Remerciements

Nous remercions tout d'abord notre tuteur de projet **M. Yann Mathet**, enseignant-chercheur à l'université de Caen et responsable des projets de deuxième année du Master, pour ses conseils et sa disponibilité.

Ensuite nous tenons à remercier tous particulièrement **Jean Boudet**, notre collègue de promotion pour sa participation lors de nos tests de réception des signaux GPS.

Merci aussi aux **administrateurs système** de l'université d'avoir été présents lorsque nous avions besoin d'eux.

Enfin, nous tenons à remercier tous **nos collègues de promotion** dans laquelle règne une atmosphère agréable de solidarité et d'entraide.

Table des matières

Introduction

Ce projet prend place dans le cadre de notre Master Document Numérique en Réseau Ingénierie de l'Internet (DNR2I). Il sera fait au cours de ce rapport l'avancement de notre projet concernant le développement d'une application servant à la gestion de trajets pour les sportifs amateurs de course à pieds. Nous verrons au cours de ce rapport les enjeux du projet, nous analyserons ensuite les technologies employées ainsi que nos choix d'implémentations. Puis nous verrons en détail la réalisation du projet. Et enfin, nous terminerons par un bilan généralisant notre travail sur ce projet.

Première partie

Présentation et objectifs

Chapitre 1

Présentation du projet

1.1 Contexte

De nos jours, les smartphones disposent tous d'une puce GPS permettant leur géolocalisation à chaque instant avec une précision de plus en plus grande. C'est donc l'outil idéal pour les joggers et cyclistes qui souhaitent enregistrer et analyser leurs performances. De nombreuses applications ont ainsi vues le jour ces dernières années pour répondre à ces besoins. On peut notamment évoquer le leader sur le marché des applications mobiles dans la catégorie santé et remise en forme qu'est Runkeeper ou encore un de ses concurrents Runtastic.

