Faculdade de Tecnologia de Sorocaba – José Crespo Gonzales
Tecnologia em Análise e Desenvolvimento de Sistemas

RELATÓRIO FINAL - Iniciação Científica

Marcos Rogério Franci Ferreira Leão Filho

Beaxer – Análise de Ferramentas e Desenvolvimento

Sorocaba
Julho-2017

Faculdade de Tecnologia de Sorocaba – José Crespo Gonzales
Tecnologia em Análise e Desenvolvimento de Sistemas

Beaxer – Análise de Ferramentas e Desenvolvimento

Marcos Rogério Franci Ferreira Leão Filho

Orientadora: Profª Mª Denilce de Almeida Oliveira Veloso

Sorocaba
Julho-2017

RESUMO

A proposta deste projeto é a pesquisa de ferramentas necessárias e o desenvolvimento do protótipo de um aplicativo fundamentado no marketing utilizando *beacons*. A partir da instalação do aplicativo nos *smarthphones* dos clientes e de *Beacons* estrategicamente localizados na loja, será possível enviar notificações a um cliente em potencial assim que ele entra na loja, essas notificações podem exibir desde promoções e descontos até detalhes sobre determinado produto. *Beacon* em português, farol, são pequenos dispositivos que emitem sinais *Bluetooth* para dispositivos que possuem a tecnologia *bluetooth* 4.0 ou superior, conhecido como *Bluetooth Low Energy (*BLE). A diferença entre a versão do *bluetooth* 4.0 em relação as anteriores é simplesmente seu baixo consumo de energia, devido ao fato de que o foco da versão atual não seja a transferência de arquivos, e sim a emissão de UUIDs, códigos *Minors* e *Majors*, códigos esses que realizam a identificação única de um *beacon*. A aplicação será desenvolvida para dispositivos Android. Será utilizado um IDE (Ambiente de desenvolvimento integrado) da empresa Embarcadero, atual responsável pelo desenvolvimento de IDE's para a linguagem Object Pascal, popularmente conhecida como Delphi.

Palavras-Chave: App, Marketing, Beacons, Android, Delphi.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

ABNT – Associação Brasileira de Normas Técnicas.

BLE – *Bluetooth Low Energy* (Bluetooth de baixa energia).

GPS – *Global Position System* (Sistema de posicionamento global).

HTML - *HyperText Markup Language* (Linguagem de Marcação de Hipertexto).

IDE - Ambiente de desenvolvimento integrado.

IMEI – *International Mobile Equipment Identity* (Identidade internacional de equipamento móvel).

UUID – *Universally Unique Identifier* (Identificador único universal).

SUMÁRIO

1 INTRODUÇÃO E JUSTIFICATIVA

Para a efetividade de um *beacon* é necessário que haja uma aplicação para realizar a identificação do mesmo, ou seja, um *beacon* é apenas um emissor de sinal sem utilidade quando não há um aplicativo que realize funções pré-programadas ao reconhecer estes identificadores. Também é possível realizar cálculos de distância de emissão de sinal, desta forma sendo possível saber de maneira aproximada qual a distância de um determinado *beacon*, e com esta informação realizar diferentes funções para cada intervalo de distância de um *beacon* e o dispositivo que está recebendo seu sinal. O alcance de um *Beacon* varia de acordo com seu fabricante, os *beacons* da empresa Estimote, também conhecidos como *iBeacons* variam de 7 a 200 metros de alcance.

Os *Beacons* são utilizados neste projeto para realizar:

- Reconhecimento de *beacons* próximos ao dispositivo que possui a aplicação instalada.
- Exibição da descrição do local onde o usuário se encontra.
- Exibição do IMEI do dispositivo ao reconhecer determinado *beacon* próximo ao dispositivo que possui o aplicativo.
    - O envio de notificações de possíveis promoções e cupons de descontos para usuários que estiverem passando com seus dispositivos dentro da região de emissão de sinal do *beacon*, sendo o objetivo desta funcionalidade a atração de possíveis

consumidores para o estabelecimento que possuir esta tecnologia.

Simulando um cenário onde um potêncial cliente caminha em um shopping qualquer com a aplicação (da loja x) instalada no seu *smarthphone*, com a opção *bluetooth* ligado. Este cliente caminha sem a intenção de ir na loja x, porém, ao passar em frente a loja, o aplicativo reconhece o *Beacon* e envia uma notificação ao cliente informando que ele acaba de ganhar um *vouncher* de desconto de 50% em determinado produto, isso aumentaria significativamente a possibilidade do cliente adentrar esta loja e consumir até mesmo mais do que o produto no qual ele ganhou o desconto, e ao começar a percorrer os corredores da loja x. A aplicação poderá estar contando o tempo em que ele permanece em um corredor e enviar estas informações para um banco de dados, assim gerando estatísticas de demanda de produtos e até mesmo criando um perfil para aquele consumidor específico, perfil este que poderá ser utilizado para envio de promoções exclusivas para este cliente.

É possível desenvolver o aplicativo em questão com diversas ferramentas e linguagens de programação, sendo algumas delas:

- Android studio: IDE desenvolvido pela própria Google, desenvolvedora do sistema operacional Android, sistema este que rodará a aplicação que se comunicará com o *Beacon*, utiliza a linguagem de programação Java e o layout da aplicação é realizado através de componentes nativos do IDE e através de XML e sua licença é gratuita.

- Eclipse: Eclipse era o principal IDE para desenvolvimento de aplicativos Android antes do lançamento do Android Studio em 2013, utiliza a linguagem Java para o desenvolvimento e XML para o layout das aplicações.

