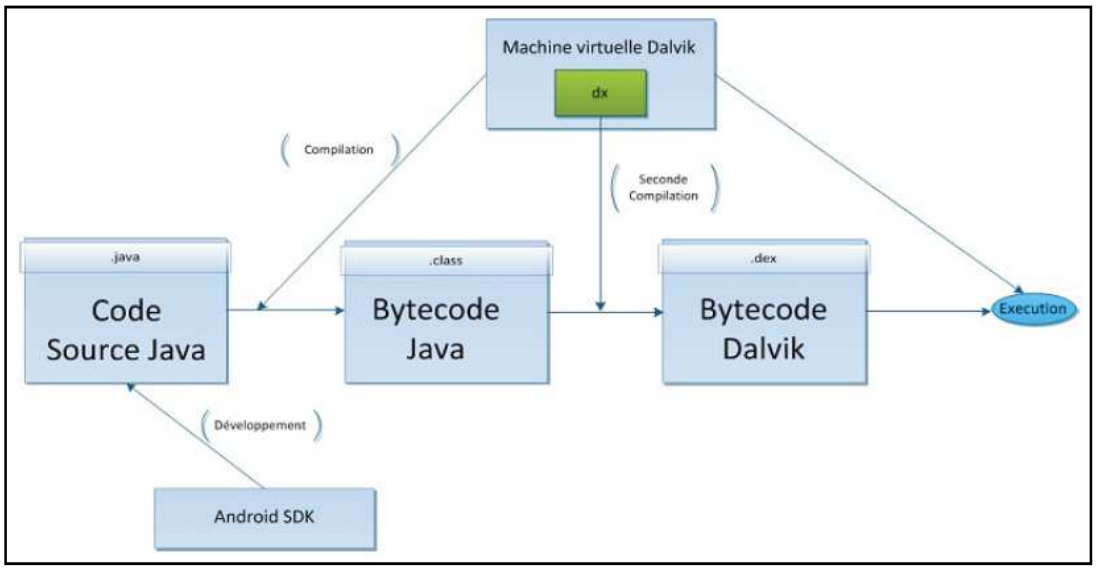


Figure 12: La machine virtuelle Dalvik.

3.2.3 Troisième niveau : module de développement d'applications

Le Framework est situé au dessus de l'Android Runtime et des librairies. Il fournit des API permettant aux développeurs de créer des applications riches. A ce niveau on distingue deux types de service :

3.2.3.1 Core Plateform Services

Android introduit la notion de services. Un service est une application, qui n'a aucune interaction avec l'utilisateur et qui tourne en arrière plan. Les services cœurs de la plateforme (Core Plateform Services) fournissent des services essentiels au fonctionnement de la plateforme :

- **Activity Manager** : gère le cycle de vie des applications et maintient une pile de navigation permettant d'aller d'une application à une autre ;
- **Package Manager** : utilisé, par l'Activity Manager, pour charger les informations provenant des fichiers (.apk) (android package file) ;
- **Window Manager** : il gère les fenêtres des applications (quelle fenêtre doit être affichée devant une autre à l'écran) ;
- **Resource Manager** : gère tous ce qui n'est pas du code, toutes les ressources (images, fichier audio, etc.) ;
- **Content Provider** : gère le partage de données entre applications.
- **View System** : fournit tous les composants graphiques : listes, grilles, boutons, etc.

3.2.3.2 Hardware Services

Les services matériels (Hardware Services) fournissent un accès vers les API matérielles :

- **Telephony Service** : permet d'accéder aux interfaces téléphoniques (GSM, 3G, etc.) ;
- **Location Service** : permet d'accéder au GPS ;
- **Bluetooth Service** : permet d'accéder à l'interface Bluetooth ;
- **WiFi Service** : permet d'accéder à l'interface Wifi ;
- **USB Service** : permet d'accéder aux interfaces USB ;
- **Sensor Service** : permet d'accéder aux capteurs (capteur de luminosité, de proximité, d'accélération, etc.).

3.2.4 Quatrième niveau : applications

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

4. Développement sous Android

4.1 L'environnement de développement sous Android

Afin de développer des applications sous Android, un ensemble d'outils est nécessaire. Vue que les procédures d'installation de ces outils sont assez longues et fastidieuses, alors les décrire, pas à pas, risquerai de prendre énormément de place et ainsi beaucoup de pages. Alors, on se contentera juste d'évoquer les outils et leur intérêt.

4.1.1 Le JDK (Java Developpment Kit)

Les applications développées pour Android étant essentiellement écrites en langage java ; un langage de programmation orienté objet qui a la particularité d'être très portable. Cela signifie qu'un programme Java, fonctionnant sur Windows (par exemple), pourra facilement tourner sur Mac ou GNU/Linux.

Cette petite prouesse vient du fait que Java s'appuie sur une machine virtuelle pour s'exécuter (appelée la JVM). Pour avoir une JVM sur votre ordinateur, il vous faut télécharger le JRE. Ce dernier contient, en plus de la JVM, des bibliothèques Java standards.

La JVM ne lit pas directement le code Java. Elle lit un code compilé (le bytecode). Pour passer du code Java, que le développeur écrit, au code compilé, lu par la JVM, des outils spéciaux sont nécessaires. Ces outils sont inclus dans le JDK. De plus, le JDK contient le JRE (et donc la machine virtuelle), ce qui est bien pratique

Pour résumer, on dira que:

- Pour un simple utilisateur de Java : il doit avoir le JRE ;
- Pour un développeur : il aura besoin des outils du JDK (22).

4.1.2 Le SDK *(*Software Development Kit*)* Android

Un SDK, c'est-à-dire un kit de développement logiciel, est un ensemble d'outils que met à disposition un éditeur afin de permettre de développer des applications pour un environnement précis. Le SDK Android permet, donc, de développer des applications pour Android et uniquement pour Android.

Au premier lancement du SDK, un écran semblable à la figure suivante s'affichera :

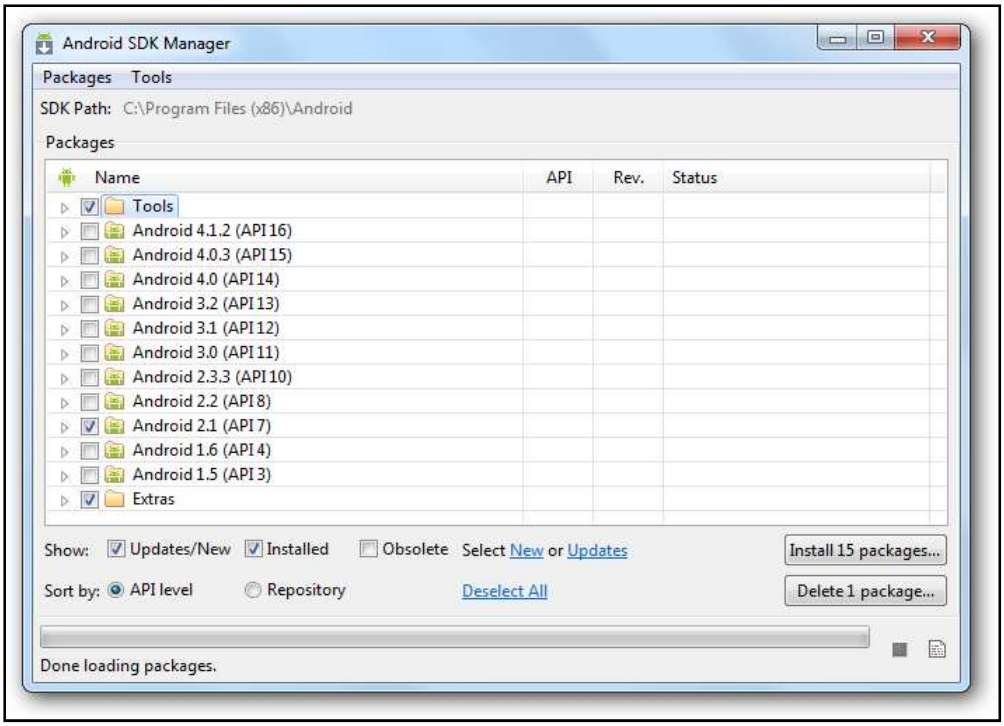


Figure 14: Le SDK Android.

En regardant bien le nom des paquets, vous remarquerez qu'ils suivent tous un même motif. Il est écrit à chaque fois : Android [un nombre] (API [un autre nombre]). La présence de ces nombres s'explique par le fait qu'il existe plusieurs versions de la plateforme en circulation. Le premier nombre correspond à la version d'Android et le second à la version de l'API Android associée. Quand on développe une application, il faut prendre en compte ces numéros, puisqu'une application développée pour une version précise d'Android ne fonctionnera pas pour les versions antérieures.

4.1.3 L'IDE Eclipse

Eclipse est un environnement de développement intégré. C'est un logiciel qui permet d'écrire un programme beaucoup plus facilement qu'avec le simple Bloc-notes. Outre la coloration du code, il permet d'apporter des outils très pratiques pour compiler vos programmes, les déboguer, etc. Il peut être utilisé pour programmer avec n'importe quel type de langage, mais nous l'utiliserons pour faire du Java.

De plus, Eclipse est conçu pour pouvoir être complété avec des plugins (extension). Ainsi, il existe un plugin pour développer des applications Android que nous verrons dans la partie suivante.

4.1.4 Le plugin ADT pour Eclipse

Google fournit un plugin pour Eclipse, nommé ADT (Android Development Tools). La fonction principale de ce plugin est de créer un pont entre Eclipse et le SDK Android.

4.1.5 L'émulateur de téléphone : Android Virtual Device

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

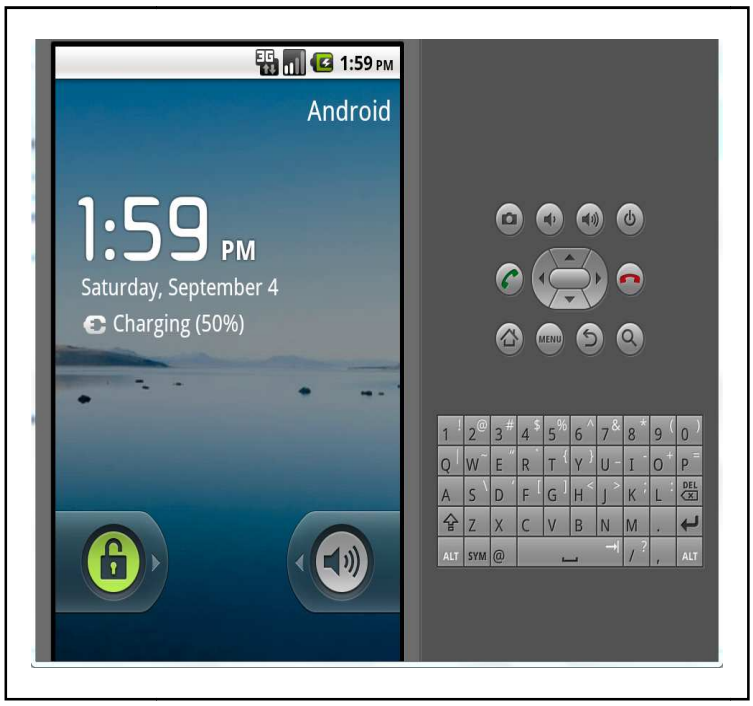


Figure 15: L'émulateur.

4.2 Structure d'un projet

Le système de construction d'un programme Android est organisé sous la forme d'une arborescence de répertoires spécifiques à un projet, exactement comme n'importe quel projet Java. Les détails, cependant, sont spécifiques à Android. Voici un rapide tour d'horizon de la structure d'un projet :

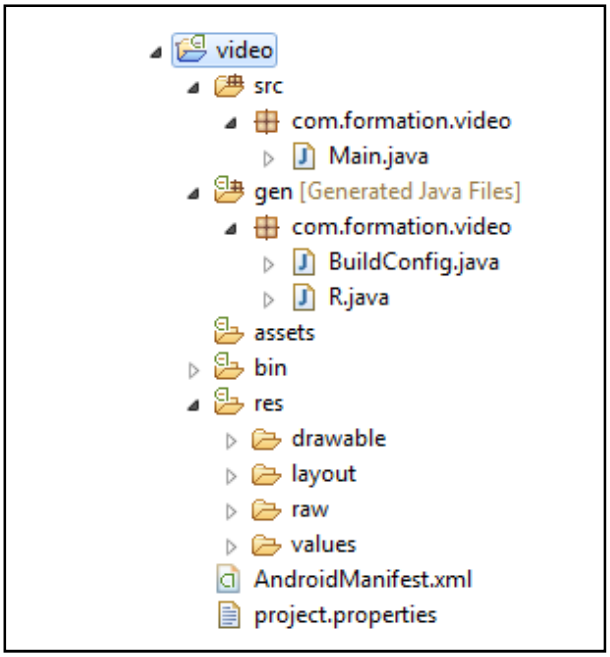


Figure 16: L'arborescence d'un projet.

Tout comme bon nombre de technologies actuelles, les sources d'une application Android possèdent une structure bien définie qui doit être respectée. Ces arborescences permettent non seulement de rendre les projets plus lisibles et organisés, mais aussi de simplifier le développement. Lors de la création d'un nouveau projet, voici l'arborescence qui est automatiquement générée :

- src : répertoire contenant l'ensemble des sources du projet. Il contient les classes, de type activité, qui gèrent entre autre le cycle de vie, mais aussi les classes permettant de piloter les différentes fonctions de l'application ;
- gen : répertoire contenant l'ensemble des fichiers générés par le plugin correspondant à l'environnement de développement. Aucune modification ne doit être faite dans ces fichiers ;
- androidManifest.xml : fichier XML décrivant l'application et ses composants, tels que les activités, les services, etc. Lors de la création d'une activité, une erreur courante pour un premier projet Android est d'oublier de la déclarer dans le fichier Manifest. C'est une étape indispensable pour le fonctionnement de l'application. Le Manifest est, en quelque sorte, la carte d'identité de l'application, et permet d'autoriser l'exécution des activités et autres actions de l'application.
- res : répertoire contenant toutes les ressources telles que les images, les vues de l'interface graphique, etc., nécessaires à l'application. Ce répertoire est structuré par défaut de la manière suivante :

- res/drawable : contient les ressources de type image ;
- res/layout : contient les descriptions des interfaces graphiques au format XML (les vues) ;
- res/xml : contient les fichiers XML supplémentaires (non présents par défaut) ;
- res/menu : contient la description des menus, composants très courants d'une vue ;
- res/values : contient diverses ressources, telles que les textes, qui sont empaquetées sans aucun traitement.

Au moment de la compilation du projet, l'application finale est générée au format APK, dans le répertoire bin de l'arborescence. C'est ce fichier qu'il faut ensuite déployer sur les équipements, afin de pouvoir faire tourner l'application.

4.3 Composantes d'une application Android

Une application Android se compose de plusieurs éléments. Dans ce qui suit, nous allons essayer de découvrir les plus importants :

4.3.1 Les Activités

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

Plusieurs choix se proposent pour mettre en place l'interface visuelle :

- Utiliser un fichier XML pour décrire l'interface ;
- Créer les éléments de l'interface à l'intérieur du code java.

4.3.2 Les Services

Un Service est, en fait, un programme tournant en tâche de fond et n'ayant pas d'interface graphique. L'exemple commun illustrant cette notion, est celui du lecteur mp3. Un lecteur mp3 ne nécessite pas, pour la plupart du temps, d'interface graphique et doit tourner en tâche de fond, laissant la possibilité aux autres applications de s'exécuter librement. Un service peut être lancé à différents moments :

- Au démarrage du téléphone ;
- Au moment d'un événement (arrivée d'un appel, SMS, mail, etc.) ;
- Lancement de l'application ;
- Action particulière dans application.

4.3.3 Les BroadcastReceivers

Un BroadcastReceiver, comme son nom l'indique, permet d'écouter ce qui se passe sur le système ou sur votre application et de déclencher une action que vous aurez prédéfinie. C'est souvent par ce mécanisme que les services sont lancés.

4.3.4 Les Content providers

Les ContentProvider sont, comme l'exprime leur nom, des gestionnaires de données. Ils permettent de partager l'information entre applications. Vous pouvez accéder :

- Aux contacts stockés dans le téléphone ;
- A l'agenda ;
- Aux photos ;
- Ainsi que d'autres données depuis votre application grâce aux content providers.

4.3.5 Les Intents

Les Intents sont des objets permettant de faire passer des messages contenant de l'information entre composants principaux. La notion d'Intent peut être vue comme une demande de démarrage d'un autre composant, d'une action à effectuer.

4.4 Cycle de vie d'une application Android

Le diagramme d'état suivant présente les principaux états du cycle de vie d'une activité Android, il est suivi d'une description des principales méthodes événementielles du cycle de vie d'une activité.

