UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

TÔ AQUI: APLICATIVO PARA GEORREFERENCIAMENTO EM AMBIENTES RESTRITOS

DIEGO TONDIM ROCHA

BLUMENAU
2015

2015/1-09

DIEGO TONDIM ROCHA

TÔ AQUI: APLICATIVO PARA GEORREFERENCIAMENTO EM AMBIENTES RESTRITOS

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Ciência da Computação do Centro de Ciências Exatas e Naturais da Universidade Regional de Blumenau como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Prof. Aurélio Faustino Hoppe, Mestre – Orientador

BLUMENAU
2015

2015/1-09

TÔ AQUI: APLICATIVO PARA GEORREFERENCIAMENTO EM AMBIENTES RESTRITOS

Por

DIEGO TONDIM ROCHA

Trabalho de Conclusão de Curso aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II pela banca examinadora formada por:

Presidente:	Prof. Aurélio Faustino Hoppe, Mestre – Orientador, FURB
Membro:	Prof. Dalton Solano dos Reis, Mestre – FURB
Membro:	Prof. Francisco Adell Péricas, Mestre – FURB

Blumenau, 6 de julho de 2015

Dedico este trabalho a todos que de alguma forma contribuíram para a realização deste.

AGRADECIMENTOS

A Deus, por guiar as minhas escolhas.

A minha família, pelo apoio e incentivo que me concederam em todos os momentos, especialmente aos meus pais, Paulo e Rosi, a minha mulher, Taiani Correia Pereira Duarte, por todo o amor, apoio e compreensão.

Aos amigos e colegas de trabalhos, todos que de alguma forma contribuíram para que fosse possível realizar este trabalho.

Aos amigos da faculdade que acompanharam a ideia desde o início e compartilharam conhecimentos, experiências e informações de grande ajuda.

Ao professor Dalton Solano, pela ideia de trabalhar com os Beacons e todo o apoio durante a pesquisa e o desenvolvimento.

Ao meu orientador, Aurélio Hoppe, pelo apoio emocional, pela tranquilidade transmitida nos momentos de dificuldade, e principalmente pelo conhecimento compartilhado possibilitando a realização deste projeto.

A mente que se abre a uma nova ideia jamais voltará ao seu tamanho original.

Albert Einstein

RESUMO

Este trabalho apresenta um aplicativo que visa aprimorar o georreferenciamento em ambientes restritos permitindo ao usuário locomover-se pelos ambientes e visualizar a sua localização através de um mapa, disponibilizado em dispositivos móveis com os sistemas operacionais Android e iOS. O aplicativo exibe um mapa do Google Maps e executa a integração com o AGPS e os Beacons para determinar a localização do usuário. Também foi utilizado o recurso de síntese de voz para prover acessibilidade ao aplicativo. Todos os recursos foram implementados através de *plugins* escritos em JavaScript. Foi utilizado o *framework* Cordova, que possibilitou o uso de HTML5, Javascript e CSS na aplicação sem a necessidade de utilização de linguagens específicas do sistema operacional. A partir dos experimentos realizados, comprovou-se que o aplicativo facilita a locomoção dos usuários por um ambiente interno, conforme a avaliação dos usuários, todos conseguiram chegar ao local determinado utilizando as informações de localização e as notificações de eventos geradas pelo aplicativo e, 90% avaliaram o aplicativo como bom.

Palavras-chave: Dispositivos móveis. Beacon. Cordova. Multiplataforma. AGPS.

ABSTRACT

This work presents an application that aims to improve the georeferencing in restricted environments and your location using a map made available on mobile devices running Android and iOS operating systems. The application displays a map from Google Maps and performs integration with AGPS and Beacons to determine the user's location. It was also used the speech synthesis feature to provide accessibility to the application. All features have been implemented through plugins written in JavaScript. Cordova framework was used, which allowed the use of HTML5, JavaScript and CSS in the application without the need to use operating system specific languages. Through the experiments, it was found that the application facilitates the mobility of users in an indoor environment, as rated by users, all managed to arrive at the designated location using the location information and event notifications generated by the application, 90 % assessed the application as good.

Key-words: Mobile devices. Beacon. Cordova. Cross-platform. AGPS.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AGPS - *Assisted Global Positioning System*

AoA - *Angle of Arrival*

AP - *Access Point*

API – *Application Programming Interface*

BLE - *Bluetooth Low Energy*

GPS - *Global Positioning System*

RSSI - *Received Signal Strength Indicator*

TDOA - *Time Difference of Arrival*

WBLS - *Wireless Based Location System*

WIFI - *Wireless Fidelity*

SUMÁRIO

1 INTRODUÇÃO

Atualmente com o advento de novas tecnologias e a constante evolução nos estudos de localização, os meios utilizados também evoluíram e receberam melhorias, tornando-os versáteis e mais precisos (GATTERMANN, 2013, p. 15). Os dispositivos móveis (celulares, *smartphones*, *tablets*), exemplificam este avanço, onde hoje é possível reproduzir áudio, reproduzir vídeo, fotografar, armazenar grandes massas de dados, realizar comunicação entre dispositivos e até mesmo saber onde está através dos recursos de georreferenciamento dos dispositivos. Este último é um dos recursos que está sendo cada vez mais explorado, principalmente quando se diz respeito à localização física do usuário, seja no carro, no trânsito engarrafado precisando de uma rota alternativa, até mesmo dentro de um Shopping procurando uma loja.

A partir disso, pode-se observar que o ambiente da universidade é passível para aplicar a ideia de localização em ambientes restritos, pois é um ambiente descentralizado, amplo, com vários eventos ocorrendo no mesmo momento todos os dias. Porém, os atuais mecanismo s de localização não fornecem a posição exata do usuário. O *Global Positioning System* (GPS) possui algumas limitações quando se está em ambientes internos e utilizando dispositivos móveis, o consumo de bateria e o bloqueio de sinal são alguns dos problemas frequentemente relatados. O GPS então, muitas vezes não consegue fornecer a precisão necessária para realizar uma navegação interna. Nos dispositivos móveis o recurso possui uma melhoria chamada de *Assisted Global Positioning System* (AGPS), que é a união dos recursos de GPS, rede sem fio e rede de celular móvel para aumentar a disponibilidade e principalmente diminuir o tempo de resposta da localização do usuário. Uma forma de diminuir estas limitações seria usar o AGPS associado ao uso do Beacon, buscando aumentar a qualidade da localização do usuário em relação a pontos mapeados neste espaço interno.

Através dos recursos de hardware disponíveis nos dispositivos móveis atuais, é possível desenvolver aplicativos para qualquer plataforma que utilize os mesmos, seja iOS, Android, Windows Phone, porém, existe o custo para desenvolvimento para cada uma. O custo e tempo de desenvolvimento pode ser reduzido utilizando uma plataforma de desenvolvimento única, e que permite realizar o *deploy* para várias plataformas distintas. Uma das plataformas mais difundidas atualmente, é o *framework* Cordova da Apache, com a distribuição Phonegap. Utilizar o Phonegap permite que seja utilizada uma única linguagem de programação, que seria o Javascript, e usufruir de todos os recursos de hardware disponíveis nos dispositivos, através de *plugins* disponibilizados pela comunidade.

Diante do exposto, foi desenvolvido um aplicativo que permite navegar pela FURB, através de uma referência em um mapa representando a universidade. Para tanto, o diferencial do aplicativo, é a busca pelo aprimoramento da localização em ambientes restritos, utilizando o dispositivo Beacon para auxiliar os recursos de localização existentes nos dispositivos móveis e disponibilizar para as plataformas iOS e Android.

1.1 OBJETIVOS

O objetivo deste trabalho consiste em desenvolver um aplicativo que possibilite a navegação pelas dependências da FURB, assim como receber notificações dos locais onde o usuário está no momento.

Os objetivos específicos do trabalho são:

a) aprimorar a localização em ambientes restritos com o intuito de obter mais precisão, através dos Beacons e o AGPS;
b) disponibilizar um aplicativo multiplataforma para dispositivos móveis, usando o *framework* Cordova;
c) facilitar a percepção do usuário em relação a sua localização, através de um mapa que represente o ambiente da universidade e um indicador do ponto em que o usuário se encontra.

1.2 ESTRUTURA

O trabalho foi dividido em quatro capítulos: no primeiro, consta uma introdução sobre o assunto. O segundo capítulo apresenta a fundamentação teórica necessária para o desenvolvimento do aplicativo, passando pelo funcionamento do dispositivo *Bluetooth*, o padrão iBeacon e o *framework* do Cordova. O terceiro capítulo traz informações sobre o desenvolvimento do trabalho, requisitos, especificação, implementação, operacionalidade e resultados. No quarto capítulo são relatadas as conclusões do trabalho e suas possíveis extensões.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo está organizado em três seções: a seção 2.1 tem por objetivo esclarecer os conceitos relacionados a georreferenciamento, georreferenciamento em ambientes restritos utilizando dispositivos móveis e o dispositivo Bluetooth Beacon. A seção 2.2 apresenta como é feito o desenvolvimento multiplataforma com o Cordova. Ao final, a seção 2.3 apresenta os trabalhos correlatos.

2.1 GEORREFERENCIAMENTO

Georreferenciamento é ter uma imagem ou mapa, ou qualquer outra forma de representação geográfica e tornar suas coordenadas conhecidas num dado sistema de referência (BRUM, 2006). Normalmente quando o assunto é georreferenciamento, fala-se de localização de algo ou alguém. Segundo Michaelis (2014), uma das definições da palavra seria "[...] Determinação, por meio do radar ou outros aparelhos, da posição exata de aviões ou submarinos, amigos ou inimigos.", nesse caso estaríamos tratando de diferentes meios para determinar a posição do elemento. No mundo real, conforme ensinado em sala de aula, temos o conceito definido por Mendonça (2013 apud GATTERMANN, 2013, p. 20) "As coordenadas geográficas permitem a localização num determinado ponto do globo terrestre em relação à linha do equador ou ao meridiano de Greenwich; e a localização cartográfica permite localizar um ponto num plano cartográfico, após a escolha de uma projeção cartográfica".

Nas próximas seções serão explanados os meios disponíveis nos dispositivos móveis atualmente para georreferenciamento, desde o AGPS que é a combinação do GPS, Rede de Celular e *Wireless Fidelity* (WIFI) para aumentar a disponibilidade do recurso, assim como descreve Broering (2010) "Sistema AGPS (*assisted*-GPS, em português GPS assistido) foi desenvolvido para alcançar lugares que o GPS se mostrava muito suscetível a indisponibilidade, como em grandes centros urbanos, em ambientes indoor, como edificações e carros", até atualmente os Beacons usando Bluetooth.

2.1.1 AGPS

O AGPS tem como função fornecer informações ao aparelho que recebe o sinal GPS, sem aprimorar sua precisão. O que pode ocorrer algumas vezes é a assistência fornecer ao aparelho informações que o levam mais rapidamente a posição precisa, sendo assim, seria um equívoco dizer que o AGPS é mais rápido que o GPS, ele apenas utiliza os recursos de rede de

celular e rede sem fio disponíveis para melhorar a disponibilidade do recurso (BROERING, 2010).

O Sistema de Posicionamento Global (GPS) é uma das tecnologias mais utilizadas atualmente para localização, sendo o principal componente do AGPS. O GPS funciona com 27 satélites ao redor da Terra. O exército americano desenvolveu essa rede de satélites como um sistema de navegação militar, mas logo a disponibilizou às para uso civil. As órbitas propiciam que sempre haja pelo menos quatro satélites "visíveis" no céu. A função de um receptor GPS é localizar quatro ou mais desses satélites, determinar a distância para cada um e utilizar esta informação para deduzir sua própria posição (BRAVO; BASTOS, 2014).

A rede sem fio IEEE 802.11, que também é conhecida como rede Wi-Fi, é um padrão que opera em faixas de frequências que permite que dispositivos como os notebooks, celulares, *smartphones*, *tablets*, possam se conectar à internet sem a necessidade de cabos físicos. Ela pode determinar a localização de um dispositivo pela localização do ponto de acesso, e a força do sinal, podendo utilizar também mais de um ponto de acesso para aumentar a precisão da localização (GATTERMANN, 2013).

As técnicas de localização mais empregadas atualmente nos sistemas baseados em redes Wi-Fi podem ser divididas em grupos. As que utilizam do ângulo de chegada do sinal de RF, conhecida como *Angle of Arrival* (AoA), utilizadas em sistemas que usam triangulação com antenas direcionais. Para estimar a posição, eles utilizam de três pontos de referência no raio de acesso ao dispositivo, calculando o ângulo e o tamanho das arestas do triângulo formado. Técnica semelhante é a triangulação, onde são necessários dois pontos distintos de medição em relação a um ponto fixo para estimar a posição do dispositivo. As que se baseiam na potência do sinal de Rádio Frequência (RF), *Received Signal Strength Indicator* (RSSI) mapeando o ambiente para posterior comparação em tempo real, com um banco de dados pré-armazenado. Por fim, as técnicas que utilizam *Time Difference of Arrival* (TDOA) onde a diferença do tempo de propagação do sinal de RF é medida para estimar a posição do dispositivo (FAGUNDES, 2008).

O princípio das redes celulares é baseado na subdivisão de uma área geográfica, coberta pela rede, num número de pequenas subáreas denominadas células. Em cada célula, existe uma estação fixa que atua como um transmissor-receptor e que serve todas as estações móveis, localizadas na vizinhança da célula, cada célula possui um identificador, e uma estação base que cobre um conjunto de células. (SVERZUT, 2008 apud GATTERMANN, 2013, p. 22).

2.1.2 Beacon

O *Bluetooth* é um meio utilizado para a comunicação entre dispositivos móveis, computadores, entre outros dispositivos eletrônicos, utilizando sinais de radiofrequência com alcance de 1m, 10m ou 100m. A frequência é baixa e consequentemente a potência é baixa assim como o custo. Atualmente maioria dos dispositivos eletrônicos que se tem contato no cotidiano possui *Bluetooth*, desde um *smartphone*, *notebook*, computador, *tablet* até uma geladeira. A comunicação entre os dispositivos com *Bluetooth* ocorre através do processo de emparelhamento, transferindo os dados com segurança (TANEMBAUM, 2011 apud GATTERMANN, 2013 p. 25).

O *Bluetooth* está na versão 4.1, que comparado as versões antigas está bem mais estável e veloz. No caso de localização em ambientes internos utilizando *Bluetooth*, os dispositivos devem estar dentro de uma área de alcance do dispositivo móvel e auxiliam na localização, possibilitando assim melhorar a precisão.

Bluetooth Low Energy (BLE) é uma tecnologia relativamente nova, originalmente lançada pela Nokia em 2006 sob o nome Wibree. Em 2007, o *Bluetooth Special Interest Group* (SIG) anunciou que a especificação Wibree se tornaria parte do *Bluetooth Core Specification* Version 4.0 como o que hoje é conhecido como *Bluetooth Low Energy*. O baixo consumo de energia da tecnologia BLE e sua capacidade de se conectar a *smartphones* tornam mais adequado para dispositivos de baixa potência, como monitores de frequência cardíaca, computadores de ciclismo e termômetros. É também uma tecnologia adequada para uso nas chamadas casas inteligentes, pois ele se comunica usando transmissões de rádio na frequência de 2.4 - 2.48ghz. O BLE não sofre com os mesmos problemas como outras tecnologias utilizadas em casa. O infravermelho em controles remotos é um exemplo, pois necessitam de linha de visão para o dispositivo que está sendo controlado. Um resultado interessante é que o Bluetooth utiliza ondas de rádio e a intensidade do sinal pode ser utilizada para calcular a proximidade entre dois dispositivos *Bluetooth* (ANDERSSON, 2014, p. 1, tradução nossa).

O Beacon é um dispositivo BLE, que envia mensagens para os dispositivos para possibilitar a identificação e o cálculo da distância entre os dois. O alcance dos Beacons é de 50 metros, média da tecnologia BLE, alguns fabricantes recomendam 10 metros, sendo assim, ter-se-ia uma cobertura de 100m² para cada dispositivo, o custo/benefício nesse ponto pensando em ambientes grandes seria considerável, além de que, o espaço ocupado por esse dispositivo é mínimo, como pode-se ver na Figura 1.

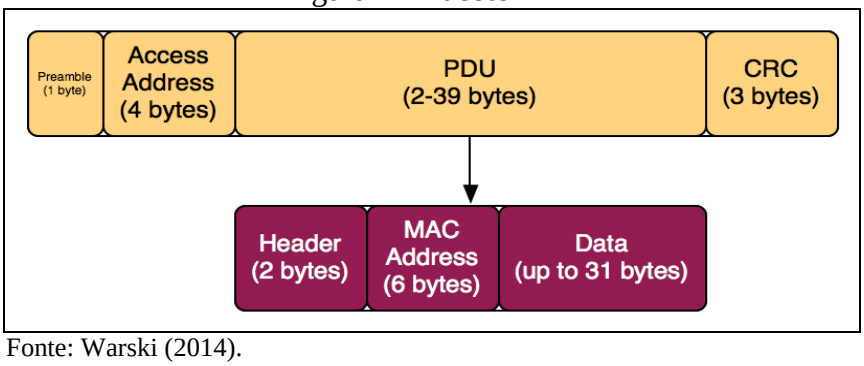
Figura 1 – Beacon StickнFind

Os dispositivos com BLE possuem dois papéis, um é chamado de *central* e o outro de *peripheral*. Pode-se comparar a estrutura Cliente-Servidor, onde o dispositivo *central* seria o cliente e o *peripheral* o servidor. Nesse caso o Beacon é um dispositivo *peripheral* enquanto o dispositivo móvel é *central*. Para os dispositivos *peripherals* serem descobertos, emitem um pacote de *broadcast* em um tempo determinado (normalmente em milissegundos). Eles disparam os pacotes a todo momento, mesmo quando não existem outros dispositivos na área (ANDERSSON, 2014, p. 2, tradução nossa).