- Delphi: IDE mantido pela empresa Embarcadero, sendo sua licença paga para empresas e possuindo versão gratuita para estudos e pesquisas, possui suporte para desenvolvimento de aplicação *mobile* em suas versões mais recentes, sendo possível o desenvolvimento de aplicações para sistemas operacionais Windows Phone, mantido pela Microsoft, Android, mantido pela Google e iOS, mantido pela Apple, utiliza a linguagem de programação C++ e Object Pascal, popularmente conhecida como Delphi.

Para o desenvolvimento desta aplicação será utilizado o IDE Delphi 10.1 Berlin, devido a facilidade de desenvolvimento e por se tratar de uma tecnologia pouco utilizada para o desenvolvimento de aplicações mobile.

Uma desvantagem de se utilizar o *beacon* é a necessidade do *bluetooth* dos dispositivos móveis estarem ligados, algo que ainda não é comum em nossa realidade devido ao fato de que versões anteriores do *bluetooth* consumirem muita energia e a falta de segurança.

Um exemplo de tecnologia que não era frequentemente ativa nos dispositivos móveis é o GPS, tanto pelo fato do consumo excessivo de bateria e devido a não necessidade de utilização, porém, com o lançamento de um

jogo desenvolvido pela empresa Niantic para *smartphones* chamado Pokemon Go, diversos usuários começaram a utilizar a função GPS de seus aparelhos.

2 REVISÃO BIBLIOGRÁFICA

2.1 Beacons

Beacons são pequenos dispositivos que realizam a emissão de sinal *Bluetooth* 4.0, são enviados através desta emissão de sinal códigos identificadores do *Beacon*, tais como o UUID, *minor* e *major*, sendo o UUID um código composto por 16 *bytes* podendo conter letras e números, sua sigla em português significa Identificador Universal Exclusivo, este código é responsável por diferenciar os *beacons* de um fabricante para outro, por exemplo: *Beacons* da empresa Estimote possuem o mesmo UUID, porém seus identificadores *Majors* e *Minors* são diferentes dos outros, e um *Beacon* de outra empresa não poderá ter o mesmo UUID que um *beacon* da empresa Estimote. O código *Major* e o *Minor* possuem apenas 2 *bytes*, estes códigos são responsáveis por diferenciar um *beacon* do outro, é possível que *Beacons* do mesmo fabricante possuam o mesmo Código *Major* ou o Código *Minor*, mas nunca os dois iguais. (Ibeacon, 2017).

Segundo Pinheiro (2004), que também é projetista e gestor de redes, membro da BICSI (*Advancing the information and communications technology community*), autor dos livros Guia Completo de Cabeamento de Redes, Cabeamento Óptico, Infraestrutura Elétrica para Redes de Computadores e

Biometrica nos Sistemas Computacionais – Você é a Senha, o *Bluetooth* é uma tecnologia *Wireless* (sem fio) e móvel que permite que dispositivos eletrônicos como computadores, celulares, impressoras, etc, se comuniquem em curta distância sem utilização de nenhum cabeamento. (Pinheiro, 2004).

O *bluetooth* 4.0 ou superior, também conhecido como BLE (*Bluetooth Low Energy*) é a versão de baixo consumo de energia das versões *Bluetooth* anteriores, aparelhos que não necessitam da transmissão de grande quantidade de dados tem seu gasto energético reuzido em 10% em relação as versões anteriores, devido a esta nova versão consumir tão pouca energia, a bateria dos dispositivos *Beacons* podem até 2 anos em funcionamento contínuo. Se tratando de dispositivos móveis como *smartphones*, notebooks, impressoras e outros dispositivos que podem realizar a transferência de uma quantidade maior de dados em relação as informações transmitidas pelos *beacon*s, a tecnologia *Bluetooth Low Energy* possui uma função de economia de energia, ou seja, quando o *bluetooth* está ligado em algum destes dispositivos e nenhuma transferência ou recebimento de dados está ocorrendo, automaticamento o *bluetooth* entra em modo *sleep,* desta forma não gastando uma quantidade significativa de energia.(*Bluetooth low energy*, 2017).

A duração da bateria de dispositivos *Beacons* pode chegar a mais de dois anos devido ao seu baixo consumo de energia, sendo um fator importante para a viabilidade da utilização desta tecnologia, desta forma não havendo necessidade de manutenção frequente nos dispositivos, sendo necessário apenas configurar a aplicação em relação ao que será exibido ao cliente através de seus identificadores e distâncias dos dispositivos *beacons* e móveis.

2.2 Linguagem de Programação

Para Bolton (2016), linguagem de programação é usada para desenvolver programas e aplicativos para equipamentos eletrônicos, alguns exemplos de linguagem de programação são a linguagem Pascal, a versão orientada a objetos do Pascal que é a Object Pascal, também conhecida como Delphi, C, C++, Java, Python, PHP, C#, entre outras. Dentre estas linguagens existem as linguagens compiladas e interpretadas, o Object Pascal é uma linguagem compilada.(Bolton, 2016).

A programação orientada a objetos é um tipo de programação que realiza a abstração do mundo real para o digital, visa criar classes que possuam características que definem objetos na vida real, por exemplo: se o objetivo fosse construir uma cadeira com uma linguagem de programação orientada a objetos teria-se a classe Cadeira com seus atributos pernas, assento, estofado e encosto. Neste caso tem-se a classe Cadeira, mas podem exister vários tipos de cadeira, desta forma, utilizando uma linguagem de programação orientada a objetos não seria necessário criar uma classe para cada tipo de cadeira, seria possível construir cadeiras passando suas características, tais como cadeira A possui 3 pernas, encosto alto, assento de madeira e estofado de algodão, e utilizando a mesma classe, poderia-se construir a cadeira de 4 pernas, assento de metal, encosto baixo e sem estofamento. Já na linguagem de programação estrturada este processo seria realizado para cada tipo de cadeira que fosse criada, desta forma repetindo código e deixando o desenvolvimento mais pesado. (Abílio, 2016)