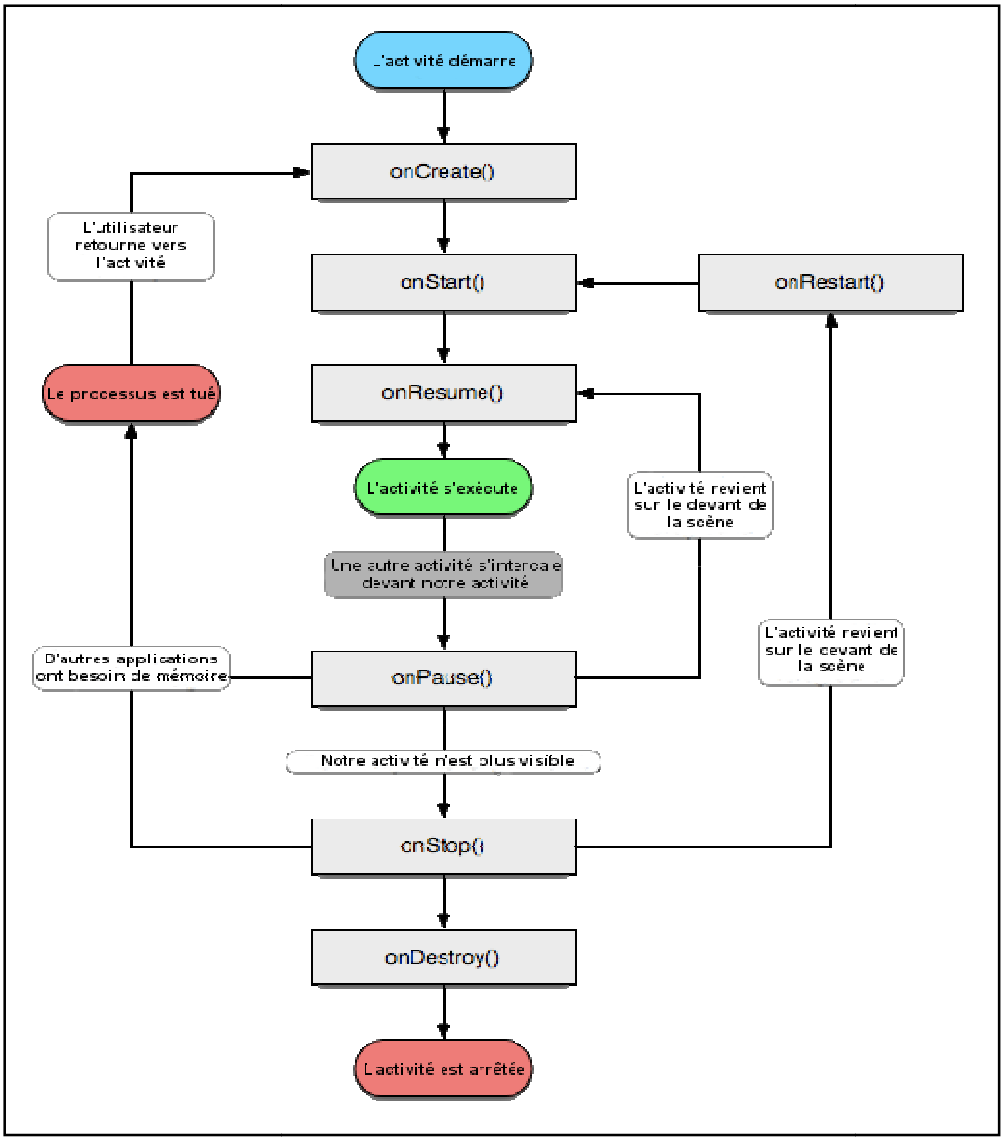


Figure 17: Le cycle de vie d'une Activité.

La méthode onCreate() est appelée à la création de votre activité. Elle sert à initialiser l'activité ainsi que toutes les données nécessaires à cette dernière. Quand la méthode OnCreate() est appelée, on lui passe un Bundle en argument. Ce Bundle contient l'état de sauvegarde enregistré lors de la dernière exécution de l'activité.

La méthode onStart() signifie le début d'exécution de l'activité (début du passage au premier plan). Si l'activité ne peut pas aller en avant plan, pour une quelconque raison, elle sera transférée à OnStop().

La méthode onResume() est appelée lorsque l'activité commencera à interagir avec l'utilisateur juste après avoir été dans un état de pause.

La méthode onPause() est appelée au passage d'une autre activité en premier plan. L'intérêt d'un tel appel est de sauvegarder l'état de l'activité et les différents traitements effectués par l'utilisateur. A ce stade, votre activité n'a plus accès à l'écran, vous devez arrêter de faire toute action en rapport avec l'interaction utilisateur (désabonner les listeners).

La méthode onStop() est appelée quand l'activité n'est plus visible quelque soit la raison. Dans cette méthode vous devez arrêter tous les traitements et services exécutés par votre application.

La méthode onDestroy est appelée quand votre application est totalement fermée (Processus terminé). Les données non sauvegardées sont perdues (20).

5. Exploitation des fonctionnalités d'Android : les capteurs

La majorité des appareils modernes sont bien plus que de simples outils pour communiquer ou naviguer sur internet. Ils possèdent des capacités sensorielles, matérialisées par leurs capteurs. Ces derniers peuvent être divisés en trois principales catégories :

5.1 Les capteurs de mouvements

En mesurant les forces d'accélération et de rotation sur les trois axes, ces capteurs sont capables de déterminer dans quelle direction se dirige l'appareil. On y trouve l'accéléromètre, les capteurs de gravité, les gyroscopes et les capteurs de vecteurs de rotation.

5.2 Les capteurs environnementaux

Ce sont trois capteurs (baromètre, photomètre et thermomètre) qui mesurent la température ambiante, la pression atmosphérique, l'illumination et l'humidité.

5.3 Les capteurs de position

Evidemment, ils déterminent la position de l'appareil. On trouve ainsi les capteurs d'orientation et le magnétomètre.

D'un point de vue technique, on trouve deux types de capteurs. Certains sont des composants matériels, (c.-à-d. composant physique présent sur le terminal), qui fournissent des données en prenant des mesures. Les autres capteurs sont uniquement présents d'une manière logicielle. Ils se basent sur des données fournies par des capteurs physiques pour donner des données nouvelles (20).

La figure suivante donne le repère associé aux capteurs à trois dimensions :

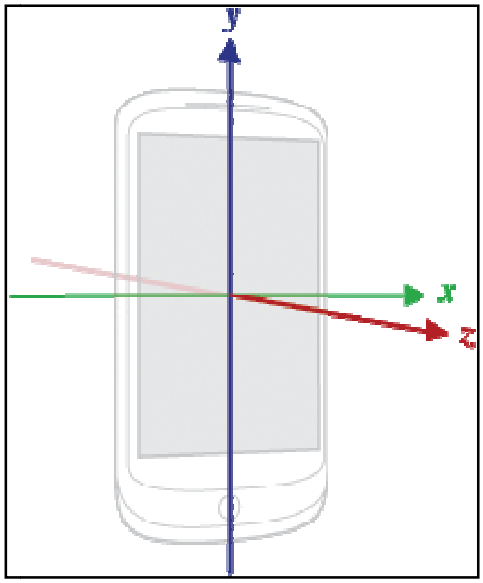


Figure 18: Le repère associé aux capteurs.

Le tableau suivant recense la majorité de capteurs embarqués par un Smartphone :

Capteur	**Type**	**Description**	**Utilisation typique**
Accéléromètre	Matériel	Mesure la force d'accélération appliquée au terminal sur les trois axes (x, y et z).	Détecter les mouvements.
Gyroscope	Matériel	Mesure le taux de rotation sur chacun des trois axes en radian par seconde (rad/s).	Détecter l'orientation de l'appareil.
Photomètre	Matériel	Mesure le niveau de lumière ambiante en lux (lx).	Détecter la luminosité pour adapter celle de l'écran de l'appareil.
Magnétomètre	Matériel	Mesure le champ géomagnétique sur les trois axes en micro tesla (µT)	Créer un compas.
Orientation	Logiciel	Mesure le degré de rotation que l'appareil effectue sur les trois axes.	Déterminer la position de l'appareil.
Baromètre	Matériel	Mesure la pression ambiante en hectopascal (hPa) ou millibar (mbar).	Surveiller les changements de pression de l'air ambiant.
Capteur de proximité	Matériel	Mesure la proximité d'un objet en centimètres (cm).	Détecter si l'utilisateur porte le téléphone à son oreille pendant un appel.
Thermomètre	Matériel	Mesure la température de l'appareil en degrés Celsius (°C).	Surveiller la température.

Tableau 2 : Les capteurs embarqués.

6. Conclusion

Dans ce deuxième chapitre nous avons essayé de lever le voile sur cette nouvelle race de terminaux en nous basant sur la plateforme Android. L'approche faite durant cette étude était beaucoup plus logicielle. Dans ce qui suit, nous allons essayer de comprendre les concepts fondamentaux de deux réseaux largement utilisés dans les applications pour Smartphone en l'occurrence : Le Bluetooth et le GPS.

Chapitre III : Le réseau Bluetooth et la géolocalisation par GPS

1. Introduction

Dans ce troisième chapitre, et après avoir vu l'essentiel sur tout ce qui entoure le développement d'applications pour la plateforme Android, nous allons nous intéresser au réseau sans fils Bluetooth et à la géolocalisation par le système GPS en tachant d'insister sur l'essentiel.

2. Le réseau sans fils Bluetooth

Bluetooth est une technologie de réseau personnel sans fil (noté WPAN pour Wireless Personal Area Network), mise au point par le géant suédois Ericsson en 1994. En 1998, un groupe d'intérêt baptisé Bluetooth SIG (Bluetooth Special Interest Group) a été fondé par Ericsson, IBM, Intel, Toshiba et Nokia. Aujourd'hui, plus de 2500 entreprises ont rejoint le groupe.

Le but de cette technologie est de permettre la communication à courte distance, entre plusieurs appareils et sans le moindre câble, en utilisant les ondes radio comme support de transmission (bande de fréquence située autour des 2,45 GHz), apportant ainsi l'une des meilleures solutions à l'informatique mobile qui permet aux utilisateurs de se déplacer tout en restant connectés au réseau.

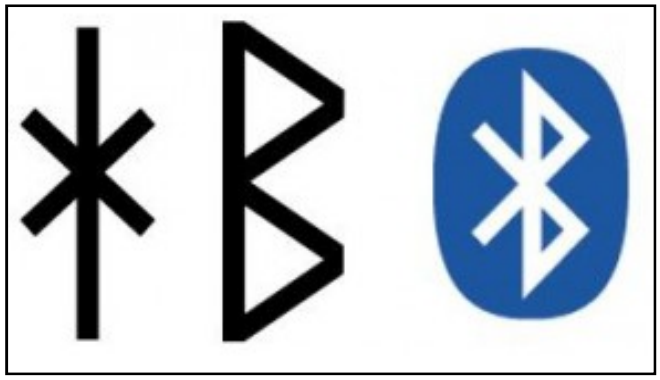


Figure 19: Le logo de Bluetooth.

L'origine de l'appellation Bluetooth fait référence à un roi danois Harald «Dent bleue» (dû à son goût immodéré pour les mûres), qui aurait unifié les différents royaumes nordiques à la fin du Moyen Âge, de même que Bluetooth SIG s'est créé autour d'intérêts communs (23). Le logo Bluetooth est constitué des initiales d'Harald en alphabet runique.

2.1 L'origine de la technologie Bluetooth

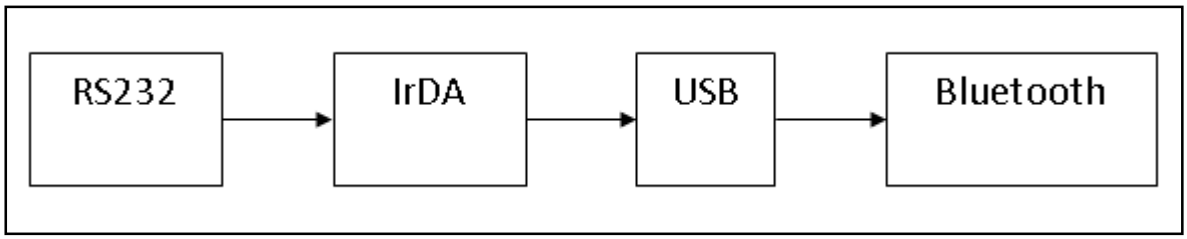


Figure 20: L'origine de la technologie Bluetooth.

Pour remonter aux origines de la technologie Bluetooth, il nous faut faire un lointain retour en arrière, aux années 60 : le port série. Le port série a été inventé afin de relier des périphériques (clavier, terminaux, matériels de mesure) à des ordinateurs : c'est le standard RS232.

Son évolution sans fil est l'IrDA, un protocole qui utilise les ondes lumineuses infrarouges pour la transmission de données. Cependant, tout comme le protocole RS232, il est possible de relier un seul périphérique à la fois. C'est pourquoi a été conçu l'USB (Universal Serial Bus) qui permet de relier plusieurs périphériques en série. Une évolution sans fil de ce protocole a naturellement vu le jour : Bluetooth. Cette technologie est donc une évolution lointaine sans fil du RS232 (23).

2.2 La topologie du réseau

Bluetooth est un réseau de type «ad-hoc», ce qui signifie qu'il ne dispose pas de station de base tel que le réseau cellulaire GSM :

- Ce réseau est auto-configurable: deux machines mobiles, se retrouvant dans le même secteur, peuvent se reconnaître puis échanger des données ;
- Chaque périphérique peut échanger des informations avec n'importe quelle autre périphérique;
- Les périphériques peuvent échanger des données uniquement lorsqu'ils sont à portée de réception l'un par rapport à l'autre (24).

2.2.1 Le Piconet

Un Piconet est un réseau qui se crée de manière instantanée et automatique quand plusieurs périphériques Bluetooth sont dans un même rayon (10 m environ).

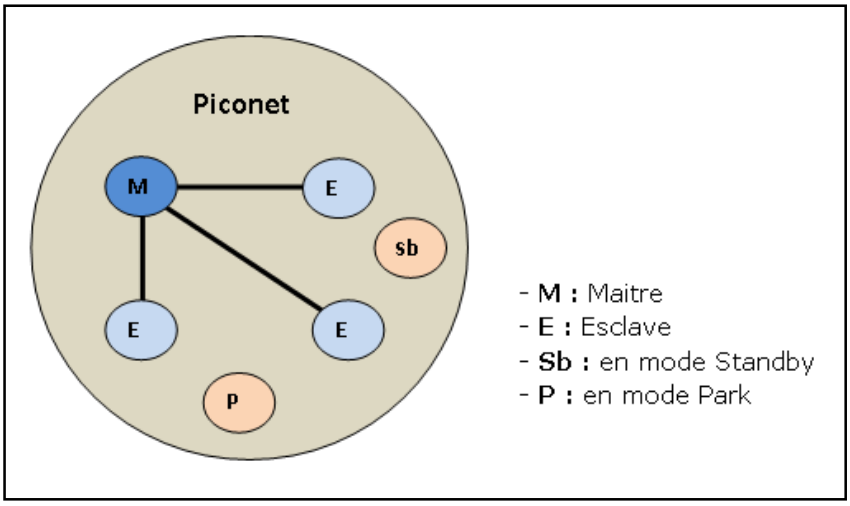


Figure 22: Le Piconet.

L'appareil qui initie l'échange joue le rôle de maître, tandis que le ou les autres sont dits esclaves.

Ce réseau suit une topologie en étoile : 1 maître et plusieurs esclaves. Un périphérique maître peut administrer jusqu'à 7 esclaves actifs ou 255 esclaves en mode parked (=inactif) et c'est lui qui détermine la fréquence de saut en fréquence pour tout le Piconet. La communication est directe entre le maître et un esclave. Tous les esclaves du Piconet sont synchronisés sur l'horloge du maître et ne peuvent, en aucun cas, communiquer entre eux.

2.2.2 Les Scatternets

Les Scatternets sont, en fait, des interconnexions de Piconets. Ces interconnexions sont possibles car les périphériques esclaves peuvent avoir plusieurs maîtres. Les différents Piconets peuvent donc être reliés entre eux.

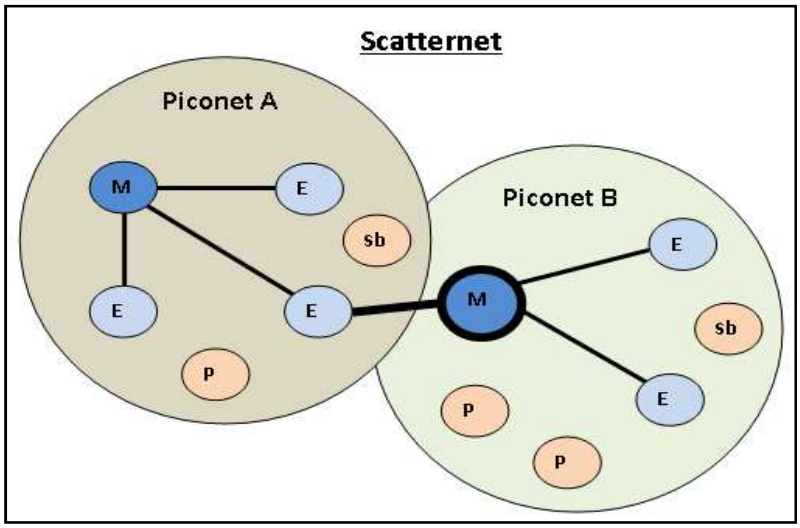


Figure 23: Le Scatternet.

