UNIVERSIDADE DE CAXIAS DO SUL
Centro de Computação e Tecnologia da Informação - CCTI
Curso de Bacharelado em Ciência da Computação

Luís Gustavo Tessari

APLICATIVO DE LOCALIZAÇÃO DE LUGARES POR GPS PARA IPHONE

Caxias do Sul

2010

Luís Gustavo Tessari

APLICATIVO DE LOCALIZAÇÃO DE LUGARES POR GPS PARA IPHONE

Trabalho de Conclusão de Curso para obtenção do Grau de Bacharel em Ciência da Computação da Universidade de Caxias do Sul.

Helena Graziottin Ribeiro
Orientadora

Caxias do Sul

2010

AGRADECIMENTOS

Aos professores, pelos ensinamentos e cobranças as quais me fizeram sempre lutar e estudar para passar pelos obstáculos.

Agradeço a todas as amizades que eu fiz durante o decorrer do curso e que estiveram comigo nessa longa jornada.

Ao pessoal da Logidados, que durante todo esse tempo me transmitiram conhecimentos pertinentes, os quais indubitavelmente foram indispensáveis para a minha formação acadêmica.

A minha namorada pelo incentivo em não me deixar desistir e pela paciência em vários momentos.

Aos meus amigos e familiares que sempre me apoiaram. Gostaria de citar em especial ao meu irmão por todas as ajudas que ele me deu. E por último, mas não menos importante, ao meu pai e minha mãe. Vocês dois sabem o quão difícil o foi e sem o apoio e carinho de vocês eu definitivamente não estaria aqui.

RESUMO

Este trabalho apresenta a elaboração e implementação de um aplicativo de localização de lugares por GPS para iPhone, convergindo algumas tecnologias em um único software voltado para a distribuição mundial do mesmo. Os telefones celulares disponibilizam uma série de serviços aos seus usuários. É possível implementar novos aplicativos para incorporar e criar novas soluções.

Palavras-chaves: programação em iPhone, GPS, aplicativo, busca.

ABSTRACT

This work presents the elaboration and implementation of a search application of points of interest by GPS for iPhone, mixing some technologies into a single software with the intent of global distribution. The cell phones provide a lot of services to its users. It is possible to develop new applications to merge and create new solutions.

Keywords: iPhone programming, GPS, application, search.

LISTA DE ILUSTRAÇÕES

Figura 1: Disposição do teclado QWERTY 11
Figura 2: Um diagrama de um programa típico para iPhone 14
Figura 3: Maps para iPhone 15
Figura 4: Location Tracking GPS 16
Figura 5: A distribuição dos satélites GPS 18
Figura 6: Os segmentos do GPS 18
Figura 7: Posicionamento dos satélites GPS 19
Figura 8: Modelo de Caso de Uso 27
Figura 9: Descrição da Arquitetura 29
Figura 10: Diagrama de pacotes 30
Figura 11: Diagrama de Classes do Projeto 31
Figura 12: Iniciando o projeto no XCode 32
Figura 13: Localizador.h 35
Figura 14: Localizador.m 36
Figura 15: Splash screen 37
Figura 16: Tela inicial do Finder GPS 38
Figura 17: Telas de visualização de locais cadastrados 39
Figura 18: Inclusão de um novo local 40
Figura 19: Informações do novo local 41
Figura 20: A tela de mais opções do aplicativo 41
Figura 21: Módulos Unidade de Distância e Sobre respectivamente 42
Figura 22: Lista de idiomas do iPhone 43
Figura 23: Aplicativo em inglês e alemão 44
Figura 24: Aplicativo em espanhol e italiano 45
Figura 25: Aplicativo em português e francês 45
Figura 26: Opções disponíveis no iTunes Connect 47
Figura 27: Finder GPS disponível na Apple Store 48

LISTA DE ABREVIATURAS E SIGLAS

Sigla	**Significado**
API	*Application Programming Interface*
CORBA	*Common Object Request Broker Architecture*
CSS	*Cascading Style Sheet*
GPS	*Global Positioning System*
HTML	*HyperText Markup Language*
HTTP	*Hypertext Transfer Protocol*
MCS	*Master Control Station*
S/A	*Selective Availability*
XIB	*NeXT Interface Builder*
XML	*eXtensible Markup Language*

Sumário

1 Introdução 9
2 Smartphones 11
2.1 Introduzindo: O iPhone 12
2.1.1 XCode: O ambiente de desenvolvimento 13
2.1.2 Uma visão geral de um aplicativo para o iPhone 13
2.2 Aplicativos disponíveis com GPS na Apple Store 14
2.2.1 Maps 14
2.2.2 Location Tracking GPS 15
3 Recursos e aplicações do iPhone 17
3.1 GPS 17
3.1.1 Os segmentos do GPS 18
3.1.2 A idéia do funcionamento 19
3.1.3 A precisão do GPS 20
3.1.4 Fórmula de Haversine: calculando distâncias entre duas coordenadas 20
3.2 Objective-C: Uma breve introdução 20
3.3 Persistência de dados 22
3.3.1 Banco de dados: SQLite 22
3.3.2 Um banco de dados embutido 22
3.3.3 Portabilidade 23
3.3.4 Flexibilidade 23
3.3.5 Licença de uso 23
3.4 Web services e XML 24
3.5 Internacionalização 24
3.5.1 Internacionalização e Localização 25
4 Projeto do Finder GPS 26
4.1 Plano de Projeto 26
4.2 Requisitos do Sistema 26
4.3 Caso Manter Registros 27
4.4 Caso Buscar Local 28
4.5 Caso Configurações Adicionais 28
4.6 Descrição da Arquitetura 29
4.7 Estrutura das tabelas do Banco de Dados 29
4.8 Diagrama de Classes 30
5 Implementação da Proposta 32
5.1 Frameworks utilizados 33
5.1.1 Explorando o projeto no XCode 33
5.2 Exemplo de código: Classe para utilização do GPS 34
5.3 A Interface e exemplos de uso 36
5.4 Aplicação de Internacionalização 42
5.5 Imagens do aplicativo em outros idiomas 44
5.6 Dificuldades enfrentadas 46
5.7 Distribuição do aplicativo na Apple Store 46
6 Conclusão 49
6.1 Trabalhos futuros 49
7 Referências Bibliográficas 51

1 INTRODUÇÃO

Desde o início das eras, o homem percebeu a necessidade de se deslocar de um ponto a outro. Independentemente do modo que o ser humano utilizasse para se locomover, a navegação que se destacou, sem dúvida, foi a navegação visual. Esse método é usado até hoje nos meios de transporte comum ou em nosso dia-a-dia, por exemplo, quando vamos ao supermercado, estamos navegando, mentalizando e repetindo um caminho (Fontana, 2002).

A telefonia celular móvel foi introduzida no Brasil no final do século passado. Inicialmente muito cara e restrita, ela cresceu em um ritmo exponencial.

Atualmente a utilização do serviço de ligação é apenas uma das muitas possibilidades que um aparelho celular pode proporcionar ao usuário. Com a globalização cada vez mais evidente as pessoas estão ávidas por informações, e as consomem e as geram mais e mais. Tudo isso em tempo real. Uma dessas importantes funcionalidades é a utilização de GPS presente em alguns aparelhos celulares e *smartphones*. GPS é a abreviatura de *Global Positioning System*, ou seja, Sistema de Posicionamento Global. Por meio do receptor GPS pode-se determinar uma posição geográfica exata sobre a superfície terrestre (latitude e longitude).

Sua finalidade é informar a real posição do receptor e, a partir disso, calcular e determinar a velocidade real e o tempo estimado para pontos marcados (*waypoints*) (Fontana 2002).

Com o lançamento dos telefones móveis com GPS, a possibilidade de ter em mãos a sua localização exata abriu muitas portas para desenvolvimento de soluções que há alguns anos atrás não se pensava possível (pelo menos não para um desenvolvedor comum).

A motivação deste trabalho foi buscar um desafio e conhecimento em uma área completamente nova e com uma forte tendência de crescimento que agrega muitas tecnologias de ponta. A possibilidade de poder distribuir ele através da Apple Store e, consequentemente, ter um alcance há muito usuários foram pontos chave na escolha do assunto.

O capítulo dois descreve um pouco os *smartphones* e fornece uma breve introdução ao iPhone.

O capítulo três descreve as tecnologias necessárias e utilizadas neste projeto as quais o iPhone permite usar.

No próximo capítulo, o capítulo quatro, é realizado todo o planejamento do projeto detalhando os requisitos e casos de uso realizados.

No capítulo cinco é descrito e mostrado como foi implementado o aplicativo com telas de exemplos de uso.

Finalmente o trabalho é concluído fazendo-se algumas considerações finais e a avaliação do trabalho como um todo.

2 SMARTPHONES

Smartphone é um celular que possui um teclado QWERTY (vide figura 1), o nome vem das 6 primeiras letras, e é mais poderoso e possui mais recursos que um celular normal como GPS, câmera, acesso a internet, tela de alta resolução entre outras (ALLEN, GRAUPERA, LUNDRIGAN, 2010).

Figura 1: Disposição do teclado QWERTY

Desenvolver aplicações para dispositivos móveis pode ser um negócio complicado. Os desenvolvedores precisam utilizar ferramentas e APIs específicas e ainda mais, escrever códigos em linguagens diferentes em plataformas diferentes. É muito comum ser difícil de entender tudo o que se precisa para desenvolver e distribuir uma aplicação para um celular em específico até o momento de realmente criar um.

Hoje em dia é possível se afirmar que o telefone celular é o novo computador. O computador *desktop* vai continuar existindo mas este novo mercado está em rápida expansão. Celulares de hoje tem o poder de processamento de computadores de 10 anos atrás. Isso tudo no tamanho da sua mão. Os celulares estão sendo usados como computadores por mais pessoas e com mais propósitos. Os *smartphones* são geralmente mais baratos que computadores, muito mais convenientes principalmente pela sua mobilidade e muitas vezes mais úteis com o uso da sua funcionalidade de geolocalização. No mundo existem mais celulares conectados na Internet do que computadores. Somente no Brasil já existem mais de 194 milhões de aparelhos celulares, mais celulares que o número atual da população brasileira. Obviamente deste montante, os *smartphones* ainda são uma pequena parcela do total. Mas no mercado de celulares, os que hoje são de tecnologia de ponta podem ser na metade do ano seguinte ultrapassados. Por isso, é um mercado de grande expansão, especialmente com lucros crescendo vindos de aplicações desenvolvidas para este nicho.

Os *smartphones* vieram para mudar a visão das pessoas em relação aos computadores. Cada vez mais estão sendo adquiridos e sua expansão está sendo vertiginosa. O computador de mesa está no caminho de ser relegado a especialistas e profissionais da área, visto que a

maioria das funções exercidas hoje no *desktop* serão atividades normais que um *smartphone* já exerce ou irá exercer em breve. Mais importante ainda, novos aplicativos irão surgir para suprir as necessidades das pessoas que não usam computadores hoje. O desenvolvimento de software irá mudar o seu foco para o desenvolvimento de aparelhos celulares visto que a maioria das pessoas que usam computadores hoje irão usar eles indiretamente através dos celulares.

Existem vários *smartphones* no mercado hoje e o foco deste trabalho é o iPhone.

2.1 INTRODUZINDO: O IPHONE

Aplicativos para celulares não são de agora. Desde a expansão do uso de aparelhos no final do século passado eles existem. Já naquela época era considerado um mercado muito bom. Mas lá no início era muito confuso como baixar e instalar uma nova aplicação ou um jogo, e a maioria dos usuários não instalavam nada em seus aparelhos. Também há de se ressaltar que naquela época os desenvolvedores trabalhavam diretamente ou indiretamente para os fabricantes dos celulares. Não existiam desenvolvedores independentes.

Com a entrada do iPhone no mercado, a Apple revitalizou o desenvolvimento para celulares. Ela criou uma interface fácil de usar e navegar através de seus aplicativos e com preços atraentes. A facilidade de comprar através da Apple Store e a instalação automática chamaram a atenção do consumidor e o mais importante, promoveram esta capacidade para os seus usuários e futuros usuários.

No final de Setembro de 2009 a Apple anunciou que mais de 85 mil aplicativos estavam disponíveis através da Apple Store e, que havia também alcançado o número astronômico de 2 bilhões de downloads para um mercado, até então, de mais de 50 milhões de usuários de iPhones e iPods Touch (APPLE, 2009). A Apple transformou drasticamente o mercado de dispositivos móveis aumentando o consumo dos usuários em aplicativos e mudando o foco de desenvolvedores independentes para este lucrativo mercado.