2.3 Delphi 10.1

Delphi é um Ambiente de Desenvolvimento Integrado para desenvolvimento na linguagem Object Pascal criado pela empresa Borland. Inicialmente chamado de Turbo Pascal, o Delphi foi lançado oficialmente em 1995. Em meados de 2007, após a criação da CodeGear, divisão da Borland responsável por compiladores foi lançada a versão do Delphi chamada RAD Studio 2007, nesta época uma empresa chamada Embarcadero Technologies comprou a CodeGear e passou a deter os direitos legais sobre todos os compiladores Delphi. A Embarcadero lança em agosto de 2010 a primeira versão da família XE, chamada Delphi XE, uma versão *cross* plataforma de compiladores, onde era possível o desenvolvimento de aplicação para plataforma Amazon EC2 e Microsoft Azure, além das outras plataformas já existentes nas versões anteriores, tais como o Windows. Como forma de acompanhar o mercado, as versões posteriores foram incluindo desenvolvimento para as plataformas mais atuais, tais como a versão XE 4 com suporte para desenvolvimento para sistemas operacionais da Apple e a versão XE5 com suporte para sistemas operacionais Android (Duarte, 2015). Na versão Delphi 10.1 Berlin, a penúltima versão lançada pela empresa até o momento, novas funcionalidades surgiram, tais como componentes próprios para o desenvolvimento de aplicações que se comunicam com *Beacons*, o componente TBeacon, este componente é o responsável pela conexão entre o aplicativo e o dispositivo *Beacon*, sem a necessidade de que toda esta comunicação seja realizada de forma manual, sendo necessário apenas o desenvolvimento das funcionalidades da aplicação, assim como a exibição da

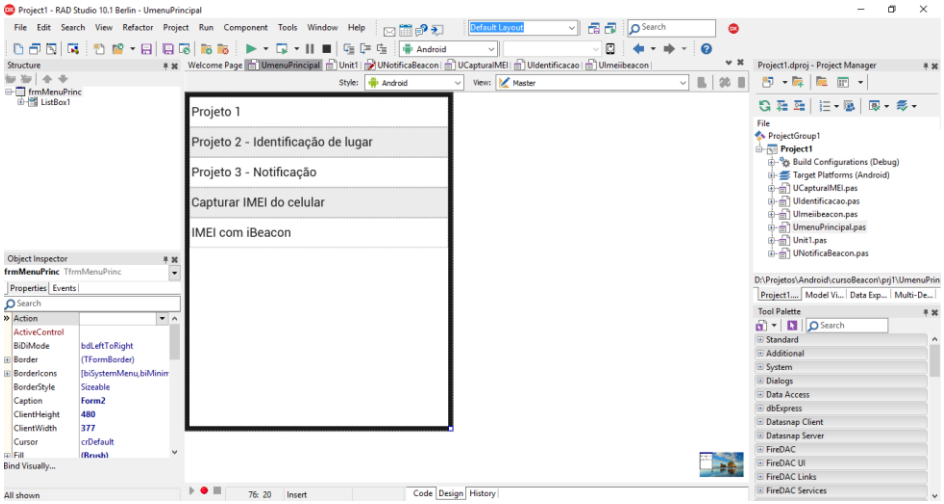
descrição de determinado produto através do reconhecimento de proximidade entre o dispositivo que contém a aplicação desenvolvida e o *Beacon*. (Delphi, 2017).

O Delphi 10.1 Berlin (figura 1) possui componentes próprios para possibilitar que a aplicação se comunique com *beacons*, este componente se chama TBeacon, sendo necessário apenas inserir o componente no projeto e ativar o componente, fazendo isso, o aplicativo reconhecerá todos os dispositivos *beacons* ao redor do ambiente em que está sendo executado, respeitando o limite de distância de cada dispositivo. O TBeacon (figura 2) possui a opção de filtrar quais *beacons* serão reconhecidos pela aplicação, este filtro é realizado atráves de um pré cadastro dos identificadores existentes nos *beacons*, sendo o identificador UUID, código hexadecimal que comumente possui 32 digitos, sendo este código um padrão para cada fabricante, ou seja, todos os beacos de uma determinada empresa possui o mesmo UUID, código *major* e *minor*, identificadores numéricos que variam do 0 ao 65535, estes códigos são responsáveis pela diferenciação de beacos do mesmo fabricante, nunca haverá mais de um *beacon* do mesmo fabricante com estes dois códigos iguais.

É através destes códigos identificadores que é possível realizar as funções programadas, por exemplo: um supermercado possui dois dispositivos *beacons*, um se encontra no setor de chocolates e o outro no setor de bebidas, o *beacon* do setor de chocolates possui o UUID x, código *Minor* y e código *Major* z, o *beacon* do setor de bebidas possui UUID a, código *Major* b e código *Minor* c. Então, um consumidor específico possui o aplicativo do supermercado

