Sistema Dinâmico para Acompanhamento de Treino Esportivo

Alex Henrique Scapin[1], Reiner Franthesco Perozzo[1]

[1]Curso de Sistemas de Informação – Centro Universitário Franciscano - Caixa Postal 97010-032– Santa Maria – RS – Brasil

alex.scapin@redes.ufsm.com, reiner.perozzo@unifra.br

Abstract. *This paper presents a proposal for a software application to assist athletes or practitioners of physical activities during the course of training. This application is based in mobile devices with Android operating system and GPS technology in order to determine the current position by obtaining the geographic coordinates. With the use of this training system, it will be possible view information about the current performance of the athlete, taking into account data from previous performances, when there coincidence records at the point where the device is located, visualizing the current position of the device on a Google map fragment*

Resumo. *Este trabalho apresenta o desenvolvimento de uma aplicação para auxiliar os atletas ou praticantes de atividades físicas durante a realização dos treinamentos. Esta aplicação tem como plataforma base dispositivos móveis com sistema operacional Android e tecnologia GPS para determinar a posição atual através da obtenção das coordenadas geográficas. Com a utilização deste sistema de treinamento, será possível visualizar informações sobre o desempenho atual do atleta, levando em consideração dados de desempenhos anteriores, quando houver a coincidência de registros no ponto em que se encontra o dispositivo, visualizando ainda a posição atual do dispositivo em um fragmento de mapa do Google.*

1. Introdução

Devido ao aumento na utilização de dispositivos móveis como *smartphones*, *tablets*, *notebooks* e outros *gadgets* houve, também, um aumento no número e desenvolvimento de novos aplicativos para suprir as necessidades dos usuários. Nesse cenário crescente surgem aplicações para o sistema operacional Android que tem como objetivo auxiliar atletas no gerenciamento das informações de treinamentos esportivos, apresentando-as de forma clara e buscando, muitas vezes, incentivá-los a alcançar novos desafios.

Android é uma plataforma *open-source* de desenvolvimento para dispositivos móveis como *tablets* e *smartphones* que inclui um sistema operacional baseado no Linux, contando com uma interface gráfica capaz de oferecer diferentes recursos visuais, como *browser* para navegação na web, sistema de GPS, suporte multimídia, entre outros [Lecheta 2010]. Essa plataforma permite que usuários contribuam de forma significativa para o seu crescimento, tanto na evolução de sistemas e soluções já existentes, quanto na elaboração de novas ferramentas e aplicações, a fim de solucionar, facilitar ou contornar os mais variados tipos de problemas e necessidades, de forma colaborativa ou comercial.

Segundo um estudo realizado pelo Portal Brasil (2014) com o passar dos anos, vem aumentando também o número de pessoas que praticam algum tipo de exercício

físico durante o tempo livre, seja pelo fator saúde e a conscientização sobre a importância da prática de atividades físicas, ou simplesmente pelo auxílio à estética corporal.

Dessa forma, este trabalho visa o desenvolvimento de uma aplicação para auxiliar no controle e visualização das informações referentes às atividades físicas realizadas pelos usuários durante a realização das mesmas (em tempo de execução), buscando unir a utilização dos dispositivos móveis no auxílio e na monitoração à prática de atividades físicas.

1.1 Objetivo Geral

Desenvolver um aplicativo voltado à dispositivos móveis para auxiliar os usuários/atletas durante a atividade física, proporcionando uma visão sobre dados referentes ao desempenho atual e histórico através de indicadores de velocidade atual posição geográfica, de forma que o usuário seja capaz de controlar seu esforço para obter melhores resultados.

1.2 Objetivos Específicos

O trabalho possui os seguintes objetivos específicos:

- Investigar o funcionamento do sistema GPS em dispositivos móveis, verificando os pontos necessários para satisfazer as necessidades da aplicação;
- Planejar o desenvolvimento da aplicação através de etapas seguindo uma metodologia específica para alcançar o objetivo;
- Desenvolver as funcionalidades da aplicação prevendo o modo de operação e comportamento do produto final;
- Implementar as interfaces para Android, buscando atender os requisitos e as funcionalidades previstas, validando e realizando testes do funcionamento da aplicação desenvolvida.

2. Referencial Teórico

Nesta seção é apresentada a revisão bibliográfica sobre o tema escolhido para este trabalho, conceituando e visando buscar características importantes de tecnologias e trabalhos relacionados que servem como base para a aplicação proposta. Esta Seção traz, portanto, uma breve descrição da plataforma Android, do *Global Positioning System* (GPS) e do ambiente de desenvolvimento Android Studio.

2.1. Plataforma Android

O Android foi inicialmente desenvolvido pela Android Inc. sendo depois, adquirida pelo Google e mantida pela Open Handset Alliance (OHA), grupo constituído pelo Google e outras empresas do mercado de telefonia como a Samsung, Motorola, LG, Intel, Toshiba, HTC, Sony Ericsson, ASUS, entre outros [Petroni *et al.* 2014].

Segundo Lecheta (2010), Android é uma plataforma *open-source* de desenvolvimento para dispositivos móveis como *tablets* e *smartphones* que inclui um sistema operacional baseado no Linux que conta com recursos para interface gráfica, *browser* para navegação na web, sistema de GPS, suporte a multimídia, entre outros.

O sistema Android conta com diversas versões já desenvolvidas, estando em constante evolução. Isto se deve diretamente a sua flexibilidade e poder de inovação que

vem conquistando a aceitação dos usuários. Essas versões têm nome de doces e seguem a sequência das letras do alfabeto, sendo a mais recente o Android 6.0, denominada *Marshmallow* [Android 2015].

2.2. GPS

Global Positioning System ou, simplesmente, GPS é um sistema composto por uma constelação de satélites na órbita terrestre, projetada de maneira que pelo menos quatro deles fiquem visíveis a partir de qualquer ponto do planeta [Morimoto 2009].

Os satélites que compõe o sistema de GPS emitem sinais para dispositivos receptores, e através deles, pode-se descobrir a localização do receptor na superfície terrestre. A partir das informações recebidas, pode-se calcular também a velocidade, sentido de uma rota, altitude em relação ao nível do mar e estimativas sobre tempo de percurso [Garret 2014].

2.3. Internet Móvel 3G/4G

Ao se abordar o tema de telefonia móvel de terceira e de quarta geração (3G e 4G), que serão importantes para a elaboração deste estudo, é necessário citar a existência de tecnologias precursoras sendo, respectivamente, as tecnologias 1G, 2G, 2,5G e 2,75G, em que as duas últimas são padrões de transição para a 3G.

Diante de todas as evoluções nessas redes, talvez a mais significativa tenha sido a 3G, que surgiu para permitir acesso à internet de alta velocidade e vídeo-telefonia utilizando redes de telefonia celular, fornecendo transmissão de dados além de chamadas telefônicas [Neto *et al.* 2013].