Para encontrar um dispositivo *peripheral*, o *central* procura através dos pacotes recebidos e após descobrir o dispositivo ele estabelece conexão com o *peripheral*. Se a conexão é estabelecida com sucesso, os dados são lidos através do *Generic Attribute Profile* (GATT). Na maioria dos casos não é necessário estabelecer conexão entre os dispositivos, os dados do pacote que o *peripheral* dispara (nome do dispositivo, informação especifica do fabricante) são o suficiente para o primeiro contato. Não são permitidas múltiplas conexões do Beacon para outros dispositivos móveis, porém, um dispositivo móvel pode estar conectado a vários Beacons, e o Beacon não consegue notificar outros dispositivos enquanto estiver conectado (ANDERSSON, 2014, p. 2, tradução nossa).

O pacote transmitido pelos dispositivos BLE tem 47 bytes, dispostos conforme a Figura 2 (WARSKI, 2014, tradução nossa).

Figura 2 – Pacote BLE



Fonte: Warski (2014).

O PDU tem seu próprio cabeçalho (2 bytes: tamanho da carga útil e seu tipo - se o dispositivo suporta conexões, etc.) e a carga real (até 37 bytes). Os primeiros seis bytes de carga útil são o endereço MAC do dispositivo, e a informação real pode ter até 31 bytes. Os 31 bytes variam conforme o fabricante do dispositivo. (WARSKI, 2014, tradução nossa).

Existem várias aplicações possíveis para os Beacons. Mais comumente estão sendo utilizados no mercado varejista para notificar usuários de *smartphones* sobre informações de produtos dentro do ambiente em que o cliente está conforme o setor. Também é possível aplicar em feiras e eventos, disponibilizando informações sobre os expositores, mostrando que é um dispositivo que permite ser explorado para vários ramos diferentes.

2.1.3 iBeacon

Os iBeacons usam somente o canal de notificações. Como o nome "Beacon" sugere, eles transmitem pacotes de dados em intervalos regulares, e esses dados podem então ser apanhados por dispositivos como *smartphones*. iBeacons são simplesmente dispositivos que seguem um padrão específico para notificações usando BLE, com alguns recursos adicionais na plataforma iOS (WARSKI, 2014, tradução nossa). No Quadro 1 um exemplo do pacote de dados transmitido no padrão iBeacon conforme a fabricante Estimote.

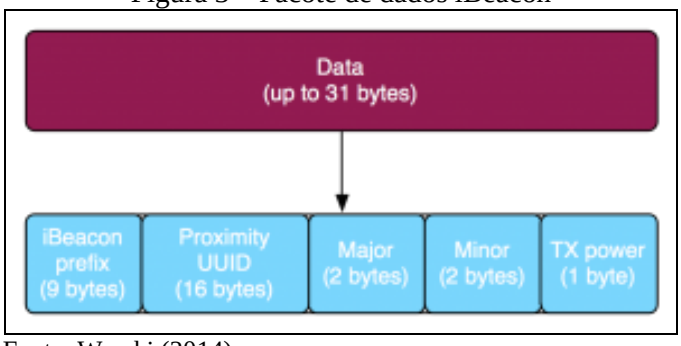
Quadro 1 – Exemplo pacote de dados iBeacon

1	`02 01 06 1A FF 4C 00 02 15: iBeacon prefix (fixed)`
2	`B9 40 7F 30 F5 F8 46 6E AF F9 25 55 6B 57 FE 6D: proximity UUID`
3	`00 49: major`
4	`00 0A: minor`
5	`C5: 2's complement of measured TX power`

Fonte: Warski (2014).

O padrão iBeacon define como os 31 bytes de dados da PDU serão organizados, na Figura 3 pode-se visualizar como é organizado o pacote, que conforme a especificação utiliza apenas 30 bytes.

Figura 3 – Pacote de dados iBeacon



Fonte: Warski (2014).

Os nove primeiros bytes são utilizados para identificar o padrão iBeacon. O UUID é um identificador que deve ser usado para distinguir os dispositivos dos outros. O número

maior (2 bytes) é usado para agrupar um conjunto relacionado de Beacons. Por exemplo, todos os Beacons de uma loja irão ter o mesmo número. Dessa forma, o aplicativo vai saber em qual loja o cliente está. O número menor (2 bytes) é usado para identificar Beacons individuais. Cada um em uma loja terá um número diferente, de modo que identifique o local que o dispositivo está na loja. O campo final, intensidade de TX, é usado para determinar a distância que você está do Beacon. Isto pode ser apresentado, de forma simplificada (imediata / longe / fora de alcance) ou como uma medição mais precisa em metros. A intensidade de TX, é a força do sinal medido a 1 metro do dispositivo RSSI. À medida que a força do sinal diminui à medida que aumenta a distância, sabendo a RSSI a 1 metro, e o RSSI atual (temos a informação em conjunto com o sinal recebido), é possível calcular a diferença. A plataforma iOS tem este recurso, para outras plataformas deve ser implementado manualmente. Obstáculos como móveis, pessoas ou congestionamento de comunicação podem enfraquecer o sinal. Por conseguinte, a distância é apenas uma estimativa. (WARSKI, 2014, tradução nossa).

2.2 DESENVOLVIMENTO MULTIPLATAFORMA COM CORDOVA/PHONEGAP

No projeto é utilizado o Phonegap, uma das distribuições mais disseminadas do Cordova, que é o núcleo da biblioteca mantido pela Apache (PALUDO, 2014). O Phonegap é capaz de transformar um aplicativo *web*, desenvolvido nos padrões para web, em um aplicativo nativo, um aplicativo desenvolvido especialmente para uma determinada plataforma. Atualmente o Phonegap possibilita a criação de projetos para as principais plataformas, entre elas: iOS, Android e Windows Phone (BERLIM; REICHERT; TEODORO, 2013).

Este *framework* permite ao desenvolvedor criar soluções nativas, independentes de plataforma, utilizando um serviço de empacotamento de aplicações chamado PhoneGap Build. Desta forma, o aplicativo pode ser configurado universalmente para executar e trocar informações com diferentes dispositivos e sistemas operacionais (MUNRO, 2012 apud SILVA, 2013, p. 2).

O Phonegap é um *framework* de desenvolvimento de aplicações multiplataforma para dispositivos móveis que permite a utilização de tecnologias Web Padrão como HTML5, CSS3 e Javascript evitando linguagem de desenvolvimento nativa das plataformas móveis. Os aplicativos são executados em pacotes direcionados para cada plataforma e dependem de padrões compatíveis com sua *Application Programming Interface* (API) para acessar recursos de cada dispositivo (ADOBE, 2014).

Na Figura 5 é apresentada a arquitetura do framework Cordova. A camada superior representa o código fonte da aplicação. A camada central é composta por bibliotecas Javascript e nativas, sendo esta camada responsável pela interface entre a aplicação Web e a aplicação nativa (SANTOS; SILVA, 2014, p. 4).

Figura 4 – Camadas da arquitetura do *framework* PhoneGap

Fonte: adaptado de Santos e Silva (2014).

O Phonegap é composto basicamente pela camada de aplicação WEB onde a interface do usuário é renderizada, as camadas mais internas da API Javascript e nativa, onde concentram-se as classes e padrões que abstraem os sistemas operacionais. A organização em camadas permite aos desenvolvedores manter um padrão de desenvolvimento, caso exista algum recurso não disponível na API, existe a possibilidade de extensão do *framework* para atender à necessidade, e por último o acesso nativo ao sistema operacional para o qual está sendo desenvolvido o aplicativo.

O Cordova deve ser instalado utilizando o Node.js, que deve ser configurado previamente na máquina. Na linha de comando do Node.js, o comando `npm install -g cordova` para instalar o Cordova. Após a instalação do Cordova, via linha de comando é possível criar projetos, gerenciar plataformas e *plugins* e fazer o *deploy* do aplicativo (PALUDO, 2014).

A criação de um projeto utilizando o Cordova, na linha de comando do Node.js inicia-se com o comando `cordova create`, seguido pelos parâmetros da pasta onde a aplicação deve ser armazenada, de uma identificação de domínio reverso (opcional) e do nome da aplicação (opcional). Exemplo: `cordova create hello com.example.hello HelloWorld` (APACHE, 2015).

É necessário incluir pelo menos uma plataforma móvel para que o *build* do aplicativo seja possível. Para isso, deve-se acessar a pasta do projeto e então adicionar as plataformas desejadas com o comando `cordova platform add` seguido por uma das plataformas disponíveis: ios, android, entre outros. Exemplo: `cordova platform add android`. Com pelo menos uma plataforma adicionada já é possível executar a aplicação, através de um emulador ou de um dispositivo da plataforma escolhida (APACHE, 2015).

O projeto criado deve conter uma pasta `www`. No projeto criado consta o arquivo `index.html` que é chamado como página inicial por padrão ao executar a aplicação. Existem outras pastas que são criadas por padrão para organizar os arquivos JavaScript, CSS e entre outros (APACHE, 2015). O *build* é feito através do comando `cordova build`. É possível também gerar o *build* de uma plataforma específica adicionando o nome da plataforma no final do comando: `cordova build android` (APACHE, 2015). Para executar a aplicação, é utilizado o comando `cordova run` seguido pela plataforma. Exemplo: `cordova run android`. Este comando faz o *deploy* da aplicação no emulador caso nenhum dispositivo móvel da plataforma informada esteja disponível. Caso ele identifique um dispositivo móvel conectado, o comando fará o *deploy* no dispositivo (APACHE, 2015).

2.3 TRABALHOS CORRELATOS

A seguir serão descritos dois trabalhos e um produto com características semelhantes aos principais objetivos deste trabalho. A seção 2.3.1 detalha “Técnicas de Localização de Dispositivos Móveis em Redes Wifi – TDOA” de Fagundes (2008). Na seção 2.3.2 é descrito o produto “BinarioBeacon” da BinarioMobile (2014), a seção 2.3.3 descreve a dissertação “WBLS: um sistema de localização de dispositivos móveis em redes Wi-Fi” de Moura (2007). Por fim, uma análise entre os trabalhos correlatos na seção 2.3.4.

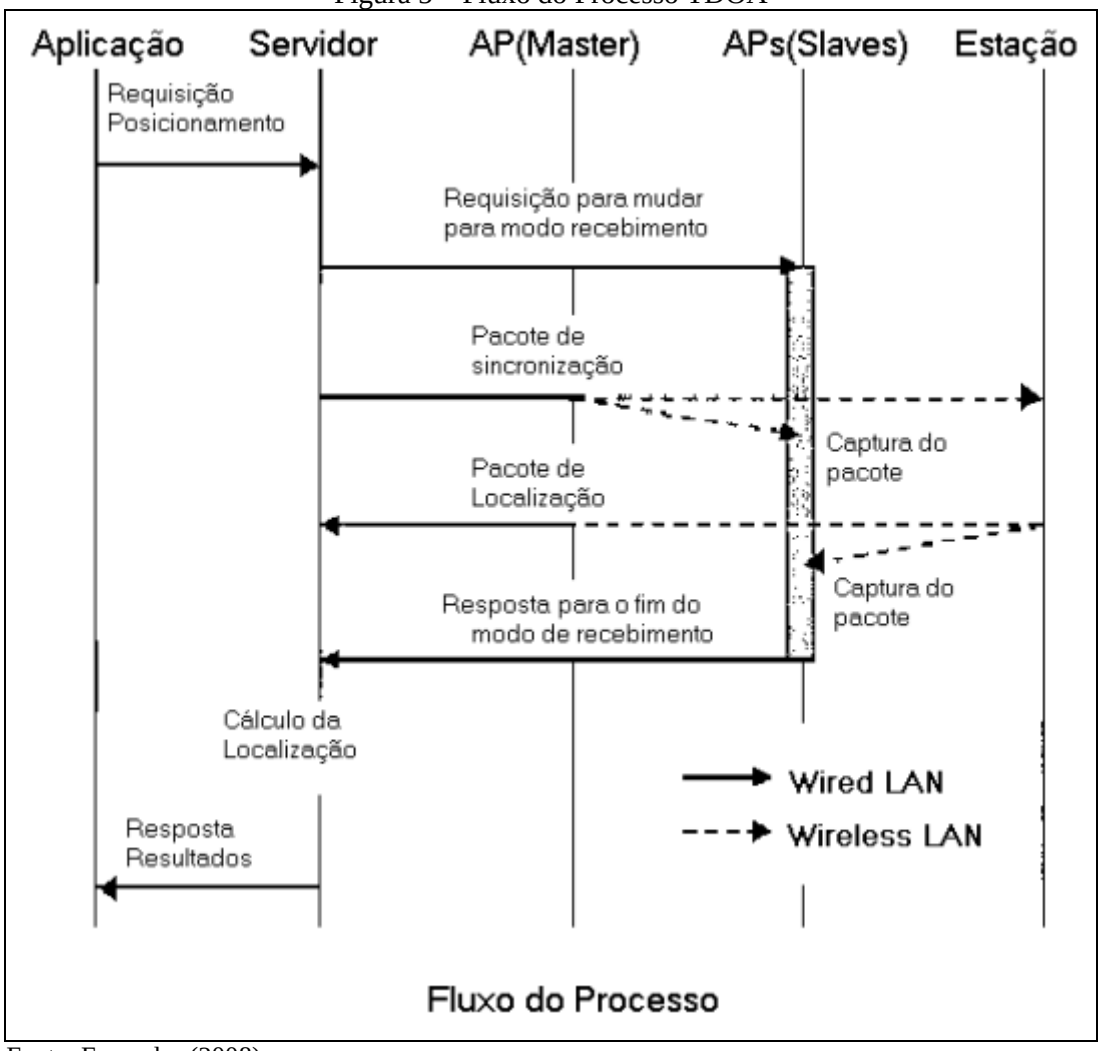
2.3.1 Técnicas de Localização de Dispositivos Móveis em Redes Wifi – TDOA

O trabalho desenvolvido por Fagundes (2008) descreve as técnicas mais utilizadas de localização baseada em redes sem fio. Utiliza como parâmetro os ambientes internos, visando obter desde a localização do dispositivo móvel dentro da rede, até melhorar a segurança na rede interna considerando conexões de distâncias maiores do que a do ambiente de uso da rede.

A principal técnica abordada por Fagundes (2008) é a TDOA, pois ela tem uma característica específica, como descreve Fagundes (2008, p. 8) “[...] realiza um

monitoramento do tempo que o sinal de Rádio Frequência (RF) leva entre o transmissor e o receptor e, por este motivo, vem contemplar um requisito importantíssimo relacionado à segurança do sistema [...]". A Figura 5 representa o fluxo de operação do TDOA.

Figura 5 – Fluxo do Processo TDOA



Fonte: Fagundes (2008).

O fluxo apresentado na Figura 5 inicia com a aplicação solicitando a localização atual ao servidor. O servidor envia uma requisição aos *Access Points* (APs) secundários, para mudar o modo de recebimento. Os APs recebem a requisição e mudam o modo de operação. O servidor envia um pacote de sincronização para a estação, os APs secundários recebem e capturam os pacotes de sincronização. A estação envia um pacote de localização para o servidor, então os APs secundários recebem e capturam os pacotes. Os APs secundários enviam o final do modo de captura e enviam o pacote de dados ao servidor. O servidor calcula baseado nos pacotes de sincronização e localização a posição da estação, e o servidor retorna o resultado para a aplicação (FAGUNDES, 2008).

A vantagem da técnica de TDOA em relação às outras técnicas apresentadas no trabalho de Fagundes (2008), é que ela ao estimar a posição mediante o tempo de propagação do sinal, pode detectar tentativas de conexão ou invasão de áreas fora do perímetro de cobertura da rede sem fio. Outra vantagem da técnica, é que ela não depende de um mapeamento prévio e armazenamento num banco de dados de informações.

Segundo Fagundes (2008), das diversas técnicas testadas, um sistema de localização de dispositivos móveis pode fornecer eficaz segurança ao sistema e ainda vir a complementar um sistema *Wireless Intrusion Detection Systems*, pois auxiliará além da identificação de acessos indesejados, a localização de dispositivos que compõem a própria rede, detectando outros dispositivos estranhos ou indesejados à rede sem fio.

2.3.2 BinarioBeacon

BinarioBeacon é uma plataforma nova no mercado brasileiro, que possui um *web* site para gerenciamento dos locais e um aplicativo para *smartphones* e *tablets*. Através do sistema de gerenciamento, é configurado o mapa do local, e também as mensagens e notificações que o aplicativo deve exibir. O princípio do produto é o dispositivo Beacon, desenvolvido sobre a plataforma Java e com versões dos aplicativos para iOS, Android e Windows 8 (BINARIOMOBILE, 2014).