E sim, o iPhone continua sendo um iPod. Mas o iPod é apenas uma de suas muitas funcionalidades. Além de tocar músicas ele é o celular que melhor utiliza o poder da Internet. Ele mostra emails totalmente formatados (e com anexos), e mostra páginas inteiras com seu conteúdo e fontes intactos. Possui também um sensor de inclinação, sensor de aproximação, wi-fi, sensor de luz, *bluetooth* e definitivamente o melhor *touch screen* do mercado.

Além do mais ele possui um calendário e uma lista de contatos com a tecnologia de sincronização. Por exemplo se o usuário possui uma conta no Gmail, esta conta irá espelhar

exatamente todos os contatos que possuir no celular no Google *Contacts*. Independentemente de onde cadastrar no iPhone ou no próprio *website*.

Além disto é também uma calculadora, um alarme, um relógio e prevê o tempo. Mas graças a Apple Store é possível elevar as possibilidades. A partir da integração/utilização de novos aplicativos de graça ou quase de graça é possível jogar jogos divertidos em 3D, transformá-lo em um controle remoto, ler livros e mais em uma infinidade de opções.

2.1.1 XCode: O ambiente de desenvolvimento

Quando o iPhone foi lançado não havia nenhum ambiente nativo de desenvolvimento aberto ao público. A Apple afirmou que não era preciso um e que as aplicações criadas para ele deveriam ser baseadas na *web* e construídas usando CSS, JavaScript e HTML. Isso não foi o suficiente para acalmar os desenvolvedores que queriam acesso direto ao hardware em si.

Apenas alguns meses depois deste anúncio, a comunidade *open source* conseguiu um feito que muitos diziam ser impossível. Os desenvolvedores ganharam acesso irrestrito ao aparelho usando técnicas de engenharia reversa e quebraram o código de segurança, o chamado *jailbreak*. Logo em seguida estavam criando uma ferramenta de desenvolvimento de código aberto para desenvolver os seus próprios aplicativos diretamente no iPhone. Detectando isso como uma ameaça em meados de 2008 a Apple lançou o seu próprio ambiente de desenvolvimento para iPhone integrado ao já existente XCode.

2.1.2 Uma visão geral de um aplicativo para o iPhone

Quando o usuário pressiona o ícone do aplicativo na tela principal do iPhone, o método main() é chamado. Este método chama o método UIApplicationMain o qual é o controlador principal do aplicativo e responsável por tratar os eventos. Internamente ele possui uma *thread*, onde são tratados os eventos de toques em tela, avisos de notificação, mudanças de orientação do celular, entre outras. Quando o usuário realiza uma operação que irá fazer com que o aplicativo termine, o *framework* UIKit notifica a aplicação e termina o processo.

Todo o aplicativo ao ser criado possui uma classe delegate, geralmente com o nome do seu projeto na frente, por exemplo: FinderAppDelegate. O delegate da aplicação é a classe principal do aplicativo e é ela quem recebe as mensagens da *thread* principal. É também a

responsável por tratar erros críticos.

Também há a classe principal controladora de *views*. Esta classe tem como principal objetivo apresentar ao usuário as *views* criadas pelo desenvolvedor. *Views* são as telas que são mostradas ao toque de um botão ou que respondem a certas situações. Elas fornecem a navegação ao usuário (vide figura 2).

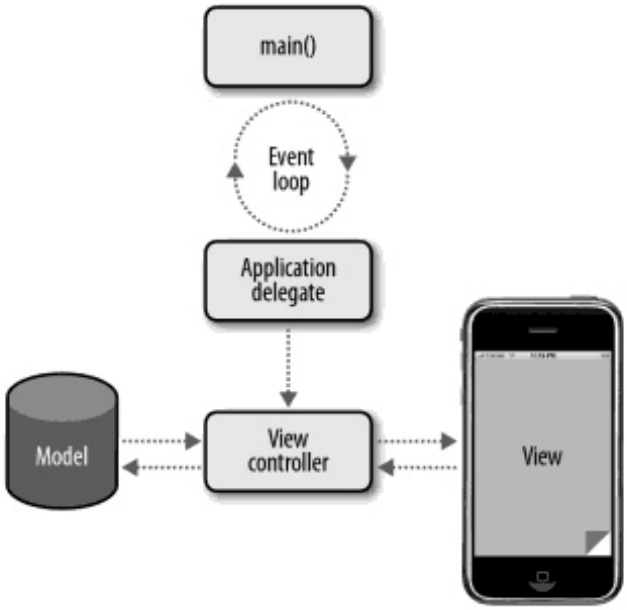

Figura 2: Um diagrama de um programa típico para iPhone

2.2 Aplicativos disponíveis com GPS na Apple Store

Antes de iniciar o desenvolvimento do aplicativo foram pesquisados alguns aplicativos com idéias similares para ver o seu funcionamento e com o que se basear. Esses aplicativos são apresentados a seguir.

2.2.1 Maps

Primeiramente o próprio Maps que vem com o iPhone. Um aplicativo completo que implementa várias funcionalidades interessantes, como o melhor caminho a se fazer a pé ou de carro (vide figura 3).

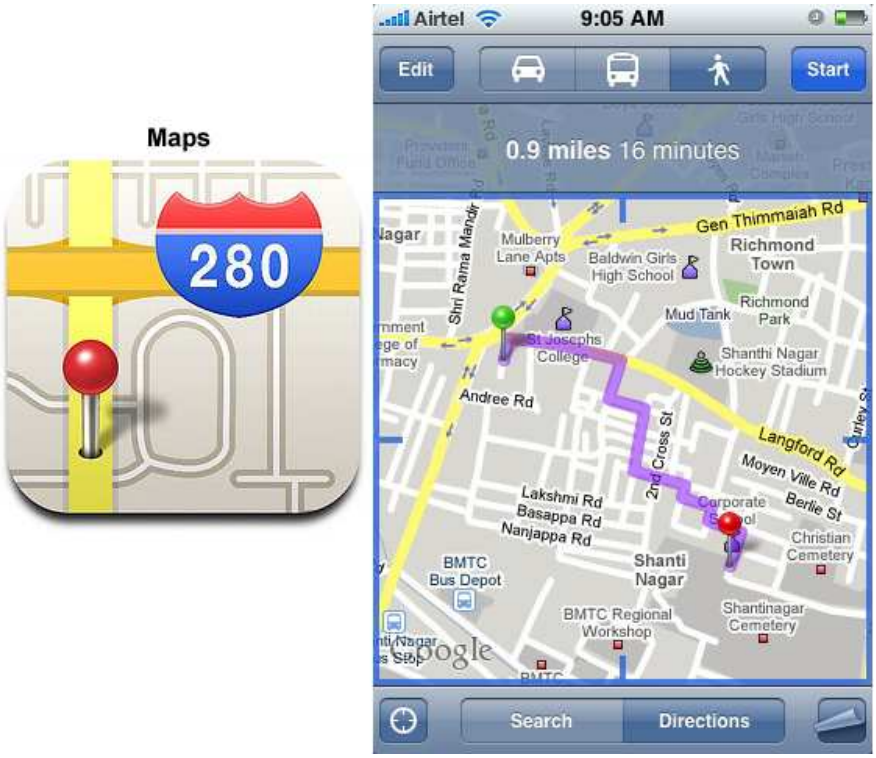

Figura 3: Maps para iPhone

2.2.2 Location Tracking GPS

Um aplicativo bastante interessante também que utiliza o poderio do MapKit. Suas principais características são (vide figura 4):

- Permite mudança de orientação do iPhone;
- Permite salvar o percurso que o usuário está fazendo, permitindo customizar cores do trajeto percorrido;
- Baixar e utilizar mapas em modo *offline*;
- Exportar os dados e enviar por email.

Figura 4: Location Tracking GPS

3 RECURSOS E APLICAÇÕES DO IPHONE

Este projeto visa demonstrar como proceder para analisar e desenvolver um aplicativo para celular utilizando o recurso do GPS embutido no iPhone. Para tanto, é necessário mostrar um estudo das tecnologias envolvidas e necessárias, o porquê da utilização de cada uma e suas principais características que se encaixam no assunto proposto a banca avaliadora. As próximas seções são introduzidas com este próposito, contando um pouco da história de cada uma.

3.1 GPS

É da natureza humana o medo do desconhecido, o medo de se perder ou de não se achar o que procura. O ato de ir com a dúvida do como chegar torna-se enormemente facilitado com o auxílio da tecnologia para localização disponível atualmente e, mais ainda, com a dupla segurança de como voltar e, como se não bastasse, a localização do indivíduo (Rocha, 2003).

O Sistema de Posicionamento Global (GPS) é um sistema de navegação baseada por satélites o qual foi desenvolvido pelo departamento de defesa dos Estados Unidos no começo dos anos 70 com o objetivo claro de suprir as necessidades militares. Mas somente mais tarde foi disponibilizado para o mundo. O GPS fornece informações contínuas de posicionamento em qualquer lugar do mundo sob qualquer situação climática.

Basicamente o Sistema de Posicionamento Global é constituído por uma rede de 24 satélites em seis planos de órbita sobre a Terra a uma altitude de 20.200 km aproximamente. Esta constelação foi concluída em julho de 1993.

Como o seu serviço alcança um número gigantesco de indivíduos e como também é utilizado por razões de segurança, o GPS funciona de modo passivo, ou seja, os usuários podem somente receber os sinais do satélite (RABBANY 2002).

Para garantir cobertura de sinal continuamente ao redor do mundo, os satélites GPS são distribuídos de forma que quatro satélites são posicionados em cada um dos 6 planos orbitais (vide figura 5).

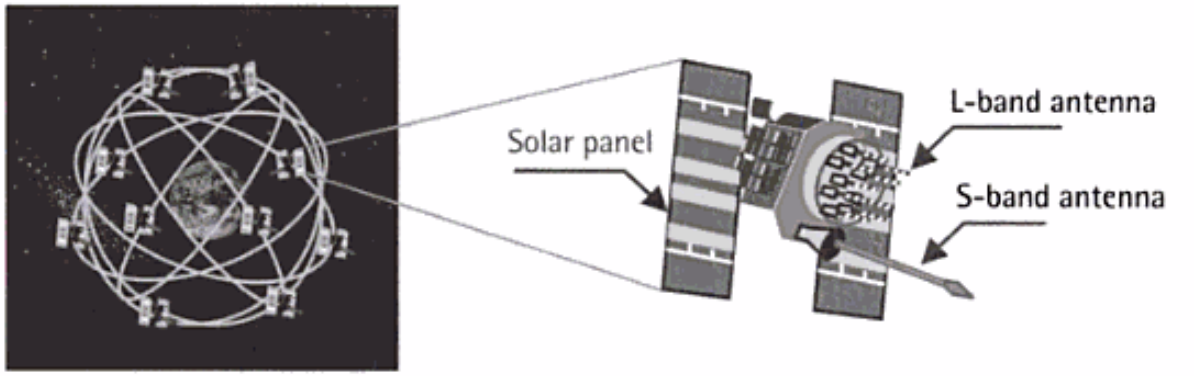

Figura 5: A distribuição dos satélites GPS

3.1.1 Os segmentos do GPS

O GPS consiste em três segmentos: o segmento do espaço, o segmento de controle e o segmento do usuário (vide figura 6). O segmento do espaço é constituído pelos 24 satélites descritos anteriormente. Cada um transmite um sinal, o qual possui um número de componentes: duas ondas sinoidais (também comumente conhecido como transportadora de frequência), dois códigos digitais e uma mensagem de navegação. Os códigos e a mensagem são adicionados ao transportador como modulações bifásicas binárias. Sua principal

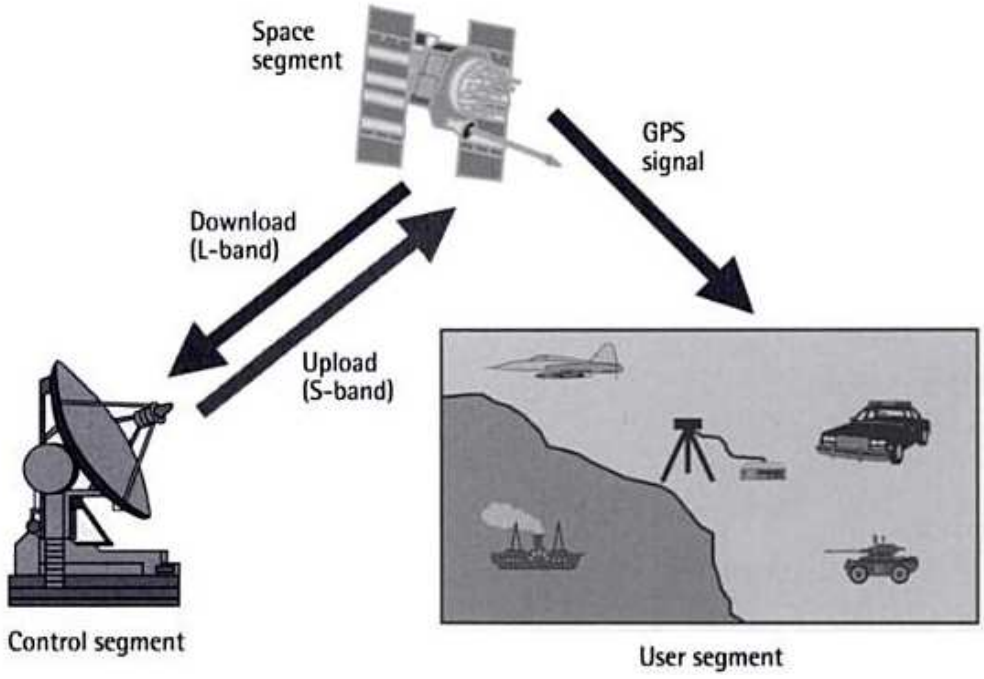

Figura 6: Os segmentos do GPS

funcionalidade é de determinar a distância do usuário receptor ao satélite. A mensagem de navegação contém além de outras informações, as coordenadas exatas dos satélites.