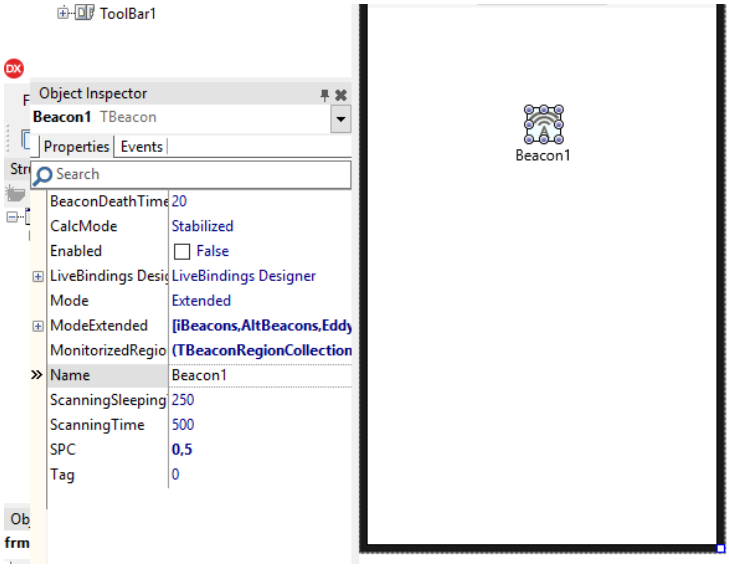
em seu aparelho celular e seu *bluetooth* está ligado, quando este consumidor chega na região de emissão de sinal do *beacon* localizado em um desses dois setores, a aplicação reconhece estes códigos identificadores, verifica para qual setor estes códigos foram cadastrados e análisa se o consumidor se encontra na distância mínima pré cadastrada deste dispositivo para reconhecer que ele efetivamente se encontra em determinado setor do estabelecimento, após realizar estas funções, a aplicação consegue enviar para um banco de dados quantos clientes passaram por aquele local, qual seu tempo de permanência no local e até mesmo enviar uma notificação para o dispositivo do cliente contendo alguma informação ou desconto de algum produto daquele setor, e com estes dados é possível gerar estratégias de negócios atráves de estatísticas capturadas de forma rápida, sem a necessidade de realizar pesquisas diretamente com o cliente e ainda aumentar a probabilidade do mesmo consumir algo após receber a informação de que algum produto se encontra em promoção.

Figura 1 – Ambiente de desenvolvimento integrado (RAD Studio 10.1 Berlin).



Fonte: (Autor, 2017).

Figura 2 - Componente TBeacon utilizado para a comunicação com bracons.



Fonte: (Autor, 2017).

3 OBJETIVO

O objetivo deste projeto de iniciação científica é a pesquisa de ferramentas e desenvolvimento de um protótipo de uma aplicação fundamentada em marketing que realize a comunicação com dispositivos *Beacons*, onde será possível o envio de notificações contendo promoções e informações sobre determinados produtos previamente cadastrados.

4 MATERIAIS E MÉTODOS

Inicialmente foi realizada uma pesquisa para descobrir quais ferramentas e materiais poderiam ser utilizados para o desenvolvimento do aplicativo.

Após o período inicial de instalação das ferramentas, partiu-se para a criação dos arquivos do projeto e início do desenvolvimento do protótipo.

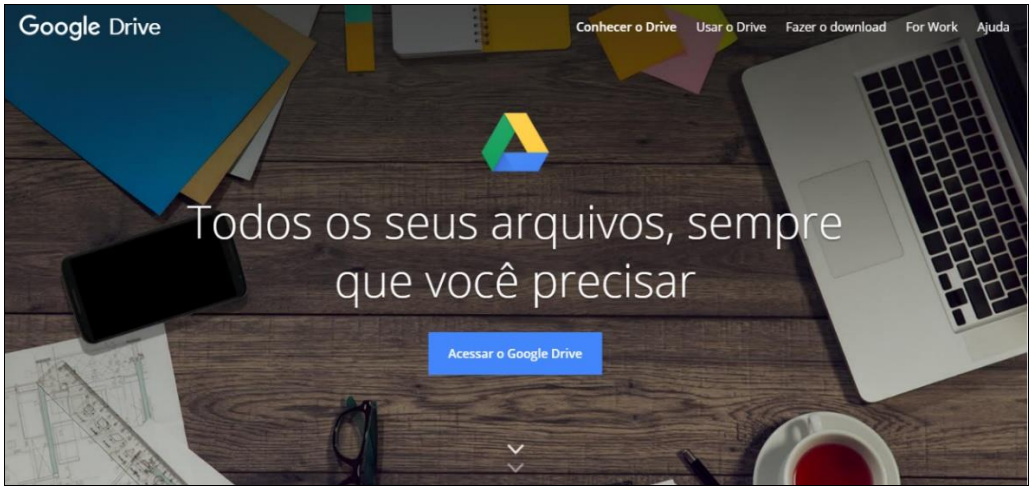
4.1 INTEGRATED DEVELOPMENT ENVIRONMENT (IDE) OU AMBIENTE DE DESENVOLVIMENTO INTEGRADO.

A ferramenta escolhida para desenvolvimento foi o Delphi 10.1 (cf 2.3)

4.2 COMPARTILHAMENTO DE ARQUIVOS

O Google Drive (figura 3) foi utilizado para armazenamento (figura 4), compartilhamento e edição de arquivos e documentos que foram desenvolvidos no projeto. Google Drive é uma ferramenta de armazenamento e compartilhamento de arquivos online, desenvolvida pela empresa Google em 24 de Abril de 2012.