BinarioBeacon tem algumas aplicações como base para os clientes, como por exemplo, a navegação em aeroportos, os Beacons podem guiar o usuário através do aeroporto com a visualização de uma planta do aeroporto e enviar alertas quando o avião estiver disponível para embarque, além de permitir que a companhia aérea localize os passageiros que ainda não embarcaram. A Figura 6 demonstra a imagem do aplicativo apresentado pela empresa BinarioMobile (BINARIOMOBILE, 2014).

Figura 6 – Aplicativo BinarioBeacon

Fonte: adaptada de Lopes (2014).

Na Figura 6 é possível visualizar o mapa interno do aplicativo, e a indicação da localização atual do usuário conforme os portões de embarque no aeroporto. Caso o usuário deseje chegar em algum portão especifico, a aplicação exibe a rota para o usuário. Todas as informações são configuradas pelo site do sistema de gerenciamento.

2.3.3 WBLS: um sistema de localização de dispositivos móveis em redes Wi-Fi

A dissertação de Moura (2007) explora o *Wireless Based Location System* (WBLS), que, diferente de outras técnicas, não considera apenas uma probabilística. É utilizado um mapa com as informações de potência recebida RSSI e a frequência do sinal coletada de vários pontos de acesso. As redes sem fio possuem algumas falhas de transmissão devido a diversos fatores, com isso ocorrem ruídos nas transmissões que afetam as técnicas para determinar a localização. Sendo assim, o WBLS visa desconsiderar a frequência de presença de sinal no processo de estimativa, visando eliminar os ruídos.

O método WBLS proposto por Moura (2007) é baseado na execução do algoritmo do sistema básico, que é formado resumidamente por três componentes: o módulo de estimação, módulo de transição e módulo de observação. A proposta do WBLS é alterar o sistema básico, visando o módulo de observação. Os cálculos probabilísticos realizados nesse módulo seriam limitados ao produto das probabilidades apenas dos PAs com sinal, diferente do cálculo original, onde são considerados todos os PAs do mapeamento (MOURA, 2007).

Moura (2007, p.54) descreve WBLS:

> Seu diferencial é que ele procura lidar com o fato de que a informação obtida nas medições realizadas para a construção do mapa de RSSI sobre as frequências de presença de sinais pode não ser confiável. A modificação que ele propõe é uma nova forma para calcular as probabilidades de observação dado um estado.

A aplicação mostrou-se uma boa técnica para um ambiente com ruídos na rede e mais robusta para casos onde ocorrem falhas na transmissão ou perda de sinal dos pontos de acesso, sendo assim superior a algumas técnicas de localização para ambientes internos utilizando redes sem fio. A desvantagem é que a sua implementação é muito mais complexa, considerando que a estrutura existente normalmente não satisfaz a técnica, diferente da técnica de RSSI, custando muito mais, sendo viável em redes onde se deseja maior segurança, o uso da bateria também é comprometido, pois conforme Moura (2007), nos testes, foram considerados apenas 4 horas diárias, pois a bateria do celular era facilmente descarregada, necessitando parar para recarregar.

2.3.4 Análise e Comparação dos Trabalhos Correlatos

Os trabalhos correlatos apresentados nas seções 2.3.1, 2.3.2 e 2.3.3 demonstram pesquisas que visam melhorar a localização em ambientes internos, analisando cada um, é possível perceber três áreas nas quais podemos compará-los, assim possibilitando extrair fatores que reforçam a ideia da proposta apresentada. No Quadro 2 são comparados o custo, estrutura física do ambiente e a implementação.

Quadro 2 – Análise e comparação dos trabalhos correlatos

Trabalho	Custo	Estrutura Física	Implementação
Fagundes (2008)	Custo baixo quando existe uma rede pronta, caso seja necessário montar a rede sem fio, com os *Access Points* (AP), o sistema acaba se tornando caro.	Os APs devem ser colocados em locais estratégicos, pois como se trata de RF, o sinal pode sofrer interferências e prejudicar a estimativa de distância.	O aplicativo torna-se complexo, pois é necessário executar muitos controles a nível de hardware. O aplicativo comunica-se apenas com um servidor que dispara a rotina de localização para os APs que executam a técnica e retornam a posição.
BinarioMobile (2014)	Custo baixo do dispositivo, considerando que ele auxilia a melhorar a precisão dos outros recursos que o dispositivo móvel já possui, como o GPS.	A estrutura torna-se simples com o uso dos Beacons, mapeando o ambiente e colocando-os em locais estratégicos facilita a leitura e processamento das informações.	Desenvolvimento simplificado, pois com o uso do *framework* do fabricante os tratamentos a nível de aplicação são reduzidos.
Moura (2007)	Custo baixo assim como o trabalho da seção 2.3.1, pois utiliza redes sem fio existentes.	A estrutura exige APs dispostos em locais estratégicos assim como o trabalho apresentado na seção 2.3.1.	A implementação é mais complexa, pois é implementado um grafo para controlar a diferença de potência de sinal entre os APs, e não a diferença de frequência de sinal como a técnica da seção 2.3.1.

Conforme características apresentadas no Quadro 2, pode-se observar que as técnicas que utilizam as redes sem fio possuem custo baixo. Porém, são mais complexas de implementar e tem precisão aproximada de 2 metros. Já os aplicativos que utilizam o Beacon são mais fáceis de implementar e a precisão pode chegar a escala de centímetros, o dispositivo tem um custo elevado ainda, comparado a um roteador sem fio por exemplo, porém o tamanho do dispositivo é menor, e o custo para manutenção é mais baixo, considerando que a bateria de um Beacon pode durar na média de 9 anos conforme modelo e fabricante.

3 DESENVOLVIMENTO

Neste capítulo são apresentadas as etapas do desenvolvimento do aplicativo. A seção 3.1 traz os requisitos principais do trabalho. Na seção 3.2 consta a especificação, com modelos e diagramas. A implementação, com técnicas, ferramentas utilizadas e comentários sobre a operacionalidade da implementação estão na seção 3.3. A seção 3.4 detalha os resultados obtidos.

3.1 PRINCIPAIS REQUISITOS DO PROBLEMA A SER TRABALHADO

O aplicativo deverá atender os seguintes requisitos:

a) permitir que o usuário visualize sua localização dentro de um mapa interno (Requisito Funcional - RF);
b) permitir receber notificações conforme a localização dentro da universidade (RF);
c) implementar em Javascript com o *framework* PhoneGap (Requisito Não-Funcional - RNF);
d) disponibilizar o aplicativo para as plataformas iOS e Android (RNF);
e) utilizar o dispositivo Beacon para melhorar a precisão da localização (RNF).

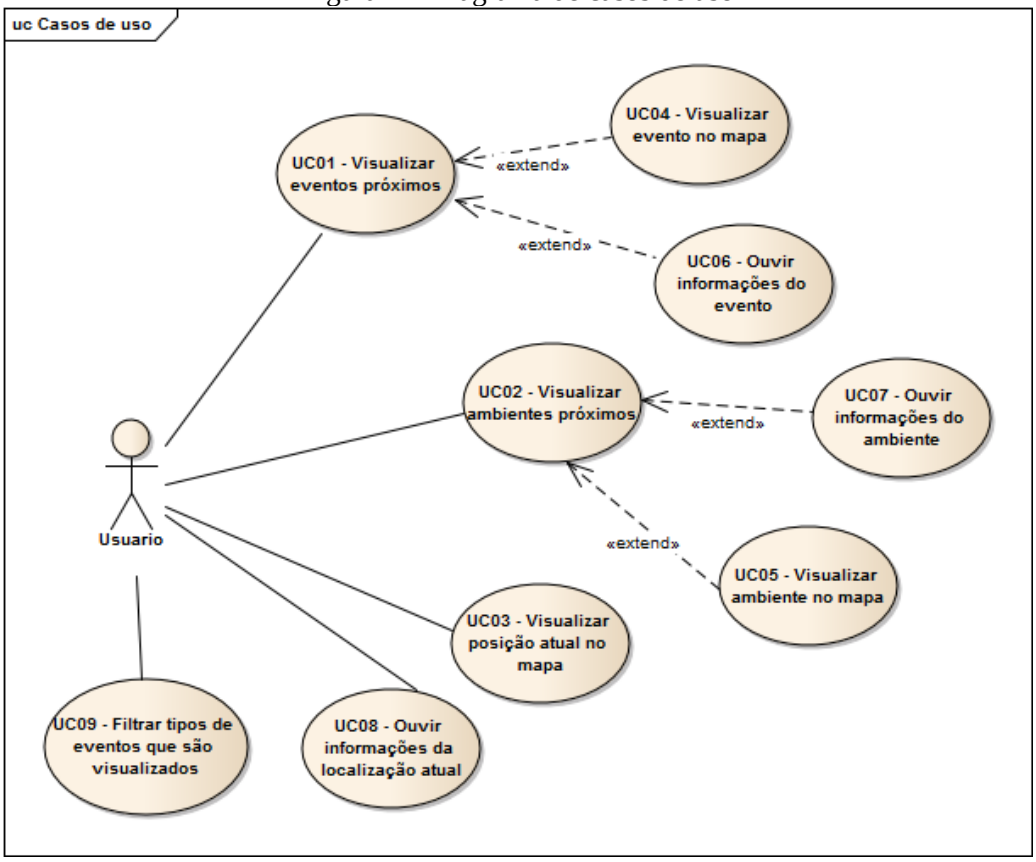
3.2 ESPECIFICAÇÃO

A especificação do aplicativo foi criada utilizando a ferramenta *Enterprise Architect* (EA). Neste trabalho foram elaborados os diagramas de classes, casos de uso e atividades utilizando o EA.

3.2.1 Diagrama de casos de uso

A Figura 7 exibe o diagrama de casos de uso com as ações disponibilizadas pelo aplicativo para o ator `Usuário`.

Figura 7 – Diagrama de casos de uso



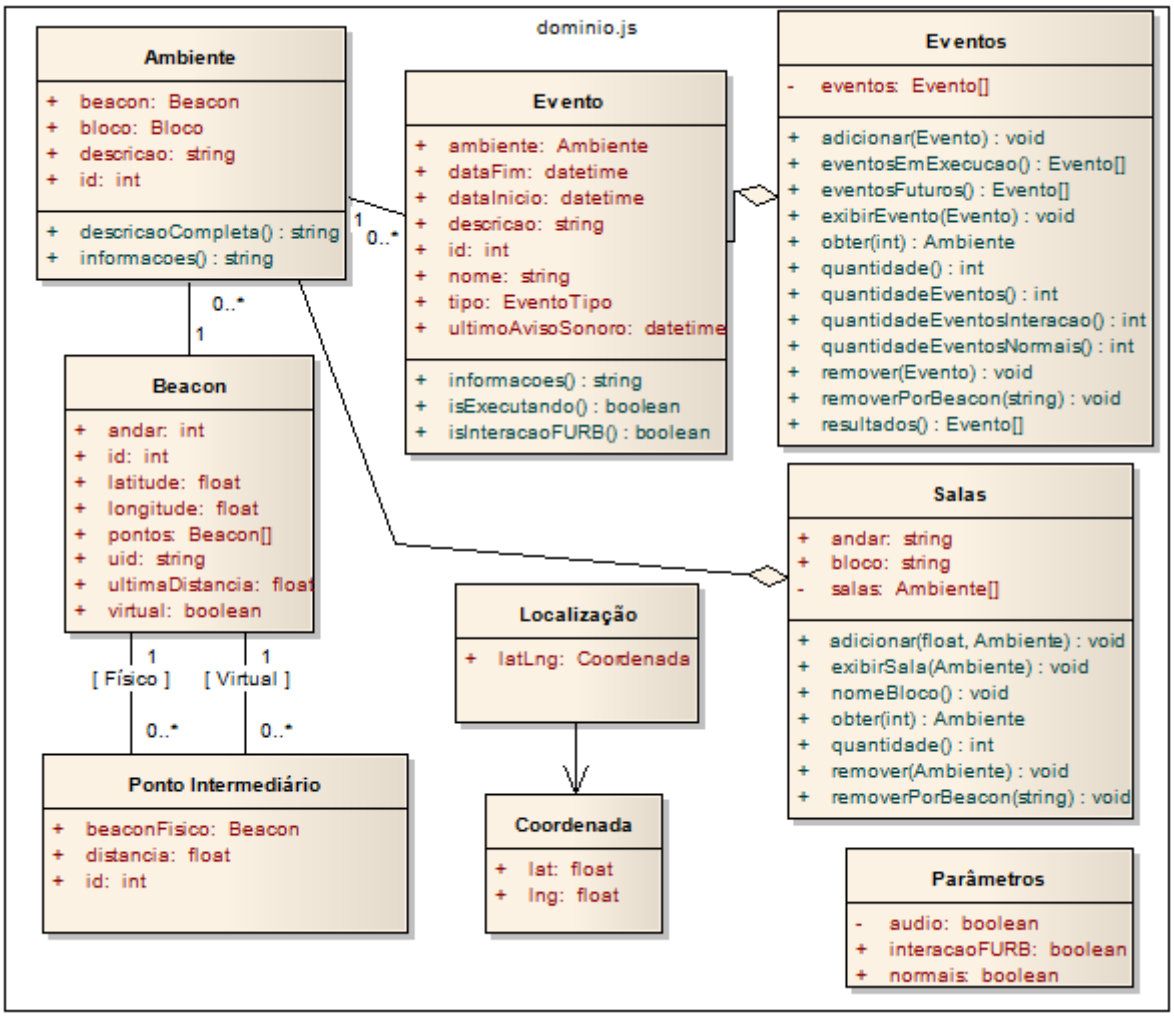
O caso de uso de `UC01 – Visualizar eventos próximos` é utilizado quando o usuário deseja visualizar os eventos que estão ocorrendo próximos a sua localização atual. No caso de uso `UC04 – Visualizar evento no mapa` ele pode selecionar um evento e verificar onde é a localização aproximada do mesmo, no caso de uso `UC06 – Ouvir informações do evento` ao selecionar o evento o aplicativo fala as informações do mesmo. No caso de uso `UC02 – Visualizar ambientes próximos` o usuário pode verificar quais ambientes estão próximos a sua localização atual. No caso de uso `UC07 – Ouvir informações do ambiente`, ao selecionar o ambiente o aplicativo fala informações do ambiente. No caso de uso `UC05 – Visualizar ambiente no mapa`, ao seleciona o ambiente o aplicativo exibe um marcador com a posição aproximada do ambiente. No `UC08 – Ouvir informações da localização atual`, o aplicativo fala a cada 1 minuto o bloco e andar que o usuário está, e logo após os eventos que já iniciaram e os que iniciarão na próxima 1 hora. No caso de uso `UC03 – Visualizar posição atual no mapa`, o aplicativo exibe um marcador no mapa com a sua posição, bloco e andar atual. No caso de uso `UC09 – Filtrar tipos de eventos que são visualizados`, o usuário pode escolher se os dois tipos de eventos suportados serão exibidos no aplicativo.

O detalhamento dos casos de uso da aplicação está no Apêndice A. Nele constam os objetivos, cenários e fluxos alternativos.

3.2.2 Diagrama de classes

Nesta seção constam os diagramas de classe do projeto, com relacionamentos e estruturas dos objetos. A Figura 8 mostra as classes de domínio do aplicativo.

Figura 8 – Diagrama de classes do domínio



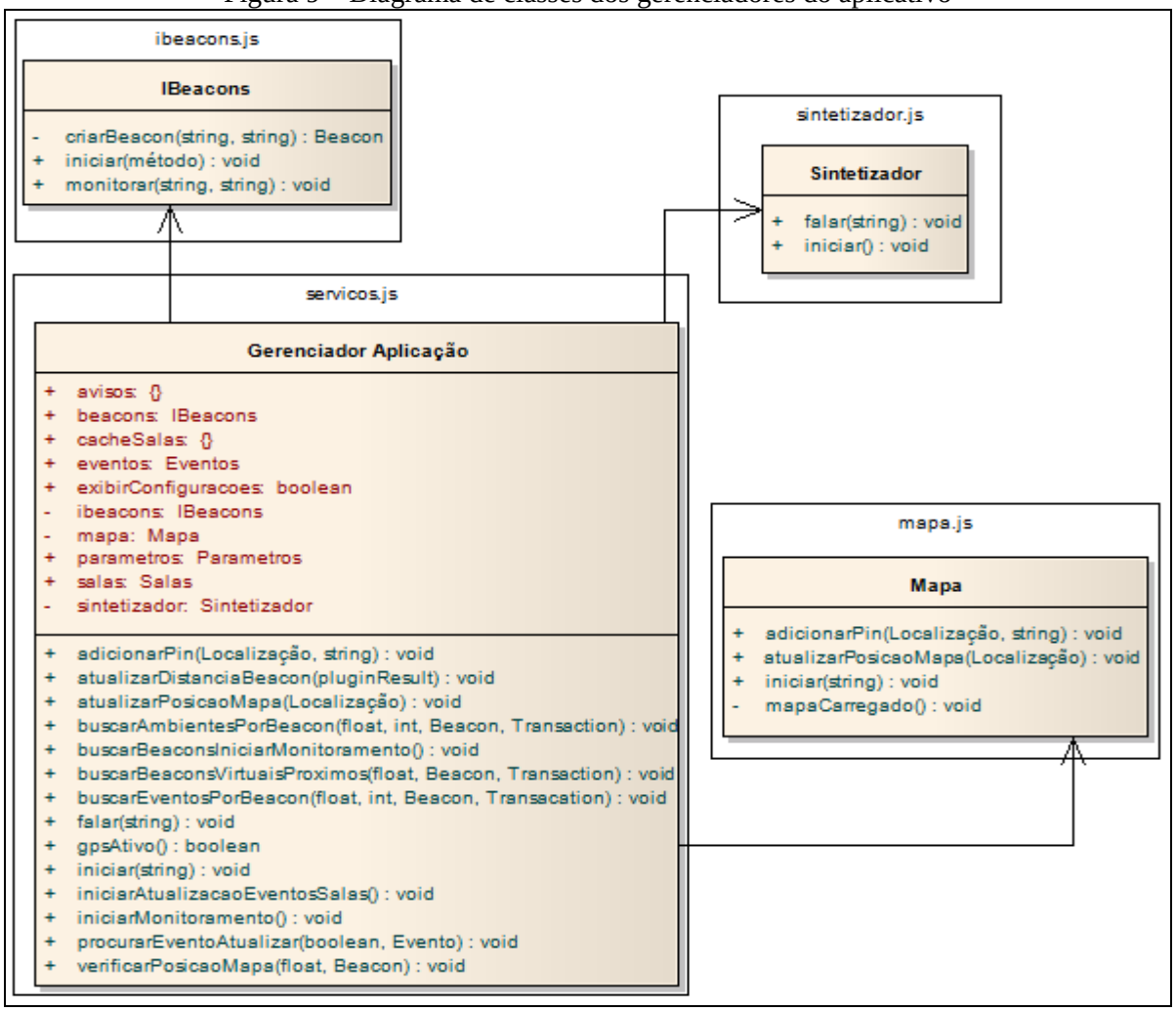
A Figura 8 exibe os modelos de domínio para a aplicação. As classes de domínio contêm as informações necessárias para a persistência dos dados no banco de dados SQLite, conforme a estrutura do Apêndice B. O domínio tem como base a classe Beacon, que possui as informações mapeadas de localização para cada dispositivo físico nos ambientes, através dos atributos latitude, longitude e andar. O atributo virtual permite classificar se o Beacon cadastrado é físico ou virtual, ou seja, será calculado conforme os pontos intermediários existentes.

