LUCAS HENRIQUE MENDES LAZARO
ENLAI LIMA CHENG
LUCAS NEGRÃO COUCEIRO

BOXES : INTERAGINDO COM CIDADES HÍBRIDAS POR MEIO DE DISPOSITIVOS MÓVEIS

São Paulo
2015

LUCAS HENRIQUE MENDES LAZARO
ENLAI LIMA CHENG
LUCAS NEGRÃO COUCEIRO

BOXES : INTERAGINDO COM CIDADES HÍBRIDAS POR MEIO DE DISPOSITIVOS MÓVEIS

Trabalho apresentado à Escola Politécnica
da Universidade de São Paulo para a
Graduação em Engenharia de Computação.

São Paulo
2015

LUCAS HENRIQUE MENDES LAZARO
ENLAI LIMA CHENG
LUCAS NEGRÃO COUCEIRO

BOXES : INTERAGINDO COM CIDADES HÍBRIDAS POR MEIO DE DISPOSITIVOS MÓVEIS

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para a Graduação em Engenharia de Computação.

Orientador:
Profa. Dra. Lucia Vilela Leite Filgueiras

Co-orientador
Prof. Dr. James Ohene-Djan

São Paulo
2015

DEDICATÓRIA

Aos meus pais, José Alberto e Rosemeire, ao meu irmão Junior e a minha Família que tanto me apoiaram nas decisões tomadas nesta jornada.

Aos tantos amigos e colegas que tornaram essa experiência muito mais divertida e educativa do que eu esperava.

- Lucas Henrique Mendes Lazaro

Dedico este trabalho ao meus pais Cheng e Anamaria e minha namorada Fernanda, que tanto me ajudaram nessa longa jornada, principalmente nos meses finais do curso.

Aos amigo que compartilharam até os momentos mais difíceis ao longo do curso e deste trabalho.

- Enlai Lima Cheng

Aos meus pais, Waldemar e Vânia, e irmão, Matheus, que me apoiaram durante todos os momentos dessa jornada.

Aos amigos que dividiram momentos bons e ruins durante toda a graduação e, principalmente, durante a execução deste trabalho.

- Lucas Negrão Couceiro

AGRADECIMENTOS

À Professora Lucia Vilela Leite Filgueiras pela paciência, discussões, orientação e disposição.

Aos nosso pais que nos apoiaram ao longo de toda a jornada na Escola Politécnica e em tantos outros momentos da vida.

À equipe do Parque Cientec pela disposição com que nos recebeu para conhecer o parque e realizar atividades referentes a este trabalho.

Aos amigos Luiza Brandão, Diogo Gomes, Priscila Ritschel, Camila Lopes e Jéssica Maria que ajudaram direta ou indiretamente na realização desse projeto.

RESUMO

BOXES é um recurso para fornecer informação nos locais onde ela é relevante e em momentos oportunos. É uma forma alternativa de distribuição de conteúdo diferente do atual modelo de um usuário seguir geradores de conteúdo e obter as informações conforme elas vão sendo publicadas na rede.

BOXES entrega conteúdo baseado na localização de um usuário por meio da utilização de links geográficos programáveis, chamados de Geolinks. Boxes contribui para aumentar a interação com espaços físicos reais criando ambientes inteligentes e propicia ambientes híbridos.

Para aumentar a interatividade de um espaço físico, os Geolinks são relacionados com conteúdo digital registrado para tal localização. Este conteúdo pode ser texto, foto, conteúdo de áudio ou um link para outra aplicação ou página na internet.

Disponível em dispositivos móveis com sistema operacional Android, possíveis aplicações de BOXES são campanhas publicitárias, guia para áreas abertas como parques ou até transformar uma avenida como a Paulista em galeria de arte com a colocação de fotos e outras mídias visuais em boxes ao longo dela.

Palavras-chave: Smart City. Hybrid City. Geolink. Serviços baseados em localização.

ABSTRACT

Boxes is an environment that provides relevant geographical contextualized information to its users. It's an alternative form of content distribution in contrast to the standard where consumers follow content providers and wait for new content to be generated and delivered to them.

Boxes delivers content based on a user's location by using programmable geographical links, hereby referenced as Geolinks. Boxes contributes to augment physical spaces by creating smart environments that enables hybrid environments.

In order do increase a physical space's interactivity, these geolinks are combined with a custom digital resource related to that specific location. This digital resource can take a variety of forms such as text, photo, audio, video or a link to another application or website.