A ANATEL (Agência Nacional de Telecomunicações) definiu um prazo de implantação da tecnologia 4G no Brasil até o final de 2014, a qual utiliza frequência de 2,5 GHz [Olhar Digital 2013]. Analisando a velocidade teórica com relação à 3G+ (padrão que alcança até 21 Mbps), a tecnologia atual oferece velocidades entre 50 Mbps e 1 Gbps [Hamann 2013].

2.4. Ambiente Android Studio

Android Studio é um ambiente de desenvolvimento integrado, do inglês *Integrated Development Environment* (IDE) criado pelo Google e apresentada no Google I/O 2013. Essa ferramenta surgiu para auxiliar os desenvolvedores de aplicativos para Android, levando em conta as dificuldades de realizar corretamente algumas tarefas que parecem simples, mas que são muito importantes para o funcionamento de novos programas [Barros 2013].

Tendo como base para seu desenvolvimento o *IntelliJ IDEA* (IDE em Java para desenvolvimento de *software*), o Android Studio usa o *Gradle* para compilar aplicações. Permite armazenar os projetos em sistemas de controle de versão como o GIT, Subversion e Mercurial possuindo, ainda, um gerenciador de dispositivo virtual do Android para configurar e administrar os emuladores. As características mais marcantes dessa ferramenta são: visualização de recursos como *strings*, ícones e cores, análise de código baseado nas anotações da API (*Application Programming Interface*) do Android, pré-visualização do *layout* aplicando as mudanças simultaneamente e construtor de *layout*, que permite arrastar e soltar os componentes [Avram 2013].

2.5. Trabalhos Correlatos

Nesta seção são apresentados trabalhos relacionados que servem como base para organização, motivação e definição da proposta deste trabalho. Assim, foram selecionados trabalhos e aplicativos que se aproximam da abordagem proposta, permitindo uma visão sobre tecnologias existentes e disponíveis no mercado.

2.5.1. Aplicação móvel e plataforma web para suporte à estimação do gasto energético em atividade física

No trabalho realizado por Pires (2012) foi proposta a criação de uma aplicação para auxiliar e incentivar a prática de exercícios físicos utilizando dispositivos que muitas vezes contribuem para o sedentarismo. Nesse estudo, por exemplo, foram utilizados *smartphones* com sistema operacional Android. A solução utiliza dados coletados através de sensores e do GPS para estimar o gasto energético do usuário com base na distância percorrida e no cálculo do tempo de voo dos saltos realizados durante a realização da atividade física.

Esse estudo aborda, ainda, a elaboração de uma aplicação web, com sistema de cadastro de usuários e resultados do treinamento, permitindo a gestão dos dados do utilizador, do treinamento e dos objetivos a serem atingidos.

2.5.2. Strava

Strava é um aplicativo/site que permite ao usuário coletar informações como velocidade, altimetria do terreno, tempo percorrido, entre outros. Através da utilização de um *smartphone* ou dispositivos com sistema de GPS, esse aplicativo permite que, ao final de uma atividade física, seja realizado o *upload* dos dados registrados durante o exercício para o site, o qual opera como uma rede social formada por ciclistas e corredores. Além disso, há uma organização dos dados em *ranking* permitindo a realização de desafios esportivos entre usuários (ou a si mesmo) a fim de incentivar a prática de atividades físicas [Strava 2015]. A Figura 1 ilustra algumas das funcionalidades disponibilizadas pelo Strava quando utilizado em *smartphones.*

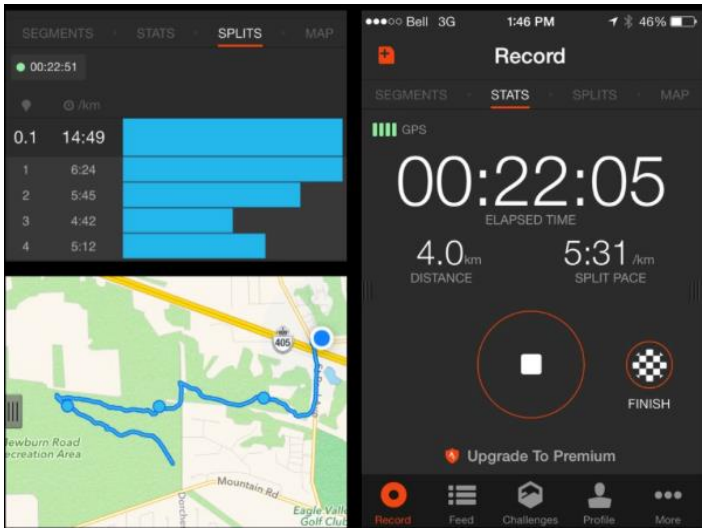

Figura 1 - Ilustração de algumas funcionalidades do Strava.

2.5.3. Runtastic

Runtastic é um aplicativo para dispositivos móveis que possuem GPS. Semelhante ao Strava, ele permite a seleção de um tipo de atividade física dentre as várias

disponibilizadas em sua interface gráfica, com o objetivo de coletar informações e estatísticas durante o percurso ou exercício. No *website* é possível, também, salvar os treinos e resultados obtidos durante os exercícios, comparar os resultados com os amigos e analisar as estatísticas de forma mais detalhada [Runtastic 2015]. A Figura 2 ilustra algumas das funcionalidades disponibilizadas pelo aplicativo Runtastic.

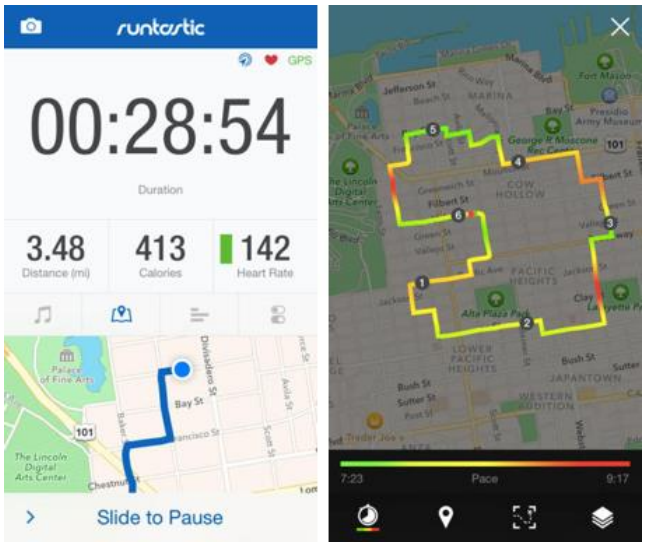

Figura 2 - Exemplo de funcionalidades presentes no Runtastic.