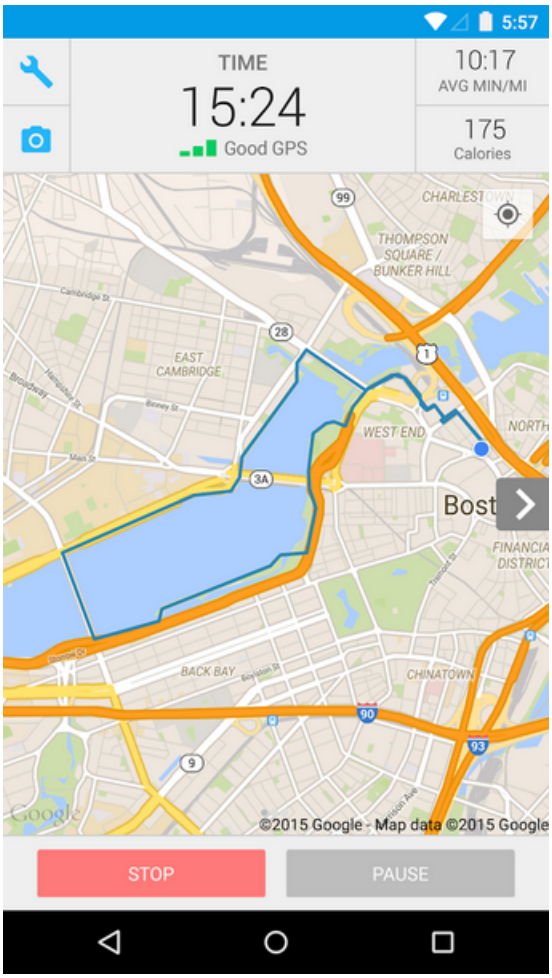

FIGURE 1.1 – Application RunKeeper

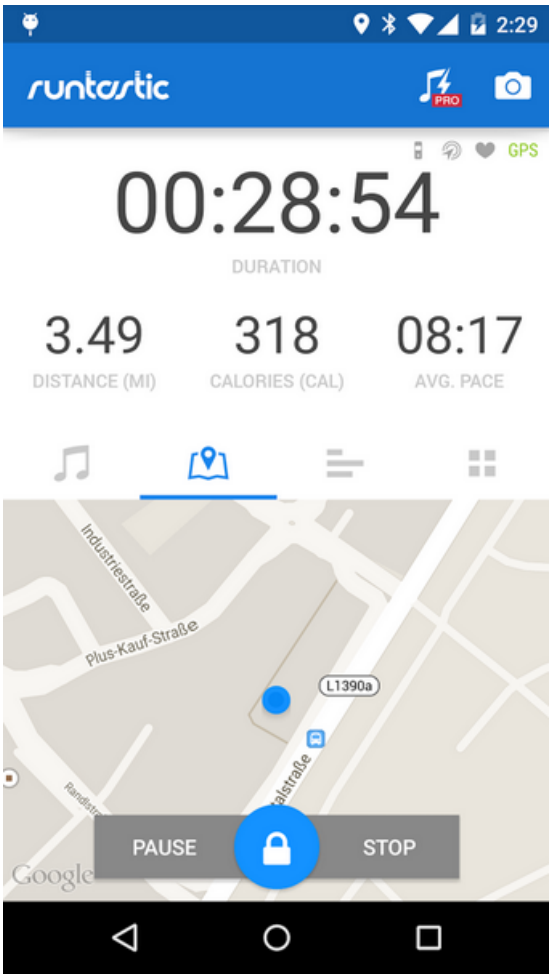

FIGURE 1.2 – Application Runtastic

Classiquement, ces deux applications propose d'enregistrer les trajets de l'utilisateur, d'afficher son parcours sur une carte et de calculer en temps réel les différentes données de la course. Mais certaines fonctionnalités qui pourrait se montrer trés intéressantes pour l'utilisateur ne sont pas proposées.

C'est dans ce contexte que notre tuteur, M. Mathet nous a proposé de concevoir et de développer une application mobile permettant d'enregistrer les tracés GPS des parcours de l'utilisateur et d'analyser ses performances. Mais aussi, et c'est ici que tous l'enjeu du projet et la nouveauté se situe, de comparer en temps réel les performances de l'utilisateur par rapport aux précédentes effectuées sur le même parcours. Concrètement, il s'agit d'afficher l'avance ou le retard en secondes qu'il a par rapport à une performance antérieur définie (la meilleure ou la moyenne des dernières).

Chapitre 2

Objectifs

Les objectifs de ce projet sont multiples. D'abord il s'agit de réaliser une application mobile pour smartphone Android qui dispose des fonctionnalités suivantes :

• Enregistrement des coordonnées GPS du trajet • Visualisation du trajet sur une carte • Analyse des données pour obtenir : o le temps du trajet o la distance parcourue o la vitesse moyenne o l'allure (temps pour parcourir un kilomètre)

Le second objectif est certainement le cœur du projet puisque c'est dans celui-ci que réside la nouveauté. Il s'agit de regrouper les trajets identiques pour pouvoir les comparer. Un parcours représente alors un ensemble de trajet qui suivent à peu de chose près la même trajectoire.

Deuxième partie

Analyse et implémentations

Chapitre 3

Détails sur l'implémentation

3.1 Représentation du modèle

3.1.1 Parcours

Comme expliqué au chapitre précédent un parcours est constitué de plusieurs trajets. Donc la classe **Parcours** possédera une liste de trajets. Il regroupe également des informations qui lui sont propres tel qu'un nom et un identifiant. On remarquera également qu'un parcours possède un trajet de référence. Ce trajet est cruciale car il permettra la comparaison avec le trajet qui est effectué. Comme celui-ci pourra être modifié, on le représentera dans la classe **Parcours** par son identifiant. On ajoutera à cette classe tout un tas d'attributs servant à déterminer le meilleur temps, la vitesse moyenne, la vitesse maximale sur l'ensemble des trajets effectués.

3.1.2 Trajet

La classe **Trajet** représente le chemin qu'un utilisateur emprunte lorsqu'il effectue une course. On y retrouve une liste de points (**Location**) nécessaire à la visualisation du trajet ainsi qu'à la comparaison avec le trajet de référence. On gardera également des informations concernant la date à laquelle le trajet a été réalisé ainsi qu'un identifiant pour faciliter le stockage en base.