Available on mobile devices with Android operating system, possible applications to BOXES are marketing campaigns, tour guide to open areas like parks or turn a big avenue like Paulista in an art galery by placing BOXES with photos and others visual midia along it.

Keywords: Smart City. Hybrid City. Geolink. Location based services.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIAÇÕES

API	*Application Program Interface*
Cientec	Parque de Ciência e Tecnologia
CSS	*Cascading Style Sheets*
GPS	*Global Positioning System*
HTTP	*Hypertext Transfer Protocol*
IAG	Instituto de Astronomia, Geofísica e Ciências Atmosféricas
IDE	*Integrated Development Environment*
IP	*Internet Protocol*
JSON	*JavaScript Object Notation*
MVC	*Model-View-Controller*
OMS	Organização Mundial da Saúde
ORM	*Object Relational Mapping*
PaaS	*Platform as a Service*
PEFI	Parque Estadual das Fontes do Ipiranga
QR Code	*Quick Response Code*
REST	*Representational State Transfer*
RFID	*Radio-Frequency Identification*
SPA	*Single Page Application*
TCP	*Transmission Control Protocol*
USP	Universidade de São Paulo
URI	*Uniform Resource Identifier*
WWW	*World Wide Web*

SUMÁRIO

1. Introdução

O objetivo deste trabalho foi desenvolver um sistema que fornecesse proativamente conteúdo com base na localização de um usuário. Os tópicos a seguir descrevem a justificativa para o desenvolvimento do mesmo, os objetivos gerais e específicos que este trabalho se propôs, como se deu o desenvolvimento do mesmo e como está estruturado este texto.

1.1. Justificativa

Atualmente a produção de conteúdo é constante e intensa, veículos de informação se multiplicam, porém esse grande volume e distribuição acabam dificultando o acesso ao que de fato é relevante por parte dos usuários. A distribuição de conteúdo na internet atualmente se baseia em usuários seguindo geradores de conteúdo e obtendo as informações conforme elas vão sendo publicadas na rede (por exemplo: Facebook, Twitter, Buzzfeed). É importante desenvolver maneiras de fornecer informação relevante em momentos oportunos

Utilizar um sistema para tornar o conteúdo sensível ao contexto (*context aware*) é uma maneira de agregar a ele valor e tornar este conteúdo mais relevante para seus usuários. O principal objetivo de um sistema sensível ao contexto é fornecer conteúdo ou serviços a um usuário baseado em informações como localização, horário, aproximação, situação do usuário e capacidades de rede disponíveis em seu dispositivo. Dentre esses, a localização é o fator de contexto mais explorado. (VAN SETTEN; POKRAEV; KOOLWAAIJ, 2004)

Dispositivos móveis como *smartphones* e *tablets*, altamente presentes no cotidiano de grande parte da população, possuem diversos recursos e sensores que permitem obter a localização do usuário sendo uma ferramenta adequada para utilização de uma aplicação com o propósito de oferecer conteúdo com relevância de localização.

Munson e Gupta (2002) já haviam proposto o desenvolvimento de um serviço de propósito geral para fornecer notificações baseadas em localização, listando diversas aplicações que seriam beneficiadas.

Serviços que se baseiam em localização são aqueles que, por meio de dispositivos móveis com capacidade de fornecer a localização e conectados a uma rede de telecomunicações móveis, fornecem informação ao usuário (JAGOE, 2003; UNNI and HARMON, 2007; VIRRANTAUS et al., 2001).

A associação de conteúdo que possa ser acessado por dispositivos móveis a um local expande os métodos de interação com o ambiente permitindo a utilização de objetos virtuais no mundo real, tornando o ambiente híbrido. Como definido por Streitz (2009), uma cidade híbrida - conceito associado também com cidades inteligentes - é aquela com seus habitantes e entidades reais existindo em paralelo com uma cidade virtual que possui homólogos aos elementos reais.

Existem dois desafios associados ao uso de aplicações que enviam notificações e conteúdos baseado na localização que são questões relacionadas ao tráfego de dados e privacidade (GARZON e DEVA, 2014).

Dessa maneira se torna interessante o desenvolvimento de uma ferramenta que permita a inserção de objetos virtuais no mundo real para associar conteúdo a uma localização e que notifique o usuário de maneira proativa a respeito da existência desses objetos e conteúdos de acordo com seu interesse, fazendo uso de uma arquitetura que reduza o tráfego de dados e aumente a privacidade do usuário.