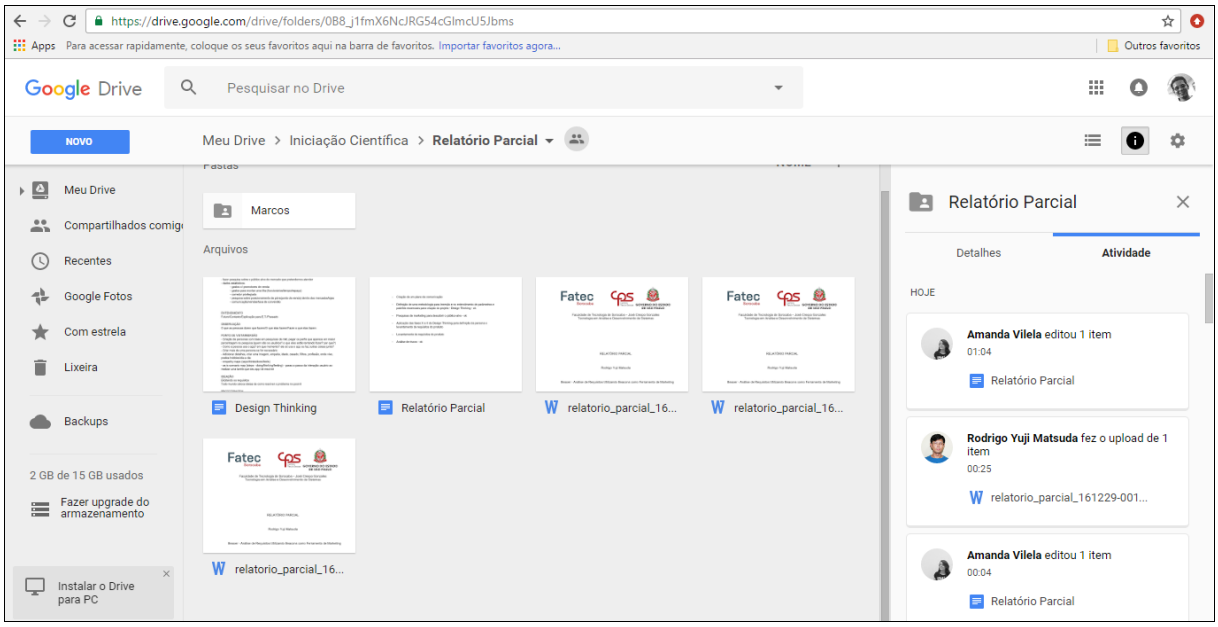
Figura 3 – Página inicial Google Drive.



Fonte: (Autor, 2016).

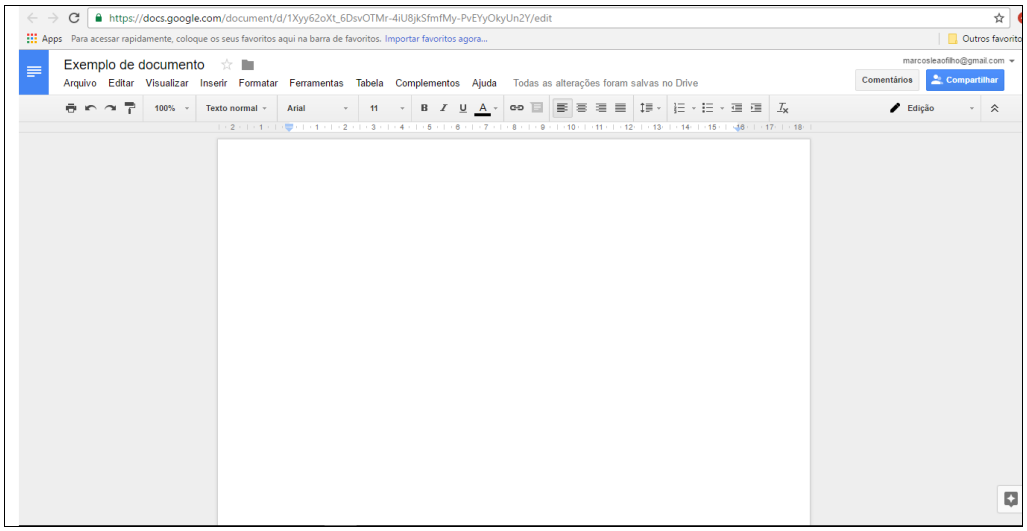
Com esta ferramenta é possível a criação de documentos de texto (figura 5) compatíveis com os aplicativos de edição da Microsoft, o pacote Office, sendo necessário apenas realizar a formatação para os padrões ABNT após a finalização do documento, pois por se tratar de uma ferramenta online as opções de formatação são reduzidas. Outra função interessante do Google drive é o armazenamento de resultados da ferramenta de formulários da Google, o Google Forms. Ao se realizar alguma pesquisa com esta ferramenta, automaticamente será armazenado um arquivo de planilha compatível com o Excel da Microsoft.

Figura 4 – Diretório de armazenamento e compartilhamento de arquivos no Google Drive.



Fonte: (Autor, 2016).

Figura 5 – Editor de texto online do Google Drive.



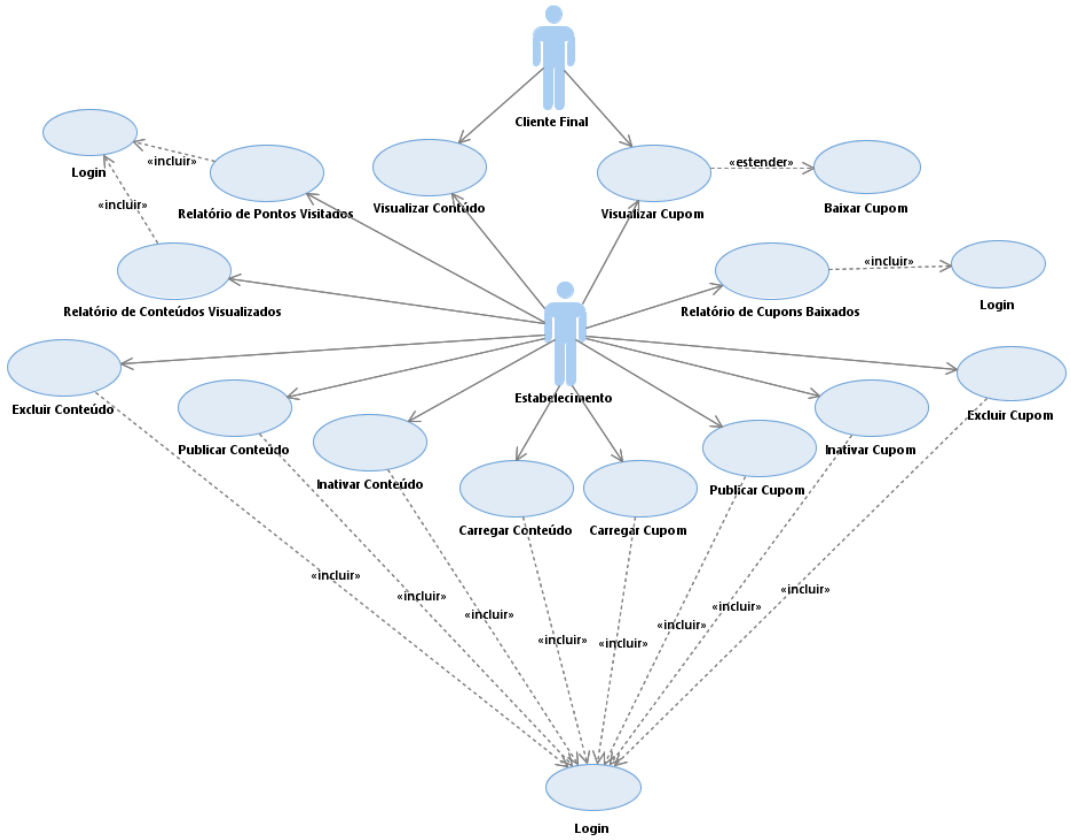
Fonte: (Autor, 2016).