O controle de segmento consiste de uma rede de monitoramento ao redor do mundo com uma estação de controle mestre (MCS) localizada na cidade de Colorado Springs, Colorado, EUA. A sua função principal é de localizar os satélites, determinar sua posição, verificar a integridade do sistema, verificar o comportamento dos relógios atômicos dos satélites, dados atmosféricos, entre outras informações.

E por último o segmento do usuário. Com um receptor GPS conectado a uma antena GPS um usuário pode receber sinais GPS, as quais podem ser usadas para determinar com exatidão a sua coordenada em qualquer lugar do mundo (RABBANY, 2002).

3.1.2 A idéia do funcionamento

A idéia do GPS é bastante simples. Se a distância para um ponto na Terra para três satélites é conhecido juntamente com a posição dos satélites, então a localização do ponto, ou receptor, pode ser determinada simplesmente utilizando o conceito de intersecção de um ponto. Teoricamente, apenas três distâncias localizadas simultaneamente são necessárias. Neste caso o receptor é localizado pela intersecção de três esferas cada uma com um raio de distância de um satélite receptor e centralizada naquele satélite em particular (vide figura 7).

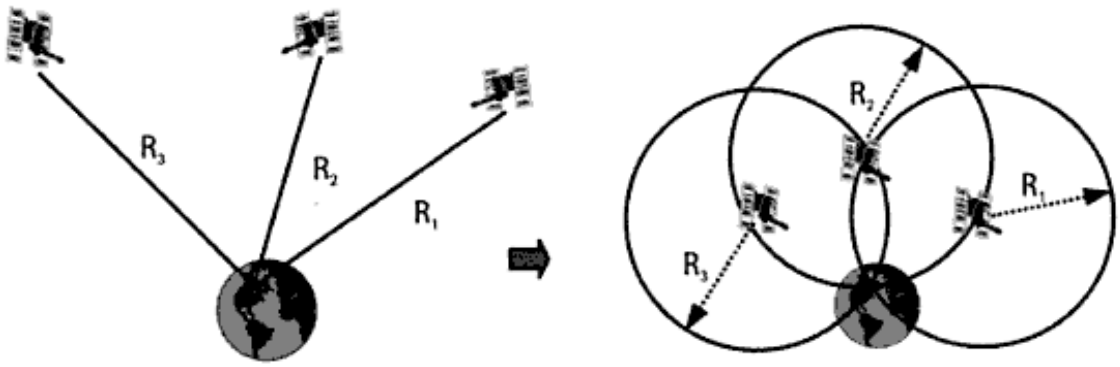

Figura 7: Posicionamento dos satélites GPS

Uma vez calculada a posição em qualquer lugar da Terra, o receptor GPS terá sempre de cinco a doze satélites em vista. O receptor, continuamente, selecionará os melhores satélites em vista para o cálculo das posições a uma taxa de uma nova posição por segundo. O receptor GPS fornece coordenadas, altitude, velocidade e hora.

3.1.3 A precisão do GPS

Os usuários não autorizados sofriam a influência da disponibilidade seletiva (S/A). O fator S/A era uma degradação proposital, elaborado pelo Departamento de Defesa americano, da precisão do GPS para algo em torno de 100 metros na horizontal, 156 metros na vertical e 340 nanosegundos no tempo. Esta influência foi eliminada no dia primeiro de maio de 2000 pelo então presidente norte-americano, Bill Clinton. A partir desta data a precisão das posições eram de 15 metros ou menos (ROCHA, 2003).

3.1.4 Fórmula de Haversine: calculando distâncias entre duas coordenadas

Para calcular duas distâncias em métodos de navegação, existe uma fórmula básica de trigonometria para tal. A fórmula de Haversine calcula a distância entre dois pontos, através das coordenadas de ambos (latitude e longitude). Algo para se lembrar é que ignora altitudes, é somente a distância em linha reta entre as duas coordenadas.

Fórmula de Haversine (os ângulos precisam ser passados em radianos para função trigonométrica):

R = o raio da Terra (6.371km)

$\Delta lat = lat_2 - lat_1$

$\Delta long = long_2 - long_1$

$a = seno^2(\Delta lat/2) + coseno(lat_1).coseno(lat_2).seno^2(\Delta long/2)$

$c = 2.atan2(\sqrt{a}, \sqrt{(1-a)})$

distância = R.c

3.2 Objective-C: Uma breve introdução

Dennis Ritchie da AT&T foi um dos pioneiros no desenvolvimento da linguagem de programação C no início dos anos 70. Somente no final dos anos 70 o C começou a ganhar popularidade visto que no seu início era um código fechado e somente o laboratório de pesquisas e desenvolvimento da AT&T tinha acesso. O aumento da popularidade na época pode ser creditado ao também aumento da popularidade do sistema operacional UNIX, o qual era quase que inteiramente escrito na linguagem C (KOCHAN, 2009).

Quem criou a linguagem Objective-C foi Brian Cox no início dos anos 80. Ele era baseado em uma linguagem chamada SmallTalk-80, na qual o Objective-C criava uma camada sobre o C, ou seja, eram criados extensões que eram adicionadas a linguagem C para criar uma variação e consequentemente uma nova linguagem de programação onde a utilização de orientação a objetos era possível. Hoje é possível dizer que ele se tornou uma variação do C++.

No início dos anos 80 a empresa NeXT Software licenciou o uso da linguagem e desenvolveu um ambiente chamado NEXTSTEP. No final dos anos 90 a Apple adquiriu a empresa NeXT Software e o ambiente NEXTSTEP tornou-se o ambiente central de desenvolvimento para o seu próximo grande lançamento: o sistema operacional Mac OS X.

Mais alguns anos se passaram e em 2007 a Apple lançou mais um produto revolucionário: o iPhone. Assim como o Mac OS X, o iPhone possui o seu sistema operacional iOS implementado em Objective-C.

Logo após o lançamento do iPhone os desenvolvedores praticamente imploraram a Apple para liberar o desenvolvimento de aplicativos para o celular. No primeiro momento a Apple permitiu o desenvolvimento de aplicativos baseados na *web*, ou seja, o usuário precisava acessar pelo navegador Safari o aplicativo. Logo foi visto as inúmeras limitações e privações da capacidade que o iPhone dispunha e os desenvolvedores clamaram pela liberação do desenvolvimento de aplicativos nativos no iPhone. Um aplicativo nativo é aquele que roda diretamente no sistema operacional iOS, assim como qualquer outro que vem instalado junto com o aparelho.

Percebendo a grande oportunidade que surgiu, não demorou muito e a Apple lançou um ambiente de desenvolvimento para o iPhone chamado XCode. Com o XCode é possível desenvolver aplicativos para iPhone tanto como para Mac OS X. Mas para tanto existem alguns pré-requisitos para tal.

Primeiramente é preciso ter um sistema operacional Mac OS X Snow Leopard versão 10.6.2 rodando. Baixando e instalando o XCode, juntamente com as suas atualizações, já é possível desenvolver e testar o aplicativo utilizando o iPhone Simulator. Para testar diretamente no celular é preciso ingressar no programa de desenvolvedores da Apple para o iOS. Para entrar no programa e testar diretamente no iPhone é preciso comprar uma licença por U$ 99,00. Esta licença dá direito ao cliente de desenvolver e distribuir aplicativos por um ano e de sonhar em ver seu aplicativo sendo usado mundialmente (APPLE DEVELOPER, 2010).

3.3 Persistência de dados

Ao pesquisar sobre persistência de dados para o iPhone, foram localizadas algumas soluções. A primeira e mais básica de todas seria salvar os dados no sistema de arquivos do próprio sistema operacional iOS., uma solução fácil de implementar, mas com um custo elevado de processamento. Ao buscar novamente por outra solução foi encontrada uma opção melhor e que se encontra na grande maioria dos aplicativos atuais para iPhone e que a própria Apple espera que se torne o *backbone* principal para guardar os dados: utilizar o *framework* que implementa o banco de dados SQLite (COOPER, 2008).

3.3.1 Banco de dados: SQLite

SQLite é uma biblioteca *open source* que implementa um banco de dados relacional embutido. Lançado originalmente em 2000 foi concebido para prover uma forma conveniente para as aplicações utilizarem um banco de dados sem a preocupação que geralmente vem com uma instalação de um banco de dados dedicado. Com o passar dos anos o SQLite firmou uma reputação de extrema portabilidade, facilidade de uso, leve, confiabilidade e de extrema eficiência (COOPER, 2008).

Mas há de se levar em consideração seus própositos. Não se pode comparar um banco de dados SQLite com um Oracle ou SQL Server. Os objetivos de cada um são distintos. O SQLite pode ser muito conveniente em vários casos de rápida solução onde é preciso construir, montar, guardar e manipular dados de uma forma eficiente e com o mínimo esforço. Uma boa forma de se comparar seria dizer que enquanto o Oracle foi desenvolvido para milhares de usuários, o SQLite foi desenvolvido para milhares de usos. E realmente ele acaba se destacando por vários motivos aos quais serão listados nas seções a seguir.

3.3.2 Um banco de dados embutido

Como o próprio nome sugere, o SQLite é uma versão ultra leve de um banco de dados relacional. Cada vez mais os desenvolvedores vem utilizando esta excelente alternativa para guardar dados em sistemas pequenos, *web sites* (especialmente com a adoção do SQLite no PHP 5) e também em celulares. Ao invés de rodar como um processo no Sistema Operacional em que está presente, ele coexiste dentro da aplicação que o está utilizando, dentro do seu

espaço de processamento. O seu código é embutido como parte do programa que o hospeda.

Uma das principais vantagens em embutir o banco de dados dentro do aplicativo é que não é necessário configurar nenhuma rede. Mesmo que a rede não esteja disponível o banco de dados continua rodando firme e forte. O cliente e o servidor rodam ambos no mesmo processo. Isso reduz problemas de latência de rede, simplifica a administração do banco de dados e o torna mais fácil de administrar.

3.3.3 Portabilidade

SQLite foi concebido com a idéia de portabilidade na ponta do lápis. Ele compila e roda nos sistema operacionais Windows, Linux, Mac OS X, BSD, Unix, Solaris e muitos outros. Ele funciona em ambientes 16, 32 e 64 bits. E mais, os arquivos do banco de dados também são portáveis. O formato do arquivo é binário compatível entre qualquer sistema operacional e arquiteturas de hardware. É possível criar um banco de dados no Windows e usá-lo no Mac OS X sem precisar mudar uma única linha de código. Cada base de dados pode ter até 2 terabytes de dados (lembrando que existe esse limite por causa do tamanho máximo de cada arquivo do sistema operacional).

3.3.4 Flexibilidade

Vários fatores ajudam a elevar o nome do SQLite. Flexibilidade é um dos pontos forte deste banco de dados embutido. Ele fornece a facilidade de utilizar comandos SQL com a simplicidade de guardar todas as tabelas em um único arquivo. Não é preciso configurar nenhum servidor de banco de dados robusto com n variáveis. Não é preciso se preocupar com tráfego de rede e toda a segurança envolvida por trás como configuração de Firewall. O desenvolvedor consegue utilizar comandos SQL diretamente na aplicação.