2.3 La portée d'une liaison Bluetooth

Il existe 3 classes de modules radio Bluetooth sur le marché ayant des puissances différentes et donc des portées différentes :

Classe	Puissance	Portée
1	100 mW (20 dBm)	100 mètres
2	2,5 mW (4 dBm)	15 à 20 mètres
3	1 mW (0 dBm)	10 mètres

Tableau 3: La portée des liaisons Bluetooth.

2.4 La protection contre les brouillages

A cause des perturbations, il a fallu protéger la transmission radio contre les brouillages par une technique d'étalement de spectre, qui consiste à utiliser une bande de fréquence beaucoup plus large que celle qui est nécessaire :

- par sauts en fréquence ;
- par code binaire: avant de moduler la porteuse, on mélange le signal binaire à une séquence numérique pseudo aléatoire de débit nettement plus élevé (Wifi, CDMA, UMTS) (25).

2.4.1 La technique de sauts en fréquence

L'étalement de spectre par saut en fréquence a originalement été conçu dans un but militaire afin d'empêcher l'écoute des transmissions radio. En effet, une station ne connaissant pas la combinaison de fréquences à utiliser, ne pouvait pas écouter la communication car il lui était impossible de localiser la fréquence sur laquelle le signal était émis puis de chercher la nouvelle fréquence (25).

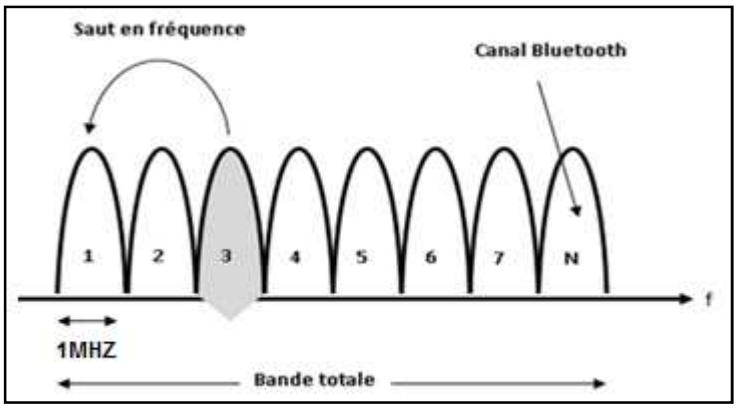


Figure 24: Le saut de fréquence dans le réseau Bluetooth.

Chaque transmission Bluetooth utilise les 79 canaux et occupe la bande dans sa totalité, soit une largeur d'environ (80 MHz) pour un débit maximal de 1 Mbits/s. L'avantage est une relative insensibilité à la présence de signaux de brouillages au prix d'un encombrement spectral supérieur.

Un cas pratique, qui permet de voir l'intérêt de la technique de saut en fréquence dans le standard Bluetooth, est lorsqu'on dispose de deux Piconets dans un même espace.

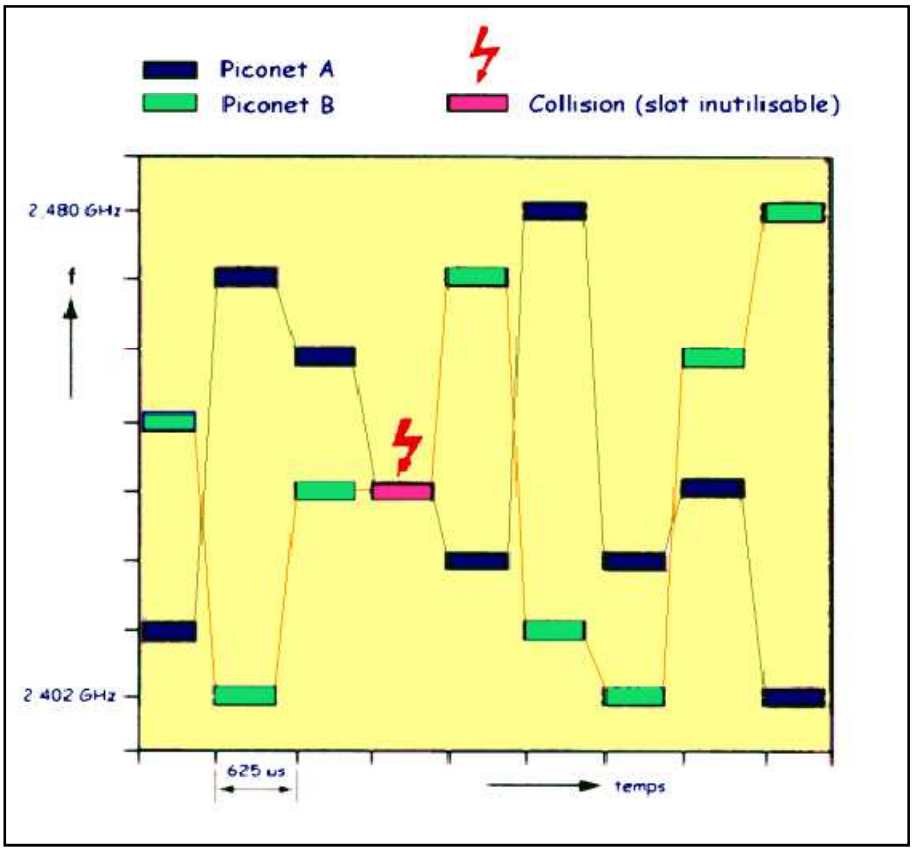


Figure 25: L'intérêt des sauts de fréquences.

L'information est d'abord transmise sur une fréquence pendant une durée de 625 µs (un time-slot), puis l'émetteur passe sur la fréquence suivante. Les sauts en fréquence (1600 sauts par seconde) sont déterminés par calcul à partir de l'adresse du maître et de l'horloge. Ils sont donc aussi connus par le récepteur qui change de fréquence de manière synchrone avec l'émetteur pour récupérer le signal transmis.

Chaque réseau Piconet utilise une succession de fréquences différentes, et le risque que 2 Piconets entrent en collision est vraiment faible. Les collisions sont, généralement, gérées par retransmission du paquet (25).

2.5 La transmission de données

Pour la transmission, les données sont regroupées en paquets et associées à des informations d'adresse et de description du paquet :

1. Code d'accès (72 bits): chaque paquet débute par un code d'accès composé du code de canal ou CAC (Chanel Access Code), propre à un Piconet, du code de composant ou DAC (Device Access Code), utilisé pour le paging, et du code de recherche ou IAC (Inquiry Access Code), si le maître recherche d'autres équipements Bluetooth du Piconet ;
2. En-tête (54 bits): ce champ contient dans l'ordre l'adresse de l'esclave (codée sur 3 bits, soit 7 au maximum) qui échange des données ; le type de paquet et des bits de contrôle (erreurs, buffer de réception) ;
3. Données binaires: la taille de cette partie est variable et peut aller jusqu'à 240 bits.

Bluetooth peut utiliser des paquets de données courts (1 time-slot, 240 bits au maximum), moyens (3 time-slot, 1480 bits max) ou longs (5 time-slot, 2745 bits max).

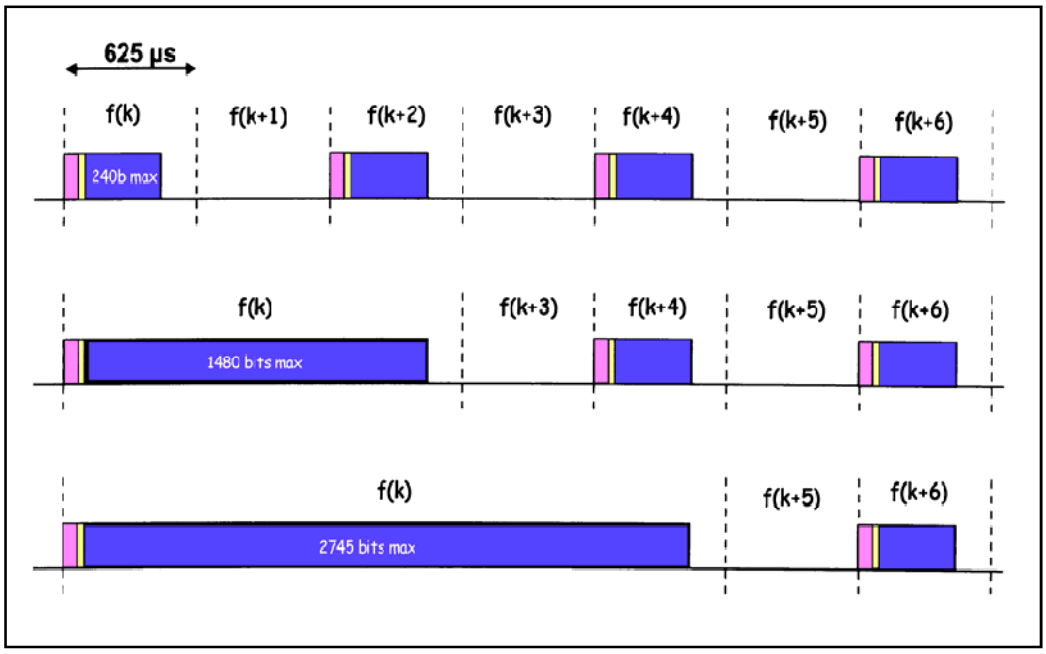


Figure 26: La transmission de données par paquets.

En fonction de la taille des paquets utilisés, le débit peut varier dans une large mesure:

- Paquet long dans un sens, court dans l'autre :

  D1=2745 bits/6x625µs = 732 Kbits/s ;

  D2= 240 bits/6x625µs = 64 Kbits/s.

- Paquet moyen dans un sens, court dans l'autre :

  D1= 1480 bits/4x625µs = 592 Kbits/s ;

  D2= 240 bits/4x625µs = 96 Kbits/s.

- Paquet long dans les deux sens :

  D1= 2745 bits/10x625µs = 439.2 Kbits/s = D2 (25).

2.6 La mise en place d'une liaison

Le schéma suivant donne un aperçu sur les différents états que peut prendre un périphérique Bluetooth :

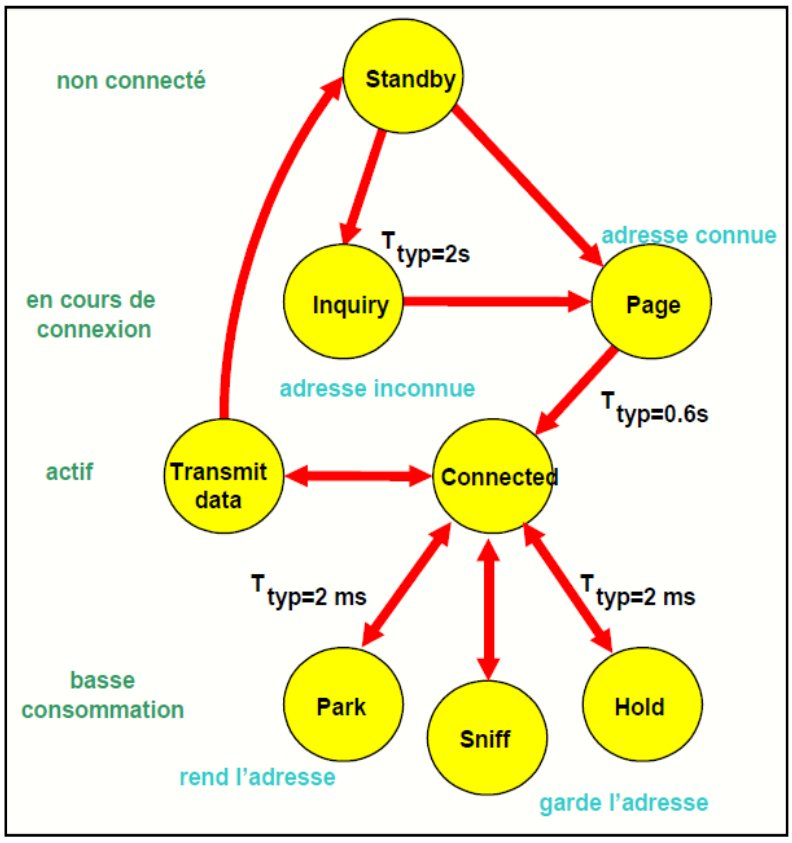


Figure 27: Les différents états du réseau Bluetooth.

Les modules Bluetooth entrent en liaison de la façon suivante:

- Ils sont, au départ, en mode d'attente et cherchent la présence de transmissions à proximité toutes les 1,28 secondes, en écoutant successivement 32 «porteuses d'éveil» parmi les 79 fréquences ;
- Le module Bluetooth qui souhaite communiquer envoie des requêtes en mode Inquiry si les adresses sont inconnues et en mode Page si elles sont connues par les porteuses d'éveil aux quelles répondent les modules situés à proximité. Il devient alors le maître du Piconet, et son adresse définit la suite des sauts en fréquence suivie par les esclaves ;
- Les autres modules (esclaves) se synchronisent sur l'horloge du maître, et répondent avec leur numéro d'identification. Le maître sait maintenant quels sont les modules présents, connaît leur adresse respectives et peut sélectionner le dispositif qui l'intéresse ;
- S'il veut échanger des données, le maître passe en mode d'appel et définit, avec le module esclave visé, le type et les caractéristiques de la liaison qu'il veut mettre en place ;
- La liaison sera établie (état : Connected) au bout de 0,6 secondes ; tout est prêt alors pour la transmission des données (Transmit Data) ;
- Après l'échange de données, le module peut retourner en mode d'attente ou adopter l'un des trois états à faible consommation:
    1. Le mode de maintien (Hold), au cours duquel l'appareil reste actif dans le Piconet. Lorsque le temporisateur interne de l'esclave vient à expiration, l'esclave s'annonce brièvement au maître avant de recommencer le compte à rebours ;
    2. Le mode renifleur (Sniff), l'esclave est programmé pour se mettre périodiquement à l'écoule du Piconet afin de déterminer si ce dernier désire lui envoyer des données ;
    3. Le dernier mode à faible consommation fait appel au parcage des esclaves (Park) : l'esclave se retire du Piconet, rend son adresse MAC (Media Access Control, de 0 à 7) et se resynchronise périodiquement sur le maître (25).

3. Définition de la géolocalisation

La géolocalisation peut être définie comme étant un procédé permettant de positionner un objet (une personne, un véhicule, etc.) sur un plan ou une carte à l'aide de ses coordonnées géographiques.

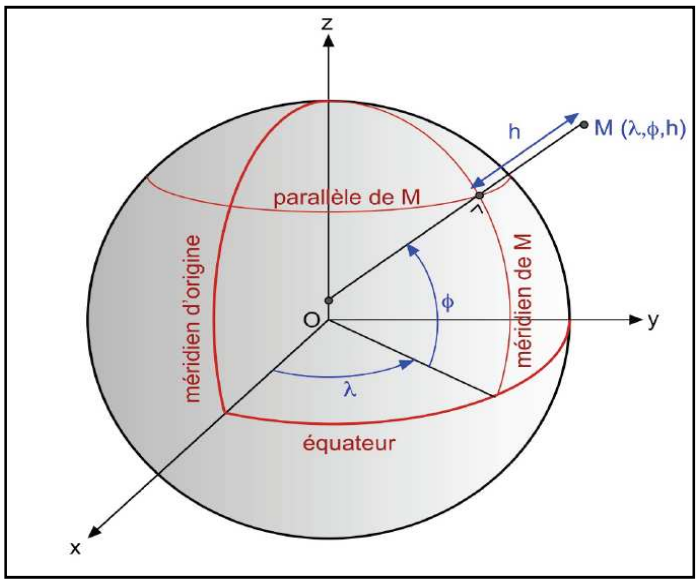


Figure 28: Les coordonnées géographiques.

Les coordonnées géographiques d'un point M de la surface de la terre sont essentiellement :

- La longitude: angle orienté entre le plan méridien origine et le plan méridien contenant le point M. Le méridien d'origine est celui de Greenwich ;
- La latitude: angle orienté entre le plan de l'équateur et la normale à l'ellipsoïde passant par le point M ;
- La hauteur : distance algébrique entre le point M et l'ellipsoïde (26).

3.1 Les techniques de géolocalisation

De nos jours, il existe plusieurs techniques de localisation. La plus connue est l'utilisation des récepteurs GPS. La précision de 10 à 20 mètres qu'offre le GPS, dans les milieux dégagés, a permis un large déploiement des services de localisation en extérieur. Cependant, l'extension de ces services en intérieur reste limitée. Afin de se libérer des limitations du GPS dans les environnements urbains denses ou intérieurs, des solutions reposant sur des capteurs ou des réseaux sans fils ont fait leur apparition.