A classe `Ambiente` contém um `Beacon`, sendo assim cada `Beacon` pode ter mais de um `Ambiente` associado. A classe `Evento` possui um `Ambiente`, possui os atributos de `dataInicio` e `dataFim` para controlar o período do `Evento`. Na classe `Evento`, é possível definir se o mesmo está em execução, se o evento é do tipo `InteracaoFURB`, e também obter o texto através do método informações para executar a síntese de voz.

As classes `Salas` e `Eventos` gerenciam as listas das respectivas classes, `Ambiente` e `Evento`, onde é possível adicionar ou remover um novo item, e automaticamente atualizar as informações na tela do aplicativo.

As classes do pacote de domínio são utilizadas pelo `Gerenciador Aplicação`, que inicia todos os *plugins*, e controla através das informações de cada Beacon, a atualização dos eventos, dos ambientes, do posicionamento do mapa e a execução da síntese de voz, conforme é demonstrado na Figura 9.

Figura 9 – Diagrama de classes dos gerenciadores do aplicativo



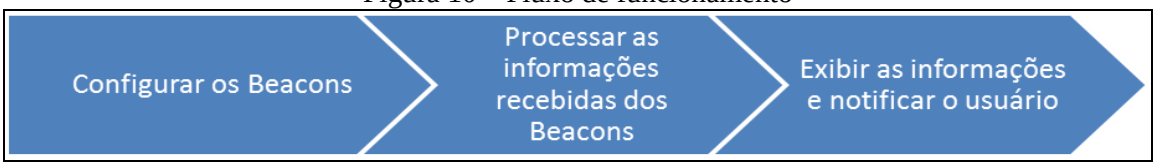
A classe `IBeacons` encapsula o *plugin* de integração com os Beacons, e o `Gerenciador Aplicação` contém uma instância do mesmo. A classe `Sintetizador`

encapsula o *plugin* de síntese de voz, através do processo controlado pelo `Gerenciador Aplicação`, o mesmo controla a instância do sintetizador. A classe `Mapa` encapsula o *plugin* de integração com o Google Maps, através dele é possível atualizar a posição atual do usuário e adicionar marcadores.

3.2.3 Diagrama de atividades

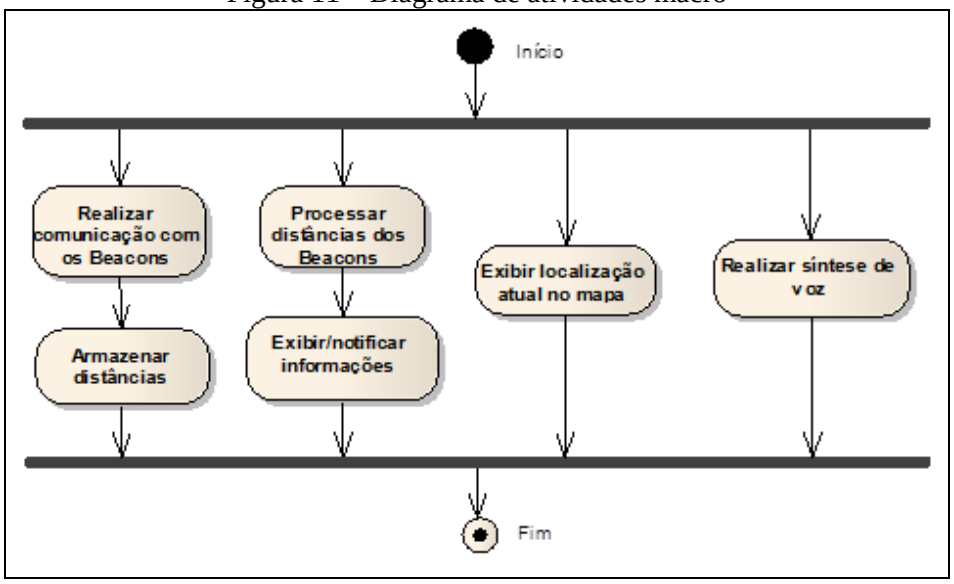
O fluxo representado na Figura 10 demonstra o processo com uma visão macro, desconsiderando os processos internos da aplicação.

Figura 10 – Fluxo de funcionamento



Conforme o fluxo apresentado a Figura 10, pode-se verificar que a primeira etapa que deve ser realizado é a configuração dos Beacons, pois isto influencia diretamente o comportamento da aplicação. Após a configuração, na segunda etapa, com a aplicação em funcionamento, as informações dos Beacons são recebidas e processadas, realizando a comparação com a base de dados interna. Por fim, na terceira etapa, as informações processadas são exibidas ao usuário através da interface do aplicativo e notificações. Na Figura 11 são demonstradas as duas últimas etapas do fluxo em um diagrama de atividades com visão macro.

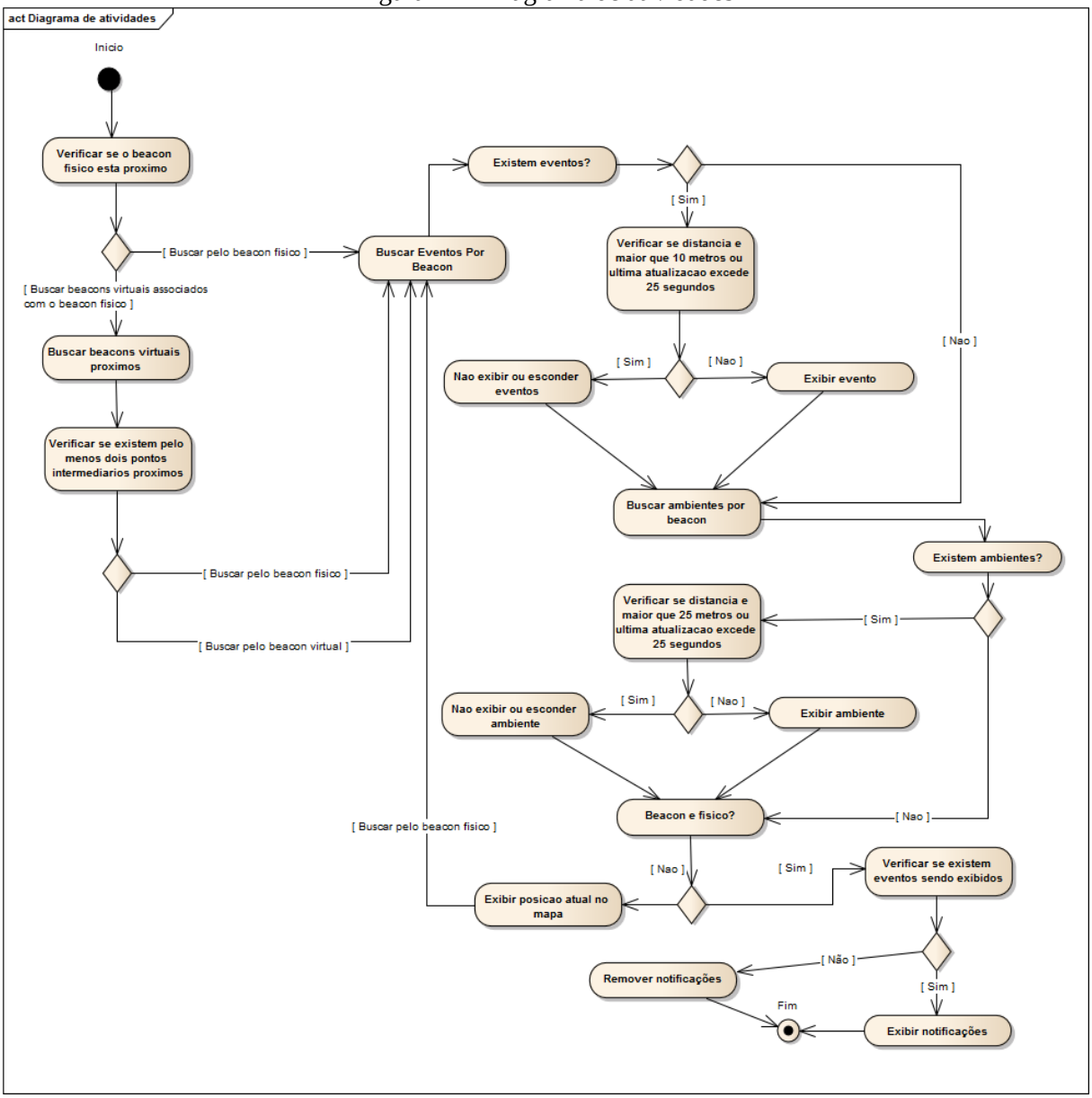
Figura 11 – Diagrama de atividades macro



O diagrama de atividades da Figura 11 demonstra os principais processos realizados pelo aplicativo. O aplicativo deve possuir quatro linhas de execução. A primeira linha de

execução deve receber as informações dos Beacons e armazenar em memória. A segunda linha de execução deve processar as distâncias e atualizar as informações em tela. A terceira linha de execução deve atualizar a posição do usuário no mapa, conforme as informações do GPS ou do Beacon mais próximo. A quarta linha de execução deve realizar a síntese de voz automática referente ao ambiente em que ele está. O diagrama de atividades representado na Figura 12 mostra o processo de atualização dos eventos e ambientes na tela do usuário, esse processo ocorre a cada 3 segundos e faz parte da segunda linha de execução, executando a iteração através dos Beacons cadastrados no aplicativo.

Figura 12 – Diagrama de atividades



A Figura 12 exibe o diagrama de atividades do processo de atualização dos eventos e ambientes em tela. O processo inicia quando o aplicativo itera sobre os Beacons físicos

cadastros na base de dados, o fluxo inicia verificando se o Beacon físico está próximo (qualquer distância reconhecida), sendo verdadeiro é iniciada a procura por Beacons virtuais associados a esse Beacon físico. Neste caso para um Beacon virtual existir é necessário que o usuário esteja em um ponto onde pelo menos dois Beacons físicos satisfaçam a condição determinada nos pontos intermediários cadastrados. Após validar o Beacon virtual, é possível exibir os eventos, ambientes e a localização atual do usuário conforme o Beacon virtual. Caso não seja encontrado um Beacon virtual, a aplicação faz o mesmo processo exibindo os eventos associados ao Beacon caso esteja em um raio de 10 metros, ou removendo os mesmo da tela, os ambientes são exibidos considerando um raio de 25 metros, caso o beacon esteja fora desse raio os ambientes são removidos da tela. Caso o Beacon não esteja próximo no início do processo, o restante é executado considerando apenas o Beacon físico, para possibilitar o aplicativo remover os eventos e salas que podem estar sendo exibidos. Após as validações, o resultado é processador e são exibidas as notificações de quantos eventos estão próximos ao usuário.

3.3 IMPLEMENTAÇÃO

A seguir são mostradas as técnicas e ferramentas utilizadas na implementação do aplicativo Tô Aqui, a descrição do desenvolvimento do trabalho e a operacionalidade do aplicativo.

3.3.1 Técnicas e ferramentas utilizadas

Para o desenvolvimento foi utilizada a linguagem Javascript, recursos do Android *Software Development Kit* (SDK) e do iOS SDK, do Cordova e os *plugins* que serão explanados nas próximas seções. A ferramenta para desenvolvimento utilizada foi o Notepad++, pois o código desenvolvido não necessitou de uma *Integrated Development Environment* (IDE) específica para uma plataforma, o aplicativo é multiplataforma, sendo possível compilar para Android e iOS.

Para definir a estrutura da aplicação e a comunicação entre o dispositivo móvel e o Beacon, foram pesquisadas e analisadas várias alternativas. Primeiramente foram realizados testes com o desenvolvimento de uma aplicação em iOS utilizando a SDK do fabricante do Beacon, permitindo verificar a complexidade da mesma e como seria tratada a comunicação à nível de aplicação entre os dois. Foi estudada uma aplicação Android utilizando a SDK do fabricante e realizados testes para verificar se o comportamento era o mesmo nos dois ambientes.

Através do conhecimento adquirido com os testes iniciais de integração e desenvolvimento, foi verificado que era possível fazer uma aplicação multiplataforma, desenvolvendo apenas uma interface. Sendo assim foi cogitado desenvolver um *plugin* para utilizar a SDK nativa do fabricante e criar uma camada acima para manipular através do código Javascript, porém, limitaria o uso a apenas um fabricante, sendo assim, foi pesquisada um *plugin* Cordova que atendesse o padrão iBeacon e multiplataforma, resultando no iBeacon Plugin desenvolvido por Peter Metz e mantido no GitHub. O mesmo foi escolhido pois está em constante atualização e possui muitas contribuições de usuários com sugestões e validações.

Foram utilizados outros *plugins* para Cordova visando complementar e aprimorar a experiência do usuário no aplicativo: Phonegap Google Maps Plugin para integração com o Google Maps, Cordova *Local Notification Plugin* para notificações e TTS Plugin para síntese de voz visando melhorar a acessibilidade do aplicativo.

Para a interface do aplicativo foi utilizado o JQuery Mobile, pois a biblioteca atende com eficácia as necessidades do aplicativo, e possui uma documentação extensa e detalhada sobre os recursos disponibilizados. A biblioteca Knockout também foi utilizada para a interface, ela permite realizar atualização das informações de uma forma mais simples e legível, facilitando o desenvolvimento.

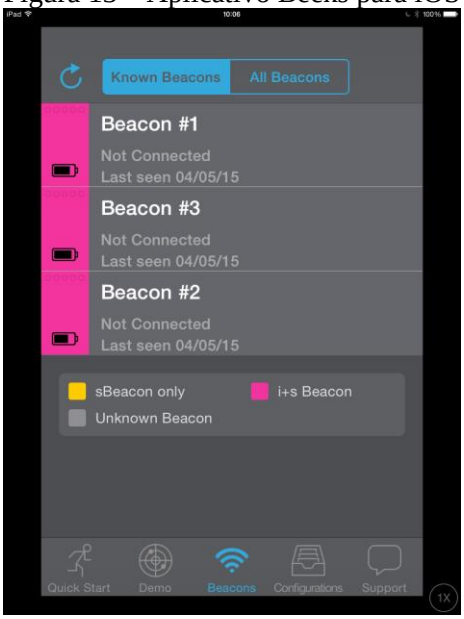
3.3.2 Etapas do desenvolvimento

A seguir serão detalhadas cada uma das implementações, conforme o fluxo apresentado na Figura 10 e o diagrama de atividades apresentado na Figura 11, que contemplam a configuração dos Beacons, integração com os Beacons, integração com o Google Maps, utilização de síntese de voz, notificações ao usuário e a integração com a interface da aplicação.

3.3.2.1 Configuração dos Beacons

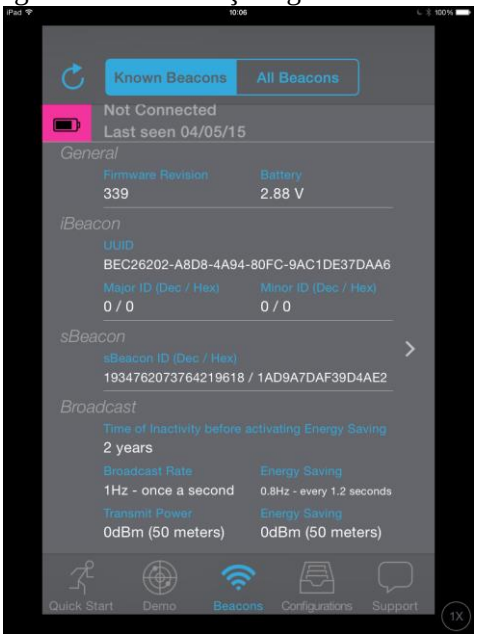
A configuração dos Beacons influencia diretamente o desempenho dos mesmos em relação ao aplicativo. Para realizar as alterações no hardware é necessário utilizar o aplicativo Beeks para iOS, disponibilizado pela fabricante StickNFind por e-mail após a compra dos dispositivos. A Figura 13 apresenta a tela inicial do aplicativo onde pode-se visualizar os Beacons já configurados ou próximos, conforme a legenda disponibilizada, os Beacons estão todos habilitados com o padrão iBeacon e são identificados pela cor rosa.