2.5.4. SIAF: Um Sistema de Informação de Atividade Física

Nesse estudo, Portocarrero (2010) propõe o desenvolvimento de um sistema de coleta de informações e dados fisiológicos de pessoas que realizam exercícios físicos promovidos pelas unidades de saúde da cidade de São Carlos - SP. Esse sistema utiliza basicamente uma rede de sensores para monitorar estatísticas do corpo humano, redes sem fio e dispositivos móveis para coleta e envio *online* dos dados para o sistema de gerenciamento das atividades e controle de sessões que foram desenvolvidos em linguagem Java.

2.5.5. Conclusão dos trabalhos correlatos

Os trabalhos correlatos foram pertinentes para a formulação da proposta inicial, possibilitando a criação de uma base sólida de conhecimento para formular e pontuar as funcionalidades que compõem a aplicação desenvolvida. Tendo em vista que, assim como os trabalhos correlatos, a aplicação proposta busca auxiliar atletas e praticantes de atividades físicas através da obtenção e exibição de dados dos treinamentos, optando apenas por exibir essas informações de forma diferenciada, em tempo de execução, possibilitando ao usuário tomar decisões sobre o emprego do esforço físico conforme as estatísticas apresentadas pela aplicação.

3. Proposta

Este trabalho teve como proposta o desenvolvimento de um aplicativo para dispositivos móveis capaz de auxiliar o atleta ou praticante de alguma atividade física durante a prática dos exercícios. A aplicação tem como principal característica a captura de dados sobre a performance do usuário e exibição de informações importantes, em tempo de execução, sobre o seu desempenho em determinada localização do globo terrestre, levando em consideração as suas coordenadas geográficas.

Para auxiliar e informar o atleta, o aplicativo utiliza uma gama de informações adquiridas através do sistema de GPS do aparelho como latitude, longitude e velocidade. Ao definir uma nova rota, ou selecionar um percurso já existente, o sistema de posicionamento global deverá capturar informações referentes à localização geográfica, definindo no mapa da aplicação a localização do dispositivo.

Com o funcionamento correto do sistema de GPS, assim que o usuário se deslocar, a aplicação deverá alterar a identificação da posição atual conforme o deslocamento, mostrar um cronômetro exibindo o tempo de duração da atividade física, assim como exibir a velocidade atual, a média, a mínima e a velocidade máxima alcançada naquela coordenada geográfica, como ilustra a Figura 3.

Figura 3 – Visão geral da aplicação proposta.

Com a utilização desse aplicativo, é esperado que o usuário/atleta consiga visualizar, durante a prática do exercício, um comparativo entre desempenho atual e dados históricos daquele mesmo ponto do percurso. Com base nesses dados, o atleta poderá dosar o emprego de esforço físico e gasto energético durante a atividade a fim de alcançar, ao final do treinamento, um melhor resultado.

Para a elaboração deste trabalho optou-se por utilizar a metodologia *Feature Driven Development* (FDD), por oferecer suporte às necessidades e aos requisitos da proposta inicial, permitindo a inserção e ajuste de funcionalidades ao longo do desenvolvimento do tempo.

Segundo Palmer e Felsing (2012) a metodologia FDD é constituída por cinco processos, sendo eles: desenvolver modelo abrangente, criar lista de funcionalidades, planejar por funcionalidade, arquitetar por funcionalidade e construir por funcionalidade. Os três processos iniciais fazem parte da fase de Concepção e Planejamento, podendo sofrer mudanças e ajustes no decorrer do desenvolvimento do projeto. Arquitetar por funcionalidade e construir por funcionalidade fazem parte da fase de Construção, sendo executadas de forma iterativa. [Ratamal 2008].

3.1 Projeto

Considerando a metodologia utilizada no trabalho, esta Subseção apresenta o projeto da proposta, contemplando as fases de desenvolver um modelo abrangente, a construção da

lista de funcionalidades e o planejamento por funcionalidades. Inicialmente é realizado um estudo sobre o escopo do sistema e, após, são realizados novos estudos sobre o domínio de negócio, de forma mais detalhada.

Nessa etapa é gerado um modelo de objetos através de diagramas de domínio, classes ou sequência (se houver) que servirão como parâmetro para uma determinada área de domínio, podendo sofrer alterações durante as diversas iterações que o projeto vai ser executado, ficando atualizado até a conclusão do projeto [Ratamal 2008].

Para a elaboração da aplicação, com a intenção de auxiliar no desenvolvimento da mesma, é apresentado um diagrama de classes ilustrado na Figura 4, em que são destacados os atributos, os métodos, os parâmetros e a interação entre as classes que compõem o sistema.