3.3.5 Licença de uso

O código é aberto, ninguém exige direitos de cópia. Não é preciso pagar um único centavo por uma licença de uso na sua aplicação. Não é preciso pedir permissão para ninguém indepentemente do objetivo, seja ele comercial ou não. É basicamente criar a sua base de dados, criar o que é preciso e distribuir da forma que for mais conveniente.

3.4 Web services e XML

Pode se dizer que *Web Services* está na moda. Hoje em dia é difícil alguém da área da informática não ter ouvido falar no termo. Com a obrigatoriedade das Notas Fiscais Eletrônicas, baseadas na tecnologia, ela entrou de vez como um poderoso aliado do desenvolvedor brasileiro.

Web Service fornece uma camada de abstração acima de softwares existentes, trabalhando muito similarmente como a Internet. Isso o torna portável entre navegadores, sistemas operacionais, plataformas de hardware ou linguagens de programação (NEWCOMER, 2004).

A sua principal idéia é, como o próprio nome diz, de prover serviços pela internet. Uma outra forma de se colocar um adjetivo seria de prover um sistema distribuído, adaptado para a *web*. Mas diferentemente da maioria dos sistema distribuídos existentes, que acoplam protocolos de comunicação como parte do seu escopo, os Web services rodam em cima do protocolo HTTP. A rede mundial de computadores já está inserida no seu serviço.

O poder do *Web Service* está na linguagem XML. O XML resolve os problemas de depêndencia de dados para as linguagens de programação, sistemas *middleware* e sistemas de gerenciamento de banco de dados. Anteriormente a ele houve uma tentativa com o CORBA (GOKHALE). Até houve uma certa aceitação, mas a sua complexidade de utilização por ser uma alternativa paga acabaram minando as possibilidades de uma aceitação global. Ao contrário os *Web Services* trouxeram uma alternativa muito simples de entender e aberta.

O XML pode ser descrito não como uma tecnologia mas sim uma variedade de tecnologias nele mesmo. XML não apenas fornece e descreve os dados mas também contém informações importantes para mapeá-los para dentro e para fora de qualquer software ou linguagem de programação.

Os *Web Services* fornecem poder quase que ilimitado. Qualquer programa pode ser mapeado para um *Web Service* e qualquer *Web Service* pode ser mapeado para qualquer programa.

3.5 Internacionalização

Atualmente qualquer tipo de software pode possuir um público alvo de alcance global. Isso se mostra claramente presente em aplicativos para celulares. Ao finalizar o seu desenvolvimento o uso do conceito de Internacionalização traz enormes vantagens.

Vender o seu programa para certos países pode incluir uma certa adaptação para que o mesmo transpareça ter sido desenvolvido especificamente para aquela região. Apesar de o inglês ter se tornado um idioma globalizado, muitas pessoas ainda não o sabem ou simplesmente preferem usar com o seu idioma. Um usuário estrangeiro dificilmente iria usá-lo em um idioma que não entende. Similarmente, podem existir imagens que são aceitáveis em algumas culturas mas em outras não.

Este foi um problema que surgiu durante o desenvolvimento deste aplicativo. Como poder criá-lo multi idiomas? A solução a própria Apple é quem fornece com a tecnologia de Internacionalização.

A Internacionalização permite que o seu programa customize imagens, idiomas, sons, ou qualquer componente visual que o desenvolver queira. Ao invés de reescrever todo o aplicativo para cada idioma que deseja dar suporte, é possível internacionalizar ele para que suporte qualquer idioma.

O processo envolve em separar as imagens e textos visíveis do código fonte. Uma vez isoladas em arquivos separados de recursos, o desenvolvedor pode traduzir para os idiomas desejados e integrados de volta para aplicação.

3.5.1 Internacionalização e Localização

Internacionalização é o processo de concepção e construção de um software para facilitar a localização. Por sua vez localização, é a adaptação cultural e do idioma de uma aplicação internacionalizada para dois ou mais mercados culturalmente distintos. Por exemplo quando um usuário abre um programa localizado, o usuário deve se sentir confortável em manipulá-lo sem se preocupar com a origem de seu desenvolvimento.

A Internacionalização e a Localização são atividades complementares. Ao Internacionalizar uma aplicação é criada toda uma infra-estrutura necessária para suportar o conteúdo localizável. Ao tornar localizável é preciso adicionar recursos customizados que transformem então estas imagens ou textos para diferentes culturas.

4 PROJETO DO FINDER GPS

Nas seções a seguir será descrito o projeto para o aplicativo proposto.

4.1 Plano de Projeto

A idéia do projeto é de poder disponibilizar um aplicativo pronto, e não apenas um protótipo, para distribuição e possivelmente comercialização através da Apple Store. Sua principal funcionalidade será a de possibilitar ao usuário do iPhone cadastrar locais de seu interesse e em conjunto do GPS, uma funcionalidade embutida no iPhone e visualizar a distância que se encontra em linha reta dos seus locais.

4.2 Requisitos do Sistema

Os usuários que adquirirem o aplicativo terão a possibilidade de cadastrar locais de duas maneiras: utilizando a sua localização atual ou através de uma busca por um web service digitando o endereço que deseja localizar.

Os locais cadastrados estarão divididos em sub-categorias para facilitar a visualização dos locais previamente cadastrados. Também será possível visualizar todos os locais e excluir os registros.

O aplicativo terá que dar suporte a vários idiomas. Da mesma forma o usuário poderá escolher qual a unidade de distância ele quer trabalhar.

A figura 8 demonstra o modelo de caso de uso para este projeto.

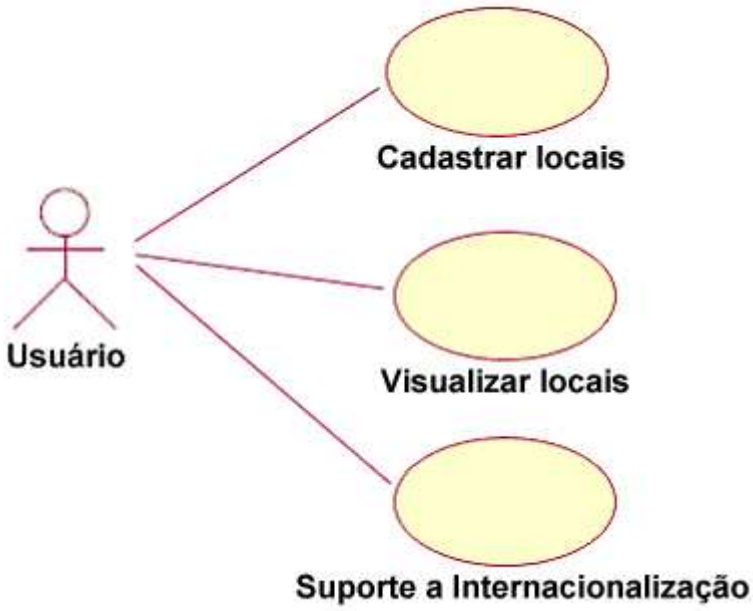

Figura 8: Modelo de Caso de Uso

O usuário do aplicativo terá a possibilidade de realizar manutenções nos seus registros cadastrados. Ele poderá incluir e excluir. Será possível também realizar uma busca por locais, caso o usuário possua um plano de dados no iPhone ou estiver conectado a uma rede wireless. A busca é feita através de um web service disponibilizado pelo Google no endereço http://maps.google.com/maps/apli/geocode. Internamente o aplicativo irá tratar o retorno do XML e disponibilizar em tela o resultado da busca. Também o usuário pode usar da sua posição corrente, sem precisar usar a Internet para esta opção.

4.3 Caso Manter Registros

O usuário cadastra um local selecionando a opção de usar a localização atual informando a descrição para o local e o tipo de lugar em que ele se enquadra. Ao cadastrar o local será possível visualizá-lo na tela principal selecionando o respectivo tipo de lugar.

O usuário poderá visualizar dentro de uma sub-categoria os registros e excluir os que desejar.

Foram previamente definidas 15 categorias para o aplicativo as quais são listadas a seguir:

- Todos os lugares;
- Meus Lugares;
- Restaurante;

- Posto de Gasolina;
- Supermercado;
- Igreja;
- Hospital;
- Hotel;
- Café;
- Banco;
- Aeroporto;
- Bar;
- Compras;
- Amigos;
- Estabel. Comercial.

Vale ressaltar que estas categorias podem vir a crescer futuramente em novas versões do aplicativo.

4.4 Caso Buscar local

Ao clicar no botáo incluir, o usuário cadastra um local digitando um endereço de busca. Caso o aplicativo obtenha retorno ao selecionar um registro o usuário irá informar a descrição para o local e o tipo de lugar em que ele se enquadra. Um cenário alternativo será poder cadastrar um local utilizando a funcionalidade de aproveitar a localização atual do usuário. Esta opção se encontrará na mesma tela da busca.

4.5 Caso Configurações Adicionais

Ao clicar no botão Mais, o usuário poderá selecionar a unidade de medida com que deseja trabalhar: metros ou quilômetros. Também será possível, através das configurações de idioma do iPhone, usuário poderá selecionar o idioma do aplicativo para o que lhe melhor convier.

4.6 Descrição da Arquitetura

A arquitetura que o sistema utiliza é local, porém fortemente ligado a web. A atualização do mapa requer conexão com a internet assim como a busca por novos locais digitando a partir de um endereço ambos usando protocolo HTTP.

O banco de dados SQLite é local e roda junto com o aplicativo do iPhone (vide figura 9).

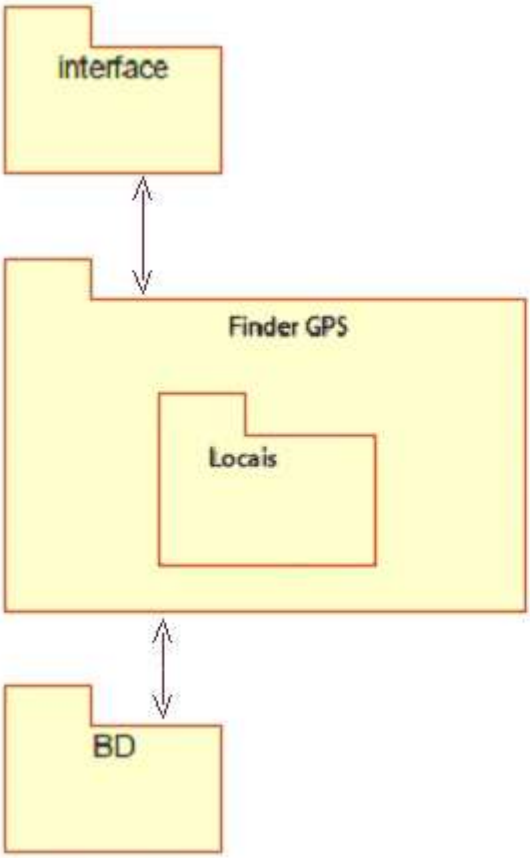

Figura 9: Descrição da Arquitetura

4.7 Estrutura das tabelas do Banco de Dados

Para que o aplicativo ficasse 100% funcional foi necessário agregar algumas tecnologias ao desenvolvimento.

Primeiramente a escolha de onde guardar os dados foi direcionada rapidamente ao SQLite. Visto a sua facilidade de uso foi criado uma pequena base de dados interna que roda junto ao aplicativo. Esta pequena e simples base de dados pode ser vista no diagrama de

pacotes da figura 10.