4.3 ANÁLISE DE REQUISITOS

O protótipo foi desenvolvido utilizando-se de informações disponíveis de (Matsuda,2017). O autor disponibilizou diversos artefatos que podem ser utilizados para o desenvolvimento (implementação) do aplicativo.

Para o desenvolvimento do protótipo da aplicação foi utilizada a ideia de exibição de cupons de desconto (figura 6) através de notificações enviadas pelo aplicativo para o dispositivo do usuário, sendo possível através do desenvolvimento da aplicação completa, obter relatórios de pontos visitados através da captura do IMEI do dispositivo do usuário ou através do email do usuário cadastrado na aplicação. Para o nível de cliente, em uma aplicação completa, somente seria exibido a visualização de conteúdo existente na aplicação, sendo este conteúdo possíveis produtos pré cadastrados e visualização de cupons liberados pelo administrador da aplicação.

Figura 6 – Diagrama de Caso de Uso de Alto Nível



Fonte: (Matsuda, 2017).

4.4 EXPERIÊNCIA DO USUÁRIO E INTERFACE

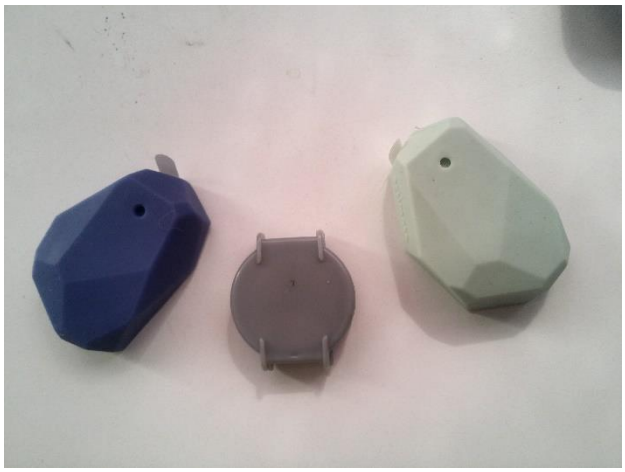
Como o objetivo principal deste projeto era desenvolver uma aplicação em Object Pascal visando principalmente a conexão com os *beacons* e exibição de notificações contendo informações de produtos e/ou cupons promocionais, inicialmente a tela ficou simples, mas funcional. Em um próximo passo poderão ser aproveitadas a parte de Interface e Experiência do Usuário propostos por (Almeida,2017).

5 RESULTADOS E DISCUSSÕES

5.1 Funcionalidades da aplicação

A aplicação desenvolvida realiza a comunicação entre os dispositivos móveis e *beacons* (figura 7), buscando entender e testar as funcionalidades de um *beacon*. A aplicação identifica o local onde o usuário se encontra exibindo a descrição do local previamente programada, exibe detalhes sobre determinado produto ao se aproximar do *beacon*, captura o identificador do *smartphone* (IMEI), mostra a distância entre o *smatphone* que contém a aplicação e o *beacon*, realiza a contagem de *beacons* encontrados pela aplicação e eventualmente envia uma notificação para o usuário contendo uma promoção.

Figura 7 - Beacons utilizados para o desenvolvimento e teste da aplicação.



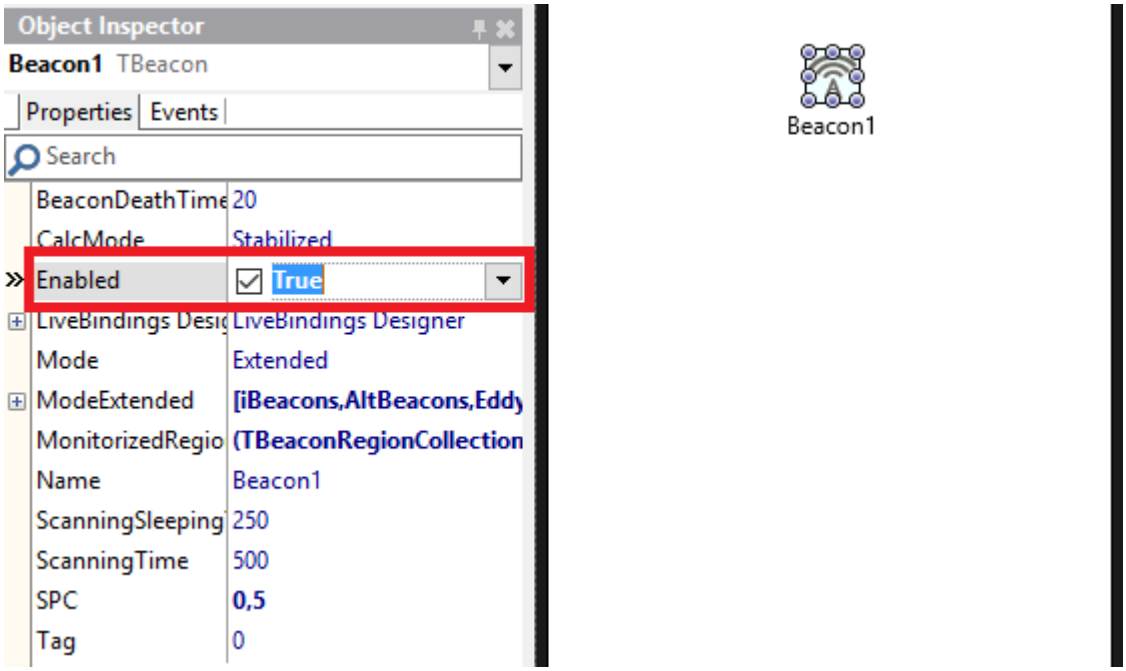
Fonte: (Autor, 2017).