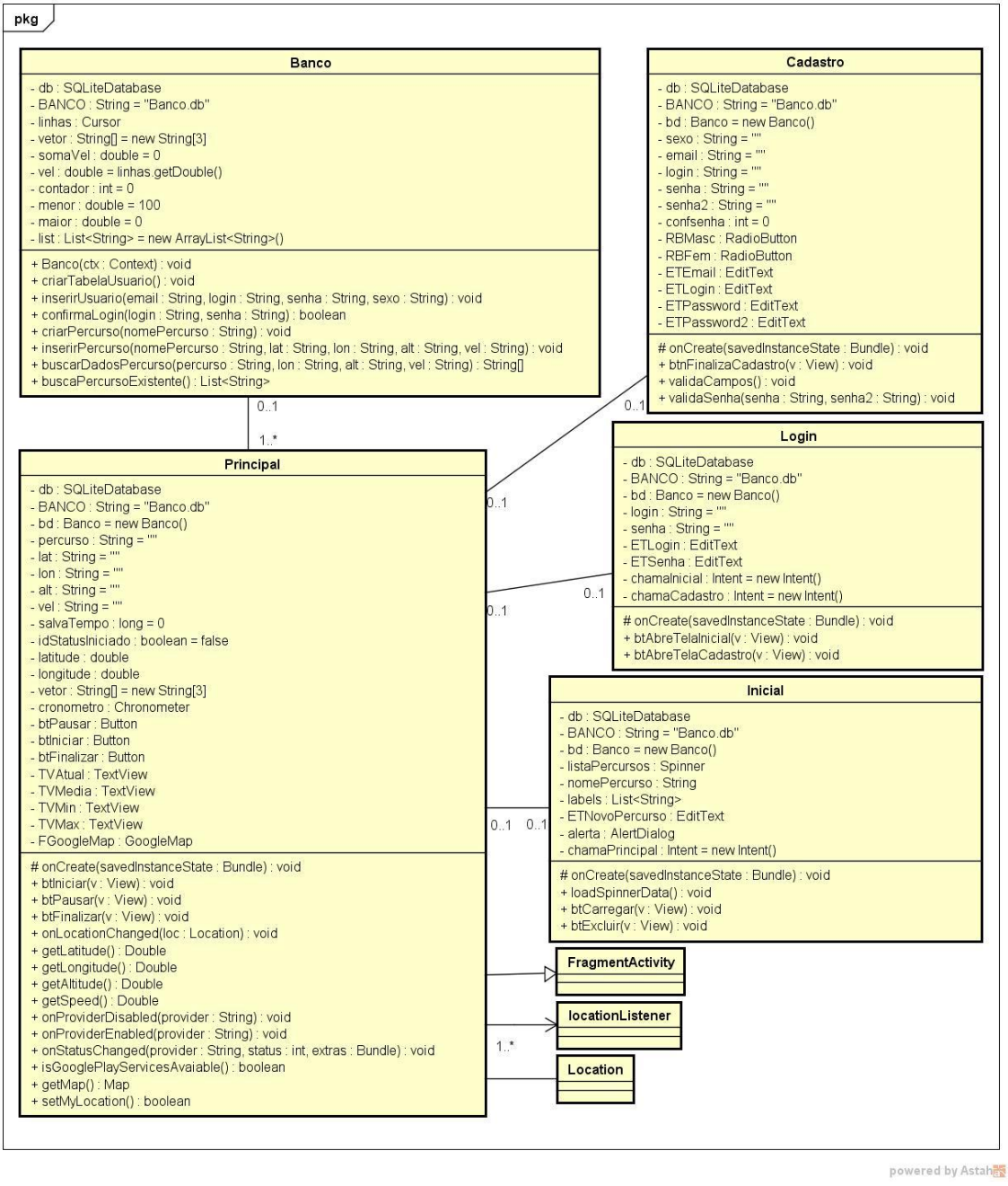

Figura 4 - Diagrama de classes da aplicação.

Conforme ilustrado na Figura 4, a Classe Principal do projeto herda características da *FragmentActivity*, que é utilizada para criar telas com um nível maior de compatibilidade entre dispositivos e versões do Android. É, também, a classe Principal que implementa o serviço *LocationListener* e sempre que houver mudança de localização baseada nas coordenadas do GPS realiza chamada dos métodos da classe Banco para realizar operações de inclusão, alteração e exclusão de registros, assim como verificar se a posição atual de um percurso coincide com dados já existentes na base para a realização do cálculo de média e atribuição da maior e menor velocidade da posição correspondente. A relação e a descrição dos métodos e dos atributos que constituem o diagrama de Classes, ilustrado na Figura 5, estão presentes nos Apêndices.

Após a criação do modelo de objetos, pode-se iniciar a criação da lista de funcionalidades, as quais estarão presentes no sistema atendendo aos requisitos. Na metodologia FDD, as funcionalidades são funções granulares que expressam algum valor agregado à pessoa interessada no sistema ou *software* [De Luca 2002]. As funcionalidades previstas para o sistema deste trabalho são:

- Gerenciar informações de cadastro e *login* do usuário;
- Capturar dados de deslocamento e posição do dispositivo;
- Gerenciar dados e informações de treinamentos;
- Mostrar comparação de desempenho em tempo de execução;
- Verificar conectividade ao sistema de GPS;
- Gerenciar informações de treinamentos da base de dados;

Nessa etapa, o sistema deverá possibilitar que o usuário insira dados relevantes ao seu cadastro, escolha entre definição de um novo percurso ou seleção de um percurso já realizado anteriormente. O atleta deverá indicar sempre o nome do percurso que irá fazer na realização da atividade, pois é através dele que o sistema verificará a existência de dados históricos na base de dados para realização de comparações e exibição das informações ao usuário.

3.2 Implementação

Esta Subseção engloba os dois últimos processos da metodologia FDD, contemplando o detalhamento e a construção por funcionalidades. Ao final do processo de detalhamento por funcionalidades são criados diagramas de sequência e o modelo de objetos gerado no primeiro processo. Há uma revisão das classes, assim como dos métodos inseridos detalhando as funções [De Luca 2002].

Cada percurso criado na aplicação é uma tabela na base de dados, sendo que os dados de localização e velocidade pertencentes a cada ponto geográfico são registrados na tabela correspondente ao percurso definido. A aplicação implementa na classe Principal o serviço *LocationListener*, responsável por verificar a alteração de coordenadas referentes à localização no mapa geográfico. Sempre que existir esta mudança de localização, a aplicação coleta os dados fornecidos pelo sistema de GPS e faz a inserção e a comparação de latitude e de longitude na base de dados.

Quando houver a coincidência de dados de localização atual com registros existentes, será realizado o cálculo da média entre as velocidades daquela coordenada, memorização da maior e menor velocidade naquele ponto e as informações são exibidas na interface da aplicação para conhecimento do atleta.

Contemplando a construção por funcionalidades, destaca-se nessa etapa alguns módulos (interfaces gráficas e trechos de código) que são relevantes quanto ao desenvolvimento da aplicação. A Figura 5 ilustra duas interfaces gráficas que foram construídas para a manipulação do aplicativo proposto.

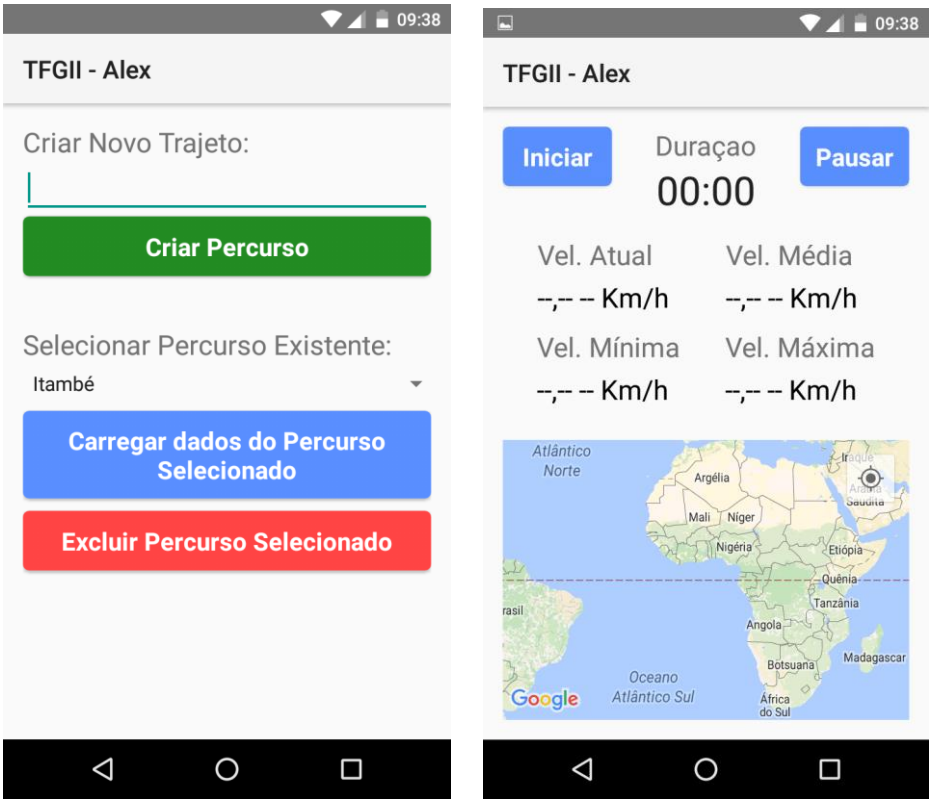

Figura 5 – Interfaces do aplicativo.