Figura 13 – Aplicativo Beeks para iOS



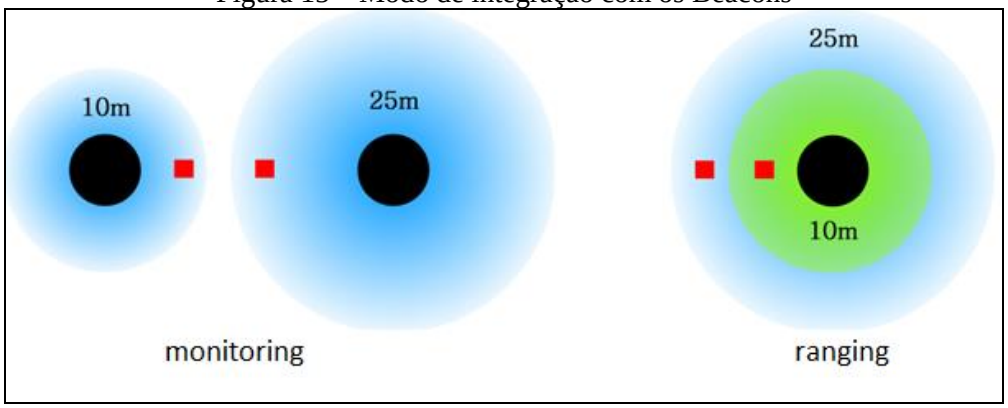
Ao selecionar um Beacon na lista exibida na Figura 13, são exibidas as informações básicas do Beacon como pode-se visualizar na Figura 14. Versão do *firmware*, bateria, informações do iBeacon caso esteja habilitado, informações utilizadas para a transmissão dos pacotes em modo normal e modo de baixo consumo de bateria.

Figura 14 – Informações gerais do Beacon



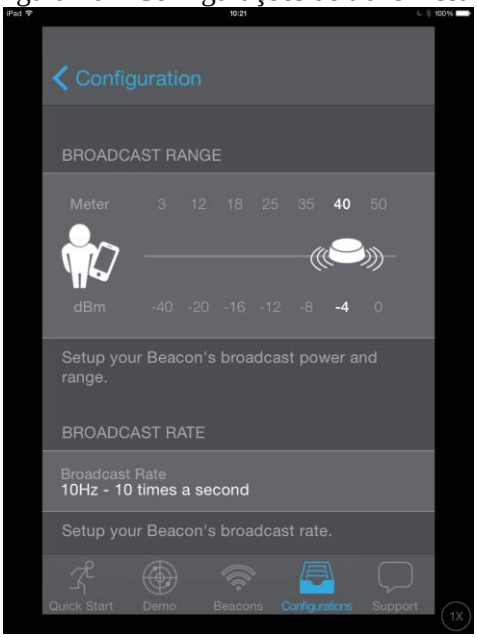
Para realizar a configuração de *broadcast* é necessário avaliar o modo com o qual será realizada a integração com os Beacons no aplicativo desenvolvido, sendo possível utilizar o *ranging* ou *monitoring*. Na Figura 15 são demonstrados os dois modos de operação para integração com o aplicativo.

Figura 15 – Modo de integração com os Beacons



Conforme a Figura 15, é possível visualizar a diferença entre os modos de integração. Ao utilizar o *monitoring* a configuração da distância alcançada pelo Beacon é de extrema importância, pois o aplicativo receberá a informação se está reconhecendo o Beacon ou não, sem a informação de qual a distância para o mesmo. Ao utilizar o *ranging*, é possível trabalhar diretamente com a distância entre o Beacon e o dispositivo móvel, controlando com apenas um Beacon diferentes tratamentos conforme os valores recebidos. Para o desenvolvimento foi avaliado que o modo *ranging* aplica-se as necessidades do aplicativo. No aplicativo de configuração, após selecionar as informações de *broadcast*, pode-se visualizar a tela representada na Figura 16, onde podemos definir o *Broadcast Range*, que define qual distância o Beacon deve atingir.

Figura 16 – Configurações de transmissão



Conforme o objetivo, ele ajusta a potência de transmissão do sinal. Os Beacons utilizados no desenvolvimento utilizaram a configuração para 40 metros, e o *Broadcast Rate* que define quantas vezes por segundo ele deve realizar o envio dos pacotes é de 1 vez por

segundo (1Hz). Após salvar as informações no aplicativo, o mesmo sincroniza com o Beacon e grava as informações.

3.3.2.2 Integração com os Beacons

Para realizar a integração com os Beacons foi utilizado o iBeacon Plugin disponibilizado no GitHub. Este *plugin* permite realizar o monitoramento dos Beacons de duas formas, sendo a primeira onde o mesmo notifica a proximidade do Beacon através de uma enumeração que possui distâncias pré-definidas como base, e a segunda que foi utilizada no aplicativo, que retorna a distância do usuário para o Beacon. No Quadro 3 é apresentado um trecho que demonstra como iniciar o monitoramento do Beacon.

Quadro 3 – Monitoramento do Beacon

```
1  function IBeacons(){
2  this.criarBeacon = function(uuid, identifier){
3    return new cordova.plugins.locationManager.BeaconRegion(
4        identifier, uuid, 0, 0);
5  };
6
7  this.monitorar = function(uid, identificador){
8    cordova.plugins.locationManager.startRangingBeaconsInRegion(
9       this.criarBeacon(uid, identificador)).done();
10 };
11
12 this.iniciar = function(callback){
13   cordova.plugins.locationManager.requestWhenInUseAuthorization();
14   var delegate = new cordova.plugins.locationManager.Delegate();
15
16   delegate.didDetermineStateForRegion = function (pluginResult) {};
17
18   delegate.didStartMonitoringForRegion = function (pluginResult) {};
19     delegate.didRangeBeaconsInRegion = callback;
20     cordova.plugins.locationManager.setDelegate(delegate);
21 };
22 }
```

O *plugin* foi adicionado ao projeto utilizando o comando `cordova plugin add https://github.com/petermetz/cordova-plugin-ibeacon.git` com o Node.js. O Quadro 3 representa a classe `IBeacons`, responsável por gerenciar diretamente as classes do *plugin.* A função `iniciar` é a responsável por criar a instância do gerenciador do *plugin* e configurar as funções que serão executadas pelo mesmo retornando os Beacons encontrados, na linha 13 é chamada a função `requestWhenInUseAuthorization` serve apenas para iOS, onde é necessário realizar a solicitação de segurança ao usuário, logo depois é criada a instância do `Delegate` e configuradas as funções de retorno, a única utilizada pela aplicação é a `delegate.didStartMonitoringForRegion` na linha 19 que recebe uma função que é executada na classe `GerenciadorAplicacao`.

A função `criarBeacon` na linha 2 até 5, retorna uma instância de `BeaconRegion` criada utilizando o identificador do dispositivo físico (`uid`) e o identificador lógico (`identificador`) que está sendo utilizado na aplicação, esse objeto representa um Beacon, então na linha 7 a 10 a função `monitorar` é responsável por iniciar o monitoramento, adicionando o objeto ao gerenciador do *plugin* através da função `cordova.plugins.locationManager.startRangingBeaconsInRegion` na linha 8.

O Quadro 4 demonstra a função `atualizarDistanciaBeacon` pertencente à classe `GerenciadorAplicacao`, executada a cada 1 segundo pelo *plugin* para cada `BeaconRegion` monitorado.

Quadro 4 – Função que atualiza as distâncias dos Beacons

```
1  this.atualizarDistanciaBeacon = function(pluginResult){
2    var jsonData = JSON.stringify(pluginResult);
3    var jsonPData = JSON.parse(jsonData);
4
5    var distancia = -1;
6    $.each(jsonPData.beacons, function(index, item){
7     distancia = item.accuracy;
8    });
9    var identificador = jsonPData.region.identifier;
10   var beacon = GerenciadorAplicacao.beacons[identificador];
11   if((GerenciadorAplicacao.salas.andar() == null || distancia <= 8) &&
12 distancia > -1)
13    GerenciadorAplicacao.salas.andar(identificador.andar());
14
15   if((GerenciadorAplicacao.avisos[identificador] != undefined &&
16     Math.abs(new Date() - GerenciadorAplicacao.avisos[identificador]) >
17     15000 && distancia == -1) || distancia > -1){
18   beacon.ultimaDistancia(distancia);
19   }
20 };
```

O parâmetro recebido pela função `atualizarDistanciaBeacon` é um objeto complexo que permite identificar o Beacon através do identificador lógico, e caso o mesmo esteja próximo, a propriedade `beacons` retorna uma coleção que possui o Beacon e suas informações de distância, caso a coleção esteja vazia o Beacon não está sendo reconhecido. Na linha 6, é executada a iteração e caso exista um elemento, a variável de controle `distancia` é atualizada. Na linha 9 é obtido o identificador lógico do Beacon para obter o Beacon cadastrado na aplicação. Na linha 11 é aplicada a validação da distância, considerando um valor inferior a 5 metros, para então alterar no `GerenciadorAplicacao` o andar atual do usuário que está contido no `beacon`. Na linha 14 é verificado se a última atualização da distância na aplicação faz mais de 15 segundos e a `distancia` é `-1` ou a `distancia` possui um valor maior que `-1`, sendo uma das duas condições verdadeiras é atualizada a distância atual do Beacon na aplicação e atualizada a data da última atualização do mesmo para

controle da aplicação. No Quadro 5 é apresentado o código referente a função que busca os Beacons cadastrados no banco de dados e inicia o monitoramento através das funções já apresentadas nesta seção.

Quadro 5 – Função que busca os Beacons na base de dados e inicia o monitoramento

```
1  this.buscarBeaconsIniciarMonitoramento = function(){
2    db.transaction(function(tx){
3    //adicionar os beacons em memória e cria os listeners
4    tx.executeSql('SELECT * FROM BEACON WHERE VIRTUAL = "false"',
5      [], function (tx,result){
6          if (result.rows.length > 0) {
7            for(var i=0; i<result.rows.length; i++) {
8              var row = result.rows.item(i);
9              var beaconViewModel = new Beacon(row['ID'],row['UID'],
10             row['LATITUDE'], row['LONGITUDE'], row['ANDAR']);
11
12           GerenciadorAplicacao.beacons[beaconViewModel.id()] =
13 beaconViewModel;
14           }
15           try {
16              GerenciadorAplicacao.iniciarMonitoramento();
17           }catch(e){
18                 alert(e.message);
19           }
20         }
21   }, function(e){alert("erro "+e.message);});
22   }, errorCB);
23 };
```

No Quadro 5 o código da função `buscarBeaconsIniciarMonitoramento` da classe `GerenciadorAplicacao` que é executada quando a aplicação é iniciada. A função é responsável por abrir uma transação com o banco de dados SQLite na linha 2, e então na linha 4 executar um comando SQL para procurar os Beacons físicos cadastrados. Na linha 6, caso a consulta retorne resultados, é iniciada a iteração sobre as linhas na linha 7, onde o objeto de domínio Beacon é criado e adicionado a lista de controle interno da aplicação na classe `GerenciadorAplicacao`. Após carregar todos os Beacons em memória, é iniciado o monitoramento através da chamada `GerenciadorAplicacao.iniciarMonitoramento`.

3.3.2.3 Integração com Google Maps

Para realizar a integração com o Google Maps foi utilizado o Phonegap GoogleMaps Plugin disponibilizado no GitHub. Este *plugin* permite realizar integração com o Google Maps nativo instalado no dispositivo móvel, sendo assim, o mesmo possui os mesmos recursos que as bibliotecas disponibilizadas para as plataformas Android e iOS. No Quadro 6 é exibido o trecho do código da classe `Mapa` responsável por abstrair e gerenciar o *plugin*.

Quadro 6 – Classe que manipula o plugin do Google Maps

```
1  var self = this;
2  var map;
3  var marker;
4  var pin;
5  this.gpsAtivo = false;
6
7  this.iniciar = function(elemento){
8    self.map = plugin.google.maps.Map.getMap($(elemento)[0]);
9    self.map.addEventListener(plugin.google.maps.event.MAP_READY,
10     self.mapaCarregado);
11 };
```

O *plugin* foi adicionado ao projeto utilizando o comando `cordova plugin add https://github.com/wf9a5m75/phonegap-googlemaps-plugin.git` com o Node.js. No Quadro 6 inicia-se com a declaração das variáveis que são gerenciadas pela classe, e na linha 7 o método que carrega o mapa visualmente no aplicativo, ele recebe como parâmetro um `elemento` HTML onde o mapa deve ser carregado, e logo na linha 8 é instanciado o objeto do mapa que é armazenado na variável `map`, todos os controles são realizados nessa variável, logo após instanciar é necessário adicionar um evento ao mapa, conforme a linha 9, para quando o mesmo for carregado sejam iniciados os tratamentos de posicionamento, como é demonstrado no Quadro 7 e Quadro 8.

Quadro 7 – Função executada quando o mapa é carregado

```
1  this.mapaCarregado = function () {
2    self.map.setIndoorEnabled(true);
3          var onSucesso = function(location) {
4                  GerenciadorAplicacao.mapa.gpsAtivo = true;
5                  self.atualizarPosicaoMapa(location);
6          };
7          var onErro = function(msg) {
8                  GerenciadorAplicacao.mapa.gpsAtivo = false;
9          };
10         setInterval(function(){
11                 try {
12                         self.map.getMyLocation(onSucesso, onErro);
13                 }catch(e){alert(e.message);}
14         }, 8000);
15 };
```

No Quadro 7 a função `mapaCarregado` é executada quando a instância do mapa é criada, na linha 10 é iniciado um `setInterval` que executa o `map.getMyLocation` a cada 8 segundos, caso o GPS esteja ativo, a função declarada na linha 3 é executada, na linha 4 a mesma salva na variável de classe `gpsAtivo` o valor `true` e na linha 5 executa a função `atualizarPosicaoMapa` declarada na linha 1 do Quadro 8, caso o GPS não esteja habilitado ou o usuário não permita o aplicativo utilizar a função declarada na linha 7 é executada e o valor `false` é salvo na variável `gpsAtivo`. A variável `gpsAtivo` é utilizada para controlar o marcador em casos onde o GPS encontra-se desabilitado, sendo assim, ao encontrar um

Beacon virtual ou físico, é obtida a coordenada dele e é executada a função `atualizarPosicaoMapa`.

Quadro 8 – Função que atualiza a posição no mapa

```
1  this.atualizarPosicaoMapa = function (localizacao){
2          if (localizacao == null)
3                  return;
4          self.map.addMarker({
5                  'position': localizacao.latLng,
6                  'title': "Você está aqui!"
7          }, function(m) {
8                  if(marker)
9                        marker.remove();
10                 marker = m;
11                 marker.showInfoWindow();
12         });
13         var position = new plugin.google.maps.LatLng(
14                 localizacao.latLng.lat, localizacao.latLng.lng);
15         self.map.moveCamera({
16                 'target': position,
17                 'zoom': 18
18         }, function() {});
19 };
```

No Quadro 8 a função `atualizarPosicaoMapa` adiciona o marcador na tela da posição atual do usuário na linha 4. Caso já exista um marcador o mesmo é removido e a instância atual é armazenada na classe. Na linha 15 a câmera é movida para o ponto do marcador.

3.3.2.4 Síntese de voz

Para realizar a síntese de voz foi utilizado o TTS Plugin disponibilizado no GitHub. Este *plugin* permite sintetizar um texto em voz. No Quadro 9 é exibido o trecho do código da classe `Sintetizador` responsável por abstrair e gerenciar o *plugin*.

Quadro 9 – Trecho de código da classe `Sintetizador`

```
1  function Sintetizador(){
2    this.iniciar = function(){
3          ttsPlugin.initTTS(function(){
4          ttsPlugin.setRate("0.2"); // Set voice speed : default is "0.2"
5          ttsPlugin.setLanguage("pt-BR"); // Set voice language : default
6  is "en-US"
7      }, function(){
8          alert("Não foi possível iniciar o plugin de síntese de
9  voz.\r\n");
10         });
11   }
12   this.falar = function(frase){
13         ttsPlugin.speak(frase);
14         ttsPlugin.stopSpeaking();
15   }
16 }
```

O *plugin* foi adicionado ao projeto utilizando o comando `cordova plugin add https://github.com/steevelefort/cordova3-ios-tts-plugin` com o Node.js. Conforme o trecho de código do Quadro 9, a linha 2 a função `iniciar` executa a função `initTTS` onde são configuradas a velocidade de fala na linha 4 e a linguagem na linha 5. A função abstraída `falar` é responsável por executar a função `speak` onde é passado como parâmetro o texto que deve ser sintetizado. No Quadro 10 a função demonstra a síntese de voz aplicada aos avisos que o aplicativo emite ao usuário a cada 1 minuto referente a sua localização e os eventos que estão ocorrendo perto dele.