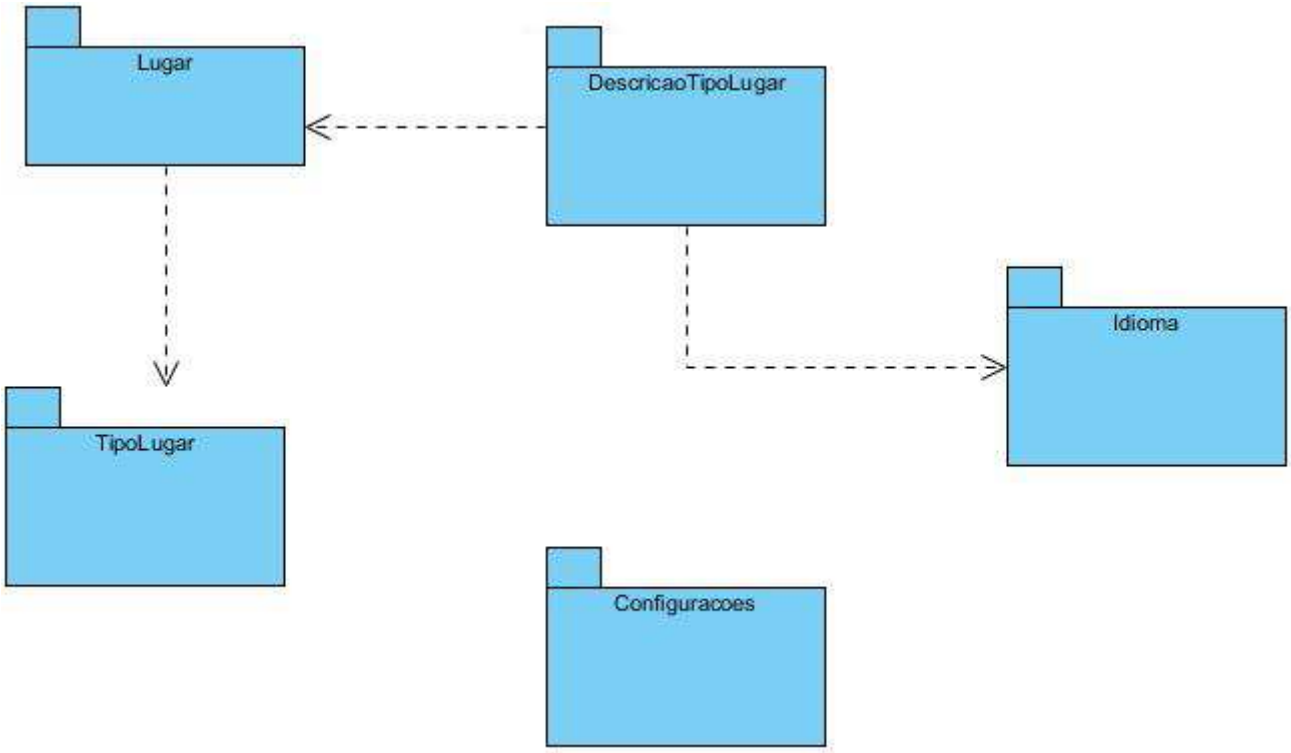

Figura 10: Diagrama de pacotes

4.8 Diagrama de Classes

Na imagem 11 é visto o diagrama de classes do projeto.

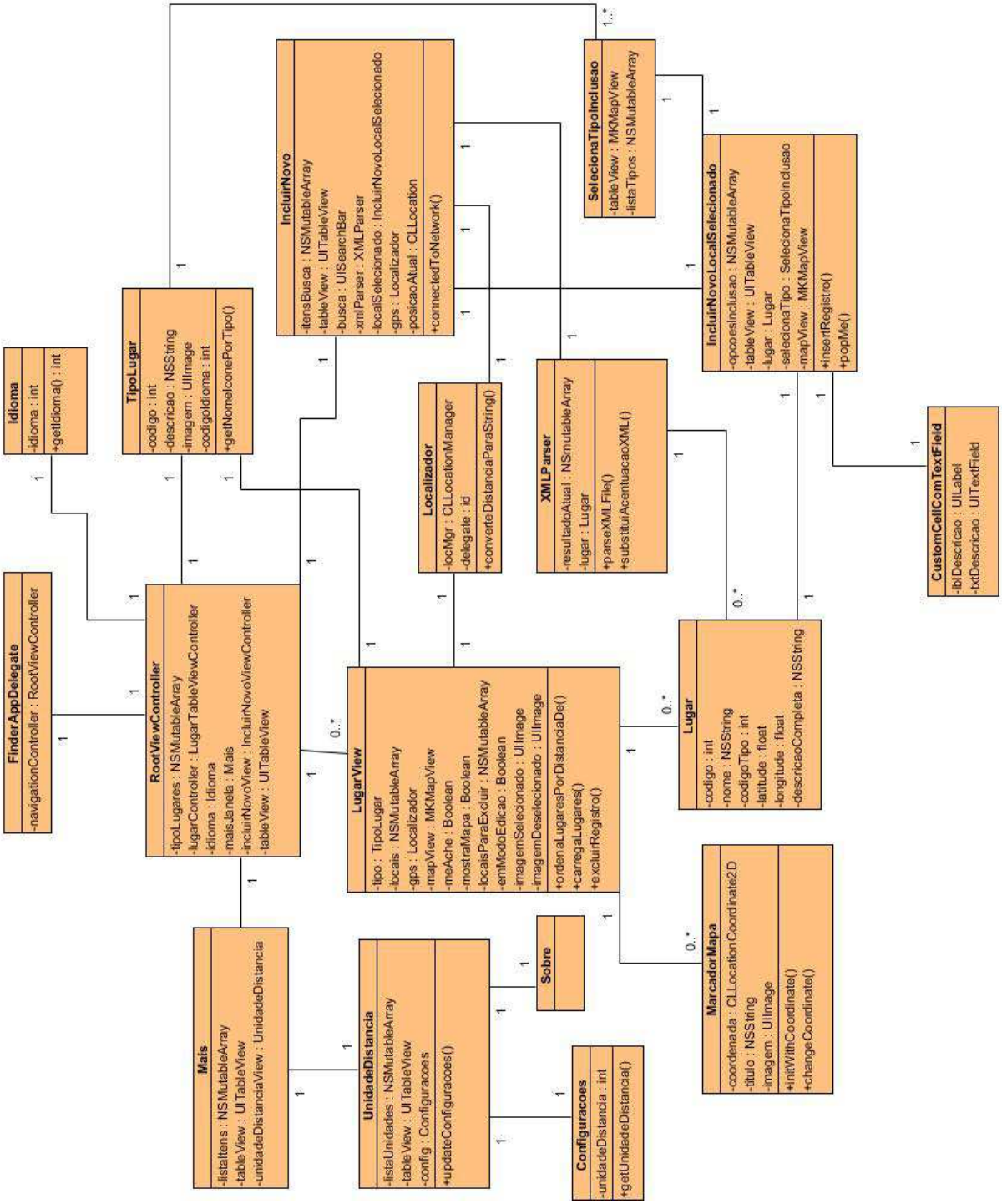

Figura 11: Diagrama de Classes do Projeto

5 IMPLEMENTAÇÃO DA PROPOSTA

Ao iniciar a implementação da proposta, foi necessário um certo tempo para se acostumar com a forma de programar, assim como entender a lógica do ambiente de desenvolvimento. Após o entendimento básico de como funciona o XCode foi escolhido que o desenvolvimento da aplicação seria através de um aplicação por navegação (vide figura 12). A navegação para este projeto no XCode pode ser descrita basicamente como programar em uma pilha. A classe controladora da navegação quando precisa mostrar uma nova *view* empilha esta nova tela e ao voltar ele desempilha.

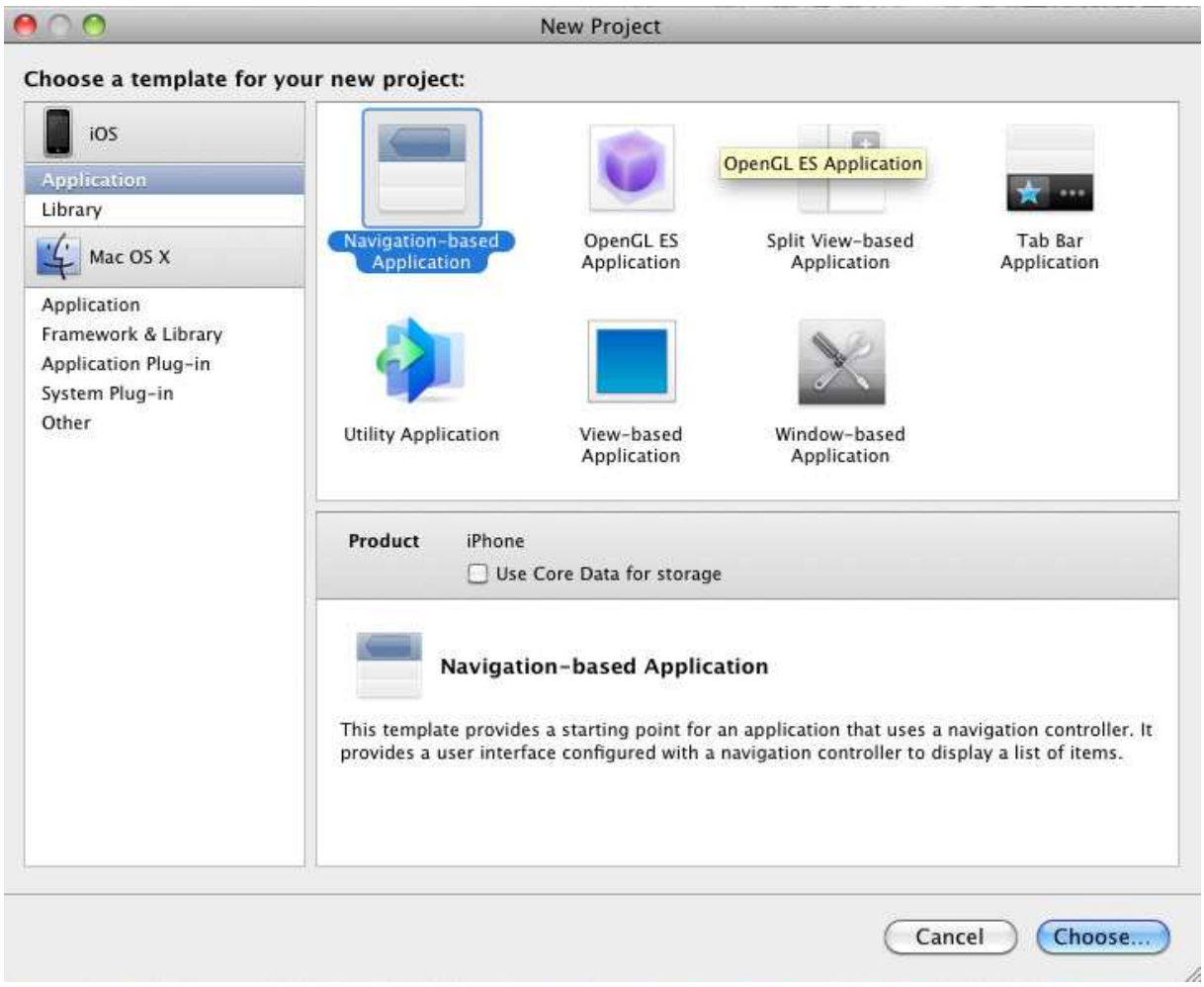

Figura 12: Iniciando o projeto no XCode

5.1 Frameworks utilizados

Os principais *frameworks* utilizados para este projeto foram os seguintes:

- **UIKit.framework** – O *framework* primário utilizado pelo iPhone para construir e mostrar os elementos de interface para o usuário;
- **MapKit.framework** – Este *framework* fornece uma interface para embutir mapas diretamente em componentes *window* ou *views*. Este *framework* também fornece suporte a *annotations* (ou também conhecido como pinos) e de executar *reverse-geocoding* para determinar a latitude e longitude através de um ponto 2D no mapa;
- **libxml2.dylib** – *Framework* que fornece suporte a tratamento de arquivos XML;
- **Foundation.framework** – O *framework* padrão utilizado por todos os aplicativos do iPhone. Este *framework* contém classes frequentemente usadas como NSString, NSInteger e NSArray;
- **libsqlite3.dylib** – *Framework* que implementa um banco de dados relacional diretamente em um arquivo;
- **SystemConfiguration.framework –** Fornece funções que detectam a acessibilidade do usuário para redes de celular e de internet;
- **CoreLocation.framework –** E por último o *framework* que permite acesso ao GPS embutido no iPhone. Permite detectar a atual posição, calcular distâncias por coordenadas, calcular velocidade de deslocamento e altitude onde se encontra o usuário.

5.1.1 Explorando o projeto no XCode

Ao criar um projeto do zero, o XCode cria vários arquivos e pastas automaticamente para o usuário. Na verdade um aplicativo para iPhone pronto para compilar. Ele cria apenas a estrutura básica para o desenvolvedor começar a implementar e se familiarizar com o ambiente. A seguir uma descrição breve do que é criado pelo XCode:

Classes

O grupo *Classes* que é criado contem as classes (.m) e os arquivos header (.h). Aqui é onde o desenvolvedor implementa a parte lógica, juntamente com a parte de apresentação do aplicativo.