A Figura 5 é composta pela atividade Inicial, responsável pela definição de um novo percurso quando pretende-se percorrer um trajeto pela primeira vez, a escolha por um percurso já existente quando o percurso pretendido já existir, ou a exclusão de um percurso já existente. A atividade Principal exibe as informações do treinamento em tempo de execução do sistema, permitindo que a atividade física seja iniciada, paralisada, continuada ou finalizada.

Na atividade Principal existe um elemento de *layout* do tipo *Fragment* que é responsável por exibir o mapa com a localização atual, obtido através do serviço fornecido pelo *GooglePlayServices*. A atividade principal é responsável também por definir o melhor provedor de dados referentes à localização do dispositivo, representado pela *String bestProvider*, tornando as informações obtidas através do sistema de GPS mais confiáveis e precisas, assim como verificar o *status* de funcionamento do GPS informando o usuário sobre o seu funcionamento.

Cabe a atividade Principal solicitar também a atualização da localização atual conforme ilustrado no trecho de código apresentado na Figura 6.

```
FGoogleMap = supportMapFragment.getMap();
FGoogleMap.setMyLocationEnabled(true);
LocationManager locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
Criteria criteria = new Criteria();
String bestProvider = locationManager.getBestProvider(criteria, true);
Location loc = locationManager.getLastKnownLocation(bestProvider);
locationManager.requestLocationUpdates(bestProvider, 0, 0, this);
locationManager.addGpsStatusListener(new GpsStatus.Listener() {
    @Override
    public void onGpsStatusChanged(int event) {
        switch (event) {
            case GpsStatus.GPS_EVENT_STARTED:
                Toast.makeText(getApplicationContext(), "GPS Ativado!",
                    Toast.LENGTH_LONG).show();
            break;
            case GpsStatus.GPS_EVENT_STOPPED:
                Toast.makeText(getApplicationContext(), "GPS Desativado," +
                    " ATIVE para iniciar!", Toast.LENGTH_LONG).show();
            break;
        }
    }
});
```

Figura 6 – Exibição do mapa e detecção de mudança na localização.

A Figura 7 ilustra o método *OnLocationChanged* que é executado sempre que há mudança na localização do dispositivo através do método *requestLocationUpdates* presente na Figura 6.

```
@Override
public void onLocationChanged(Location loc) {
    String vetor[]=new String[3];
    double latitude = loc.getLatitude();
    double longitude = loc.getLongitude();
    alt = String.format("%.4f", loc.getAltitude());
    vel = String.format("%.2f", ((loc.getSpeed() * 3600) / 1000));
    vel = vel.toString().replace(",", ".");
    lon = String.format("%.4f",longitude);
    lat = String.format("%.4f",latitude);

    LatLng latLng = new LatLng(latitude, longitude);
    FGoogleMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
    FGoogleMap.animateCamera(CameraUpdateFactory.zoomTo(15));

    if(idStatusIniciado ){
        bd.criarPercurso(nomePercurso);
        bd.inserirPercurso(nomePercurso, lat, lon, alt, vel);
        vetor = bd.buscaDadosPercurso(nomePercurso, lat, lon);
        TVAtual.setText(vel + " Km/h");
        if (vetor[0] == null && vetor[1] == null && vetor[2] == null) {
            //não encontrou dados da posição atual
        }else {
            TVMedia.setText(vetor[0] + " Km/h");
            TVMin.setText(vetor[1] + " Km/h");
            TVMax.setText(vetor[2] + " Km/h");
        }
    }
}
```

Figura 7 – Método *OnLocationChanged*.

A Figura 7 exibe a obtenção dos valores da localização atual (latitude, longitude, altitude e velocidade no formato Km/h), a definição da localização atual no mapa exibido na atividade principal e a manipulação de informações na base de dados. Nota-se, ainda, que ocorre a atribuição da velocidade atual ao elemento correspondente na interface gráfica (TVAtual) e a atribuição dos valores de velocidade média (TVMedia), mínima (TVMin) e máxima (TVMax). Todos os valores são retornados no formato de vetor, com três posições, obtidos pelo método buscaDadosPercurso, da Classe Banco, ilustrado na Figura 8.

```
public String[] buscaDadosPercurso(String nomePercurso, String lat, String lon) {
    String vetor[] = new String[3],media = "";
    double somaVel = 0, menor = 1000, maior = 0, vel;
    int contador = 0;

    db = ctx.openOrCreateDatabase(BANCO, Context.MODE_PRIVATE, null);
    Cursor linhas = db.rawQuery("SELECT velocidade FROM " + nomePercurso
            + " where latitude='" + lat + "' and longitude='" + lon + "'", null);

    if (linhas.moveToFirst()) { //retorna false se não há registros
        do {
            vel = linhas.getDouble(0);
            if (vel > maior) {
                maior = vel;
            }
            if (vel < menor) {
                menor = vel;
            }
            somaVel = vel + somaVel; //somador de velocidades coincidentes
            contador++; //somador de quantas vezes coincidem
        }
        while (linhas.moveToNext());//laço até a última linha da tabela
        //Calcula media limitando em 2 casas após a virgula
        media = String.format("%.2f",(somaVel / contador));
        media = media.toString().replace(",",".");  //substitui a , por .
        vetor[0] = media;
        vetor[1] = String.valueOf(menor);
        vetor[2] = String.valueOf(maior);
    }
    db.close();
    return vetor;
}
```

Figura 8 – Método *BuscaDadosPercurso da Classe Banco*.

Na Figura 8 há uma consulta utilizando os dados recebidos por parâmetro, que correspondem ao percurso que está sendo realizado e as coordenadas geográficas atuais. Se a execução da *query* encontrar registros na base, serão calculados a média naquele determinado ponto e a atribuição do menor e do maior valor de velocidade, os quais serão retornados em formato de vetor de *Strings* e utilizados na interface gráfica, como citado anteriormente na descrição da Figura 7.

A Figura 9 ilustra o diagrama de Entidade Relacionamento para a base de dados utilizada pelo aplicativo.

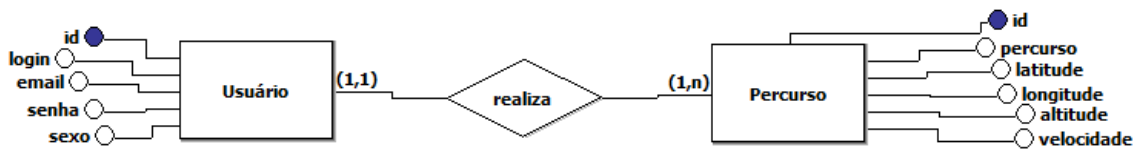

Figura 9 – Diagrama Entidade Relacionamento.