Quadro 10 – Trecho de código onde é realizada a síntese de voz

```
1  setInterval(function(){
2    try {
3          if(GerenciadorAplicacao.parametros.audio()){
4                  var mensagem = "";
5              var eventosAviso = new Eventos();
6                  var eventosAvisoFuturo = new Eventos();
7
8                  if(GerenciadorAplicacao.salas.quantidade() > 0){
9                    mensagem = GerenciadorAplicacao.salas.bloco()+
10                  " "+GerenciadorAplicacao.salas.andar()+"° andar.";
11                 }
12
13                 var emExecucao =
14                  GerenciadorAplicacao.eventos.eventosEmExecucao();
15                 mensagem = montarMensagemVozEmExecucao(
16                emExecucao, emExecucao.length, mensagem, eventosAviso);
17
18                 var futuros =
19                  GerenciadorAplicacao.eventos.eventosFuturos();
20                 mensagem = montarMensagemVozFuturo(
21                futuros, futuros.length, mensagem, eventosAvisoFuturo);
22
23                 if (mensagem !== "")
24                {
25                         GerenciadorAplicacao.falar(mensagem);
26                 }
27         }
28   } catch(e){
29         alert("Erro ao executar a síntese de voz. \r\n " +  e.message);
30   }
31 }, 60000);
```

No Quadro 10 a função que é executada a cada 1 minuto fala o bloco e andar onde o usuário está, os eventos em execução caso existam e os eventos que irão acontecer na próxima 1 hora. Na linha 3 o aplicativo verificar o parâmetro de usuário se o áudio está habilitado, então inicia na linha 5 e 6 as classes de controle dos eventos que serão pronunciados são instanciadas, a classe `Eventos` faz controle de uma lista da classe de domínio `Evento`, na linha 8 é verificado se tem alguma sala perto, sendo assim é possível definir o andar e o bloco na linha 9 e montando a mensagem para atribuir à variável. Na linha 13 são listados dos os

eventos que estão em execução e passados a função `montarMensagemVozEmExecucao`, que monta a mensagem e retorna a variável, na linha 18 o mesmo é realizado com os eventos que serão executados na próxima hora. Na linha 24, caso a `mensagem` não esteja vazia é executada a chamada do `Sintetizador.falar` através do `GerenciadorAplicacao.falar`.

3.3.2.5 Notificações

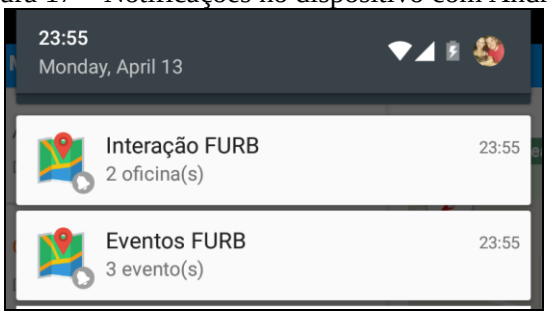
Para realizar a síntese de voz foi utilizado o Cordova *Plugin Local Notifications* disponibilizado por Sebastián Katzer (2015) no GitHub. Este *plugin* permite adicionar notificações no formato padrão do sistema operacional do dispositivo móvel. No Quadro 11 é exibido o trecho do código onde é criada uma notificação.

Quadro 11 – Exemplo de uso de notificações

```
1  cordova.plugins.notification.local.on("click", function (notification)
2  {
3    $("#left-panel").panel("open");
4  });
5
6  if(GerenciadorAplicacao.eventos.quantidadeEventosInteracao() > 0){
7    cordova.plugins.notification.local.schedule({
8          id: 1,
9          title: "Interação FURB",
10         text: GerenciadorAplicacao.eventos.quantidadeEventosInteracao()
11 +
12                 " oficina(s)",
13   });
14 } else {
15   cordova.plugins.notification.local.cancel(1, function(){}, this);
   }
```

O plugin foi adicionado ao projeto utilizando o comando `cordova plugin add https://github.com/katzer/cordova-plugin-local-notifications` com o Node.js. Na linha 1 a função `on` adiciona uma função de retorno para o evento de abertura da notificação, nesse caso quando o usuário tocar na notificação o aplicativo abre e o painel lateral esquerdo com os eventos é aberto pela função `$("#left-panel").panel("open")` na linha 3 utilizando JQuery para selecionar o elemento que contém o painel. Na linha 6 verifica-se se existem eventos do tipo interação. Se existir, na linha 7 a função schedule é executada passando como parâmetro um objeto com as informações da notificação, e a mesma é exibida para o usuário, conforme a Figura 17.

Figura 17 – Notificações no dispositivo com Android



A Figura 17 exibe os dois tipos de notificação que o aplicativo possui, sendo que são exibidas somente se existem eventos próximos, caso não existam eventos, conforme o Quadro 11 na linha 14 a função `cancel` é executada passando como parâmetro o `id` da notificação. Para diferenciar as notificações basta alterar o `id` ao criar o objeto que será adicionado ao `schedule`.

3.3.2.6 Eventos de atualização

A aplicação possui uma linha de execução que realiza a integração dos Beacons e atualiza as distâncias para cada Beacon, essa linha foi detalhada na seção 3.3.2.2. A segunda linha de execução da aplicação lê as informações de distância para cada Beacon e executa a busca dos Beacons virtuais, eventos próximos e ambientes próximos, conforme o Quadro 12.

Quadro 12 – Evento de atualização

```
1  this.iniciarAtualizacaoEventosSalas = function(){
2    setInterval(function(){
3     db.transaction(function(tx){
4          try {
5            for(var chave in GerenciadorAplicacao.beacons){
6              var beacon = GerenciadorAplicacao.beacons[chave];
7              var distancia = beacon.andar() ==
8                   GerenciadorAplicacao.salas.andar() ?
9                   beacon.ultimaDistancia() : -1;
10
11             if(distancia > -1){
12             //retorna os beacons virtuais existente na distancia
13 determinada, com variação de +-20%
14
15              GerenciadorAplicacao.buscarBeaconsVirtuaisProximos(tx,
16 beacon.id(), beacon.ultimaDistancia());
17             }
18             var temBeacons = distancia != -1 ? 1 : 0;
19           GerenciadorAplicacao.buscarAmbientesPorBeacon(tx,beacon.id(),
20               temBeacons, distancia);
21             GerenciadorAplicacao.buscarEventosPorBeacon(tx, beacon.id(),
22               temBeacons, distancia);
23
24           //notificações
25         }catch(e){alert(e.message);}
26    }, errorCB);
27   }, 3000);
28 };
```

A função `iniciarAtualizacaoEventosSalas` demonstra o fluxo apresentado na seção 3.2.3. Após realizar os tratamentos e obter os ambientes e eventos conforme as regras previamente definidas, a interface é atualizada através do *framework* Knockout, podemos verificar um exemplo de código no Quadro 13.

Quadro 13 – Atualização da interface utilizando Knockout

```
1  ko.applyBindings(GerenciadorAplicacao.eventos, $("#eventos")[0]);
2  <div id="eventos">
3     <ul id="lista-eventos" data-bind="foreach: resultados"
4  class="lista">
5            <li data-bind="click: $parent.exibirEvento">
6                  <p class="nome"
7                        data-bind="text: nome, css : { interacao:
8  isInteracaoFURB()  }"></p>
9              <p class="descricao"
10                           data-bind="text:
11 ambiente().descricaoCompleta() +' '+
12
13 dataInicio().format('HH:MM')+'h'"></p>
14         </li>
15    </ul>
16 </div>
```

Na linha 1 é executada a função `applyBindings` que é associada a propriedade `eventos` do tipo `Eventos` na classe `GerenciadorAplicacao`, a partir desse momento, qualquer alteração realizada nos objetos que pertencem a coleção interna, automaticamente refletem na interface, pois no HTML da tela, o atributo `data-bind` faz referência a função `resultados`, que retorna os eventos próximos baseados em uma coleção, essa função é do tipo `ko.computed`, sendo assim, ela é notificada a cada alteração na coleção que é do tipo `ko.observableArray`.

3.3.3 Operacionalidade da implementação

Esta seção apresenta a operacionalidade da implementação em nível de usuário, mostrando as principais funcionalidades da ferramenta para Android. As imagens da execução foram obtidas com o dispositivo LGE Nexus 4, *smartphone* com a versão 5.1 (Lollipop) do Android. Ao acessar a aplicação é apresentada a tela inicial, conforme a Figura 18, com o mapa exibindo a localização atual do usuário no centro da tela, e no topo, a informação do bloco e andar que o usuário está, caso esteja em um ambiente com Beacons configurados.

Figura 18 – Tela inicial

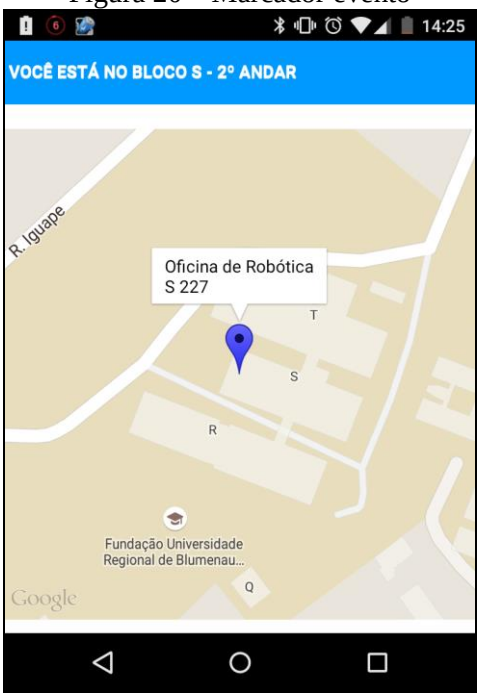
Ao arrastar o dedo na tela para a direita, é exibido o painel lateral com os `eventos`, no canto superior esquerdo são exibidos os ícones de `configuração de áudio` e `configurações`, conforme a Figura 19. O aplicativo fala a cada minuto o `bloco` e o `andar` que o usuário está, e os `eventos` que já iniciaram e os que irão iniciar na próxima 1 hora.

Figura 19 – Painel esquerdo

Os `eventos` do tipo `interação FURB` são exibidos em laranja, para diferenciar de `eventos normais`, que são exibidos em cinza. As informações dos `eventos` conforme a Figura 19 são o `nome`, o `bloco` e `sala` e o `horário de início`. Ao tocar em um `evento` da

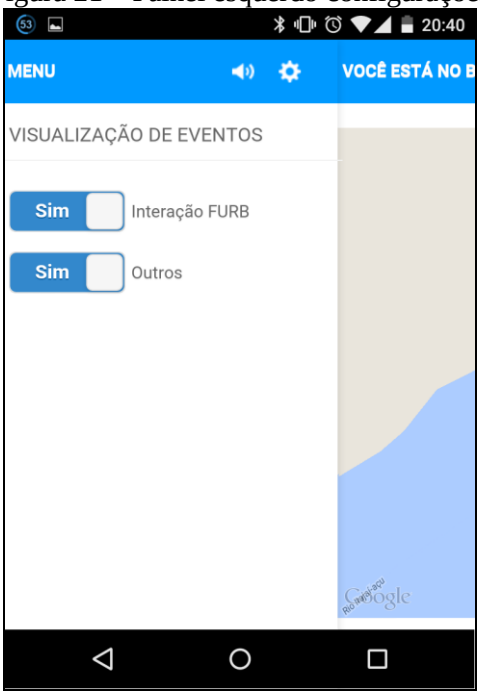
lista, a aplicação fala as informações do `evento` e exibe um `marcador azul` no `mapa` com o local aproximado conforme a Figura 20.

Figura 20 – Marcador evento



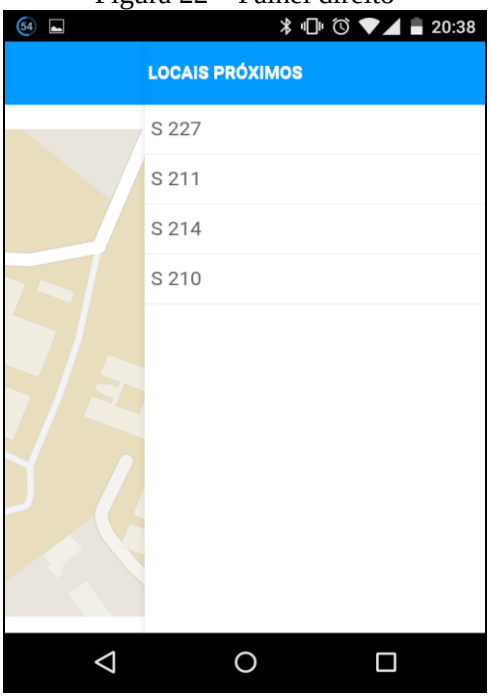
No painel esquerdo, pode-se ainda habilitar ou desabilitar o áudio através do ícone de `configurações de áudio`. O ícone de `configurações` exibe as configurações de filtros, conforme a Figura 21.

Figura 21 – Painel esquerdo configurações



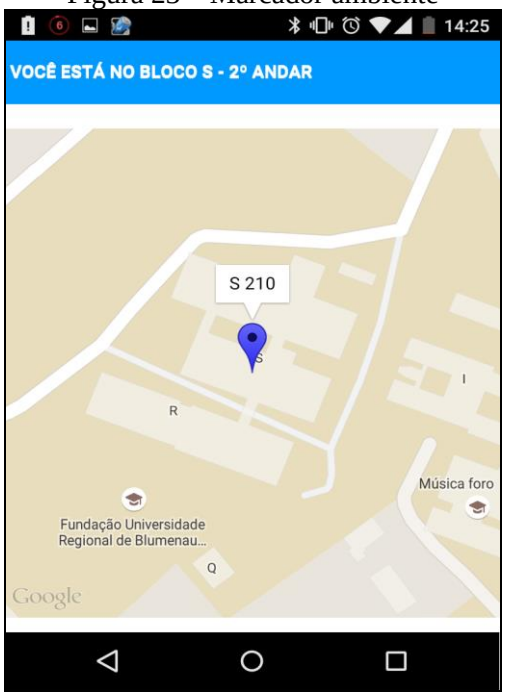
A Figura 21 exibe o painel de configurações, que permite configurar os filtros dos tipos de eventos que serão visualizados na aplicação. Ao arrastar o dedo na tela para a esquerda, é possível visualizar os ambientes próximos, conforme a Figura 22.

Figura 22 – Painel direito



Os locais próximos são exibidos com o `bloco` e o `ambiente`, conforme Figura 22. Ao tocar em um `ambiente`, o aplicativo fala o `bloco`, o `ambiente` e a `distância` caso seja maior que 2 metros, conforme mostra a Figura 23.

Figura 23 – Marcador ambiente



3.4 RESULTADOS E DISCUSSÕES

Nesta seção são apresentados os experimentos feitos com o aplicativo. Na seção 3.4.1, detalha-se o cenário, a avaliação do aplicativo, o perfil dos usuários, a aplicação dos testes, a análise e interpretação dos dados coletados. Por fim, a seção 3.4.2 faz a comparação e discussão entre o aplicativo desenvolvido e os trabalhos correlatos.

3.4.1 Experimento do aplicativo

O experimento foi realizado com dez usuários para avaliar a eficiência e a facilidade de utilização das funcionalidades implementadas no aplicativo.

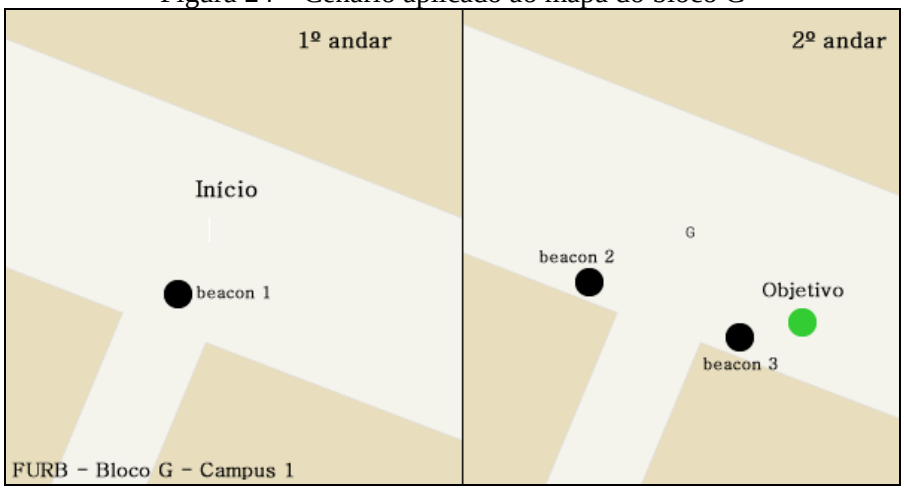
3.4.1.1 Metodologia

O experimento aconteceu durante o mês de maio por meio de testes em dupla com os usuários. Para realizar os testes foram disponibilizados dois dispositivos, iPad Mini e Nexus 4, com sistema operacional iOS 8.3 e Android 5.1 respectivamente. Nestes dispositivos também foi disponibilizado o aplicativo instalado com 3 Beacons cadastrados e configurados com eventos e ambientes do cenário para os testes. Foi fornecido a cada usuário um questionário de perfil, um objetivo e um questionário de avaliação, que estão disponíveis no Apêndice C.