3.1.1 La géolocalisation par satellite

Elle repose sur l'utilisation des ondes émises par une constellation de satellites (GPS, GLONASS, GALILEO), prévue à cet effet, pour le calcul de 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 (ex : 43° 5494 N - 1° 48472 E) et peut alors être représentée physiquement sur une carte.

3.1.2 La géolocalisation 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 de cette technique 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. Aujourd'hui, la méthode GSM la plus utilisée est celle du Cell ID (identification de la cellule radio). Cette méthode consiste à récupérer les identifiants des antennes GSM auxquelles le terminal est connecté.

3.1.3 La 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 (adresse MAC) qu'il détecte.

3.1.4 La géolocalisation par RFID

La technologie RFID (de l'anglais radio frequency identification) peut être utilisée pour la localisation en intérieur. 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 (radio-étiquette) 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.

3.2 Le système de géolocalisation par GPS

GPS (Global Positioning System) est le système de localisation par satellite le plus utilisé. Développé et entretenu par le département de la défense des Etats-Unis (DoD), dans le but de suivre la position de ses troupes et celle des troupes ennemies. le GPS est également utilisé pour des applications civiles : aide à la navigation, gestion du trafic, localisation

des téléphones portables dans des milieux dégagés, etc. Une autre application intéressante, mais moins connue de ce système, est la synchronisation en temps grâce aux horloges atomiques de grande précision embarquées dans les satellites.

Lorsque l'on parle du GPS, on pense aussitôt au récepteur GPS. Pourtant, il n'est qu'une partie de ce système. Le GPS s'articule autour de trois principaux segments :

- le segment spatial ;
- le segment utilisateur ;
- le segment de contrôle (27).

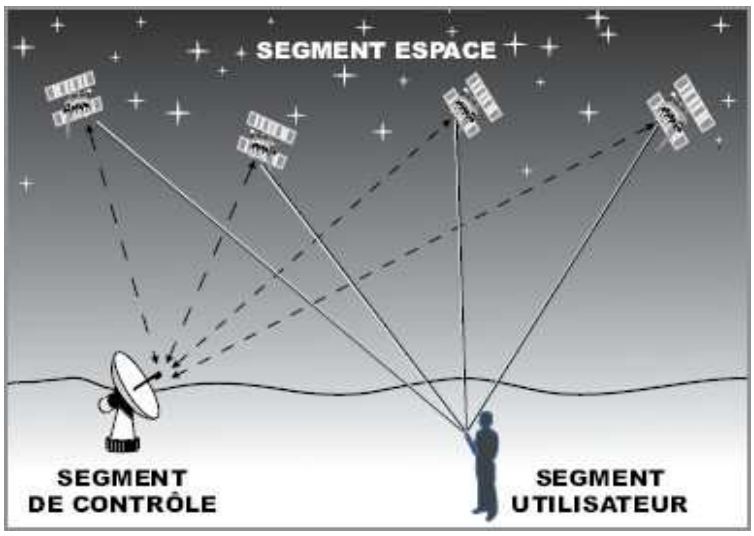


Figure 29: La constitution du système GPS.

3.2.1 Le segment spatial

Le segment spatial est constitué d'une constellation de 24 satellites NAVSTAR (Navigation Satellite Timing And Ranging) au minimum. Ces satellites sont régulièrement répartis sur six orbites circulaires, déphasées de 60°, inclinées à 55°. Ils gravitent sur une orbite polaire quasi circulaire à une altitude de 20200 km environ qu'ils bouclent en 11 heures 58 minutes. Ce segment possède des petits propulseurs pour corriger au besoin sa trajectoire (27).

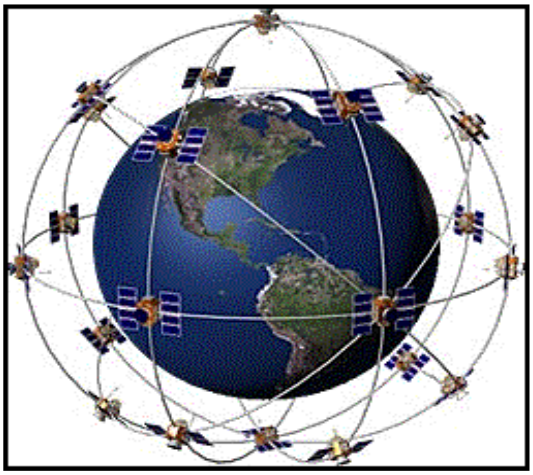


Figure 30: Le segment spatial.

Chaque satellite NAVSTAR possède plusieurs horloges atomique, chose qui lui garanti une heure précise. Ces satellites émettent en permanence sur deux fréquences L1 (1575,42 MHz) et L2 (1227,60 MHz) (28).

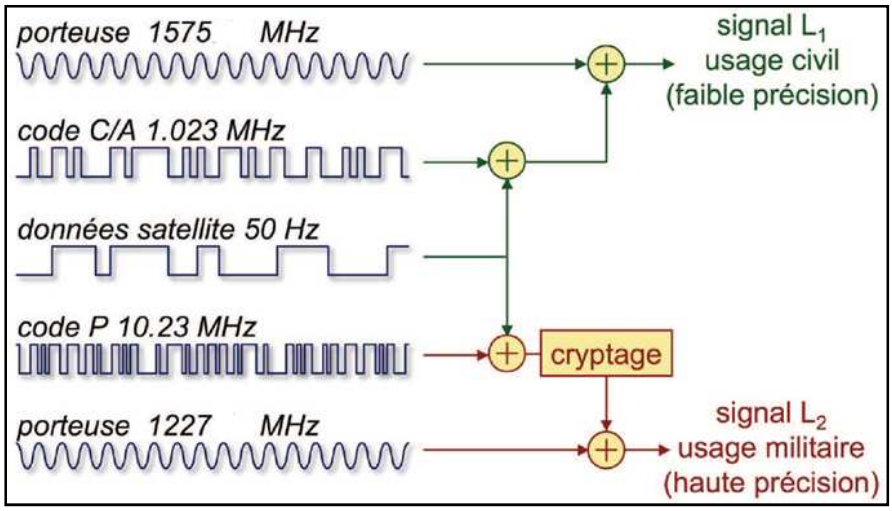


Figure 31: Les signaux émis par les satellites GPS.

Les signaux émis comportent trois types de données :

- Un almanach qui contient toutes les données sur le type de satellite, son état de fonctionnement, le calcul précis de son orbite (précision inferieure à 1m) ;
- Un code C/A (approximatif) pour un calcul approximatif du retard, destiné aux applications civiles ;
- Un code P (précis) pour un calcul plus précis du retard, destiné aux applications militaires.

3.2.2 Segment de contrôle

Le segment de contrôle est constitué de tout l'équipement terrestre qui assure le suivi et l'entretien des satellites en orbite. La station principale est située au Colorado (Falcon Air Force Base). Elle est relayée par 4 stations secondaires réparties sur des petites îles contrôlées par les USA (Hawaii, Ascension, Diego Garcia et Kwajalein). Il s'agit de s'assurer que tous les satellites fonctionnent bien, de corriger les trajectoires pour que les informations de position soient les plus correctes possibles et de synchroniser toutes les horloges des satellites, tant entre elles qu'avec l'heure de la terre (29).

Figure 32: Répartition des stations du segment de contrôle.

Si un satellite ne fonctionne pas correctement, le segment de contrôle au sol peut le déclarer «hors d'état de marche» et adopter les mesures nécessaires pour corriger le problème. Dans un tel cas, le satellite ne doit pas servir au positionnement avant d'être à nouveau déclaré fonctionnel.

3.2.3 Segment utilisateur

Constitué par l'ensemble des récepteurs susceptibles de décoder les signaux émis par la constellation de satellites et d'en déduire, par calcul, les solutions de position et de temps. Un récepteur doit être capable de sélectionner plusieurs satellites parmi ceux qui couvrent sa zone et d'acquérir les signaux GPS correspondants (29).

3.3 Principe de fonctionnement : la trilatération

La trilatération permet, grâce à trois satellites fixes qui connaissent leur position exacte, de déterminer la position d'un point précis sur Terre. Chaque satellite du réseau GPS possède une horloge atomique d'une grande précision (27).

Pour ce qui est du principe, nous pouvons le décrire comme suit : à un instant t, un satellite émet son signal vers le récepteur GPS qui le reçoit à un instant t1. Le récepteur GPS calcule ensuite, grâce à une opération simple, le temps T qu'a mis le signal pour arriver. En multipliant le temps T par la vitesse de la lumière on arrive à déterminer la distance d qui sépare le satellite et le récepteur. Donc le récepteur se trouve sur la périphérie de la sphère de rayon d et de centre le satellite en question.

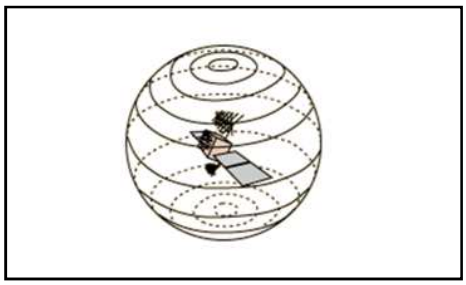


Figure 33: La position donnée par 1 satellite.

En renouvelant ce mécanisme avec un deuxième satellite, qui détermine une deuxième sphère qui coupe la 1ère en un plan sur lequel est situé le récepteur.

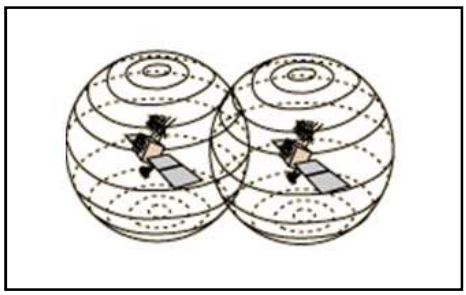


Figure 34: La position donnée par 2 satellites.

Pour terminer la trilatération, un troisième satellite est nécessaire afin de trouver sa position exacte sur ce plan. En effet lorsque la troisième sphère coupe le plan elle donne deux points.

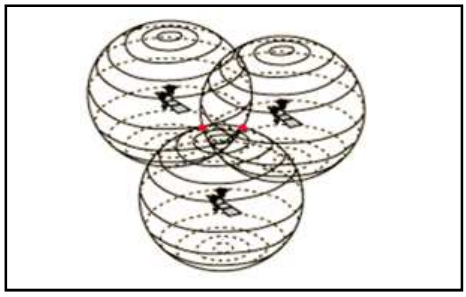


Figure 35: La position donnée par 3 satellites.

Dans le cas ou l'utilisateur se situe à la surface de la terre, seul un des 2 points est cohérent. Ainsi, on peut déduire sa position exacte en éliminant le point donnant un résultat incohérent.

Comme le récepteur n'a pas l'heure, alors l'ingéniosité des créateurs du système GPS a été d'utiliser un quatrième satellite pour synchroniser l'horloge de notre récepteur GPS. On est donc, à quatre satellites, capables de retrouver les quatre inconnues (x, y, z et t) en résolvant le système composé de quatre équations, qui ne peuvent être que les signaux émis par les quatre satellites.

Dans la pratique, le récepteur utilise entre 4 et 12 satellites pour calculer sa position. Plus le nombre de satellites est important plus la précision est meilleure.

4. Conclusion

Dans ce troisième chapitre nous avons vu l'essentiel sur deux réseaux à différentes échelles, l'un pour la transmission sans fils des valeurs de la glycémie vers le Smartphone et l'autre pour la géolocalisation du patient en cas d'urgence. Dans le prochain et dernier chapitre nous allons mettre en pratique ce qu'on vient de voir concernant ces deux réseaux.

Chapitre IV : Mise en application de la télésurveillance

1. Introduction

Cette dernière partie est consacrée à la mise en application de la solution de télésurveillance du patient diabétique. Nous donnerons au début la description fonctionnelle de la solution, puis nous passerons à la manière par laquelle nous avons procédé pour concevoir et programmer les différentes parties de notre solution qui sont, essentiellement, la réception des valeurs glycémiques par Bluetooth, la surveillance locale et les alarmes vocales, l'évolution graphique de la glycémie, les alarmes à distance et surtout la géolocalisation.

2. Schéma synoptique de la solution de télésurveillance

La figure suivante montre le schéma synoptique de la solution de télésurveillance qui vise, surtout, à illustrer le principe de fonctionnement dans ses grandes lignes. Une Surveillance locale pour aider le patient à mieux gérer son diabète et une surveillance à distance par son médecin et ses proches pour être informés en temps réel en cas d'état de détresse.

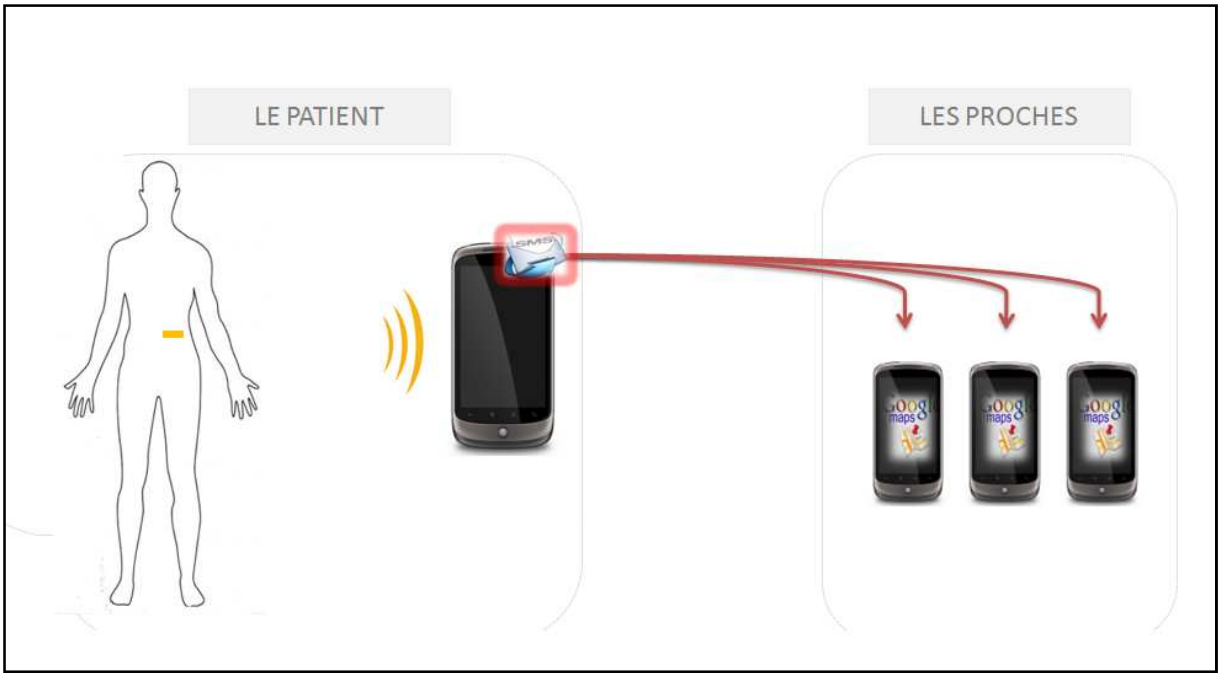


Figure 36: Le schéma synoptique de la solution de télésurveillance.

3. Le matériel nécessaire

Afin de réaliser la solution proposée, nous avons besoin, essentiellement, d'un capteur qui sera posé en sous cutané (généralement sur la partie abdominale ou à l'arrière du bras), de manière à être en contact avec le glucose interstitiel.

Le capteur interstitiel, d'une manière générale, est composé d'une électrode, recouverte de glucose oxydase, insérée en sous cutané avec une aiguille guide. La durée de vie du capteur est, donc, liée à la quantité de glucose oxydase présente sur l'électrode: entre 3 et 7 jours.

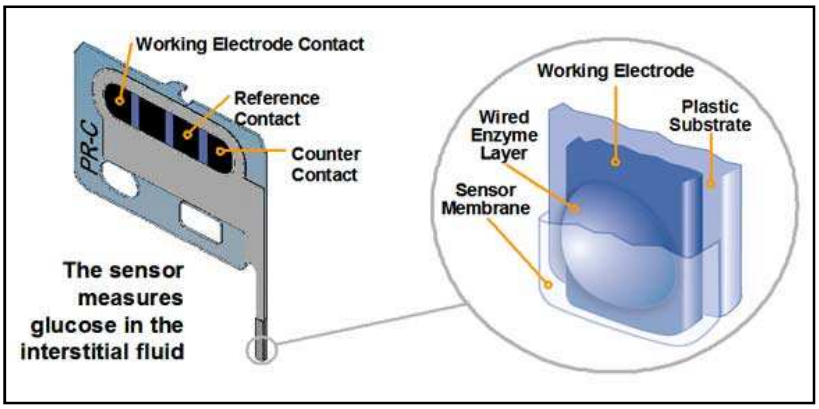


Figure 37: Le capteur de glycémie d'ABBOTT.