Na Figura 9, há o relacionamento entre as entidades Usuário e Percurso, cada uma com seus atributos e identificadores, em que um usuário pode realizar um, ou vários percursos. Para o desenvolvimento da aplicação foi utilizado o banco de dados SQLite, uma vez que o Android possui suporte nativo e permite a manipulação dos dados com comandos DDL (*Data Definition Language*) ou com comandos DML *(Data Manipulation Language*) do SQL (*Structured Query Language*) padrão.

4. Cenário de Testes e Validação

Considerando a proposta do trabalho, a qual consiste no desenvolvimento de um aplicativo na plataforma Android com a intenção de auxiliar atletas e praticantes de atividades físicas durante os exercícios, foi desenvolvida uma aplicação para dispositivos móveis, cujos testes foram realizados em um *smartphone* (modelo Moto G 3ª Geração, com Android 6.0).

Para fins de validação, utilizou-se como modalidade a prática de caminhada em ambiente urbano. O cenário escolhido para a prática da atividade foi o Parque Itaimbé, na área central de Santa Maria - RS, por ser um local de acesso público e por questões de segurança em função de não haver tráfego de veículos automotores no local. A Figura 10 ilustra as telas da aplicação em execução e o local do teste exibido no mapa.

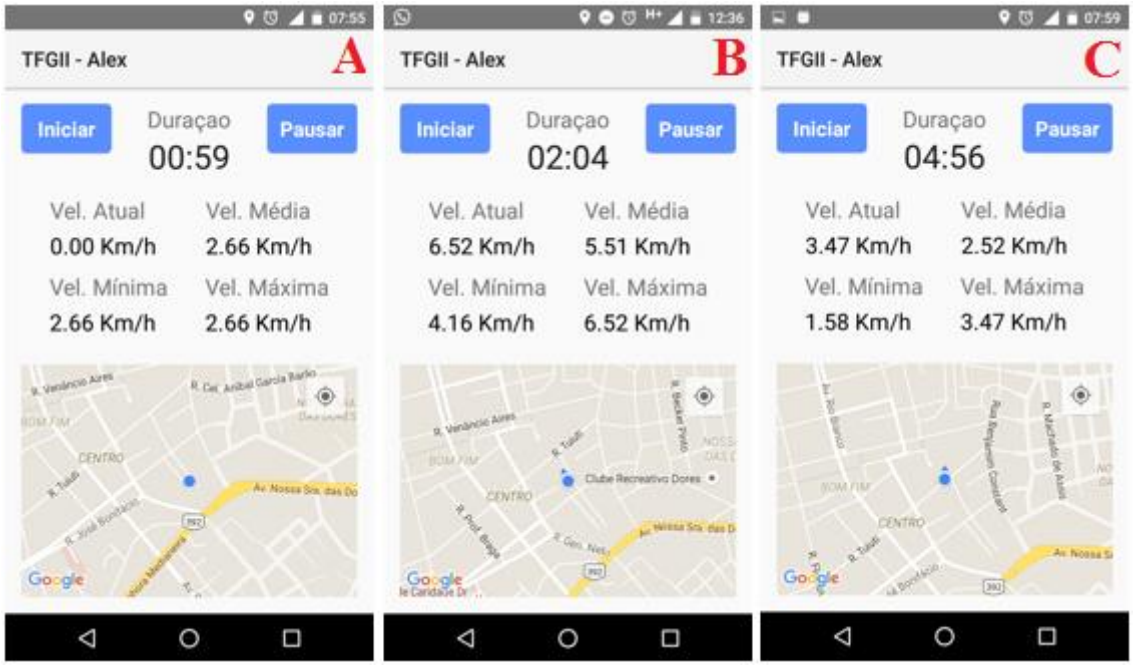

Figura 10 – Cenário de teste e validação do *software*.

Refinando a descrição da Figura 10 e detalhando os testes realizados, é ilustrado o deslocamento na realização da atividade física que serviu como teste do aplicativo. Na Figura 10.A é possível perceber que o dispositivo não está se movendo, já na figura 10.B, pode-se perceber que está em deslocamento, com velocidade de 6,52 Km/h no momento da captura da tela que, por sua vez, também é a velocidade máxima, com a média de 5,51 Km/h e mínima de 4.16 Km/h naquele ponto do percurso. Na Figura 10.C é possível notar que, mesmo com velocidade mais reduzida, o dispositivo continua em deslocamento, realizando as comparações e atribuindo os valores aos elementos da interface gráfica, conforme comportamento projetado e esperado pela aplicação quando estivesse em execução.

5. Resultados e discussões

Todo o projeto esteve focado no planejamento e desenvolvimento de um sistema que fosse capaz de auxiliar o usuário durante a prática de atividades físicas através de indicadores de desempenho, apresentados na atividade principal do aplicativo.

Durante o desenvolvimento do trabalho surgiram desafios que implicaram em alterações no planejamento inicial do projeto, principalmente no que tange à utilização de um servidor web para armazenamento dos dados de treinamentos como proposto inicialmente. Nesse ponto, vale destacar as dificuldades na troca de dados e informações entre a aplicação e o servidor web, assim como a preocupação com a utilização da conexão à internet, visto que seria necessário a utilização de grande quantia de dados móveis (internet 3G/4G) durante toda a prática da atividade física.

Em contraponto às dificuldades, optou-se pela utilização de um banco de dados no próprio aparelho, oferendo a busca e o armazenamento local dos dados obtidos sobre o trajeto, evitando, assim, um tráfego de dados desnecessário e mantendo o objetivo da proposta. Essa mudança tem como principais benefícios, a não necessidade da utilização de internet móvel e, principalmente, a não existência de requisições para envio e recebimento de informações entre o dispositivo móvel e o servidor web, o que influencia diretamente na autonomia da bateria do dispositivo, tema que ainda é um gargalo para sistemas de computação móvel.

6. Conclusão

Este trabalho apresentou a proposta de um sistema dinâmico para acompanhamento de treinos esportivos utilizando dispositivos móveis com sistema operacional Android e recursos do sistema de GPS. Dessa forma, é possível destacar que o aplicativo contempla a apresentação dos dados referentes às atividades físicas ainda durante a prática do exercício, permitindo uma visão sobre o desempenho atual (em tempo de execução), auxiliando o atleta na tomada de decisão e colaborando para a melhoria de desempenho ao final do treinamento.