3.4.1.2 Cenário do teste

O cenário utilizado no teste consiste na disponibilização do dispositivo móvel ao usuário e uma explicação sobre o funcionamento geral. Cada usuário utilizou o aplicativo por 5 a 10 minutos, para entender o funcionamento. Os usuários foram expostos ao cenário, que consistia em estar na entrada do bloco G, em frente ao balcão de informações, e o objetivo era encontrar a Oficina de Robótica, que estava entre o 1º e 2º andar do bloco G, utilizando as informações que o aplicativo fornecia aos mesmos. Na Figura 24 é demonstrado o cenário com a distribuição dos Beacons no bloco G.

Figura 24 – Cenário aplicado ao mapa do bloco G



Conforme a Figura 24, foram distribuídos três Beacons, sendo um no 1º andar, na entrada do bloco, onde o usuário iniciava a busca pelo objetivo. Os outros dois Beacons foram distribuídos no 2º andar, onde estava o objetivo do experimento.

3.4.1.3 Aplicação do teste

Para iniciar a avaliação, cada voluntário foi orientado sobre o objetivo do teste e sobre o funcionamento do aplicativo. O questionário de avaliação foi composto por dez perguntas e buscou contemplar todas as funcionalidades implementadas e disponíveis no aplicativo, as dez perguntas foram fechadas e com a possibilidade de realizar observações sobre a resposta escolhida, ou mesmo sua sugestão ou reclamação. As perguntas procuraram obter as impressões do usuário sobre o aplicativo, sobre a facilidade de utilização e a importância das funcionalidades disponíveis para chegar ao objetivo. Os resultados deste experimento são apresentados na próxima seção.

3.4.1.4 Análise e interpretação dos dados coletados

Primeiramente analisou-se dos dados coletados através do questionário de perfil de usuário. No Quadro 14 são exibidos os perfis dos dez usuários envolvidos no teste de usabilidade.

Quadro 14 – Perfis dos usuários envolvidos nos testes

Sexo	80% masculino e 20% feminino
Idade	60% entre 18 e 25 anos 30% entre 25 e 35 anos 10% mais de 35 anos
Nível de escolaridade	80% ensino superior incompleto 20% ensino superior completo
Possui algum dispositivo móvel	100% possui *smartphone*/*tablet*
Familiaridade com aplicativos de georreferenciamento	10% nunca utilizou mas conhece 40% utiliza raramente 50% utiliza diariamente

Através dos resultados tabulados, foi possível observar que todos os voluntários têm mais de 18 anos e possuem o ensino superior incompleto ou completo. Todos os voluntários possuem um *smartphone* e conhecem a finalidade do aplicativo, sendo que 50% utilizam aplicativos diariamente de georreferenciamento e navegação, como o Waze e Google Maps.

Com as informações de perfil dos usuários voluntários para os testes, prosseguiu-se para a avaliação dos resultados obtidos com o questionário de avaliação.

3.4.1.4.1Análise dos resultados do questionário de avaliação

No Quadro 15 são apresentados resultados quanto às questões envolvendo o cenário do aplicativo onde as respostas poderiam ser apenas sim e não.

Quadro 15 – Respostas questionário de avaliação sim/não

Perguntas/Respostas	sim	Não
Você conseguiu chegar no seu objetivo utilizando o aplicativo?	100%	
Você recebia as notificações e visualizava os eventos no aplicativo antes de visualizar os ambientes reais?	80%	20%
O recurso de voz lhe auxiliou no trajeto?	50%	50%
Você acredita que esse aplicativo poderia auxiliar novos alunos na universidade?	100%	
O mapa do aplicativo exibindo a sua localização atual facilitou a locomoção pela universidade?	100%	
De um modo geral, você achou o aplicativo intuitivo e fácil de usar?	70%	30%

Através dos resultados obtidos das questões objetivas foi possível concluir que todos os voluntários chegaram ao objetivo. A maioria dos voluntários utilizou o recurso de notificações e conseguia ver o evento na tela do celular antes de chegar ao destino. Porém, 20% dos voluntários não conseguiram escutar os comandos de voz, isso ocorreu devido ao atraso da atualização das informações pelo aplicativo. O recurso de voz foi prejudicado em alguns momentos, pois o áudio dos dispositivos era baixo perante o ambiente em que foi testado, sendo assim, é possível observar que 50% dos voluntários votaram que o recurso de

voz não auxiliou no trajeto, e também, porque existem outros recursos que auxiliam durante a locomoção do usuário, porém o foco do recurso de voz seriam usuários com baixa visão ou necessidades especiais, e o uso de um fone de ouvido facilitaria a utilização do recurso melhorando o retorno do áudio. A usabilidade do aplicativo foi aprovada com 70% dos votos, sendo que 30% dos voluntários responderam que o aplicativo não era intuitivo. A interface da aplicação foi planejada para fornecer um meio mais amigável ao usuário, porém, nenhum padrão específico foi utilizado para melhorar a experiência dos usuários, um estudo mais profundo dos padrões de usabilidade das plataformas Android e iOS poderia auxiliar no aprimoramento da usabilidade e interface.

No Quadro 16 são apresentados resultados quanto às questões envolvendo o cenário do aplicativo onde as respostas poderiam ser apenas sim, não e talvez.

Quadro 16 – Respostas questionário de avaliação sim/não/talvez

Perguntas/Respostas	Sim	não	talvez
Você usaria este aplicativo no seu dia a dia na universidade?	40%		60%
Você indicaria este aplicativo à outra pessoa?	90%	10%	

A maioria dos voluntários eram estudantes da graduação em estágio final, sendo assim, não havia a necessidade do aplicativo, pois já existe um conhecimento do ambiente da universidade, justificando os 60% que talvez utilizariam o aplicativo no dia a dia.

No Quadro 17 são apresentados resultados quanto às questões envolvendo o cenário do aplicativo onde as respostas poderiam ser apenas sim, não e parcialmente.

Quadro 17 – Respostas questionário de avaliação sim/não/parcialmente

Perguntas/Respostas	Sim	não	parcialmente
Você entendeu claramente os comandos de voz do aplicativo?	20%	10%	70%

A compreensão dos comandos de voz não atendeu a expectativa, sendo que 70% dos votos resultaram em um entendimento parcial. A síntese de voz executada no aplicativo não visava orientar o usuário até o objetivo, e sim, orientar a localização atual e o que estava acontecendo próximo ao mesmo, além de prover acessibilidade, com a síntese de voz auxiliando na usabilidade do aplicativo.

No Quadro 18 são apresentados resultados quanto a avaliação final do aplicativo, podendo optar entre muito bom, bom, regular e insatisfatório e as observações.

Quadro 18 – Respostas questionário de avaliação final

Perguntas/Respostas	muito bom	bom	regular	insatisfatório
Qual é a sua avaliação do aplicativo?	10%	90%		

Considerando os resultados das questões, a avaliação final foi positiva, onde 90% dos voluntários avaliaram como bom, e 10% como muito bom, foi possível concluir, que não houve grandes dificuldades para encontrar o objetivo e algumas melhorias poderiam aprimorar a experiência do usuário. Além das 10 questões objetivas, o questionário também continha espaço para anotar as suas percepções sobre as questões. As observações feitas nesta parte do experimento apontaram alguns problemas como a demora em alguns momentos para atualizar as informações e exibir as notificações e sugerem alterações para facilitar a visualização do mapa. A demora para atualizar as informações ocorreu devido a um problema no *plugin* de integração com os Beacons, onde as vezes ele tem um espaço de tempo um pouco maior para notificar à camada Javascript com as informações de distância do Beacon para o dispositivo móvel, influenciando diretamente na atualização dos eventos na tela do aplicativo. Para a visualização do mapa, foi sugerido pelos voluntários a utilização da bússola visando facilitar o senso de direção ao deslocar-se. Esta solução não foi implementada devido ao tempo limitado disponível, porém é possível incluir esta funcionalidade utilizando o *plugin* de acesso a bússola disponível no Cordova. Alguns voluntários citaram que o volume do áudio do aplicativo estava baixo, ou então que não entenderam algumas orientações do aplicativo, justificando assim o resultado de 70% de compreensão parcial dos comandos de voz. Uma última observação feita por um dos voluntários foi a de que a usabilidade do aplicativo é simples, mas ao primeiro contato ele não foi intuitivo.

3.4.2 Comparação com trabalhos correlatos e discussões

Após a avaliação dos resultados obtidos com os experimentos, o quadro de comparação dos trabalhos correlatos foi alterado, incluindo o aplicativo Tô Aqui para comparação. O Quadro 19 descreve as diferenças e semelhanças entre o aplicativo Tô Aqui e os trabalhos correlatos.

Quadro 19 – Comparação entre trabalhos correlatos e aplicativo Tô Aqui

Trabalho	Custo	Estrutura Física	Implementação
Fagundes (2008)	Custo baixo quando existe uma rede pronta, caso seja necessário montar a rede sem fio, com os *Access Points* (AP), o sistema acaba se tornando caro.	Os APs devem ser colocados em locais estratégicos, pois como se trata de RF, o sinal pode sofrer interferências e prejudicar a estimativa de distância.	O aplicativo torna-se complexo, pois é necessário executar muitos controles a nível de hardware.O aplicativo comunica-se apenas com um servidor que dispara a rotina de localização para os APs que executam a técnica e retornam a posição.
BinarioMobile (2014)	Custo baixo do dispositivo, considerando que ele auxilia a melhorar a precisão dos outros recursos que o dispositivo móvel já possui, como o GPS.	A estrutura torna-se simples com o uso dos Beacons, mapeando o ambiente e colocando-os em locais estratégicos facilita a leitura e processamento das informações.	Desenvolvimento simplificado, pois com o uso do *framework* do fabricante os tratamentos a nível de aplicação são reduzidos. É desenvolvido um aplicativo para cada plataforma.
Moura (2007)	Custo baixo assim como o trabalho da seção 2.3.1, pois utiliza redes sem fio existentes.	A estrutura exige APs dispostos em locais estratégicos assim como o trabalho apresentado na seção 2.3.1.	A implementação é mais complexa, pois é implementado um grafo para controlar a diferença de potência de sinal entre os APs, e não a diferença de frequência de sinal como a técnica da seção 2.3.1.
Rocha (2015)	O custo baseia-se apenas na compra dos Beacons, e os dispositivos móveis compatíveis são qualquer um com Bluetooth 4.0 ou superior. A possibilidade de criar pontos intermediários entre os Beacons também reduz o custo.	A estrutura torna-se simples com o uso dos Beacons, mapeando o ambiente e colocando-os em locais estratégicos facilita a leitura e processamento das informações.	A implementação utilizando Cordova, permite desenvolver apenas um aplicativo para duas plataformas. A utilização dos plug-ins facilita o desenvolvimento.

Comparando o aplicativo aos trabalhos correlatos, é possível verificar que o diferencial do aplicativo é o uso do Cordova, que permite disponibilizar o aplicativo para a plataforma Android e iOS, além da facilidade em desenvolver utilizando os Beacons.

O *plugin* para a integração com os Beacons facilitou o desenvolvimento do aplicativo, e por utilizar o padrão iBeacon, permite utilizar o aplicativo com qualquer Beacon que siga esse padrão. O *plugin* disponibiliza os dois tipos de notificação referente aos Beacons, no aplicativo foi utilizado apenas o `ranging`, que permite obter a distância entre o Beacon e o usuário. Este modo de notificação não permite receber notificações em segundo plano,

somente com o aplicativo em execução em primeiro plano. O modo *monitoring* permitiria realizar as notificações em segundo plano, mas exigiria mais Beacons para a estrutura física.

O aplicativo utiliza o Google Maps, enquanto os trabalhos correlatos não utilizam esse recurso. O aplicativo descrito na seção 2.3.2 utiliza um mapa interno customizado desenvolvido para cada aplicativo. O aplicativo também se destaca por utilizar recursos de síntese de voz para auxiliar na locomoção do usuário e a acessibilidade.

A utilização do Cordova para o desenvolvimento, possibilita o uso de ferramentas web e recursos que muitas vezes não são disponíveis na linguagem nativa do Android ou iOS, sem deixar de acessar importantes funcionalidades nativas. No protótipo desenvolvido foi utilizado o banco de dados SQLite, que é disponibilizado tanto em iOS como em Android, mas o desenvolvimento é único. O desenvolvimento para outras plataformas torna-se mais fácil e a plataforma também dispensa o conhecimento da linguagem nativa dos sistemas operacionais.

4 CONCLUSÕES

Os objetivos principais do trabalho, que eram aprimorar a precisão da localização em ambientes internos, melhorar a percepção do usuário referente a posição dele em um ambiente e disponibilizar um aplicativo multiplataforma foram alcançados.

O objetivo de aprimorar a precisão da localização em ambientes internos foi alcançado com o uso dos Beacons, comprovando que é possível obter informações do local e visualizar a localização no mapa, mesmo sem sinal de GPS.

Os recursos que o aplicativo oferece ao usuário para visualizar o que está acontecendo perto da sua posição mostraram-se eficientes, pois ele poderia visualizar em formatos diferentes a mesma informação. Através da notificação do aplicativo é possível visualizar a lista de eventos próximos à posição atual do usuário, visualizar onde fica no mapa o evento ou o ambiente desejado pelo usuário e escutar as orientações do aplicativo caso não compreenda o que está na tela ou não possa visualizar.

A possibilidade de disponibilizar o aplicativo para a plataforma iOS e para a plataforma Android com todos os recursos foi um objetivo concluído e muito importante, pois com a mesma implementação foi possível atender uma quantidade maior de usuários e a manutenção do código estar centralizada em apenas um projeto, além de validar o uso de um gerador de código, Cordova/Phonegap, para gerar um aplicativo integrado com hardware.

Conforme os resultados dos experimentos e dos questionários de avaliação foi comprovado que é possível melhorar a precisão da localização do usuário em ambientes internos através dos métodos e recursos adotados neste trabalho.

Apesar do sucesso na implementação, poderiam ter sido analisados os atrasos de notificação da biblioteca nativa para a camada Javascript do *plugin* de integração com os Beacons, que acaba interferindo diretamente em alguns momentos na usabilidade do aplicativo.

Conforme as pesquisas realizadas, existem poucos aplicativos hoje que possibilitam ao usuário realizar a navegação em ambientes internos sem depender diretamente do AGPS. A implementação de um trabalho com os objetivos propostos aqui é válida diante da necessidade identificada na própria universidade, onde o cenário possui variados eventos em grande quantidade centralizados em espaços de tempo próximos, amenizando assim o cenário de caos que é organizar e orientar as pessoas a encontrar o seu evento.

4.1 EXTENSÕES

Algumas das extensões possíveis para este trabalho são:

a) implementar controle de acesso ao aplicativo através de comunicação com o servidor da universidade;
b) implementar recurso para obter as informações que são necessárias para o funcionamento do aplicativo através do servidor da universidade;
c) aprimorar a experiência do usuário em relação a síntese de voz;
d) analisar e implementar serviço para notificação em segundo plano;
e) analisar a biblioteca de comunicação dos Beacons para Android, e verificar as melhorias que poderiam ser realizadas para estabilizar o envio das informações para a camada de aplicação do Cordova em Javascript;
f) implementar o uso da bússola para orientação do mapa.

REFERÊNCIAS

ADOBE. **PhoneGap buid**. [S.l.], 2014. Disponível em: < https://build.phonegap.com/ >. Acesso em 16 set. 2014.

ANDERSSON, Tim. **Bluetooth low energy and smartphones for proximity-based automatic door locks**. 2014. 13 f. Artigo (Tese Final do Departamento de Computação e Ciência da Informação) – Linköping Universitet, Linköping, Sweden.

APACHE. **Apache cordova documentation**: the command-line interface. Version 5.0.0. [S.l.], 2015. Disponível em: <http://cordova.apache.org/docs/en/5.0.0/guide_cli_index.md.html#The%20Command-Line%20Interface>. Acesso em: 23 mai. 2015.

BERLIM, Thiago L.; REICHERT, Jennifer R; TEODORO, Fabricia. **O Uso da tecnologia móvel para o auxílio ao aprendizado de crianças com deficiência auditiva**. Itajaí, 2013. Disponível em: < http://www6.univali.br/seer/index.php/acotb/article/viewFile/6248/3510 >. Acesso em 3 ago. 2014.

BINARIOMOBILE. **Beacon, a nova geração de serviços para mobilidade.** [S.l.], 2014. Disponível em: <http://www.binariomobile.com.br/beacon.php >. Acesso em 3 ago. 2014.

BRAVO, Dante; BASTOS, Marcelo. **Sistema de posicionamento global**. 2014. 10 f. Trabalho de Conclusão de Curso (Curso Técnico em Informática) – Universidade Federal de Santa Maria (Colégio Politécnico), Santa Maria, Brasil.

BROERING, Alex W. **Localização de estações móveis praticadas sob GSM e GPS.** 2010. 90 f. Dissertação (Graduação Tecnológica) – Curso de Graduação em Sistemas de Telecomunicação, Instituto Federal de Santa Catarina, São José, Brasil.

BRUM, Everton V. P. et al. **Georreferenciamento.** 2006. 102 f. Mestrado (Curso de Especialização em Engenharia Florestal) Curso de Pós-graduação em Ciências Ambientais, Universidade de Unemat - Campus de Cáceres, Sinop: Mato Grosso, Brasil.