Au capteur, s'ajoute un transmetteur sans fils qui permettra le transfert des valeurs glycémiques vers le Smartphone via Bluetooth. L'insertion du capteur s'effectue grâce à l'inséreur que les concepteurs mettent à disposition des utilisateurs (30).

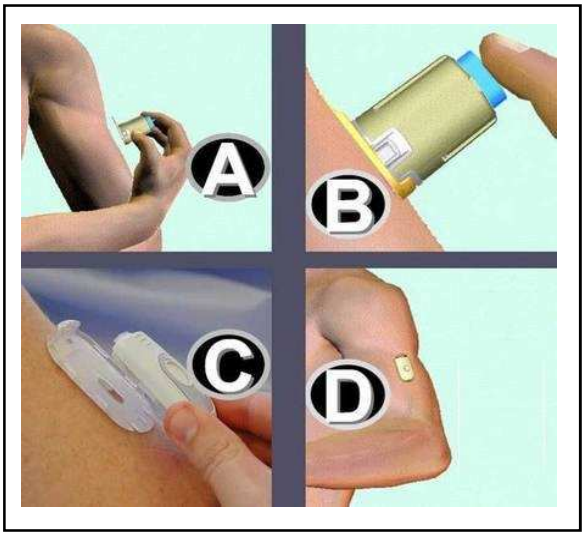


Figure 38: L'insertion du capteur.

Enfin le Smartphone, qui constitue le cerveau de notre solution, c'est d'ailleurs lui qui se chargera du reste du boulot. Notre étude est beaucoup plus axée sur ce dernier maillon car, pour ce qui est des éléments cités précédemment, on a qu'à les acheter et suivre les instructions, pas à pas, pour l'insertion et la mise en service.

4. Description fonctionnelle de la solution proposée

Comme nous pouvons le voir sur le schéma synoptique précédent, du point de vue fonctionnel, la solution proposée pour la télésurveillance du patient diabétique peut être subdivisée en deux principales parties : une pour un suivi et une surveillance locale et une autre pour le suivi à distance ou la télésurveillance.

4.1 Description de la surveillance locale

Pour la surveillance locale du patient diabétique, la stratégie adoptée repose sur l'exploitation de notre compagnon intelligent pour le suivi en continu et en temps réel de l'évolution de la glycémie.

Tout à bord, à la réception de chaque nouvelle mesure de glycémie, le Smartphone se met à la lire vocalement en même temps qu'il effectue l'affichage numérique de la mesure et l'instant de la prise.

Figure 39: L'affichage numérique.

Un rendu graphique de toutes les valeurs glycémiques issues des mesures prises par le capteur interstitiel et acheminées par réseau Bluetooth vers le Smartphone.

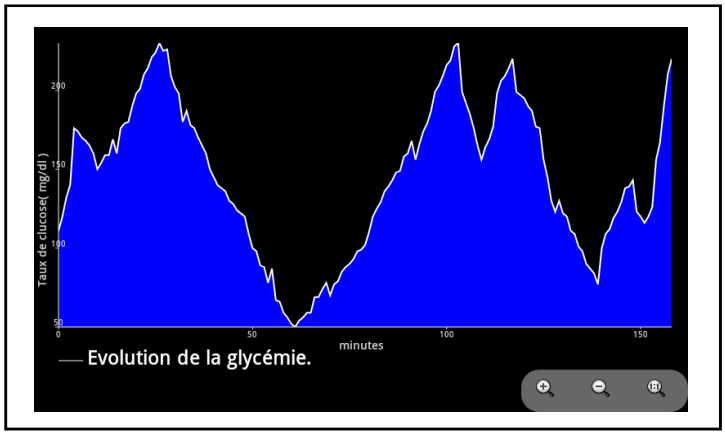


Figure 40: L'affichage graphique (Holter).

Ce graphe permet de voir la courbe de variation de la glycémie interstitielle sur une longue période, donnant ainsi la possibilité d'utiliser notre Smartphone comme "holter glycémique" qui est d'un intérêt majeure surtout que les rares mesures capillaires sont incapables de nous renseigner sur le profile glycémique réel du patient.

4.1.1 Intérêt de l'holter glycémique

Pour mieux comprendre l'intérêt de l'holter glycémique, analysons les deux figures suivantes mises en ligne par la compagnie américaine spécialisée en la matière Medtronic Inc. (31):

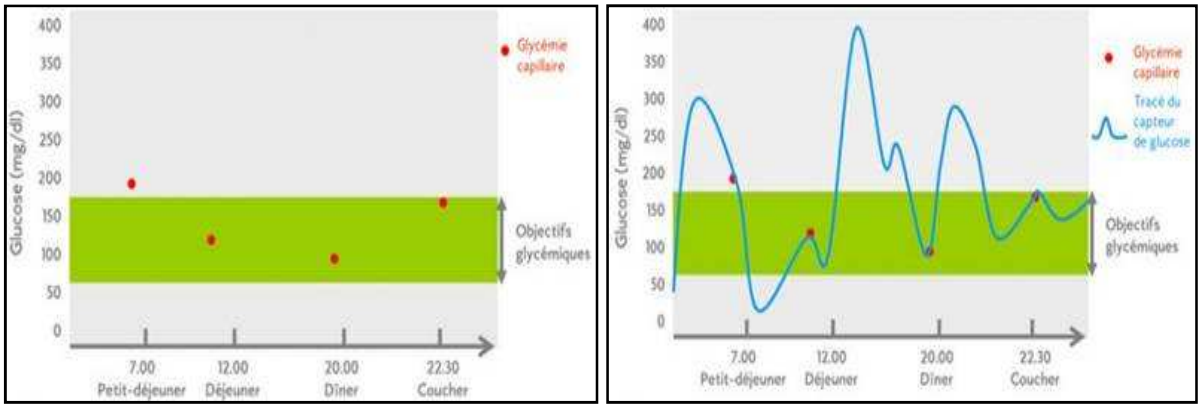


Figure 41: Le tracé à gauche est rassurant. A droite moins rassurant.

Il est clair, à partir de ces deux figures, que le port de l'holter glycémique est très bénéfique pour le patient diabétique. Il permet d'offrir un support éducatif pour le patient afin de mieux comprendre la réaction de son organisme en fonction de ses activités quotidiennes. Il permet aussi au personnel soignant d'identifier le problème et donc d'adopter une attitude thérapeutique plus spécifique à chaque patient.

Un dernier point concernant la surveillance locale est l'émission de deux types d'alarmes :

- Une alarme vocale destinée à alerter le patient en cas d'état d'hyperglycémie ou d'hypoglycémie en lui donnant son état et la dernière valeur de sa glycémie comme le montre la figure suivante :

Figure 42: Les alarmes vocales.

- Une alarme par SMS à destination du médecin et des proches du patient comportant, en plus de l'alarme qu'on a vue précédemment, les coordonnées GPS (latitude et longitude) afin de donner le lieu où se trouve le patient et son état, chose qui aura à faciliter énormément une éventuelle intervention.

4.2 Description de la télésurveillance

Pour ce qui est de la télésurveillance, elle aura essentiellement à exploiter les informations contenues dans le SMS. Le Smartphone se met en permanence à l'écoute d'un éventuel SMS en provenance du Smartphone du patient.

A la réception du SMS, la première opération qui va se dérouler est l'extraction des informations, à savoir l'alarme et les coordonnées GPS. L'alarme sera aussitôt lue vocalement et les coordonnées GPS seront exploitées pour afficher la position du patient sur Google Map.

5. Elaboration du programme

A partir de la description fonctionnelle précédente, nous pouvons déduire que les grandes lignes du programme, écrit essentiellement en langage java, des deux applications permettant la télésurveillance du patient diabétique peuvent être décrites comme suit :

5.1 Le travail en arrière plan et la gestion du multitâches

5.1.1 Le travail en arrière plan

L'un des objectifs prioritaires lors du développement d'applications Android est de travailler sur la réactivité de l'application, c'est-à-dire de faire en sorte qu'elles ne se bloquent pas sans raison apparente pendant une durée significative. En effet, Android lui-même peut repérer quand l'application n'est pas assez réactive, auquel cas il lancera une boîte de dialogue, qui s'appelle ANR, qui incitera l'utilisateur à quitter l'application. Un des cas les plus rencontrés et qui provoque l'apparition de cette boite est lorsque l'application ne répond pas à une impulsion de l'utilisateur sur l'interface graphique en moins de cinq secondes.

C'est pourquoi nous allons voir ici comment faire du travail en arrière plan, de façon à exécuter du code en parallèle de l'interface graphique, pour ne pas la bloquer quand on veut effectuer de grosses opérations qui risqueraient d'affecter l'expérience de l'utilisateur.

Afin de pouvoir exécuter du code en arrière plan, les concepteurs du système Android ont pensé à mettre en place plusieurs solutions, chacune avec ses avantages et ses limitations. Parmi ces solutions nous retrouvons :

- L'utilisation des Services ;
- L'utilisation de l'Asynctask ;
- L'utilisation des Message et Handler.

Pour la suite de cette partie, nous allons mettre de coté les deux premières solutions et nous intéresser à la dernière. Mais avant cela, nous allons voir comment parvenir à gérer le multitâche sous la plateforme Android, en définissant, au passage, certains éléments tel que le processus, le thread, etc.

5.1.2 La gestion du multitâche par Android

Un programme informatique est constitué d'instructions destinées au processeur. Ces instructions sont présentées sous la forme d'un code, et lors de l'exécution de ce code, les instructions sont traitées par le processeur dans un ordre précis. Tous les programmes Android s'exécutent dans ce qu'on appelle un processus. On peut définir un processus comme une instance d'un programme informatique qui est en cours d'exécution. Il contient non seulement le code du programme, mais aussi des variables qui représentent son état courant. Parmi ces variables, s'en trouvent certaines qui permettent de définir la plage mémoire qui est mise à la disposition du processus. Pour être exact, ce n'est pas le processus en lui-même qui va exécuter le code, mais l'un de ses constituants. Les constituants destinés à exécuter le code s'appellent des threads qu'on traduit par fils d'exécution en français. Dans le cas d'Android, les threads sont contenus dans les processus. Un processus peut avoir un ou plusieurs threads, par conséquent un processus peut exécuter plusieurs portions du code en parallèle s'il a plusieurs threads. Comme un processus n'a qu'une plage mémoire, alors tous les threads se partagent les accès à cette plage mémoire (20).

Un programme est dit multitâche s'il est capable de lancer plusieurs parties de son code en même temps et cela sans conflit. À chaque partie du code sera associé un processus pour permettre l'exécution en parallèle.

Un système monoprocesseur simule le multitâche en attribuant un temps de traitement pour chaque thread. Sur une machine multiprocesseur, des threads peuvent être exécutés sur plusieurs processeurs, donc réellement en même temps.

5.1.3 Gestion du Bluetooth en arrière plan

Tel qu'annoncé précédemment, pour la gestion de la connexion Bluetooth et la réception des valeurs glycémique du patient en continu et en temps réel, la stratégie adoptée pour accomplir cette tâche, qui représente l'un des grands morceaux de notre code, en arrière plan repose sur l'utilisation des Message et Handler. En gros ces derniers permettent d'assurer la communication entre les threads secondaires, qui s'occupent du travail en arrière plan, et le thread principal ou UI Thread, qui gère l'interface graphique, et ce par l'intermédiaire de Messages.

La figure suivante montre le schéma de principe illustrant le fonctionnement des messages et Handler pour la communication inter-threads.

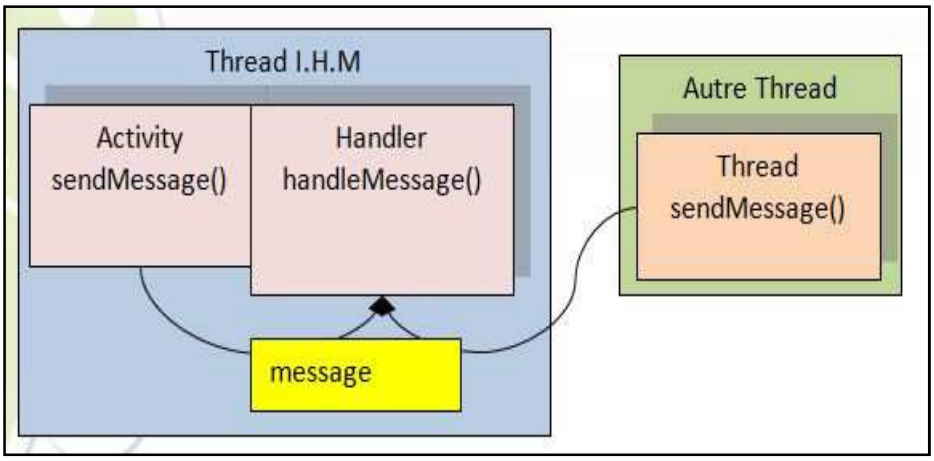


Figure 43: La communication inter-threads par Message.

En ce qui suit, les principales méthodes à appeler, que ce soit pour les threads secondaires ou le thread principal, pour bien gérer cette communication.

5.1.3.1 Thread secondaire

- Un thread secondaire qui veut communiquer avec le thread principal doit créer un nouveau Message par la méthode obtainMessage() ;
- Une fois obtenu, le thread secondaire peut remplir le message avec des données et l'insérer dans la file d'attente du handler par la méthode sendMessage().

5.1.3.2 Handler du thread principal

- Le handler utilise la méthode handleMessage() pour se mettre en attente de nouveaux messages à destination du thread principal.
- Un message extrait de la file d'attente peut, soit transporter des données à destination du thread principal, soit demander l'exécution d'objets exécutables (Runnable) avec la méthode post().
- Les principaux threads secondaires utilisés dans notre programme pour, spécifiquement, gérer la connexion Bluetooth en arrière plan sont respectivement :

    - AcceptThread : ce thread s'exécute pour mettre le Smartphone à l'écoute des connexions entrantes ;

- ConnectThread : il s'exécute pour établir une connexion via Bluetooth entre le Smartphone et le capteur de glycémie ;
- ConnectedThread : ce thread se charge de gérer la connexion une fois établie.

5.2 Exploitation de l'API Bluetooth

En ce qui suit, nous verrons comment utiliser le Bluetooth dans une application Android, en utilisant l'API fournie par Android.

Android inclus un support du Bluetooth, ce qui permet aux appareils de pouvoir échanger des données entre eux en utilisant cette technologie.

La Bluetooth API d'Android facilite, relativement, l'utilisation de cette technologie dans une application. Elle permet de :

- Scanner les autres périphériques ;
- Lier un périphérique à un autre ;
- Transférer des données d'un périphérique à un autre ;
- Gérer la multi-connexion de périphériques.

La première étape, dans l'intégration du Bluetooth dans une application, est de vérifier si le périphérique en question possède cette technologie.

Cela est très facile en utilisant l'API Bluetooth et plus particulièrement BluetoothAdapter. Cette classe permet d'exécuter les fonctionnalités basiques, comme la vérification de la présence du Bluetooth, lancer un scan des périphériques, etc.

Voici un petit bout de code, qui vérifie la présence du Bluetooth sur un appareil :

```
BluetoothAdapter bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if (bluetoothAdapter == null)
   Toast.makeText(TutoBluetoothActivity.this, "Pas de Bluetooth",
                  Toast.LENGTH_SHORT).show();
else
   Toast.makeText(TutoBluetoothActivity.this, "Avec Bluetooth",
                  Toast.LENGTH_SHORT).show();
```

Il ne faut pas oublier la permission d'utilisation du Bluetooth dans le Manifest :

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
```

La deuxième étape est d'activer le Bluetooth si nécessaire, pour cela nous disposons de deux possibilités :

- Demander à l'utilisateur de l'activer ;
- L'activer sans demander l'avis de l'utilisateur.