Para o desenvolvimento da aplicação que irá se comunicar com o *beacon* foi utilizado o IDE RAD Studio 10.1 Berlin (Imagem 1) por ser um IDE

que utiliza a linguagem Object Pascal para desenvolvimento e por não ser muito utilizada para o desenvolvimento de aplicações para dispositivos móveis.

Para que a aplicação desenvolvida reconheça os *Beacons* próximos ao *smartphone* ou outro dispositivo móvel que contenha a aplicação desenvolvida, é necessário inserir o componente TBeacon no projeto da aplicação e configurar sua propriedade “Enabled” para “true”, desta forma a aplicação reconhecerá todos os *beacons* que estiverem próximos do smartphone, conforme figura 8.

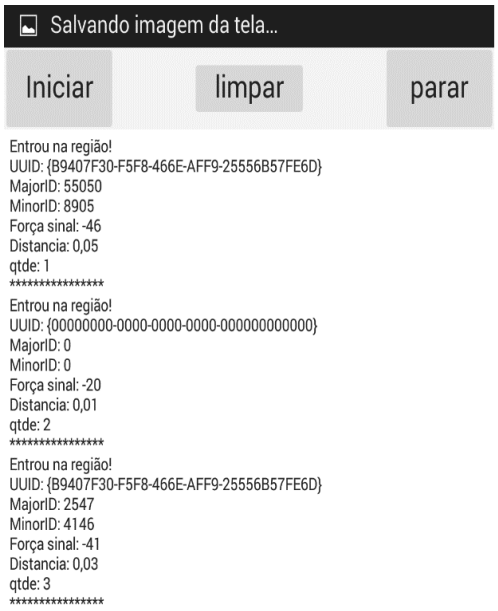
Figura 8 – Propriedade “Enabled” do componente TBeacon ativada.



Fonte: (Autor, 2017).

Como apresentado na figura 9, foram identificados três *beacons*, sendo dois do mesmo fabricante e um *beacon* genérico, cujos códigos identificadores se apresentam zerados, já nos outros dois *beacon*, o código que identifica o fabricante (UUID) são iguais para os dois, alterando apenas os códigos utilizados para distinguir os dispositivos, sendo estes os códigos *Minor* e *Major*. Também é possível verificar a quantidade de *beacons* encontrados, a força do sinal sendo recebido pelo smartphone e uma distância aproximada entre os *beacons* e o smartphone que possui a aplicação. Na figura 10, é exibida parte do código utilizado no aplicativo.

Figura 9 – Aplicativo buscando por beacons nas proximidades.



Fonte: (Autor, 2017).

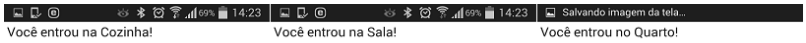
Figura 10 – Trecho de código fonte da aplicação.

```
procedure TfrmPrj1.Beacon1BeaconEnter(const Sender: TObject;
  const ABeacon: IBeacon; const CurrentBeaconList: TBeaconList);
begin
  memo1.Lines.Add('Entrou na região!');
  memo1.Lines.Add('UUID: ' + abeacon.GUID.ToString);
  memo1.Lines.Add('MajorID: ' + abeacon.Major.ToString);
  memo1.Lines.Add('MinorID: ' + abeacon.Minor.ToString);
  memo1.Lines.Add('Força sinal: ' + abeacon.Rssi.ToString);
  memo1.Lines.Add('Distancia: ' + abeacon.Distance.ToString);
  memo1.Lines.Add('qtde: ' + length(CurrentBeaconList).ToString);
  memo1.Lines.Add('*****************');
end;
```

Fonte: (Autor, 2017).

Para uma identificação precisa de locais dentro de algum estabelecimento é necessário a utilização de mais de um *beacon*, desta forma configurando a aplicação para a exibição do local em que o usuário se encontra. Para que a aplicação saiba o que exibir é necessário a utilização dos códigos identificadores dos *beacons*. No exemplo a seguir foram programados na aplicação a exibição da localização “Quarto” para quando o aplicativo reconhecer o UUID B9407F30-F5F8-466E-AFF9-25556B57FE6D, código *Major* 2547 e *Minor* 4146, para exibir a localização “Sala” foi programado para quando a aplicação reconheça o *beacon* com o UUID B9407F30-F5F8-466E-AFF9-25556B57FE6D, código *Major* 55050 e *Minor* 8905 e para exibir a localização “Cozinha” o UUID 00000000-0000-0000-0000-000000000000, *Major* 0 e *Minor* 0 (Figura 11). sendo exibido na Figura 12 um trecho do código escrito para realizar as funções acima utilizando a linguagem Object Pascal.