FAGUNDES, Leonardo P. **Técnicas de localização de dispositivos móveis em redes WiFi – TDOA**. 2008. 30 f. Trabalho de Conclusão de Curso (Curso de Especialização em Tecnologias, Gerência e Segurança de redes de Computadores) – Universidade do Rio Grande do Sul (Instituto de Informática), Porto Alegre, Brasil.

GATTERMANN. Rodrigo L. **Buscalivro**: solução móvel para apoio na localização de livros **.** 2013. 91 f. Monografia **(**Bacharel em Sistemas de Informação) – Centro Universitário Univates, Lajeado, Brasil.

LOPES, Igor. **Futurecom 2014**: Binário Beacon, uma solução de "GPS Indoor". São Paulo, 2014. Disponível em: < http://corporate.canaltech.com.br/materia/futurecom/Futurecom-2014-Binario-Beacon-uma-solucao-de-GPS-Indoor/>. Acesso em 11 jun. 2015.

MICHAELIS, Dicionário Online. **Localização**. [S.l.], 2014. Disponível em: <http://michaelis.uol.com.br>. Acesso em 5 set. 2014.

MOURA, André I. **WBLS**: a mobile device localization system on Wi-Fi networks. 2007. 134 f. Dissertação (Mestrado em Sistemas Digitais) – Curso de Mestrado em Engenharia Elétrica, Escola Politécnica da Universidade de São Paulo, São Paulo, Brasil.

PALUDO, Gabriela C. **Easyscore**: um protótipo para facilitar a leitura de partituras em dispositivos móveis. 2014. 73 f. Monografia **(**Bacharel em Ciência da Computação) – Universidade Regional de Blumenau, Blumenau, Brasil

SANTOS, Terezinha P.; SILVA, Marcelo M. da . **Os paradigmas de desenvolvimento de aplicativos para aparelhos celulares**. São Carlos, 2014. Disponível em: <http://revistatis.dc.ufscar.br/index.php/revista/article/view/86/80>. Acesso em 10 set. 2014.

SILVA, Maicon A. M. da et al. **Desenvolvimento de um caderno de campo para plataformas móveis utilizando PhoneGap**. Passo Fundo, 2013**.** Passo fundo: EMBRAPA, 2013. p. 1-6. Disponível em: <http://200.129.241.80/sbiagro/sbianais/paginas/trabalhos/118650_1.pdf>. Acesso em 10 set. 2014.

WARSKI, Adan. **How do iBeacons work?**. [S.l.], 2014. Disponível em: <http://www.warski.org/blog/2014/01/how-ibeacons-work/>. Acesso em 03 jun. 2015.

APÊNDICE A – Detalhamento dos casos de uso

A seguir é apresentado o detalhamento dos casos de uso, com uma descrição, pré e pós condições e cenário.

O caso de uso `UC01 – Visualizar eventos próximos` descreve a interação entre o usuário e a funcionalidade que permite visualizar os eventos. Detalhes sobre o caso de uso estão descritos no Quadro 20.

Quadro 20 – Caso de uso `UC01 – Visualizar eventos próximos`

Número	01
Caso de uso	Visualizar eventos próximos
Descrição	Este caso de uso permite o usuário visualizar as informações dos eventos que estão acontecendo dentro de um raio de 10 metros.
Ator	Usuário
Cenário principal	1. O usuário abre a aplicação. 2. O usuário toca na tela e arrasta para a direita abrindo painel lateral. 3. O usuário visualiza as informações dos eventos que estão sendo realizados e serão realizados próximos a sua localização atual.
Fluxo alternativo	No passo 3, caso o usuário necessite ouvir as informações, será executado o UC06 – Ouvir informações do evento No passo 3, caso o usuário necessite visualizar a posição no mapa, será executado o UC04 – Visualizar evento no mapa

O caso de uso `UC02 – Visualizar ambientes próximos` descreve a interação entre o usuário e a funcionalidade que permite visualizar os ambientes. Detalhes sobre o caso de uso estão descritos no Quadro 21.

Quadro 21 – Caso de uso `UC02 – Visualizar ambientes próximos`

Número	02
Caso de uso	Visualizar ambientes próximos
Descrição	Este caso de uso permite o usuário visualizar as informações dos ambientes que estão em um raio de 25 metros.
Ator	Usuário
Cenário principal	1. O usuário abre a aplicação. 2. O usuário toca na tela e arrasta para a esquerda abrindo painel. 3. O usuário visualiza as informações dos ambientes que estão próximos.
Fluxo alternativo	No passo 3, caso o usuário necessite ouvir as informações, será executado o UC07 – Ouvir informações do ambiente No passo 3, caso o usuário necessite visualizar a posição no mapa, será executado o UC05 – Visualizar ambiente no mapa

O caso de uso `UC03 – Visualizar posição atual no mapa` descreve a funcionalidade do aplicativo que permite o usuário visualizar a sua posição atual através do mapa. Detalhes sobre o caso de uso estão descritos no Quadro 22.

Quadro 22 – Caso de uso UC03 – Visualizar posição atual no mapa

Número	03
Caso de uso	Visualizar posição atual no mapa
Descrição	Este caso de uso permite o usuário visualizar a sua posição atual no mapa através de um marcador.
Ator	Usuário
Cenário principal	1. O usuário abre a aplicação. 2. O usuário visualiza um marcador com a sua posição atual

O caso de uso UC04 – Visualizar evento no mapa descreve a interação entre o usuário e a funcionalidade que permite visualizar a região no mapa onde o evento acontecerá ou está acontecendo. Detalhes sobre o caso de uso estão descritos no Quadro 23.

Quadro 23 – Caso de uso UC04 – Visualizar evento no mapa

Número	04
Caso de uso	Visualizar evento no mapa
Descrição	Este caso de uso permite o usuário visualizar a posição onde está o ambiente que ocorrerá ou está ocorrendo o evento.
Ator	Usuário
Cenário principal	1. O usuário seleciona um evento no painel lateral esquerdo 2. A aplicação exibe um marcador no mapa que informa o nome do evento, o local e o horário de início do evento

O caso de uso UC05 – Visualizar ambiente no mapa descreve a interação entre o usuário e a funcionalidade que permite visualizar o ponto no mapa onde o ambiente localiza-se. Detalhes sobre o caso de uso estão descritos no Quadro 24.

Quadro 24 – Caso de uso UC05 – Visualizar ambiente no mapa

Número	05
Caso de uso	Visualizar ambiente no mapa
Descrição	Este caso de uso permite o usuário visualizar a posição onde está o ambiente selecionado.
Ator	Usuário
Cenário principal	1. O usuário seleciona um ambiente no painel lateral direito 2. A aplicação exibe um marcador no mapa que informa o nome do evento, o local e o horário de início do evento

O caso de uso UC06 – Ouvir informações do evento descreve a interação entre o usuário e a funcionalidade que permite ouvir informações do evento. Detalhes sobre o caso de uso estão descritos no Quadro 25.

Quadro 25 – Caso de uso UC06 – Ouvir informações do evento

Número	06
Caso de uso	Ouvir informações do evento
Descrição	Este caso de uso permite o usuário ouvir as informações do evento selecionado.
Ator	Usuário
Cenário principal	1. O usuário seleciona um ambiente no painel lateral esquerdo 2. A aplicação fala o nome do evento, o local e o horário de início do evento

O caso de uso UC07 – Ouvir informações do ambiente descreve a interação entre o usuário e a funcionalidade que permite ouvir informações do ambiente. Detalhes sobre o caso de uso estão descritos no Quadro 26.

Quadro 26 – Caso de uso UC07 – Ouvir informações do ambiente

Número	07
Caso de uso	Ouvir informações do ambiente
Descrição	Este caso de uso permite o usuário ouvir as informações do ambiente selecionado.
Ator	Usuário
Cenário principal	1. O usuário seleciona um ambiente no painel lateral direito 2. A aplicação fala o nome do ambiente e a distância

O caso de uso UC08 – Ouvir informações da localização atual descreve a funcionalidade que permite ouvir informações sobre o que está acontecendo na região onde o usuário está. Detalhes sobre o caso de uso estão descritos no Quadro 27.

Quadro 27 – Caso de uso UC08 – Ouvir informações da localização atual

Número	08
Caso de uso	Ouvir informações da localização atual
Descrição	Este caso de uso permite o usuário ouvir as informações dos eventos que estão sendo realizados atualmente, ou serão realizados na próxima 1 hora a cada 1 minuto.
Ator	Usuário
Cenário principal	1. O usuário abre o aplicativo 2. O aplicativo fala o bloco e o andar onde o usuário está 3. O aplicativo fala o nome dos eventos que estão sendo realizados próximos a ele 4. O aplicativo fala o nome dos eventos que serão realizados na próxima hora próximos a ele
Fluxo alternativo	No passo 3, caso seja um evento: 1. O aplicativo fala o local do evento No passo 4, caso seja um evento: 2. O aplicativo fala o local do evento

O caso de uso UC09 – Filtrar tipos de eventos que são visualizados descreve a funcionalidade que permite exibir os eventos conforme o tipo (Interação FURB ou Normal). Detalhes sobre o caso de uso estão descritos no Quadro 28.

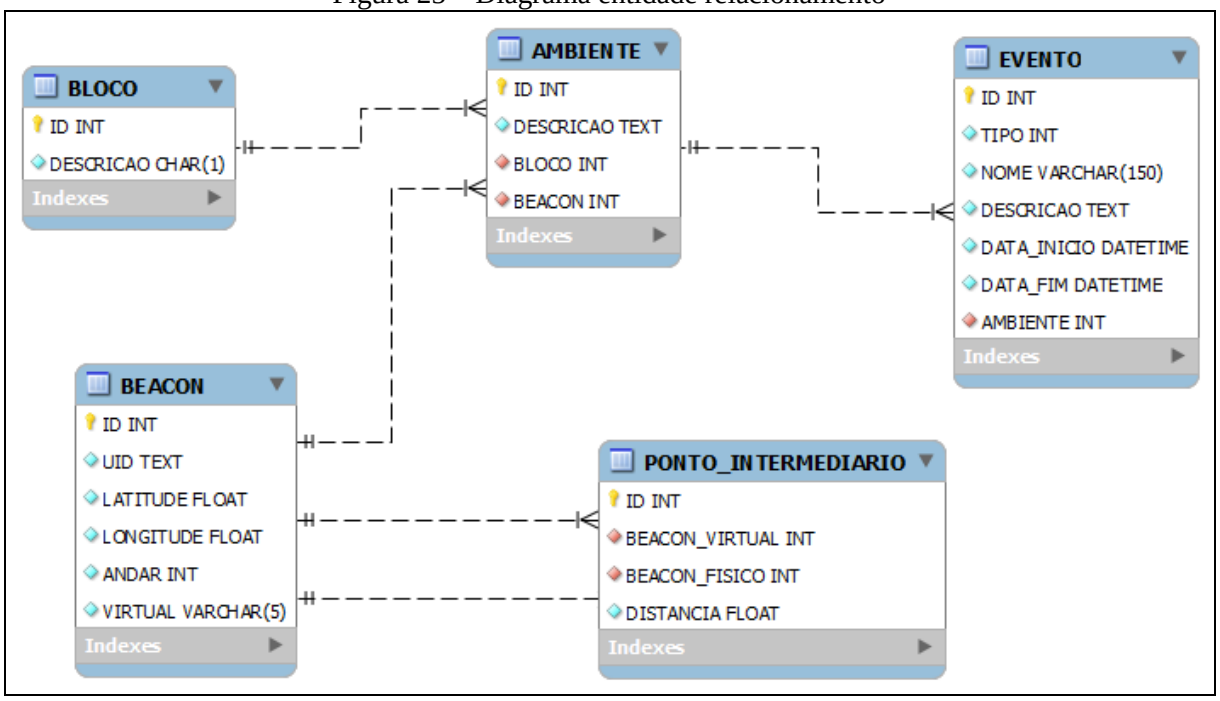
Quadro 28 – Caso de uso UC09 – Filtrar tipos de eventos que são visualizados

Número	09
Caso de uso	Filtrar tipos de eventos que são visualizados
Descrição	Este caso de uso permite o usuário filtrar quais eventos deseja visualizar na aplicação.
Ator	Usuário
Cenário principal	5. O usuário abre o aplicativo 6. O seleciona a opção de configurações 7. O usuário pode habilitar/desabilitar a visualização de eventos do tipo Interação FURB 8. O usuário pode habilitar/desabilitar a visualização de eventos do tipo Normal
Fluxo alternativo	No passo 7, caso nenhum filtro esteja habilitado: 3. A aplicação habilita o filtro novamente No passo 8, caso nenhum filtro esteja habilitado: 4. A aplicação habilita o filtro novamente

APÊNDICE B – Diagrama de Entidade Relacionamento

A Figura 25 representa o diagrama entidade relacionamento do banco de dados e também o modelo utilizado para estruturar as classes da aplicação.

Figura 25 – Diagrama entidade relacionamento



O diagrama entidade relacionamento apresentado na Figura 25, representa os dados que o aplicativo precisa para poder monitorar a região em que o usuário está e exibir as informações. A estrutura do banco de dados representa o domínio da aplicação, sendo assim, a estrutura do diagrama de classes é a mesma demonstrada no diagrama da Figura 25. A entidade base da arquitetura seria o `BEACON`, onde são armazenados o identificador único que faz a referência ao dispositivo físico utilizado para iniciar o monitoramento no aplicativo. A associação de 1-N do `BEACON` com `PONTO_INTERMEDIARIO` é utilizado para criar regiões entre os Beacons físicos. A entidade `AMBIENTE` faz uma associação com `BEACON` possibilitando criar a ligação entre o dispositivo e os ambientes próximos, o `BLOCO` é uma informação relacionada ao `AMBIENTE`, e os eventos são baseados nos ambientes, sendo assim o relacionamento 1-N entre `AMBIENTE` e `EVENTO` possibilita exibir vários eventos controlados pela `DATA_INICIO` e `DATA_FIM` no aplicativo.

APÊNDICE C – Questionário de perfil de usuário e avaliação do aplicativo

Neste apêndice constam o questionário e as instruções para os testes que os usuários seguiram. O questionário de perfil do usuário está no Quadro 29. O Quadro 30 contém as instruções para o usuário assim como o objetivo e o questionário de avaliação do aplicativo.

Quadro 29 – Questionário de perfil

PERFIL DE USUÁRIO

Observação: as informações recebidas abaixo serão mantidas de forma confidencial.

Sexo: ( ) Masculino ( ) Feminino

Idade:
( ) Tenho menos de 18 anos
( ) Tenho entre 18 e 25 anos
( ) Tenho entre 25 e 35 anos
( ) Tenho mais de 35 anos

Nível de escolaridade:
( ) Ensino fundamental incompleto
( ) Ensino fundamental completo - 1º grau
( ) Ensino médio incompleto
( ) Ensino médio completo - 2º grau
( ) Ensino superior incompleto
( ) Ensino superior completo

Você possui algum dispositivo móvel (smartphone/tablet)?
( ) Sim ( ) Não

Indique seu grau de familiaridade com aplicativos de geolocalização:
( ) Sou leigo nesses aplicativos
( ) Nunca utilizei nenhum aplicativo mas conheço a finalidade
( ) Utilizo raramente
( ) Utilizo diariamente aplicativos como Waze e Google Maps

Quadro 30 – Questionário de avaliação

INSTRUÇÕES

Com este questionário buscamos avaliar a utilização do aplicativo para georreferenciamento em ambientes restritos. Um dos objetivos do aplicativo é permitir o usuário saber onde ele está dentro da universidade, através da visualização do mapa. Outro objetivo é o usuário saber quais eventos estão ocorrendo próximos a ele, e exibir informações como o local e horário de início dos mesmos. Você pode utilizar o protótipo livremente por um período de 5 a 10 minutos para se ambientar. Ao finalizar, solicitamos que prossiga nos testes conforme as orientações abaixo.
Objetivo do experimento:
Encontrar a Oficina de Robótica entre 1º e 2º andar do bloco G.

Questionário de avaliação:

1. Você conseguiu chegar no seu objetivo utilizando o aplicativo?
   Sim ( ) Não ( )
Observações: _____
2. Você recebia as notificações e visualizava os eventos no aplicativo antes de visualizar os ambientes reais?
   Sim ( ) Não ( )
Observações: _____
3. O recurso de voz lhe auxiliou no trajeto?
   Sim ( ) Não ( )
Observações: _____
4. Você entendeu claramente os comandos de voz do aplicativo?
   Sim ( ) Não ( ) Parcialmente ( )
Observações: _____
5. Você usaria este aplicativo no seu dia a dia na universidade?
   Sim ( ) Não ( ) Talvez ( )
Observações: _____
6. Você indicaria este aplicativo à outra pessoa?
   Sim ( ) Não ( ) Talvez ( )
Observações: _____
7. Você acredita que esse aplicativo poderia auxiliar novos alunos na universidade?
   Sim ( ) Não ( )
Observações: _____
8. O mapa do aplicativo exibindo a sua localização atual facilitou a locomoção pela universidade?
   Sim ( ) Não ( )
Observações: _____
9. De um modo geral, você achou o aplicativo intuitivo e fácil de usar?
   Sim ( ) Não ( )
Observações: _____
10. Qual é a sua avaliação do aplicativo?
   ( ) Muito bom ( ) Bom ( ) Regular ( ) Insatisfatório
Observações: _____
_____