Other Sources

O grupo *Other Sources* contem apenas dois arquivos. O prefixo do cabeçalho do projeto e o main.m.

Resources

Neste grupo é onde o desenvolvedor importa todos os recursos necessários do seu projeto. Geralmente aqui se encontram os arquivos XIB (arquivo de extensão de recursos utilizado para montar a interface no Interface Builder), imagens, arquivos .db (banco de dados SQLite), Localizable.strings (para multi idiomas), sons ou qualquer outro recurso necessário para a implementação do projeto.

Frameworks

Este grupo contem todas as referências de todos os *frameworks* que o desenvolvedor necessita para o projeto.

Products

E por último este grupo contém todos os arquivos binários que são gerados ao compilar o programa.

5.2 Exemplo de código: Classe para utilização do GPS

Um dos principais obstáculos é integrar-se com o conceito de alocar e desalocar objetos em memória. *Memory leaks* podem ocorrer facilmente e especialmente em Objective-C, o que pode ser um grande problema no futuro da aplicação. Em Objective-C, e assim como na maioria das linguagens de programação derivadas do C, se faz alocação de memória e desalocação quando não se precisa mais do recurso. Algumas convenções são utilizadas e a seguir segue um exemplo de uma classe do projeto, a que fornece ao aplicativo os dados do GPS.

Ao criar uma nova classe o Objective-C cria dois arquivos: um com extensão .h e outro com a extensão .m. No .h vem toda a definição da classe e no .m a implementação da mesma. Na figura 13 é mostrado a definição e implementação da classe Localizador. Nela é basicamente definido a estrutura desta classe. É explicitamente criado um protocolo para ela e são definidas as propriedades públicas.

```
//
//  Localizador.h
//  Finder
//
//  Created by Luis Gustavo Tessari on 10/21/10.
//  Copyright 2010 __MyCompanyName__. All rights reserved.
//

#import <Foundation/Foundation.h>
#include <CoreLocation/CoreLocation.h>

@protocol LocalizadorDelegate
@required
- (void)locationUpdate:(CLLocation *)location;
- (void)locationError:(NSError *)error;
@end

@interface Localizador : NSObject <CLLocationManagerDelegate> {
    CLLocationManager *locMgr;
    id delegate;
}

@property (nonatomic, retain) CLLocationManager *locMgr;
@property (nonatomic, assign) id delegate;

NSString* converteDistanciaParaString(float distancia);

@end
```

Figura 13: Localizador.h

Uma breve explicação do funcionamento da implementação dela em Objective-C é visto na figura 14. Neste caso a classe precisa implementar os métodos delegate do *framework* CoreLocation. Isto é necessário para que quando o usuário estiver andando com o iPhone, o mesmo ative o uso da comunicação com o GPS e a classe base informe ao objeto que está implementando as notificações de atualização da localização. Também é assegurado que a classe de implementação precise obrigatoriamente implementar mais dois métodos requeridos aqui. Isso é possível através da chamada *@required*.

```
//
//  Localizador.m
//  Finder
//
//  Created by Luis Gustavo Tessari on 10/21/10.
//  Copyright 2010 __MyCompanyName__. All rights reserved.
//

#import "Localizador.h"
#import "Configuracoes.h"
#import "Constantes.h"

@implementation Localizador

// Realiza os metodos get/set das propriedades declaradas no header
@synthesize locMgr, delegate;

// Metodo padrao para alocar objetos em memoria
- (id)init {
    self = [super init];

    if(self != nil) {
        // Cria uma nova instancia do objeto locMgr
        self.locMgr = [[[CLLocationManager alloc] init] autorelease];
        // Seta o delegate para ele mesmo
        self.locMgr.delegate = self;
        // Designa a proximidade com que deseja trabalhar
        self.locMgr.desiredAccuracy = kCLLocationAccuracyBestForNavigation;     }

    return self;
}

// Metodo delegate que recebe a nova posicao
- (void)locationManager:(CLLocationManager *)manager didUpdateToLocation:
                    (CLLocation *)newLocation fromLocation:(CLLocation *)oldLocation {
    if([self.delegate conformsToProtocol:@protocol(LocalizadorDelegate)]) {
        [self.delegate locationUpdate:newLocation];
    }
}

// Metodo delegate que retorna erro ao atualizar a posicao atual
- (void)locationManager:(CLLocationManager *)manager didFailWithError:(NSError *)error {
    if([self.delegate conformsToProtocol:@protocol(LocalizadorDelegate)]) {
        [self.delegate locationError:error];
    }
}

// Metodo padrao para desalocar objetos
- (void)dealloc {
    [self.locMgr release];
    [super dealloc];
}
@end
```

Figura 14: Localizador.m

5.3 A Interface e exemplos de uso

Quando foi proposto este projeto para a banca avaliadora de professores, foi sugerido um aplicativo para iPhone que utilizasse o GPS do iPhone para mostrar locais de Caxias do Sul. A idéia inicial seria montar um servidor com os registros previamente cadastrados pelo administrador e um serviço disponibilizado na web para consumir estes dados, que neste caso seria o aplicativo do iPhone. Com o desenrolar do tempo surgiram mais idéias que vieram a

agregar mais valor ao aplicativo proposto. A idéia inicial foi modificada um pouco e o aplicativo se tornou na verdade um software mais maduro, com projeções mundiais e com um plano mais ambicioso.

Foram adicionadas funcionalidades de inclusão e exclusão personalizadas pelo próprio usuário. Uma busca através de um web service disponibilizado pelo Google que permite que o aplicativo busque por qualquer endereço no mundo. Ou mesmo que utilize a sua própria localização para salvar um local de interesse seu. Também foi implementado com a tecnologia de localização para disponibilizar em outros idiomas.

A seguir será mostrado exemplos de uso do aplicativo com as suas respectivas telas.

Primeiramente ao abrir o aplicativo é mostrado uma tela de abertura do Finder GPS (vide figura 15).

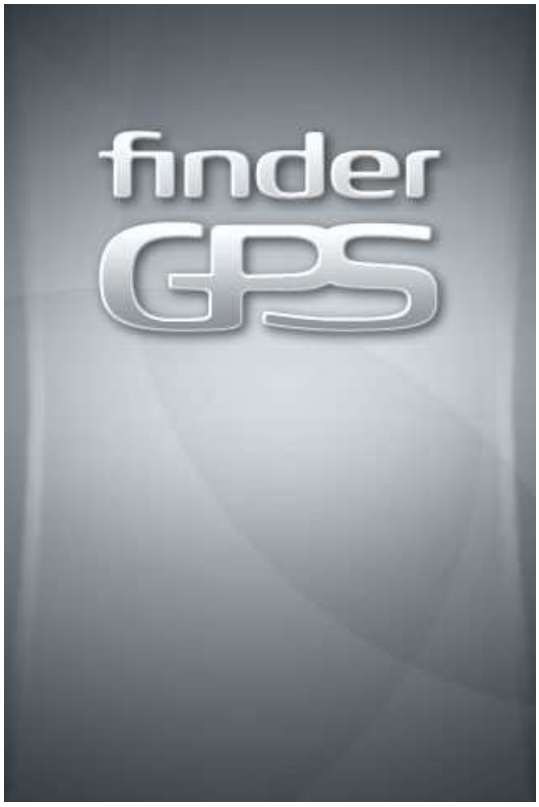

Figura 15: Splash screen

Logo após aparece a primeira tela no aplicativo. Internamente é a classe RootViewController. É a partir desta classe que todas as outras do sistema são chamadas. Nesta classe, é instanciado a classe idioma, onde se identifica através dos padrões do usuário no iPhone, e a partir desta informação é montado as subcategorias na tela. Essas categorias são previamente cadastradas no banco de dados, sendo que é necessário cadastrar um registro para cada idioma para cada tipo de lugar. Como dito anteriormente, através do *framework* de

acesso ao SQLite, o aplicativo retorna as informações do banco embutido no aplicativo e as mostra na tabela de visualização da *view* (vide figura 16).

Figura 16: Tela inicial do Finder GPS

Neste módulo é possível visualizar alguma subcategoria, incluir um novo registro ou ir nas opções adicionais. Vale lembrar que o idioma não se altera pelo aplicativo e sim o aplicativo é que se modela com as configurações padrões que o usuário definir no aparelho celular, é uma funcionalidade transparente ao usuário. Caso o usuário não utilize nenhum dos idiomas suportados pelo aplicativo o mesmo utiliza por padrão o idioma inglês.

A imagem 17 mostra a tela ao clicar em alguma subcategoria.

Figura 17: Telas de visualização de locais cadastrados

Aqui neste módulo são listados todos os locais da categoria selecionada. Ao abrir o módulo o aplicativo já foca nas coordenadas atuais do usuário. A listagem dos locais é ordenada sempre pelos locais mais próximos. Ao selecionar um local, o Finder GPS muda o status de seguir a localização do usuário e muda as coordenadas do mapa para o local pressionado. É possível dar *zoom out, zoom in* e movimentar o mapa conforme o usuário quiser. Existe uma barra de ferramentas fixa embaixo da *view*. Ali tem duas funcionalidades diferentes. A primeira permite que o usuário possa ativar a visualização do mapa ou não. A segunda permite que o aplicativo localize a posição atual do aparelho GPS e que siga a mudança das coordenadas do usuário sem que ele precise ficar monitorando. Também neste módulo é permitido através do botão editar no topo da *view*, habilitar o modo de exclusão dos registros. Esta funcionalidade permite que se selecione um ou vários locais ao mesmo tempo e que se exclua definitivamente eles do banco de dados. O cálculo da distância é realizado através de chamadas do próprio *framework* CoreLocation que implementa o cálculo da fórmula de Haversine.

Voltando a tela inicial existe uma opção de inclusão disposta no topo da tela. Nesta opção é disponibilizada uma das principais funcionalidades do aplicativo. Neste módulo é possível incluir um local baseando-se pela localização corrente de onde se encontra o

dispositivo ou através de uma busca consumindo um web service pela internet. Como é pesquisado por um web service do Google que disponibiliza todo e qualquer local do mundo, esta funcionalidade eleva o aplicativo a um novo patamar mais abrangente (vide figura 18).

Figura 18: Inclusão de um novo local

Ao selecionar o local desejado o Finder GPS traz uma nova tela. Este módulo permite que se ajuste a posição do local selecionado para que as coordenadas fiquem exatamente no ponto correto (vide figura 19). Isso é útil em casos em que o usuário digita um endereço sem todas as informações como por exemplo somente o nome da rua. Para que ele traga com mais precisão é preciso digitar o endereço o mais completo possível, como endereço + número + bairro + CEP + cidade.

Figura 19: Informações do novo local

Ao clicar em salvar o registro, volta-se para a tela inicial. Por último existe a possibilidade de alterar configurações básicas do aplicativo. Clicando no botão “mais” é mostrada a tela de mais opções (vide figura 20).

Figura 20: A tela de mais opções do aplicativo

Ao acessar Unidade de Distância é possível modificar a unidade de distância do cálculo no aplicativo. O padrão ao instalar o aplicativo é em milhas. Também é possível visualizar o módulo Sobre (vide figura 21).