```
private final static int REQUEST_CODE_ENABLE_BLUETOOTH = 0;
if (!bluetoothAdapter.isEnabled()) {
   Intent enableBlueTooth = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
   startActivityForResult(enableBlueTooth, REQUEST_CODE_ENABLE_BLUETOOTH);}
```

Tout ce que fait ce code, est de vérifier si le Bluetooth est déjà activé. Sinon, il lance une boite de dialogue pour demander à l'utilisateur d'activer le Bluetooth. Il suffit de surcharger la méthode onActivityResult() pour savoir si l'utilisateur a activé le Bluetooth ou pas :

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
       super.onActivityResult(requestCode, resultCode, data);
       if (requestCode != REQUEST_CODE_ENABLE_BLUETOOTH)
           return;
       if (resultCode == RESULT_OK) {
           // L'utilisateur a activé le bluetooth
       } else {
           // L'utilisateur n'a pas activé le bluetooth
       }
}
```

Nous aurons besoin de la permission suivante :

```
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

Voici le bout de code, qui permet d'activer le Bluetooth sans l'avis de l'utilisateur :

```
if (!bluetoothAdapter.isEnabled()) {
   bluetoothAdapter.enable();
}
```

5.2.1 Obtenir la liste des appareils déjà connus

Il est possible d'obtenir la liste des appareils déjà connus et ainsi la stocker pour une utilisation ultérieure :

```
private Set<BluetoothDevice> devices;
…
devices = bluetoothAdapter.getBondedDevices();
for (BluetoothDevice blueDevice : devices) {
  Toast.makeText(TutoBluetoothActivity.this,   "Les   périphériques   =   "   +
blueDevice.getName(), Toast.LENGTH_SHORT).show();
}
```

5.2.2 Recherche de nouveaux périphériques

Maintenant, nous allons commencer à chercher si des périphériques inconnus sont disponibles pour un éventuel échange. Cette action s'effectue en plusieurs étapes :

1. Créer un BroadcastReceiver qui sera avertit lors de la détection d'un nouveau terminal ;

Nous utilisons un BroadcastReceiver afin de recevoir l'événement qui se propage lorsqu'un nouvel appareil est détecté :

```
private final BroadcastReceiver bluetoothReceiver = new BroadcastReceiver() {
public void onReceive(Context context, Intent intent) {
  String action = intent.getAction();
  if (BluetoothDevice.ACTION_FOUND.equals(action)) {
    BluetoothDevice device =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
    Toast.makeText(Activity.this, "Nouveau périphériques = " + device.getName(),
Toast.LENGTH_SHORT).show();
  }
}
};
```

2. Enregistrer le BroadcastReceiver :

```
IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
registerReceiver(bluetoothReceiver, filter);
```

3. Lancer le scan grâce à la méthode startDiscovery() :

```
bluetoothAdapter.startDiscovery();
```

5.2.3 Rendre son appareil détectable

Pour que l'appareil puisse être trouvé et connecté à d'autres appareils, il faut le rendre visible. Pour cela il suffit d'utiliser le code suivant :

```
Intent discoverableIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
discoverableIntent.putExtra (BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, 300);
startActivity(discoverableIntent);
```

5.2.4 Etablir une connexion

Pour établir une connexion et échanger des données, il faut un côté serveur (l'appareil qui reçoit la connexion) et un côté client (l'appareil qui envoie la connexion).

Pour le côté serveur**,** voici les étapes nécessaires pour configurer un socket pour un serveur Bluetooth et accepter une connexion:

- Récupérer une instance BluetoothServerSocket en appelant la méthod*e* listenUsingRfcommWithServiceRecord() *;*
- Attendre les connexions en appelant accept();
- Si on souhaite ne plus attendre de connexions on appelle close().

```
private class AcceptThread extends Thread {
   private final BluetoothServerSocket mmServerSocket;

   public AcceptThread() {
       BluetoothServerSocket tmp = null;
       try {
            tmp    =    mBluetoothAdapter.listenUsingRfcommWithServiceRecord(NAME,
MY_UUID);
       } catch (IOException e) { }
       mmServerSocket = tmp;}

   public void run() {
       BluetoothSocket socket = null;
       while (true) {
           try {
               socket = mmServerSocket.accept();
           } catch (IOException e) {
               break;
           }

           if (socket != null) {
               manageConnectedSocket(socket);
               mmServerSocket.close();
               break;
           }
       }
   }

   public void cancel() {
       try {
           mmServerSocket.close();
       } catch (IOException e) { }
 }}
```

Pour le coté client, voici la liste des étapes nécessaires pour la connexion Bluetooth côté client :

- Récupérer une BluetoothSocket grâce à la méthode createRfcommSocketToServiceRecord() ;
- Lancer la connexion en appelant connect().

```
private class ConnectThread extends Thread {
   private final BluetoothSocket mmSocket;
   private final BluetoothDevice mmDevice;

   public ConnectThread(BluetoothDevice device) {
       BluetoothSocket tmp = null;
       mmDevice = device;
       try {
           tmp = device.createRfcommSocketToServiceRecord(MY_UUID);
       } catch (IOException e) { }
       mmSocket = tmp;
   }

   public void run() {
       mBluetoothAdapter.cancelDiscovery();
       try {
           mmSocket.connect();
       } catch (IOException connectException) {
           try {
               mmSocket.close();
           } catch (IOException closeException) { }
           return;
       }
        manageConnectedSocket(mmSocket);
   }

   public void cancel() {
       try {
           mmSocket.close();
       } catch (IOException e) { }
  }

}
```

(32).

5.3 Géolocalisation sur Google Map V2

Parmi les fonctionnalités les plus appréciées sur les plateformes mobiles modernes, la géolocalisation, qui permet de réaliser des applications innovantes grâce, notamment, à Google Maps.

La démocratisation des GPS pour l'automobile et les randonneurs a rapidement conduit les constructeurs de téléphones à ajouter des fonctions de géolocalisation à leurs

appareils pour créer des applications nomades. Sous Android, les services de géolocalisation sont divisés en deux grandes parties :

- Les API qui gèrent les plans (dans l'espace de noms com.google.android.maps) ;
- Les API qui gèrent la localisation, à proprement parler (dans l'espace de noms android.location).

Normalement dans une géolocalisation habituelle, la stratégie adoptée repose sur l'utilisation du même Smartphone pour la récupération des coordonnées GPS, grâce aux méthodes de l'API qui gère la localisation, et l'exploitation de ces coordonnées pour la représentation de la position sur les cartes de Google Map. Dans notre cas, une légère différence est à signaler. En effet, pour le besoin de notre solution, nous avons dû apporter une légère modification à cette stratégie en procédant à la décomposition de cette opération en deux parties, chacune d'elles sera exécutée sur un Smartphones à part : celui du patient pour la récupération des coordonnées GPS et celui des télé-surveillants pour la représentation de ces coordonnées sur les cartes Google Map.

5.3.1 Partie I : récupérations des coordonnées GPS

Pour ce faire, du point de vue programmation, l'une de nos Activités doit implémenter l'interface LocationListener. Nous utiliserons ensuite l'objet LocationManager pour gérer notre abonnement aux mises à jour des coordonnées GPS du téléphone.

Comme notre Activité implémente l'interface LocationListener, quatre méthodes sont à surcharger :

- onProviderEnabled() : cette méthode est appelée lorsqu'une source de localisation est activée ;
- onProviderDisabled() : cette méthode est appelée lorsqu'une source de localisation est désactivée ;
- onStatusChanged() : cette méthode est appelée lorsque le statut d'une source change ;
- onLocationChanged() : cette méthode est appelée lorsque les coordonnées GPS du téléphone changent.

Pour ne pas consommer trop de data et économiser de la batterie, il convient de s'abonner à la localisation GPS de l'utilisateur uniquement quand c'est nécessaire.

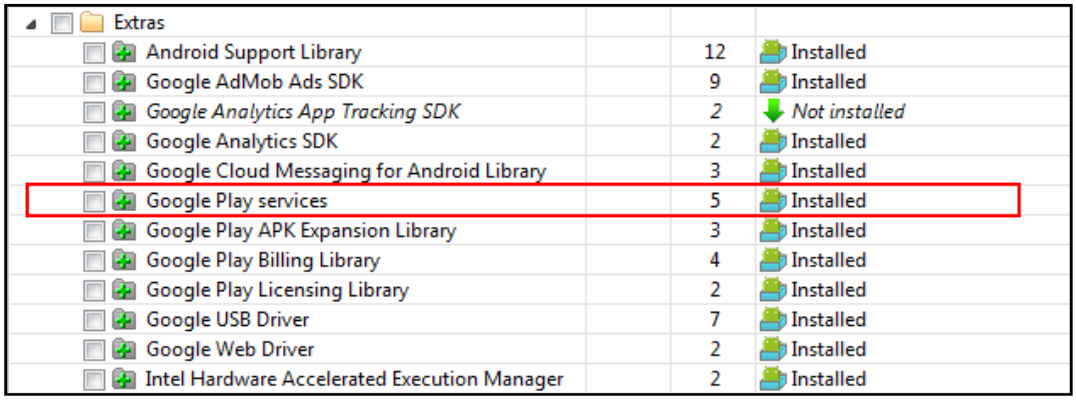
A noter qu'il est également possible de faire une géolocalisation approximative grâce aux réseaux (Wifi, GSM, etc.).

Il convient d'ajouter une autorisation, nous permettant de récupérer les coordonnées GPS de l'utilisateur. Pour ce faire, nous devons ajouter la ligne suivante au fichier Manifest:

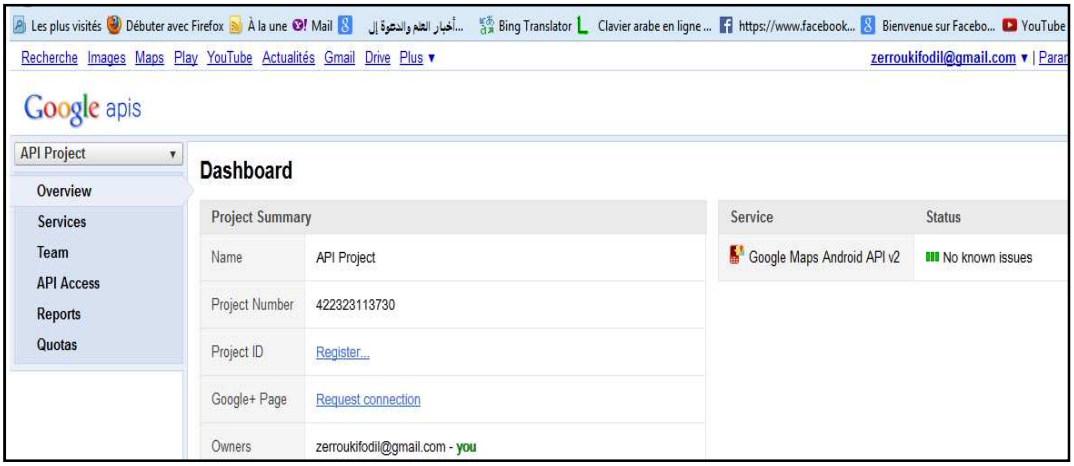
```
<uses-permission android : name= android.permission.ACCESS_FINE_LOCATION"/>
```

5.3.2 Partie II : représentation des coordonnées sur Google Map

Dans ce qui suit, nous allons essayer de décrire grossièrement les différentes étapes pour parvenir à utiliser les cartes Google Map v2. Il faut tout d'abord installer et configurer Google Play Services, disponible dans le SDK Manager.

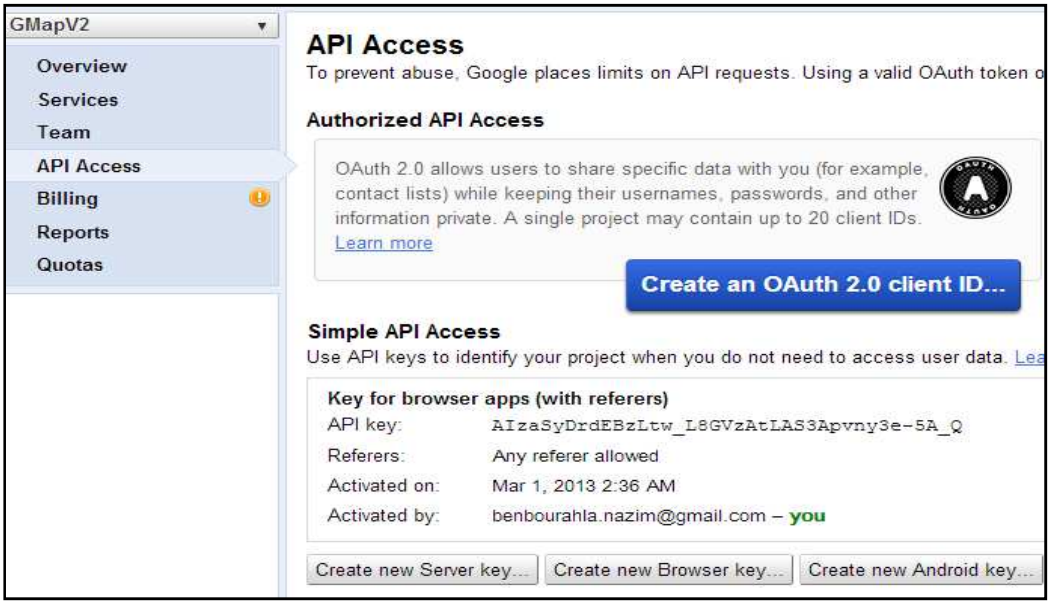


Puis, nous nous connectons avec un compte Google afin de récupérer une clé à l'adresse suivante : https://code.google.com/apis/console .

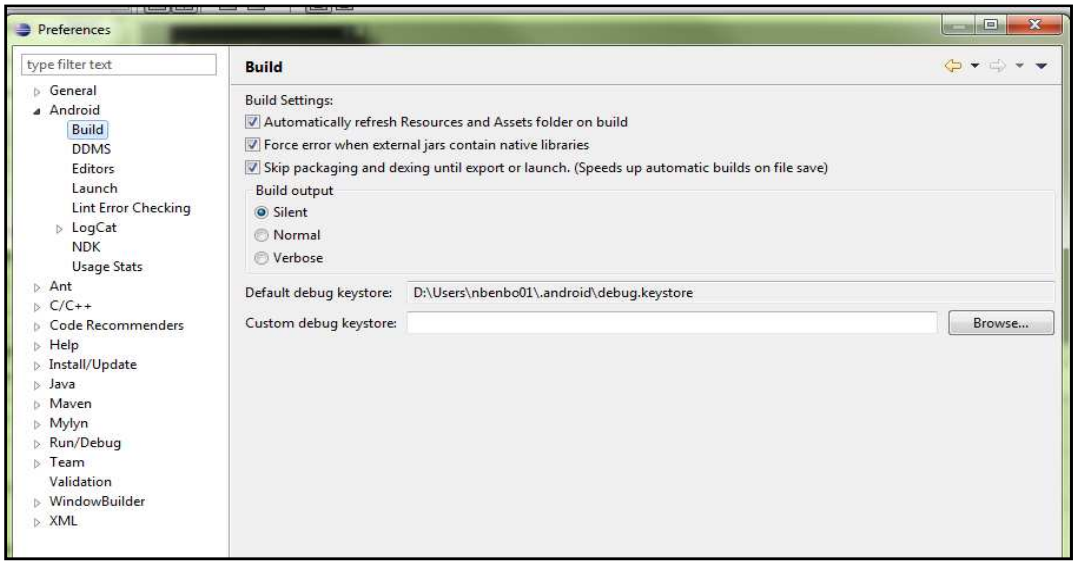


Nous créons un projet à l'aide du bouton Create, puis dans la partie Services (menu de gauche), on active le service Google Map Android API V2.

Nous cliquons sur le bouton API Access situé sur la barre de navigation (à gauche de l'écran).



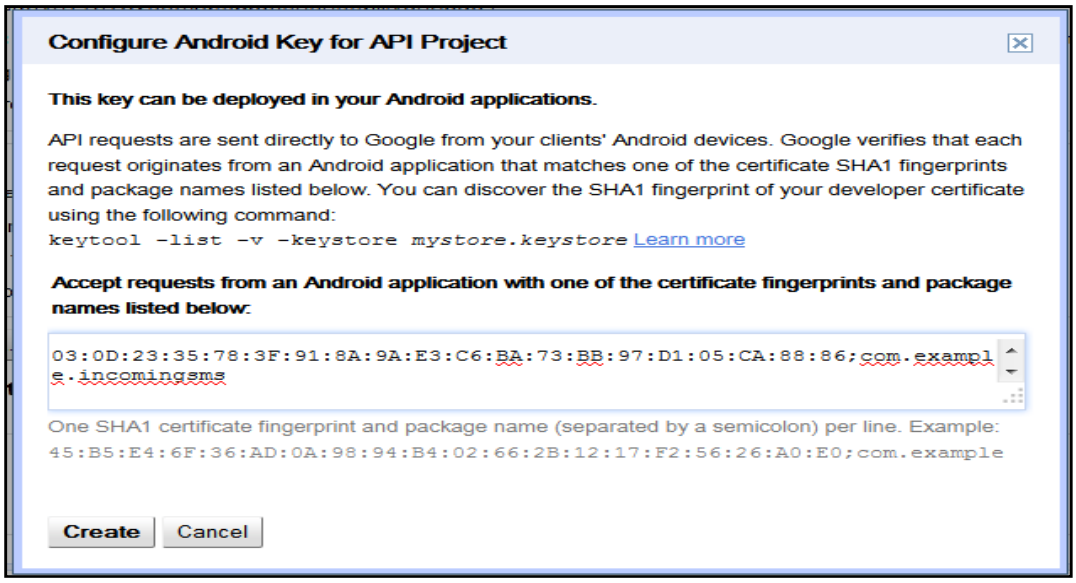
Avant d'appuyer sur Create new Android key, nous devons, d'abord, récupérez l'empreinte (SHA-1) correspondante à la clé keystore utilisée. Pour trouver l'emplacement du keystore utilisé, nous observons les préférences d'Eclipse :



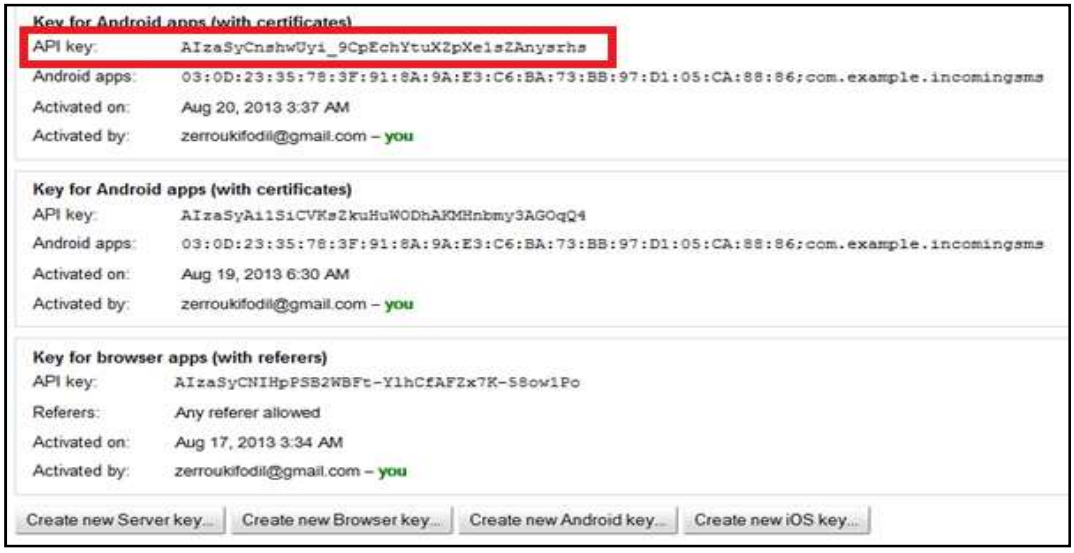
Nous récupérons la clé (SHA-1) à l'aide de la commande suivante :