Figura 11 – Aplicação identificando locais.



Fonte: (Autor, 2017).

Figura 12 – Trecho do código de identificação de locais.

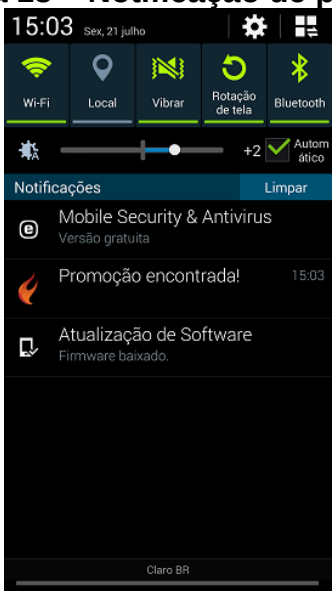
```
procedure TfrmIdentificar.Beacon1BeaconProximity(const Sender: TObject;
  const ABeacon: IBeacon; Proximity: TBeaconProximity);
begin
  if (abeacon.Major = 0) and (abeacon.Minor = 0) and
  (ABeacon.GUID.ToString = '{00000000-0000-0000-0000-000000000000}') then
  begin
    memo1.Lines.Add('Você entrou na Cozinha!');
  end
  else
  if (abeacon.Major = 2547) and (abeacon.Minor = 4146) and
  (ABeacon.GUID.ToString = '{B9407F30-F5F8-466E-AFF9-25556B57FE6D}') then
  begin
    memo1.Lines.Add('Você entrou no Quarto!');
  end
  else
  if (abeacon.Major = 55050) and (abeacon.Minor = 8905) and
  (ABeacon.GUID.ToString = '{B9407F30-F5F8-466E-AFF9-25556B57FE6D}') then
  begin
    memo1.Lines.Add('Você entrou na Sala!');
  end;
end;
```

Fonte: (Autor, 2017).

O envio de notificações para o usuário é necessário para que o mesmo não precise estar olhando para o *smartphone* a todo momento, desta forma, como uma estratégia de marketing, quando o cliente passa por perto de uma loja que possua *beacon*, o aplicativo reconhecerá e exibirá uma notificação (Figura 13) qualquer para o usuário, podendo ser alguma propaganda ou até

mesmo uma promoção relâmpago que ao ser clicada abrirá o aplicativo exibindo mais detalhes sobre o que está sendo exibido nas notificações (Figura 14). Nas imagens a seguir estão a demonstração de uma notificação e a abertura do aplicativo ao clicar na notificação.

Figura 13 – Notificação de promoção encontrada.



Fonte: (Autor, 2017).

Figura 14 – Aplicativo aberto no produto em promoção.

Fonte: (Autor, 2017).

6 CONCLUSÃO

Foi realizado o desenvolvimento de um protótipo que realiza algumas funções de conexão com dispositivos *beacon*, tais como reconhecimento de *beacons* encontrados pela aplicação, exibindo os códigos identificadores, intensidade de sinal, distância aproximada, quantidade de *beacons* encontrados, notificação de produtos em promoção e identificação de local, sendo estas funcionalidades estudadas, de total requerimento para o desenvolvimento de uma aplicação comercial real, disponibilizando ao comerciante uma ferramenta de marketing que o auxiliará com a divulgação de seus produtos, podendo publicar detalhes de seus produtos em web sites que serão acessados pela aplicação que realizará a conexão com o dispositivo *Beacon*, e para que o usuário tenha um motivo para manter a aplicação instalada em seu *smartphone*, a funcionalidade de envio de promoções e cupons de descontos exclusivos é indispensável na aplicação.

A partir de análise de ferramentas, neste projeto foi utilizado Google Drive para armazenamento e compartilhamento de documentos, Delphi 10.1 Berlin para o desenvolvimento da aplicação Mobile e pacote office para a documentação.

REFERÊNCIAS

Abílio, Igor. **Programação Orientada a Obejtos Versus Programação Estruturada.** Disponível em: < http://www.devmedia.com.br/programacao-orientada-a-objetos-versus-programacao-estruturada/32813 >. Acesso em: 13 jul. 2017

Almeida. Amanda Vilela de. **Projeto de Iniciação Científica: Beaxer – Interface e Experiência do Usuário.** Faculdade de Tecnologia de Sorocaba "José Crespo Gonzales". 2017.

Bluetooth. **Bluetooth Low Energy.** Disponível em: <http://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2012_2/bluetooth/ble.htm>. Acesso em: 13 jul. 2017.

Bolton, David. **What is a Programming Language?.** Disponível em: < https://www.thoughtco.com/what-is-a-programming-language-958332 >. Acesso em: 13 jul. 2017.

DELPHI. **Delphi: Conheça a história da ferramenta mais atual que existe, há 20 anos!.** Disponível em: < http://www.tdevrocks.com.br/2015/10/02/delphi-conheca-a-historia-da-ferramenta-mais-atual-que-existe-ha-20-anos/ > Acesso em: 13 jun. 2017.

Duarte. William. **Delphi para android e iOS. Desenvolvendo aplicativos móveis.** São Paulo: Brasport, 2015.

iBeacon. **Developer Estimote iBeacon** Disponível em: <http://developer.estimote.com/ibeacon/>. Acesso em: 13 jul. 2017.

Matsuda. Rodrigo Yuji. **Projeto de Iniciação Científica: Beaxer - Análise de Requisitos Utilizando Beacons como Ferramenta de Marketing**. Faculdade de Tecnologia de Sorocaba "José Crespo Gonzales". 2017.

Pinheiro, José Maurício Santos. **Por Dentro Do Bluetooth.** Disponível em: < http://www.projetoderedes.com.br/artigos/artigo_dentro_bluetooth.php >. Acesso em: 13 jul.2017.