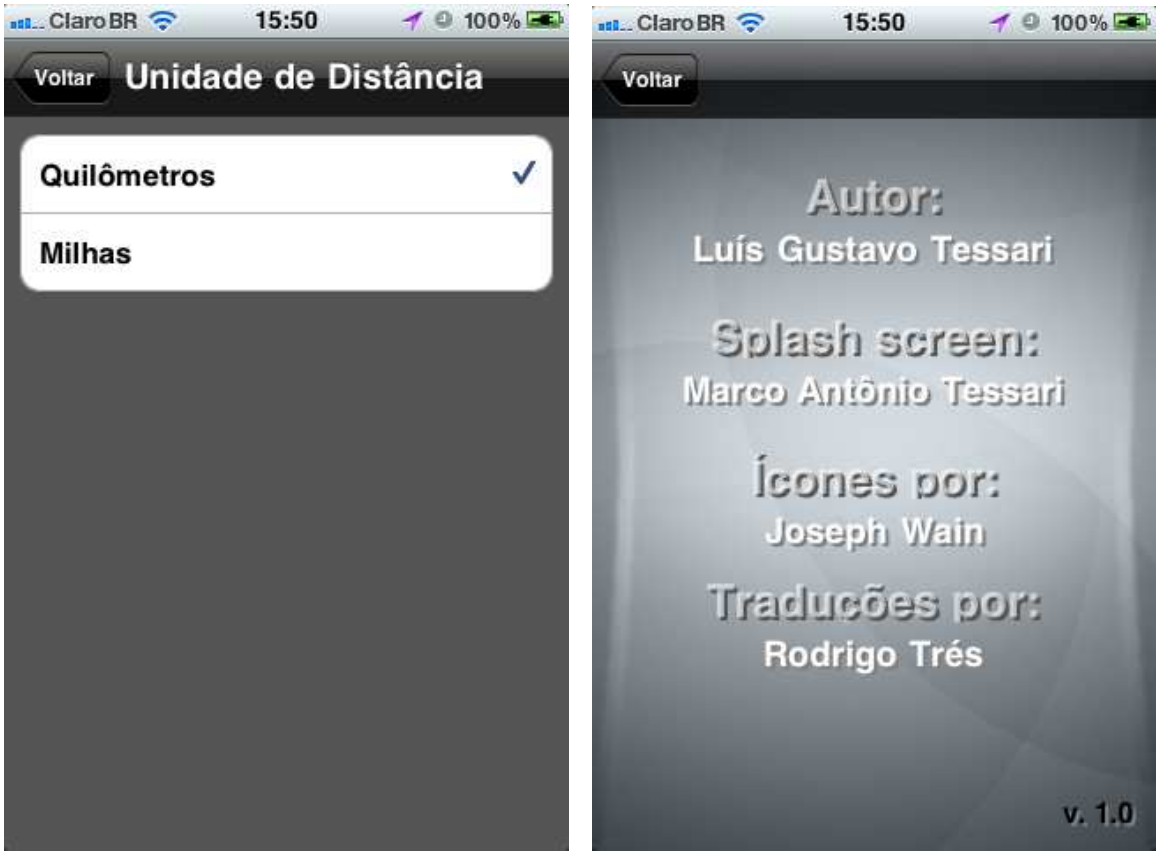

Figura 21: Módulos Unidade de Distância e Sobre respectivamente

5.4 Aplicação de Internacionalização

Com o desenrolar da implementação do aplicativo, surgiu a idéia de publicar na Apple Store. Como o iPhone possui e tem um apelo mundial, tornar a sua distribuição Internacionalizável permite uma abrangência muito maior de consumidores e de possível sucesso. O Finder GPS não possui muitas palavras e portanto agregar a ele vários idiomas não foi nada complexo. Qualquer aplicativo do iPhone procura pelo idioma padrão dentro da chave AppleLanguages do Sistema Operacional iOS (vide figura 22).

Figura 22: Lista de idiomas do iPhone

A sua primeira versão foi publicada e distribuída em 6 idiomas listados a seguir:

- Inglês (como sendo o idioma padrão);
- Português;
- Espanhol;
- Alemão;
- Italiano;
- Francês.

Mas ao transformar o aplicativo, surgiu outro pequeno detalhe. Deixar ele multi-idioma não engloba completamente o conceito explicado anteriormente. Não pelo menos

neste aplicativo. Como a idéia principal do Finder GPS é de poder mostrar a distância de onde o usuário está no momento de seus locais preferidos, ao tornar o software Internacionalizável existe uma peculiaridade. Na grande maioria dos países é utilizado o sistema métrico como unidade de distância. Porém, o maior público alvo deste aplicativo são os americanos e ingleses, cuja unidade de distância padrão é em milhas respectivamente.

Para corretamente torná-lo Internacionalizado foi criado uma opção a qual o usuário escolhe qual a unidade de distância que deseja que o aplicativo trabalhe, milhas ou quilômetros.

5.5 Imagens do aplicativo em outros idiomas

Nas figuras 23, 24 e 25 são apresentadas as telas do aplicativo em inglês, alemão, espanhol, italiano, português e francês respectivamente.

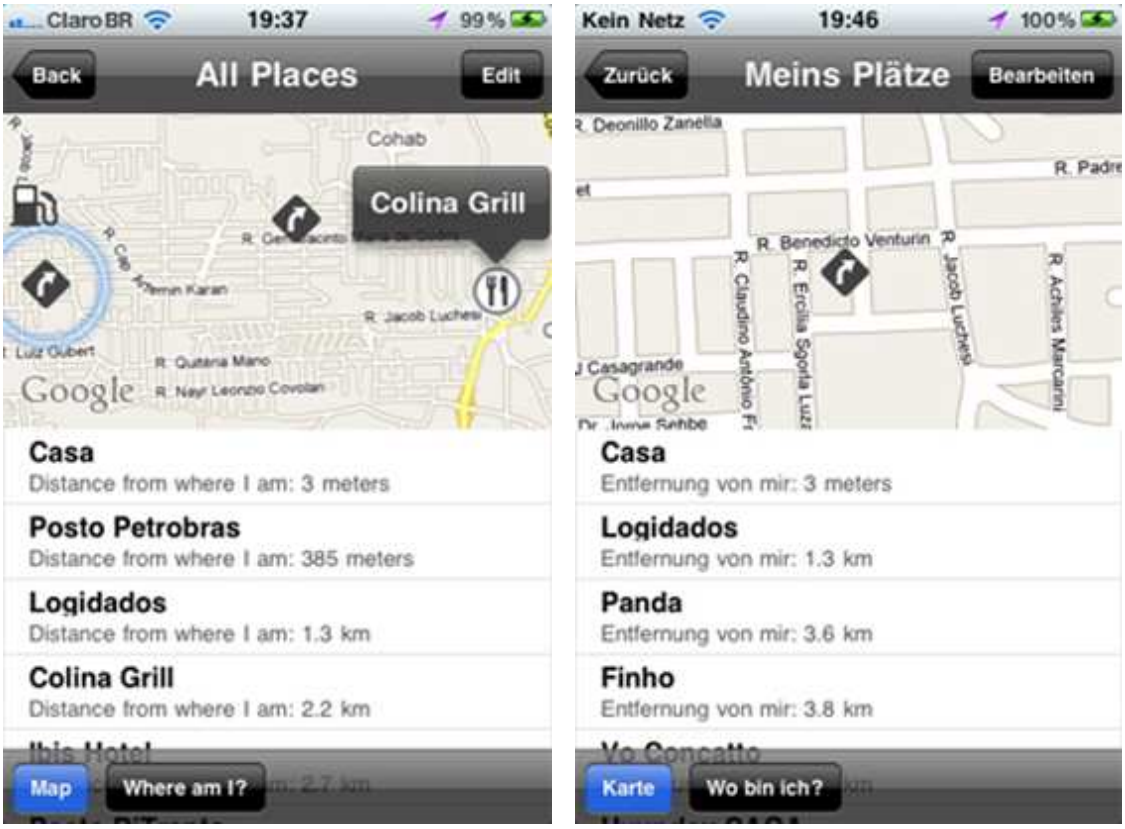

Figura 23: Aplicativo em inglês e alemão

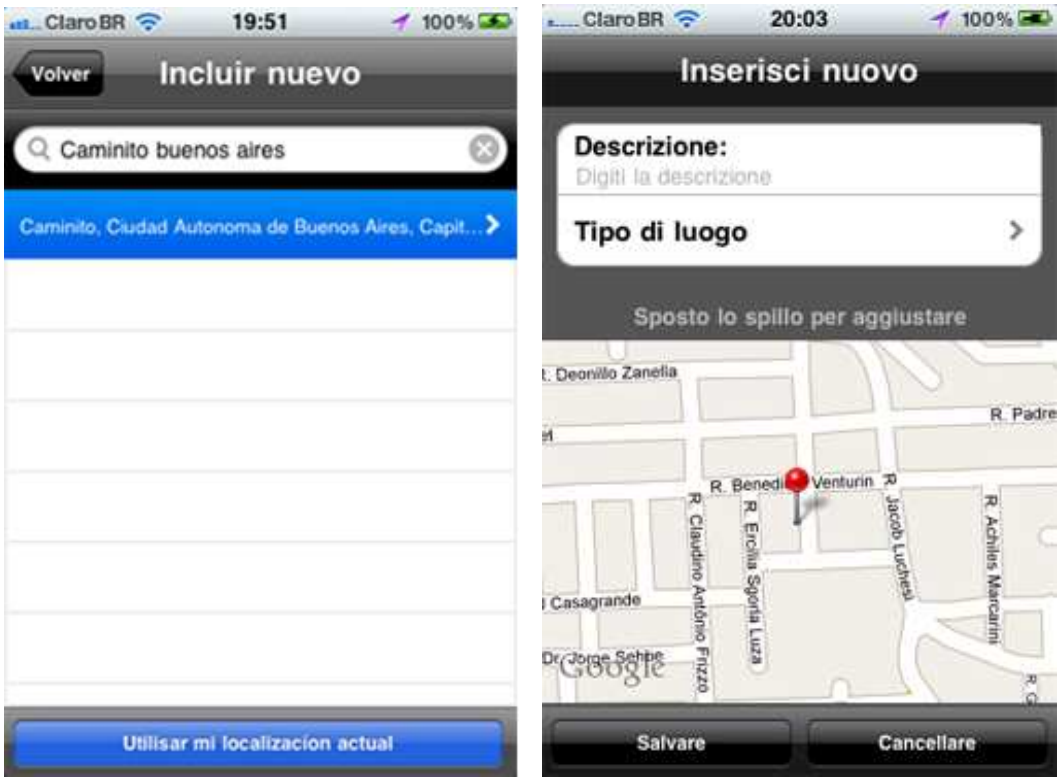

Figura 24: Aplicativo em espanhol e italiano

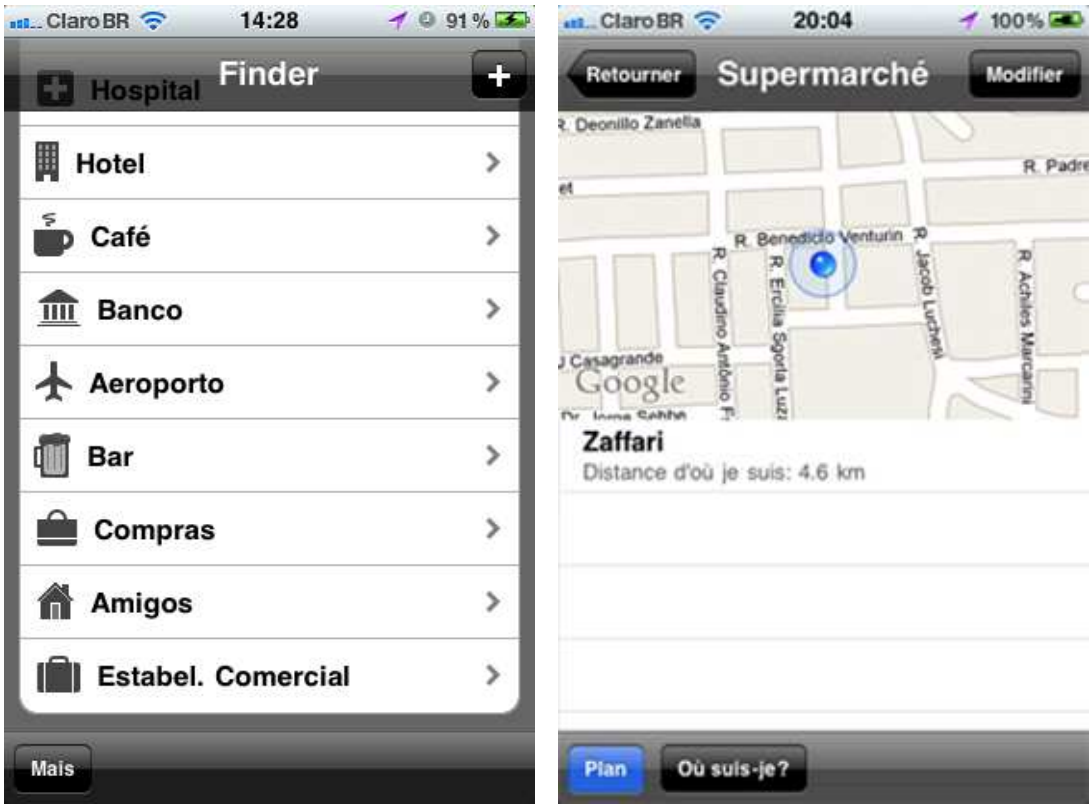

Figura 25: Aplicativo em português e francês

5.6 Dificuldades enfrentadas

Não foi uma tarefa simples desde o início este trabalho. Foi tentado baixar seis versões diferentes da imagem do sistema operacional Mac OS X e instalar em um computador com arquitetura Intel. Não foi obtido sucesso para instalar o primeiro pré-requisito após mais ou menos 10 dias. Foi necessário a aquisição de um Mac Mini e de um dvd com o *upgrade* para o Mac OS X Snow Leopard. Este processo todo até conseguir instalar todos os softwares necessários e escrever a primeira linha de código levou mais de 20 dias.