```
-list -v -alias androiddebugkey -keystore "C:\Users\Fodil Zerrouki\.android\debug.keystore" -storepass android -keypass android
```

Ensuite, nous cliquons sur le bouton Create New Android Key et nous collons notre empreinte (SHA-1) ainsi que le nom du package :



Puis nous cliquons sur le bouton Create et nous obtenons, ainsi, notre clé.



Nous retournons dans le Manifest et nous rajoutons, dans la balise application, les lignes suivantes :

```
<meta-data
        android:name="com.google.android.maps.v2.API_KEY"
        android:value="La_clé" />
```

Nous rajoutons aussi les lignes suivantes, afin de déclarer une permission nécessaire à l'utilisation de Google Map (nous remplaçons com.tutos.android.gmapv2 par le nom de notre package) :

```
<permission
        android:name="com.tutos.android.gmapv2.permission.MAPS_RECEIVE"
        android:protectionLevel="signature" />

    <uses-permission
android:name="com.tutos.android.gmapv2.permission.MAPS_RECEIVE" />
```

Ainsi que les permissions (commentées) suivantes :

```
<!-- Permission pour utiliser la connexion internet -->
    <uses-permission android:name="android.permission.INTERNET" />
    <!-- Permission permettant de vérifier l'état de la connexion -->
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <!-- Permission pour stocker des données en cache de la map -->
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
```

Afin de pouvoir utiliser Google Map V2 sur un appareil, celui ci nécessite la présence de la librairie OpenGL ES Version 2. Les lignes suivantes permettent de rendre l'application indisponible aux appareils qui ne possèdent pas cette version d'OpenGL.

```
<uses-feature
        android:glEsVersion="0x00020000"
        android:required="true" />
```

(33).

5.4 Faire parler le Smartphone : TexttoSpeech

Android fournit une fonctionnalité très intéressante de synthèse vocale appelée TexttoSpeech (TTS), qui permet de lire un texte à haute voix et dans plusieurs différentes langues. En utilisant (intelligemment) une fonctionnalité comme le TTS, nous permettons non seulement aux personnes malvoyantes de pouvoir interagir avec son Smartphone, mais également à n'importe quelle personne de pouvoir utiliser une application sans forcément devoir avoir les yeux sur l'écran.

Que ce soit pour la lecture vocale des valeurs glycémiques dans la surveillance locale (ex : pendant que le patient conduit) ou la lecture de l'alarme reçu par SMS, l'intérêt d'intégrer, à notre solution, une telle fonctionnalité devient rapidement indispensable.

Pour l'utilisation du TTS dans une application Android c'est relativement simple. Avant de pouvoir se servir du TTS, il est nécessaire de vérifier la présence d'un moteur de TTS, puisque par défaut Android ne le propose pas. Certains constructeurs comme Samsung proposent le leur, d'autres incluent, de base, une version développée par d'autres.

Pour vérifier la présence d'un moteur de TTS, Android propose, à défaut d'un moteur intégré par défaut, une action pour vérifier la présence ou non de ce moteur. Il suffit, pour cela, de surcharger la méthode onActivityResult() de son activité, et de voir le résultat :

```
Intent checkIntent = new Intent();
checkIntent.setAction(TextToSpeech.Engine.ACTION_CHECK_TTS_DATA);
startActivityForResult(checkIntent, 0x01);

private TextToSpeech mTts;
protected void onActivityResult(
        int requestCode, int resultCode, Intent data) {
             if (requestCode == 0x01) {
                     if (resultCode == TextToSpeech.Engine.CHECK_VOICE_DATA_PASS)
{
// Succès, au moins un moteur de TTS à été trouvé, on l'instancie
                         mTts = new TextToSpeech(this, this);
                     } else {
// Echec, aucun moteur n'a été trouvé, on propose à l'utilisateur d'en installer
un depuis le Market
Intent installIntent = new Intent();

installIntent.setAction(TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA);
                         startActivity(installIntent);
                     }
}}}
```

Le code précédent ne fait pas que vérifier la présence d'un moteur de TTS, il propose même de le télécharger. Une fois nous avons obtenu une instance valide du moteur, nous passons à la configuration.

Tout d'abord, nous commençons par choisir la langue dans laquelle le texte sera prononcé. En effet, bien que la langue dépende du texte lu, le résultat ne sera pas le même suivant la configuration du moteur. Par exemple : Information ne se prononce pas de la même manière en français qu'en anglais.

```
if  (mTts.isLanguageAvailable(Locale.FRANCE)==TextToSpeech.LANG_COUNTRY_AVAILABLE)
{
      mTts.setLanguage(Locale.FRANCE);
}
```

Si nous n'imposons pas un certain choix de la langue, le moteur va se mettre à lire en utilisant la langue par défaut du Smartphone. En plus de la langue, il est également

possible de configurer d'autres paramètres, comme la vitesse de lecture ou bien le ton de la voix.

Il est nécessaire d'implémenter, quelque part dans le code, l'interface OnInitListener, qui oblige d'avoir une méthode à surcharger onInit().

Enfin, pour lancer la lecture, il suffit d'appeler la méthode speak() et d'y inclure le texte à lire (il faut écrire proprement le texte).

```
mTts.speak("Bonjour tout le monde", TextToSpeech.QUEUE_FLUSH, null);
mTts.speak("Ceci est un deuxième test !", TextToSpeech.QUEUE_ADD, null);
```

(34).

5.5 Intégrer des graphiques dans une application : AChartEngine

Android ne propose nativement aucune solution pour générer des courbes, camemberts et autres graphiques lors du développement d'applications. Il est pourtant fréquent que l'on souhaite afficher ce genre d'informations. Les terminaux Android disposant, pour la plupart d'une résolution plutôt petite, il est souvent plus simple et plus efficace d'afficher les valeurs d'un tableau sous forme de graphique. Pour ce faire Il existe plusieurs API permettant d'ajouter des graphiques à une application Android. Nous avons la possibilité d'utiliser des API utilisées ordinairement dans des applications Java ou des librairies propres à l'OS Android apportant plus de fonctionnalités. L'une de ces dernières est AChartEngine, qui est une librairie permettant de générer différents types de graphiques pour Android de façon gratuite et sans nécessiter une connexion internet.

Pour l'utiliser, nous devons, d'abord, télécharger le fichier jar sur le site: http://code.google.com/p/achartengine/downloads/list , puis l'ajouter au projet Android dans un dossier nommé libs.

Comme cette librairie utilise le mécanisme d'Intent (qui dit Intent dit activité), alors les graphiques sont affichés sur des activités dédiées qu'il faudrait déclarer dans le Manifest de la manière suivante :

```
</activity>
  <activity android:name="org.achartengine.GraphicalActivity" />.
```

Pour son utilisation dans notre solution, elle sera essentiellement pour voir un certain graphe dynamique donnant le profile glycémique du patient, du lancement de l'application jusqu'au moment où nous appuyons sur le bouton pour afficher le graphique. Pour ce faire

nous avons consacré toute une activité qui se chargera, dans l'ordre, de récupérer toutes les mesures de glycémie envoyées à partir de l'activité qui gère le Bluetooth et les mettre dans un vecteur par le petit code suivant :

```
Bundle contenu_intent = this.getIntent().getExtras();

final ArrayList<Integer> axe_y = contenu_intent.getIntegerArrayList("ordonnées");
```

Une fois nous avons les mesures, nous passons à leur représentation sur une vue que nous récupèrons grâce à :

```
view =  ChartFactory.getLineChartView(context, mDataset, mRendu);
```

Nous rafraîchissons la vue pour chaque valeur grâce à la méthode repaint(), afin d'obtenir un graphe dynamique.

Pour personnaliser le rendu :

```
private XYSeriesRenderer rendu = new XYSeriesRenderer();
            rendu.setChartValuesTextSize(25);
            rendu.setColor(Color.WHITE);
            rendu.setPointStyle(PointStyle.POINT);
            rendu.setLineWidth(2);
            rendu.setFillPoints(true);
            rendu.setFillBelowLine(true);
            rendu.setFillBelowLineColor(Color.BLUE);
```

Pour personnaliser le support, sur lequel s'affiche le graphe :

```
private XYMultipleSeriesRenderer mRendu = new XYMultipleSeriesRenderer();
      mRenderer.setShowCustomTextGrid(true);
            mRendu.setAxesColor(Color.WHITE);
            mRendu.setApplyBackgroundColor(true);
            mRendu.setBackgroundColor(Color.BLACK);
            mRendu.setLegendTextSize(25);
            mRendu.setZoomButtonsVisible(true);
            mRendu.setXTitle("minutes");
            mRendu.setAxisTitleTextSize(15);
            mRendu.setYTitle("Taux de clucose( mg/dl )");
```

(35)

5.6 Démarrage automatique de l'application à la réception du SMS

L'une des particularités et des puissances de notre solution est le fait que le démarrage de l'application, déployée sur le Smartphone des télé-surveillants, se fasse d'une manière

totalement automatisée (sans aucune intervention) et ce à la réception de l'alarme, par SMS, en provenance, exclusivement, du Smartphone du patient.

Le principe est le suivant : à chaque réception de SMS, l'application démarre et procède à la récupération du numéro du destinateur. Si ce numéro correspond à celui du patient, l'application passe à l'exécution des autres étapes qui consistent à récupérer le corps du SMS et d'en extraire l'alarme et les coordonnées GPS. L'alarme sera lue à haute voix grâce à l'API TexttoSpeech. Les coordonnées GPS seront utilisées pour représenter la position du patient sur les cartes Google Map. Dans le cas où le destinateur n'est pas le Smartphone du patient, l'application s'arrête directement.

Pour la programmation, voici le code qui permet d'écouter la réception de SMS en provenance du Smartphone du patient :

```
public class IncomingSmsListener extends BroadcastReceiver {
          private String numero_patient="+213773830224";

             public void onReceive(Context context, Intent intent) {
             Bundle bundle = intent.getExtras();

              if (bundle != null) {
                  Object[] pdus = (Object[])bundle.get("pdus");
                  SmsMessage[] message = new SmsMessage[pdus.length];
                  String messageBody = null;
                  String phoneNumber = null;
                  for (int i=0; i< message.length;i++) {

                      message[i] = SmsMessage.createFromPdu((byte[])pdus[i]);
                  }
                  messageBody = message[0].getMessageBody();
                  phoneNumber = message[0].getDisplayOriginatingAddress();
                  if (message.length > -1) {

//Si le numero du destinateur correspond au numero du patient
                      if (phoneNumber.equals(numero_patient)) {

/* Executer les taches suivantes:
   1)-lecture de l'alarme vocalement
   2)-representation de la position du patient sur Google Map */}

/*Si le destinateur n'est pas le patient
Mettre fin au broadcastreceiver en appelalnt la methode finalize()*/

                      try      {finalize();}      catch      (Throwable      e)
{e.printStackTrace();}}}}}
```

Pour en finir, nous devons déclarer notre receiver dans le fichier Manifest, en lui spécifiant de démarrer à chaque arrivée de SMS et ce, en remplaçant le code suivant :

```
        <activity
            android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
```

servant à lancer l'activité principale par un simple clic sur son icône, par le code :

```
         <receiver android:name=".IncomingSmsListener">
            <intent-filter>