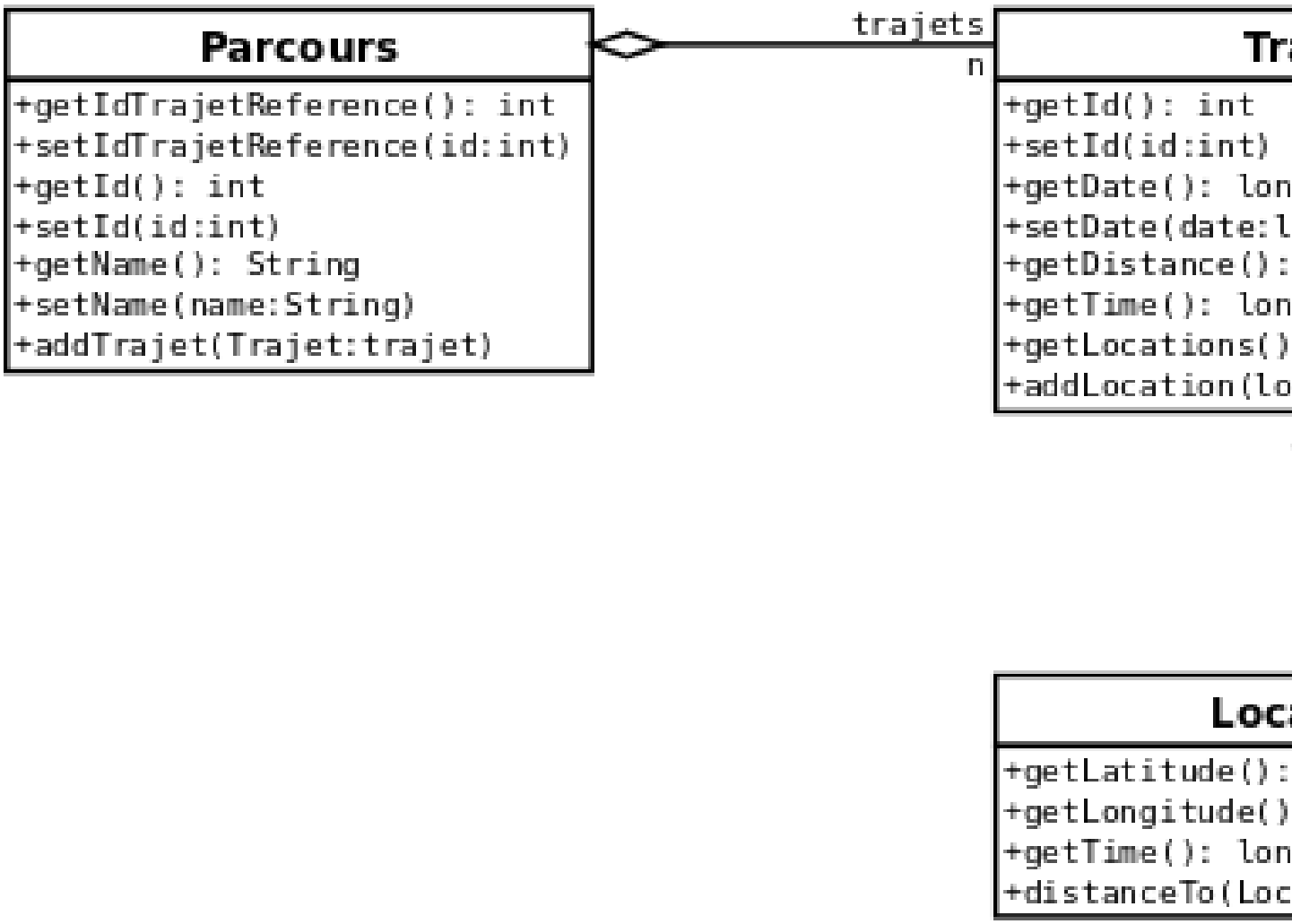

FIGURE 3.1 – Diagramme de classes

Chapitre 4

Les technologies utilisées

4.1 Le choix du langage

4.1.1 Javascript

Nous avons réalisé une étude sur les différentes technologies que nous pouvions employées pour réaliser ce projet. En effet, avec les technologies Javascript émergentes, l'utilisation de Cordova et Phonegap permettait de réaliser une application compatible avec l'ensemble des smartphones présent sur le marché actuel. Malheureusement, ces technologies sont plutôt orientées sur des technologies web. Le développement du code avec ces technologies nécessite une compilation afin d'être lisible par le smartphone sur lequel il est déployé.