Mas este foi apenas o primeiro passo. O desenvolvimento em Objective-C também é complicado. A sua forma de programar baseada em Small Talk soube ser bastante problemática. Na verdade todo o desconhecido enfrentado neste projeto foi um desafio, e por momentos foi levantado a dúvida se existiria o tempo hábil para concretizar ele por completo.

5.7 Distribuição do aplicativo na Apple Store

Ao finalizar a implementação do Finder GPS ainda tinha mais um passo a realizar: a publicação e distribuição do aplicativo através da Apple Store. Uma tarefa e tanto. Não é algo simples e é preciso realizar vários passos para ter sucesso e finalmente conseguir postar o aplicativo pronto.

Ao entrar no programa de desenvolvimento para iOS, a Apple disponibiliza acesso a um novo site, específico para manutenção e acompanhamento de estatísticas de vendas dos aplicativos do desenvolvedor. Este site é o http://itunesconnect.apple.com.

Ao colocar um aplicativo para distribuição é preciso compilar ele com algumas variáveis de configuração diferentes no XCode, assim como utilizar o certificado de distribuidor registrado disponibilizado pela Apple assim que se entra no programa de desenvolvimento.

Após isto é preciso entrar no site da iTunes Connect e ir na opção *Manage Your Applications* (vide figura 26).

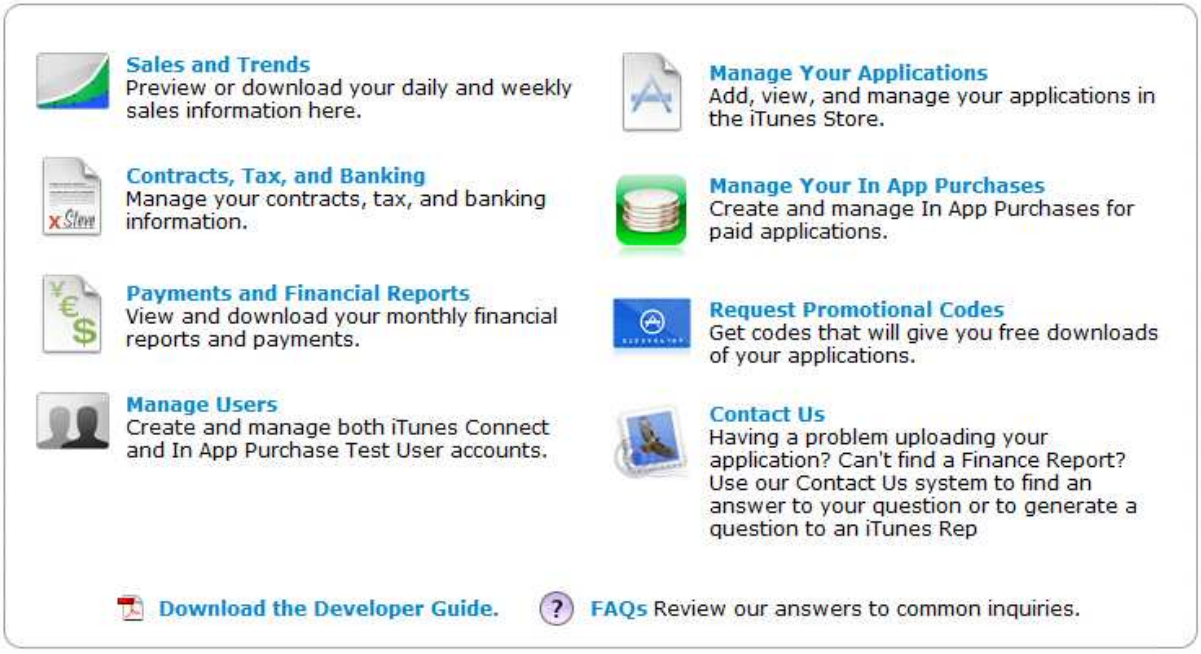

Figura 26: Opções disponíveis no iTunes Connect

É possível disponibilizar um aplicativo de graça, o que é na verdade bem fácil, ou pago, aí sim com bastante burocracia. Para o Finder GPS foi escolhido vender por U$ 0,99, o que no final foi uma verdadeira dor de cabeça. É preciso mandar documentos registrados por cartório para São Paulo, falar com representantes da Apple e ter bastante paciência. O processo todo levou quase 1 mês inteiro até tudo estar ok.

Após a Apple liberar para o desenvolvedor a possibilidade de realizar o *upload*, é preciso criar que um novo aplicativo será disponibilizado. Lá é informado descrições, tags, em que categorias ele se encaixa e as imagens que irão ser disponibilizadas na Apple Store. Ao informar isto, é preciso realizar o upload do arquivo binário compilado para distribuição com o uso do *Application Loader*, um programa fornecido pela Apple. Geralmente o processo de revisão e aprovação leva em torno de 7 dias, se tudo estiver nos conformes.

O Finder GPS foi postado na iTunes Connect no dia 22 de Novembro e entrou na loja somente no dia 02 de Dezembro. Na imagem 27 é mostrado o aplicativo disponível na Apple Store.

O aplicativo pode ser acessado através do seguinte link:
http://itunes.apple.com/us/app/finder-gps/id405126068?mt=8

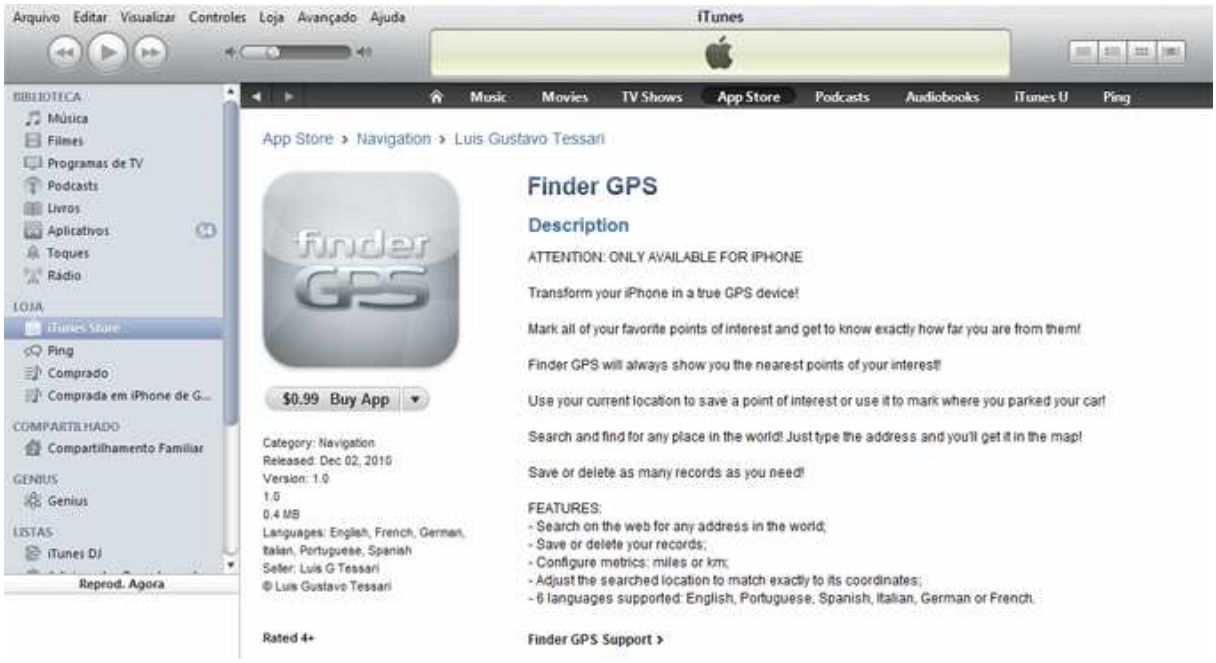

Figura 27: Finder GPS disponível na Apple Store

6 CONCLUSÃO

Este trabalho de conclusão de curso teve como principal objetivo demonstrar o grande potencial em se desenvolver aplicativos voltados para a telefonia celular, um mercado muito inexplorado por aqui.

Realmente não é algo banal e como dito anteriormente neste trabalho, só se descobre os obstáculos de implementar e distribuir o mesmo quando realmente começa a mexer com ele. O conhecimento adquirido com o andamento deste projeto em relação a vários conceitos e tecnologias novas foi bastante positivo.

O exemplo de programar para o iPhone, onde milhões de aparelhos foram vendidos no mundo, e a enorme força que a Apple Store possui, permite a desenvolvedores comuns a tentar o sucesso com aplicativos as vezes simples mas de extrema utilidade.

Gradativamente este setor irá ganhar um foco maior pelo desenvolvedor em âmbito mundial, visto sua potencialidade e a tendência é que nos próximos anos este mercado evolua com passos largos e com muitas novidades.

6.1 Trabalhos futuros

Para este projeto para um trabalho futuro, podem-se citar várias melhorias que não foram possíveis agregar nesta versão. Em futuras atualizações do Finder GPS podem ser citados os seguintes itens:

- Implementação de busca por registros previamente cadastrados;
- Associação de registros com o cadastro de contatos do usuário;
- Módulo de backup dos dados para envio por email e pontos de restauração dos dados;
- Estudar a possibilidade de cadastrar e rastrear localização de pessoas conhecidas através de um website, com o mesmo mecanismo de redes sociais com o convite e aceitação da pessoa;
- Estudar a possibilidade de mostrar o melhor caminho pelas ruas;
- Agregar o uso da bússola para saber qual a direção que o usuário está seguindo.

Também há de se ressaltar que este projeto permitiu o surgimento de novas e inovadoras idéias para desenvolver novos aplicativos em um futuro próximo. O embasamento teórico adquirido irá permitir a criação de novos aplicativos e completamente diferentes com

uma eficiência muito maior.

7 REFERÊNCIAS BIBLIOGRÁFICAS

ALLAN, Alasdair, Learning iPhone Programming. Sebastopol, CA: O'Reilly Media Inc. 2010.

ALLEN S., GRAUPERA V., LUNDRIGAN L. Pro Smartphone Cross-Platform Development: iPhone, Blackberry, Windows Mobile and Android Development and Distribution. Apress 2010.

APPLE, Apple's App Store Downloads Top Two Billion. Disponível em: http://www.apple.com/pr/library/2009/09/28appstore.html, Acessado em 28/10/2010.

APPLE DEVELOPER, iOS Dev Center. Disponível em: http://developer.apple.com/support/ios/ios-dev-center.html, Acessado em 02/12/2010.

BENNETT G., ANTE W., ASH M., JACKSON B., MIX N., PETERSON S., ROSENFELD M. iPhone Cool Projects. 1. ed. Berkeley, CA, EUA. Apress, 2009.

COOPER, Peter. 12 SQLite Resources For iPhone Developers. Disponível em: http://mobileorchard.com/iphone-sqlite-tutorials-and-libraries/, Acessado em 28/10/2010.

DUVANDER Adam. Map Scripting 101: A Guide to Building Interactive Maps and Mashups with Bing, Yahoo!, and Google Maps. No Starch Press, 2010.

FIRTMAN, Maximiliano. Programming the Web Mobile. 1. ed. 1005 Gravenstein Highway North, Sebastopol, CA, EUA. O'Reilly Media, 2010.

FONTANA, Sandro Paulo. Sistema de Posicionamento Global GPS A Navegação do Futuro. 2. ed. Porto Alegre, RS, Brasil. Mercado Aberto, 2002.

GOKHALE, Aniruddha. Reinventing the Wheel? CORBA vs. Web Services. Disponível em: http://www2002.org/CDROM/alternate/395/, Acessado em 29/10/2010.

KOCHAN, Stephen G. Programming in Objective C 2.0. Pearson Education Inc., 2009.

NEWCOMER, Eric. Understanding Web Services: XML, WSDL, SOAP and UDDI. Pearson Education Inc., 2004.

OWENS, Michael. The Definitive Guide to SQLite. Apress, 2006.

RABBANY, Ahmed El-. Introduction to GPS: The Global Positioning System. Norwood, MA: Artech House Inc. 2003.

ROCHA, José Antônio. GPS: Uma Abordagem Prática - 4.ed. rev. e ampl / 2003.