A metodologia ágil FDD escolhida para a elaboração deste trabalho, contemplou o desenvolvimento por funcionalidades, permitindo a realização de testes frequentes, repetitivos, a implementação e o acompanhamento do projeto. Além disso, a possibilidade de projetar novamente alguma funcionalidade não prevista inicialmente flexibiliza a sua utilização.

Para trabalhos futuros, poderão ser desenvolvidas funcionalidades para comparação dos dados e estatísticas dos treinamentos de forma integral, demarcação do percurso ou circuito no fragmento que contém o mapa do trajeto. Adicionalmente, poderia ser construída um sistema web, semelhante aos citados nos trabalhos correlatos, o que permitiria a visualização e acompanhamento das estatísticas não só ao ponto em que se encontra o dispositivo, mas também, o percurso de forma completa. Pode-se ainda elaborar um sistema de pontuação, que atribuirá pontos para metas alcançadas ao longo de cada treinamento, assim como a sequência da prática de atividades, o que poderá influenciar o atleta na busca por atingir melhores resultados e manter a regularidade nos treinamentos.

7. Referências

Android (2015). “História do Android”. Acesso em: 03 de Setembro de 2015. Disponível em: http://www.android.com/intl/pt-BR_br/history/.

Avram, A. ;Traduzido por Sakurai, R. (2013). “Android Studio: Uma nova IDE do Google baseada no IntelliJ IDEA”. Acesso em: 14 de Agosto de 2015. Disponível em: http://www.infoq.com/br/news/2013/05/android-studio.

Barros, T. (2013). “Android Studio é o programa do Google para desenvolver apps para Android”. Acesso em: 14 de Agosto de 2015. Disponível em: http://www.techtudo.com.br/tudo-sobre/android-studio.html.

De Luca, Jeff. Feature Driven Development Processes Download. “*Feature Driven Development*”, 2002. Acesso em: 15 de Setembro de 2015.Disponível em: http://www.featuredrivendevelopment.com/files/fddprocessesA4.pdf.

Garret, F. (2014). “Saiba o que é GPS e como funciona”. Acesso em 08 de Agosto de 2015. Disponível em: http://www.techtudo.com.br/artigos/noticia/2011/12/como-funciona-o-gps.html.

Hamann, R. (2013). “Como funcionam as redes 4G?”. Acesso em: 06 de Agosto de 2015. Disponível em: http://www.tecmundo.com.br/4g/40528-como-funcionam-as-redes-4g-ilustracao-.htm.

Lecheta, R. R. (2010) “Google Android”, In: Aprenda a criar aplicações móveis com o Android SDK, 2. ed. ver. ampl. São Paulo: Novatec, 2010.

Morimoto, C. E. (2009). “Uma introdução ao GPS”. Acesso em: 08 de Agosto de 2015. Disponível em: http://www.hardware.com.br/artigos/gps/.

Neto, C. C.; et al. (2013). “Redes 3G/4G como suporte na internet”. Acesso em: 06 de Agosto de 2015. Disponível em: http://revista.universo.edu.br/index.php?journal=1reta2&page=article&op=view&path[]=1201&path[]=905.

Olhar Digital (2013). “Conheça a diferença entre 1G, 2G, 3G e 4G”. Acesso em: 06 de Agosto de 2015. Disponível em: http://olhardigital.uol.com.br/noticia/conheca-as-diferencas-entre-1g,-2g,-3g-e-4g/34225.

Palmer, Stephen R.; Felsin, John. M. (2002) “*A Practical Guide to Feature-Driven Development.*” Prentice Hall.

Petroni, B. C.; et al. (2014). “Avaliação da usabilidade da IDE Android Studio”. Revista RETC, 14ª Edição. Abril de 2014.

Pires, I. M. S. (2012). “Aplicação móvel e plataforma web para suporte à estimação de gasto energético em atividade física”, Universidade da Beira Interior, Faculdade de Engenharia, Covilhã - Portugal.

Portal Brasil (2014). “Pesquisa revela aumento na prática de atividades físicas”. Acesso em: 23 de Outubro de 2015. Disponível em: http://www.brasil.gov.br/saude/2014/05/pesquisa-revela-aumento-na-pratica-de-atividades-fisicas.

Portocarrero, J. M. T. (2010) “SIAF: um sistema de informação, a ser integrado num ambiente de computação ubíqua, para gerenciamento de atividade física”, Universidade Federal de São Carlos, Centro de ciências exatas e de Tecnologia.

Ratamal, A. M. (2008). “Feature-Driven Development”. Acesso em: 19 de Setembro de 2015. Disponível em: http://www.heptagon.com.br/files/FDD-Processos.pdf.

Runtastic (2015). “Running, Cycling and Fitness GPS Tracker”. Acesso em: 17 de Agosto de 2015. Disponível em: http://www.runtastic.com.

Salgado, J. V. V. (2006). “Corrida de rua: Análise do crescimento do número de provas e de praticantes”. Acesso em: 12 de Outubro de 2015. Disponível em: http://fefnet178.fef.unicamp.br/ojs/index.php/fef/article/view/57/39.

Strava (2015). “Controlador de corrida e ciclismo GPS”. Acesso em: 17 de Agosto de 2015. Disponível em: http://www.strava.com.

8. Apêndices

Tabela 1. Descrição de atributos e métodos da classe Banco.

Classe Banco	
Atributos	
Nome	**Descrição**
db	Atributo responsável por indicar o tipo de base de dados.
BANCO	String que contém o nome do banco de dados.
linhas	Indicador de linhas com retorno de registros de uma consulta.
vetor	Vetor de três posições que receberá e retornará a velocidade média, mínima e máxima.
somaVel	Atributo que receberá a soma das velocidades coincidentes em um determinado ponto.
vel	Recebe a velocidade de uma localização armazenada na base de dados.
contador	Acumulador de vezes que uma localização já foi armazenada na base de dados.
menor	Recebe o menor valor de velocidade em determinada posição.
maior	Recebe o maior valor de velocidade em determinada posição.
list	Recebe o nome de percursos existentes na base de dados em formato de lista.
Métodos	
criarTabelaUsuario	Responsável pela criação da tabela de usuário.
inserirUsuario	Método responsável por inserir um usuário na base de dados.
confirmaLogin	Utilizado para verificar a existência de usuário na base de dados.
criarPercurso	Cria uma tabela com nome do percurso caso ainda não exista.
inserirPercurso	Insere dados de localização do treinamento atual na tabela do percurso em execução.
buscarDadosPercurso	Realiza a busca por coincidências de localização atual com existentes na base referentes a atividades anteriores. Retorna um vetor com a velocidade média, mínima e máxima da localização.
buscaPercursoExistente	Retorna uma lista com o nome de percursos existentes.

Tabela 2. Descrição de atributos e métodos da classe Cadastro.