                <action android:name="android.provider.Telephony.SMS_RECEIVED" />

            </intent-filter>
        </receiver>
```

qui permet de lancer le receiver à chaque réception de SMS (36).

6. Conclusion

Dans ce dernier chapitre, nous avons, dans un premier temps, fait une description fonctionnelle de la solution de télésurveillance ainsi que le matériel nécessaire à sa réalisation, puis nous somme passés à l'élaboration des programmes des deux applications constituant notre solution. Nous nous somme surtout basés dans cette dernière partie sur les grandes lignes comme le travail en arrière plan, la gestion du Bluetooth et la géolocalisation par GPS.

Conclusion générale

Conclusion générale

L'objectif de ce travail était la conception et la réalisation d'un système ambulatoire qui aura à sa charge le suivi et la surveillance d'un patient diabétique et ce, en exploitant les Smartphones sous Android.

La solution proposée, à l'issue de ce travail, permet d'assurer de façon efficace l'objectif visé.

Il faut, cependant, signaler l'indisponibilité du capteur de glycémie interstitiel (d'ABBOTT) pendant la conception de notre solution, chose qui nous a énormément ralenties.

L'alternative trouvée était d'utiliser une tablette à la place du capteur/émetteur. Pour cela, nous avons conçu une application que nous avons chargée sur la tablette. Cette application permet d'envoyer des mesures, introduites manuellement, afin de voir la réaction de notre système à ces dernières (pour le Smartphones c'est comme s'il reçoit des valeurs en provenance du capteur réel ; et si ça a marché avec la tablette ça devrait forcément marcher avec le capteur/émetteur réel).

En guise de perspectives, plusieurs travaux pourraient être envisagés pour poursuivre ce projet. Tout d'abord, nous pouvons penser à intégrer une pompe à insuline à notre solution et la commander par Smartphone. Ce dernier, et grâce à un régulateur PID, gérera la sécrétion de l'insuline afin d'assurer un certain équilibre glycémique. Ce genre de systèmes fait partie de ce qu'on appelle communément : le pancréas artificiel.

Une autre application, toujours dans le domaine du diabète et de la glycémie, consiste à concevoir une application en traitement d'image qui permettra de mesurer la glycémie grâce à l'appareil photo du Smartphone. L'idée est de lire automatiquement la couleur de la bandelette colorimétrique et d'en déduire la valeur de la glycémie.

Comme autre perspective, nous pourrons utiliser notre système comme Holter pour différentes grandeurs biomédicales. Nous pourrons même obtenir automatiquement un diagnostic primaire.

Nous pouvons aussi penser à concevoir un système plus général, sous forme d'une station qui assurera l'acquisition simultanée de plusieurs paramètres médicaux : la tension artérielle, la température corporelle, l'ECG, etc. La connexion pouvant même s'effectuer sans aucun fil, grâce à la gestion des connexions multiple par Android.

Pour clore, nous dirons que la seule et unique chose qui pourra mettre une limite aux perspectives c’est bien notre imagination.

Bibliographie

1. **Laila, Mona.** La télémédecine et les technologies d'assistance en gérontologie: Modélisation du besoin, de la prescription et du suivi. s.l. : PAF, 2012.

2. **Payen-Rousseau, Anne-Laure.** Médecine humanitaire : conception d'un projet pilote de télémédecine. s.l. : Thèse de doctorat, Université Pierre et Marie Curie, France , 2012.

3. **Ivica Klapan, Ronald Poropatich.** Remote Cardiology Consultations Using Advanced Medical Technology. s.l. : NATO Science Series, 2008.

4. **Carre, Dominique.** La santé et les autoroutes de l'information: La greffe informatique. s.l. : L'Harmattan, 2001.

5. **Duchene, Florence.** Exploitation de données multicapteurs dans un systéme de télésurveillance médicale de patients à domicile. s.l. : Rapport de stage, Université Joseph Fourier, France, 2001.

6. **World Health Organization.** Telemedecine: Opportunities and developments in Member States. s.l. : Global Observatory for eHealth series, 2010.

7. **Dufrane, Denis.** Le patch sous-cutané d'insuline : un espoir pour les diabétiques de type 1. s.l. : ABD, 2013.

8. **National Diabetes Information Clearinghouse.** The Diabetes Control and Complications Trial and Follow-up Study. s.l. : NIH, 2008.

9. **Dubois-Laforgue, Danièle et Timsit, José.** Diabète de type 1 et environnement. s.l. : M/S 2000, 2000.

10. **HALIMI, Serge.** Le diabète de type 2 ou diabète non insulinodépendant. 2005.

11. **Hadden, David.** Diabète gestationnel:ce que toute mère doit savoir. s.l. : Diabets Voice, 2002.

12. **National Diabetes Information Clearinghouse.** Hypoglycemia. s.l. : NIH, 2008.

13. **Gérard Coutouly, Emile Klein,Eric Barbieri,Mostafa Kriat.** Travaux dirigés de biochimie, biologie moléculaire et bioinformatique. [En ligne] 2006.

14. **Osbrink, Ruth.** Off the Bench - Our Glucometer Circuit . *Dashboard.* [En ligne] 2010. https://wiki.engr.illinois.edu/display/BIOE414/Off+the+Bench+-+Our+Glucometer+Circuit.

15. **Vivianne Amiet, Sierre.** Questions au spécialiste / Fragen an den Spezialisten. s.l. : Paediartica, 2003.

16. **Enejder, Annika M. K., et al.** Raman spectroscopy for noninvasive glucose measurements. s.l. : Journal of Biomedical Optics, 2005.

17. **Katika, Kamal M. et Pilon, Laurent.** Feasibility analysis of an epidermal glucose sensor based on time-resolved fluorescence. s.l. : Applied Optics, 2007. .

18. **Radermecker, Régis.** Le pancréas artificiel : un rêve devenu presque réalité. s.l. : ADB, 2012.

19. **Gartner, Inc.** Gartner Says Smartphone Sales Grew 46.5 Percent in Second Quarter of 2013 and Exceeded Feature Phone Sales for First Time. *Gartner.* [En ligne] 2013. http://www.gartner.com/newsroom/id/2573415.

20. **Espiau, Frédéric.** Créez des applications pour Android. s.l. : Le livre du zero, 2012.

21. **Mark, Murphy.** Beginning Android 3. s.l. : Apress, 2011.

22. **Cerille, Herby.** Apprenez à prengrammer en java. s.l. : Le livre du zero, 2011.

23. **Ted Coombs, Roderico DeLeon.** Basic Home Networking. s.l. : Delmar Learning, 2003.

24. **Albert S. Huang, Larry Rudolph.** Bluetooth Essentials for Programmers. s.l. : Cambridge University Press, 2007.

25. **Muller, Jean-Philippe.** Wireless LAN : techniques RF, Wifi, Bluetooth. s.l. : Scribd, 2002.

26. **Correia, Paul.** Guide pratique du GPS. s.l. : Eyrolles, 2012.

27. **Cosandier, Jean-Luc.** Principes généraux de la localisation par satellites. s.l. : Snastro, 2003.

28. **Elliott D. Kaplan, Christopher J. Hegarty.** Understanding GPS: Principles and Applications. s.l. : Artech House, 2005.

29. **El-Rabbany, Ahmed.** Introduction to GPS: The Global Positioning System. s.l. : Amazon , 2002.

30. **Abbott Diabetes Care Inc.** FreeStye NAVIGATOR continuous Glucose Monitoring System (User's Guide). s.l. : Abbott, 2008.

31. **Andreu, Nicolas.** Surveillance glycémique en continu:Une nouvelle étape vers la boucle fermée.La mesure en continu du glucose: . *Bouclons le diabète!* [En ligne] 2013. http://www.bouclonslediabete.org/techno1.

32. **Benbourahla, Nazim.** Utilisation du Bluetooth dans une application Android. *Tuto-Android.* [En ligne] http://www.tutos-android.com/utilisation-bluetooth-application-android.

33. —. Introduction à Google Map V2. *Tutos-Android.* [En ligne] http://www.tutos-android.com/introduction-a-google-map-v2.

34. **Avinash, Gupta.** Using Text to Speech (TTS) engine in an Android application. *Code project- For those who code.* [En ligne] http://www.codeproject.com/Articles/655709/Using-Text-to-Speech-TTS-engine-in-an-Android-appl.

35. **Ehret, Guillaume.** Graphique sous une application Android. *Java mind.* [En ligne] http://javamind-fr.blogspot.com/2013/01/graphique-sous-une-application-android.html.

36. **Benbourahla, Nazim.** BroadCast Receiver sous Android. *Tutos-Android.* [En ligne] http://www.tutos-android.com/broadcast-receiver-android.

Annexes

Annexe I : L’hémoglobine glyquée (HBA1C)

1 Introduction

L’hémoglobine glyquée, de son petit nom HbA1c, est la sœur jumelle de la glycémie. Elles sont toutes les deux des marqueurs du contrôle du diabète. C’est un examen (prise de sang) dont le résultat permet de juger l’équilibre glycémique pendant environ les 2 à 3 mois qui précèdent la prise de sang. Il indique le risque de complications à long terme. Il est lié à la lecture et à l’interprétation de votre carnet de surveillance.

2 Un lien entre hémoglobine et glucose

L’hémoglobine est présente dans les globules rouges du sang. Elle transporte l’oxygène dans tout l’organisme. Comme plusieurs protéines, elle se lie avec le glucose. Plus la glycémie est élevée, plus il y a du glucose lié avec l’hémoglobine. Les globules rouges vivent environ 120 jours. Le pourcentage d’hémoglobine glyquée (c’est-à-dire liée avec le sucre) ou HbA1C est fonction de l’équilibre glycémique durant une période de 2 à 3 mois.

3 Conclusion

Le résultat de cet examen est important. Vous avez une vision globale de votre équilibre durant les deux à trois mois précédents. C’est plus que la moyenne arithmétique de vos glycémies (même si le lien est fort avec cette moyenne), car l’hémoglobine glyquée est le reflet global de l’équilibre.

Annexe II : Calcul mathématique des coordonnées GPS

1 Introduction

La navigation par satellite GPS (Global Positioning System), avec de petits récepteurs portatifs, est largement utilisée par les unités militaires, les marins, les pilotes, les livreurs, les randonneurs... De plus en plus de voitures, notamment les voitures de location, en sont aussi équipées. C'est un moyen très fiable pour déterminer sa position géographique.

Nous allons nous intéresser à l'aspect mathématique du fonctionnement d'un GPS. L'idée sur laquelle est basée le GPS est une trilatération en trois dimensions. Un point sur la surface de la terre est déterminé par ses distances respectives avec 3 autres points. Le premier point est le lieu où est situé le récepteur GPS et les trois autres points sont les positions des satellites. Les distances sont mesurées grâce au temps que mettent les signaux radio pour parvenir au récepteur. Cette méthode requiert un calcul de temps très précis ainsi que l'ajout d'une petite correction pour tenir compte de la réalité, qui n'est pas une trilatération spatiale pure. Pour cela, il est en fait nécessaire d'utiliser quatre satellites, plutôt que trois, ce qui permet de calculer à la fois le lieu mais aussi le temps exact au lieu de réception du signal GPS. Pour pouvoir rendre l'exemple plus accessible, nous avons dû faire quelques simplifications :

1. On prendra les formules géométriques exactes négligeant les erreurs de mesure toujours présentes dans la réalité. En réalité, la méthode des moindres carres pour obtenir de bonnes approximations est la plus couramment utilisée.
2. On suppose la terre parfaitement ronde et on prend comme unité de longueur le rayon de la terre (6371 km = 1 rayon). Si (x, y, z) est notre système de coordonnées, avec le centre de la terre comme origine, pour tout point au niveau de la mer, on a : $x^2 + y^2 + z^2 = 1$.
3. Les signaux radio se déplacent à la vitesse de la lumière, soit 0.047 rayon/millisecondes c'est-à-dire :

$$\frac{2.99792458 \cdot 10^8 \text{ m/s}}{6371 \cdot 10^3 \text{ m}} = 0.047 \cdot 10^3 \text{ rayon/s}$$

Prenons par exemple un chalutier en pleine mer parti de Brest (France) et qui doit se rendre à Halifax (Nouvelle-Ecosse). Apres une forte tempête, le capitaine veut vérifier sa position exacte dont il n'est plus trés sûr. Pour cela, il utilise son système GPS. Les quatre satellites lui renvoient les coordonnées suivantes :

Satellite	Position(x0, y0, z0))	Temps (t0)
1	(0.94, 2.05, 0.00)	19.9
2	(1.95, 0.00, 2.06)	2.4
3	(1.02, 0.88, 1.14)	32.6
4	(2.03, 0.97, 0.00)	19.9

Tableau 1: Position et temps d'envoi du signal pour les quatre satellites.

Ci-dessus, les coordonnées x, y et z s'expriment en rayon et le temps en millisecondes. Les quatre satellites lui ont renvoyé ces informations grâce auxquelles il va chercher à déterminer sa position exacte. Comment la position de quatre satellites ainsi que le temps d'envoi de leur signal jusqu'au bateau permettent-ils de calculer une position à la surface de la terre ? La réponse à cette question revient à chercher à comprendre le principe de fonctionnement d'un système GPS. Pour cela, nous allons chercher à exprimer la distance entre le satellite et le bateau en fonction des coordonnées du bateau pour chaque satellite, ce qui nous donnera un système d'équations non linéaires. Nous montrerons ensuite qu'il est possible de rendre le système linéaire pour pouvoir utiliser des méthodes de résolution connues et nous permettre ainsi de trouver la position du chalutier.

2. Modélisation

Dans cette partie, nous allons tenter de trouver un système d'équations modélisant le fonctionnement d'un système GPS.

Nous allons tout d'abord chercher à exprimer la distance entre chaque satellite et le bateau, en fonction des coordonnées (x, y, z) du bateau et du temps t de réception du signal.

2.1 Expression de la distance

Pour calculer la distance entre chaque satellite et le bateau (lieu de réception GPS), il faut poser :

(x, y, z) = la position du bateau en rayons.

t = le temps lorsque le signal arrive en millisecondes.

Ce sont ces quatre inconnues que l'on va chercher à déterminer. Il est à noter que le temps t est unique car tous les signaux sont reçus en même temps par le chalutier. Le signal radio part au temps t0, donné par le satellite et arrive au temps t, se déplaçant, d'après nos

hypothèses, à la vitesse de la lumière, soit 0.047 rayon par milliseconde. On obtient donc la distance d entre le bateau et le satellite :

$$di = 0.047(\mathrm{t} - \mathrm{t0i}) \text{ pour i } = 1, 2, 3, 4. \quad (1)$$

D'autre part, la distance du bateau au satellite i est aussi égale a :

$$di = \sqrt{(\mathrm{x} - \mathrm{x0i})^2 + (\mathrm{y} - \mathrm{y0i})^2 + (\mathrm{z} - \mathrm{z0i})^2} \quad (2)$$

Grâce à l'équation 1 et l'équation 2, on obtient :

$$(\mathrm{x} - \mathrm{x0i})^2 + (\mathrm{y} - \mathrm{y0i})^2 + (\mathrm{z} - \mathrm{z0i})^2 = 0.0472^2(\mathrm{t} - \mathrm{t0i})^2 \quad (3)$$

Cette équation est une équation non linéaire. On peut appliquer le même raisonnement aux données de chaque satellite de manière à obtenir quatre équations non linéaires, à quatre inconnues. Tel qu'il est, ce système ne serait pas résoluble facilement. Cependant, nous allons montrer qu'il est possible de transformer en un système et ainsi de simplifier la résolution.

2.2 Détermination du système d'équations linéaires

Si on reprend l'équation 3 pour chaque satellite, d'après les données du tableau 1, on obtient :

$$(x - 0.94)^2 + (y - 2.05)^2 + (z - 0.00)^2 = 0.0472^2(t - 19.9)^2$$

$$(x - 1.95)^2 + (y - 0.00)^2 + (z - 2.06)^2 = 0.0472^2(t - 2.4)^2$$

$$(\mathrm{x} - 1.02)^2 + (\mathrm{y} - 0.88)^2 + (\mathrm{z} - 1.14)^2 = 0.0472^2(\mathrm{t} - 32.6)^2$$

$$(\mathrm{x} - 2.03)^2 + (\mathrm{y} - 0.97)^2 + (\mathrm{z} - 0.00)^2 = 0.0472^2(\mathrm{t} - 19.9)^2$$

En développant :

$$\mathrm{x}^2 + \mathrm{y}^2 + \mathrm{z}^2 - 1.88\mathrm{x} - 4.10\mathrm{y} - 0.00\mathrm{z} + 4.211 = 0.0022\mathrm{t}^2 - 0.088\mathrm{t} \quad (4)$$

$$\mathrm{x}^2 + \mathrm{y}^2 + \mathrm{z}^2 - 3.90\mathrm{x} - 0.00\mathrm{y} - 4.12\mathrm{z} + 8.033 = 0.0022\mathrm{t}^2 - 0.011\mathrm{t} \quad (5)$$

$$\mathrm{x}^2 + \mathrm{y}^2 + \mathrm{z}^2 - 2.04\mathrm{x} - 1.76\mathrm{y} - 2.28\mathrm{z} + 0.767 = 0.0022\mathrm{t}^2 - 0.144\mathrm{t} \quad (6)$$

$$\mathrm{x}^2 + \mathrm{y}^2 + \mathrm{z}^2 - 4.06\mathrm{x} - 1.94\mathrm{y} - 0.00\mathrm{z} + 4.187 = 0.0022\mathrm{t}^2 - 0.088\mathrm{t}. \quad (7)$$

On obtient un système de quatre équations non linéaires. Cependant les termes du deuxième degré sont les mêmes dans toutes les équations, on peut donc soustraire l'équation 4 a chacune des trois autres équations :

$$(5-4) \rightarrow -2.02x + 4.10y - 4.41z - 0.077t = -3.82 \quad (5')$$

$$(6-4) \rightarrow -0.16x + 2.34y - 2.28z + 0.056t = 3.44 \quad (6')$$

$$(7-4) \rightarrow -2.18x + 2.16y + 0.00z - 0.000t = 0.02 \quad (7')$$

On obtient ainsi un système de trois équations à quatre inconnues. Nous allons donc obtenir un ensemble de solutions paramétriques. Il suffira ensuite de les introduire dans l'équation 4 non linéaire que nous n'utilisons pas pour obtenir les coordonnées de la position du bateau.

3. Résolution du système

À partir du système linéaire défini par l'équation 4, l'équation 5', l'équation 6' et l'équation 7', on peut déduire la matrice augmentée et après plusieurs étapes, on obtient

$$\begin{bmatrix} -2.02 & 4.10 & -4.41 & -0.077 & \vdots & -3.82 \\ -0.16 & 2.34 & -2.28 & -0.056 & \vdots & 3.44 \\ -2.18 & 2.16 & 0.00 & 0.00 & \vdots & 0.02 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & -0.096 & \vdots & 5.40 \\ 0 & 1 & 0 & -0.097 & \vdots & 5.46 \\ 0 & 0 & 1 & -0.068 & \vdots & 3.71 \end{bmatrix}$$

Ce qui nous donne :

$$\begin{cases} x = 5.40 - 0.096t \\ y = 5.46 - 0.097t \\ z = 3.71 - 0.068t. \end{cases}$$

Les variables de position sont donc exprimées en fonction du paramètre t que l'on doit maintenant déterminer. Si on remplace ces valeurs (x, y, z) dans l'équation 4, on obtient une équation quadratique a une seule inconnue qu'on peut résoudre facilement. On trouve:

$$0.021t^2 - 1.936t + 44.436 = 0 \quad (4')$$

D'où :

$$t1 = 47.75\ \text{ms}, \quad t2 = 44.19\text{ms}$$

En remplaçant dans le système ci-dessus, on trouve les coordonnées du point P du bateau : (0.69, 0.70, 0.37) et (1.26, 1.28, 0.78).

Finalement, selon l'hypothèse 2, il faut que ces coordonnées vérifient :

$x^2 + y^2 + z^2 = 1$

Seul le premier ensemble de coordonnées vérifie l'équation de la sphère. Le chalutier se trouve donc au point P de coordonnées (0.69, 0.70, 0.37).

4. Conclusion

Le bateau a donc pu retrouver sa position. Il a pour coordonnées (0.69, 0.70, 0.37). En pratique, d'autres facteurs entrent en ligne de compte, la terre n'étant pas réellement sphérique. Ici, nous avons aussi pris comme hypothèse que le point cherche se situait au niveau de la mer, mais le système GPS doit e réalité tenir compte du relief. Malgré ces approximations, nous avons ici montré un modèle satisfaisant de la méthode du GPS.

Enfin, nous avons obtenu des coordonnées selon (x, y, z), mais le plus souvent, celles-ci sont converties en terme de latitude et longitude.