4.1.2 Java

L'avantage avec le développement sous Android vient du fait que le langage utilisé n'est autre que du Java. De plus, la compilation du langage est optimisée en fonction du téléphone sur lequel il est déployé. De ce fait, l'exécution d'une application est plus performante si elle est conçue dans le langage natif au smartphone(Java pour Android, Objectif-C pour iOS). Notre application utilisant le GPS du smartphone, nous avons opté pour un développement en Java afin d'optimiser les connexions avec le satellite servant à géolocaliser le téléphone.

4.2 Le stockage de données

Pour un bon fonctionnement, l'application doit stockées des informations concernant les parcours et les trajets.

4.2.1 SQLite

Sous Android (et autres smartphones), le stockage des données se fait via une base de données en SQLite. SQLite est une bibliothèque proposant un moteur de base de données relationnelles utilisant le langage SQL. Elle est plus légère que ses homologues MySQL et PostgreSQL car elle n'intègre pas le schéma habituel Client-Serveur. En effet, l'ensemble des données est stocké dans un fichier indépendant d'Android et directement intégré à l'application.

4.2.2 Le format GPX

Le format GPX est un format XML conçu spécialement pour représenter un ensemble de points GPS ayant une latitude et une longitude à un instant t donné. L'utilisation de fichier GPX est très intéressant dans notre projet dans la mesure où le GPS d'Android nous fournit des points devant d'être stockés mais sans encombrer la base. En effet, SQLite n'est pas très adapté au stockage de grosses données (big data). De plus, cette séparation permet d'extraire facilement les données liées à la représentation du trajet effectué.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<gpx xmlns="http://www.topografix.com/GPX/1/1" creator="MapSource 6.9.2"
   version="1.1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.
   topografix.com/GPX/1/1/gpx.xsd">
  <metadata>
    <time>2011-04-13T15:58:51Z</time>
    <bounds maxlat="49.746009" maxlon="-1.372054" minlat="49.644963" minlon=
    "-1.925924"/>
  </metadata>
  <trk>
    <trkseg>
      <trkpt lat="49.645132" lon="-1.620045">
        <ele>-46.486572</ele>
        <time>2011-04-06T08:47:37Z</time>
      </trkpt>
     ...
      <trkpt lat="49.645703" lon="-1.619700">
        <ele>-3.227295</ele>
        <time>2011-04-06T08:49:37Z</time>
    </trkseg>
  </trk>
</gpx>
```

Code 4.1 – Exemple de fichier GPX

4.2.3 L'analyseur syntaxique

L'utilisation de fichier GPX implique forcement un système de lecture et d'écriture de fichier XML (parseur). En Java, il existe deux grandes familles de parseurs XML, les parseurs utilisant SAX et les parseurs utilisant le DOM.

Les parseurs DOM

Les parseurs DOM analysent la structure même d'un document.Ils stockent en mémoire l'ensemble des balises XML (le DOM) afin de pouvoir retrouver n'importe quel élément du document. l'inconvénient vient du fait qu'un document ne peut pas être plus gros que la mémoire vive.

Les parseurs SAX

Les parseurs SAX sont événementiels. En effet, ils analysent le document et déclenchent un événement lorsqu'une balise est construite ou détruite. Cette façon de lire un document est nettement moins coûteuse que la précédente dans la mesure où rien n'est stockée par le parseur. La lecture du document se fait au fur et à mesure. Cette méthode est beaucoup plus adaptée pour notre projet dans la mesure où les fichiers GPX analysés sont volumineux et les appareils n'ont pas beaucoup de mémoire.

Glossaire

SAX (Simple API for XML) : API basée sur un mode événementiel permettant de réagir à des événements (comme le début d'un élément, la fin d'un élément) et de renvoyer le résultat à l'application.

```
public void addLocation(Location l) {
  //Si il n'y a pas de points d'enregistres
  if (locations.size() == 0) {
    distance = 0;
    locations.add(l);
    temps = 0;
  } else {

  //On recupere la derniere position
  Location last = getLastPosition();

  //On augmente la distance total du trajet
  distance += last.distanceTo(l);
  locations.add(l);

  //On incremente egalement le temps
  if (locations.size() == 1) {
    temps = 0;
  }
  temps += ((l.getTime() / 1000) - (last.getTime() / 1000));
  }
}
```

Code 4.2 – L'ajout de point à un trajet