Classe Cadastro	
Atributos	
Nome	**Descrição**
db	Atributo responsável por indicar o tipo de base de dados.
BANCO	*String* que contém o nome do banco de dados.
bd	Variável de instância da Classe Banco.
sexo	Recebe a descrição do sexo selecionado na interface com base nos valores de RBMasc e RBFem.
email	*String* que contém o email do usuário.
login	*String* que recebe o *login* definido pelo usuário para acesso à aplicação.
senha	Senha definida pelo usuário para acesso à aplicação.
senha2	Confirmação de senha informada pelo usuário.
confsenha	Indica se as senhas informadas são iguais ou não.
RBMasc	Definido como selecionado se usuário for do sexo Masculino.
RBFem	Definido como selecionado se usuário for do sexo Feminino.
ETEmail	Recebe o valor do elemento gráfico *Edit Text* responsável pelo campo de email.
ETLogin	Definido com o valor do elemento gráfico *Edit Text* responsável pelo campo de login.
ETPassword	Recebe o valor do elemento gráfico *Edit Text* responsável pela solicitação da senha.
ETPassword2	Definido com o valor do elemento gráfico *Edit Text* responsável pela solicitação da confirmação de senha.
Métodos	
onCreate	Permitirá a retomada de uma atividade interrompida durante o treinamento.
btnFinalizaCadastro	Finaliza o cadastro de usuário, passando os dados por parâmetro para o método inserirUsuario.
validaCampos	Verifica se os campos foram preenchidos.
validaSenha	Verifica a igualdade entre as senhas informadas pelo usuário.

Tabela 3. Descrição de atributos e métodos da classe Login.

Classe Login

Atributos	
Nome	**Descrição**
db	Atributo responsável por indicar o tipo de base de dados.
BANCO	*String* que contém o nome do banco de dados.
bd	Variável de instância da Classe Banco.
login	*String* que contém o nome de *login* do usuário informado na aplicação.
senha	*String* com a senha informada pelo usuário.
ETLogin	Recebe o valor do campo *Edit Text* de login da interface da aplicação.
ETSenha	Recebe o valor do campo *Edit Text* de senha da interface de login.
chamaInicial	Contém a *Intent* que chama a Classe Inicial.
chamaCadastro	Contém a *Intent* que chama a Classe Cadastro.
Métodos	
onCreate	Permitirá a retomada de uma atividade interrompida durante o treinamento.
btAbreTelaInicial	Abre a tela da Classe Inicial.
btAbreTelaCadastro	Chama a atividade de cadastro de usuário.

Tabela 4. Descrição de atributos da classe Inicial.

Classe Inicial	
Atributos	
Nome	**Descrição**
db	Atributo responsável por indicar o tipo de base de dados.
BANCO	*String* que contém o nome do banco de dados.
bd	Variável de instância da Classe Banco.
listaPercursos	Lista com nome dos percursos existentes na base de dados, exibidas no elemento *Spinner* da Classe Inicial.
nomePercurso	Contém a definição do nome do percurso pelo usuário.
labels	Recebe o nome dos percursos do banco de dados para formar a lista listaPercursos.
ETNovoPercurso	Recebe o valor do campo *Edit Text* de definição de novo percurso.
alerta	Mensagem de alerta exibido quando solicitada a exclusão de um percurso.
chamaPrincipal	Contém a *Intent* que chama a Classe Principal.

Métodos	
onCreate	Permitirá a retomada de uma atividade interrompida durante o treinamento.
loadSpinnerData	Carrega o *Spinner* com o nome dos percursos existentes na base de dados.
btCarregar	Quando pressionado define o nomePercurso com o nome do percurso existente selecionado e carrega a atividade Principal.
btExcluir	Exclui os dados do percurso selecionado da base de dados.
salvaTrajeto	Define nome o nomePercurso com o informado e Abre a tela da atividade Inicial.

Tabela 5. Descrição dos métodos da classe Principal.

Classe Principal	
Atributos	
Nome	**Descrição**
db	Atributo responsável por indicar o tipo de base de dados.
BANCO	*String* que contém o nome do banco de dados.
bd	Variável de instância da Classe Banco.
nomePercurso	Recebe o nome do Percurso definido ou selecionado para atividade.
bestProvider	Contém o melhor provedor de dados referentes à localização do dispositivo
lat	*String* com 4 casas decimais que recebe a Latitude.
lon	*String* com 4 casas decimais que recebe a Longitude.
alt	Armazena informações da altitude em relação ao nível do mar.
vel	Recebe a velocidade atual no formato de Km/h.
salvaTempo	Armazena o tempo decorrido da atividade quando o btPausar for pressionado.
idStatusIniciado	Atributo do tipo *Boolean* que indica o status da atividade (*true* = iniciado, *false* = pause).
latitude	Armazena a latitude atual no formado *Double.*
longitude	Armazena a longitude atual no formado *Double.*
vetor	Recebe a velocidade média, mínima e máxima do Método buscaDadosPercurso.

cronometro	Contém o tempo da duração de atividade para exibição na tela Principal.
btPausar	Quando pressionado define o idStatusIniciado como *false.*
btIniciar	Quando pressionado define o idStatusIniciado como *true.*
btFinalizar	Finaliza a atividade Principal, encerrando a obtenção e exibição de dados pela aplicação.
TVAtual	Elemento *Text View* da interface gráfica que recebe o valor da velocidade Atual.
TVMedia	Elemento *Text View* da interface gráfica que recebe o valor da velocidade Média.
TVMin	Elemento *Text View* da interface gráfica que recebe o valor da velocidade Mínima.
TVMax	Elemento *Text View* da interface gráfica que recebe o valor da velocidade Máxima.
FGoogleMap	Elemento *Fragment* que recebe o mapa para exibição da localização atual na atividade Principal.
Métodos	
onCreate	Permitirá a retomada de uma atividade interrompida durante o treinamento.
btIniciar	Define a atividade como iniciada.
btPausar	Define a atividade como paralisada.
btFinalizar	Finaliza a atividade e a coleta de informações da atividade Principal.
onLocationChanged	Verifica a mudança de localização com base nas informações obtidas através do sistema de GPS.
getLatitude	Utilizado para obter a latitude da coordenada atual informada pelo sistema de GPS.
getLongitude	Utilizado para obter a longitude da coordenada atual informada pelo sistema de GPS.
getAltitude	Utilizado para obter a altitude do terreno informada pelo sistema de GPS.
getSpeed	Utilizado para obter a velocidade atual informada pelo sistema de GPS.
onProviderDisable	Verifica se o GPS está desativado no dispositivo móvel.
onProviderEnable	Verifica se o GPS está ativado no dispositivo móvel.
onStatusChanged	Verifica a mudança de status de funcionamento do sistema de GPS.

isGooglePlayServicesAvaiable	Verifica se os serviços do Google estão disponíveis para obter o mapa.
getMap	Utilizado para obter o fragmento de mapa do Google.
setMyLocation	Define a localização atual no mapa.