1.2. Objetivo

O objetivo deste trabalho é construir um sistema que forneça, de forma proativa, conteúdo baseado na localização de um usuário fazendo uso de uma arquitetura que trafeguepoucos dados e mantenha a privacidade do usuário.

Para tal serão utilizados *links* geográficos programáveis, chamados de *Geolinks* segundo trabalho de autores deste projeto (LAZARO et al., 2015), para aumentar a interação com espaços físicos reais criando ambientes inteligentes que forneçam conteúdo com relevância de localização previamente associados.

Geolink é um elemento análogo a um *hyperlink* que liga um ponto de origem, definido como uma localização, a um ponto de destino que é uma informação na forma de conteúdo digital, por exemplo, texto, imagem, áudio ou vídeo.

Por meio dessa estrutura é possível consumir conteúdos associados a um local criando pontos de interesse. O usuário do sistema é acionado pelo ambiente, uma vez que a aplicação atua de forma proativa, assim comopode associar conteúdo a uma determinada localização caracterizando a inserção de objetos virtuais no mundo real, propondo um aumento da interação e criando espaços híbridos.

1.3. Metodologia

A metodologia adotada neste trabalho de formatura seguiu as seguintes etapas:

- Definição do problema: realizamos um *brainstorm* para coletar ideias a fim de definir o problema que será resolvido e auxiliar na definição de requisitos. O *brainstorm* é um formato de discussão em grupo que promove produção de ideias para resolução de problemas. A faixa etária do grupo de 13 pessoas variou de 18 a 60 anos com idade média de 35 anos, entre os participantes tivemos educadores, economistas, designers, engenheiros, estudantes, fonoaudiólogas. Os grupos se reuniram em datas distintas ao longo de Março de 2015.

- Pesquisa bibliográfica: utilizando-se os portais de artigos científicos Scopus, Periódicos da Capes e Google Scholar foram realizadas pesquisas sobre conceitos cidades híbridas e serviços baseados em localização, que apoiam o capítulo 2 e ajudaram a definir os requisitos em que se apoiou para formular a solução.
- Pesquisa de mercado e avaliação comparativa: procurou-se também encontrar dentro do mercado serviços que pudessem ter semelhanças com o sistema aqui desenvolvido. Esta etapa foi essencial para encontrar quais os diferenciais que seriam incluídos dentro do projeto.
- Definir metodologia de desenvolvimento: nesta fase estudamos as metodologias de desenvolvimento de software que são utilizadas por empresas de tecnologia e escolhemos a metodologia ágil como a forma de desenvolvimento adotada.
- Escolha de tecnologia: com o projeto definido dentro de uma área do conhecimento estudamou-se quais tecnologias podem ser utilizadas para que a solução atenda todos os requisitos definidos.
- Definição de arquitetura: após a escolha das tecnologias, definiu-se como é a arquitetura do sistema de forma a estabelecer como os diferentes componentes do sistema se comunicam.
- Desenvolvimento da solução: utilizando a tecnologia e arquitetura definidas, o sistema foi desenvolvido a fim de atender todos os requisitos definidos.
- Análise do protótipo: Com o protótipo funcional definiu-se uma aplicação teste com o objetivo de orientar pesquisas futuras.

1.4. Estrutura do trabalho

Este documento estrutura-se da seguinte forma:

O capítulo 1 é esta INTRODUÇÃO

O capítulo 2, intitulado ASPECTOS CONCEITUAIS, define todos os conceitos e tecnologias utilizados neste documento, necessários para contextualização e desenvolvimento do projeto.

O capítulo 3, intitulado ASPECTOS DA INTERAÇÃO EM AMBIENTES HÍBRIDOS, expõe uma descrição e análise dos problemas relacionados aos conceitos utilizados na elaboração deste projeto.

O capítulo 4, intitulado ESPECIFICAÇÃO DE REQUISITOS, detalha todas as características do projeto com uma descrição, análise de requisitos, casos de uso, diagrama de classe e arquitetura utilizada.

O capítulo 5, intitulado APLICAÇÃO, apresenta uma proposta de uso para a ferramenta desenvolvida envolvendo um parque de ciências e descrições para outras áreas que poderiam se beneficiar do sistema.

O capítulo 6, intitulado DISCUSSÃO, expõe as discussões relacionadas aos problemas envolvidos e os temas de notificação baseada em localização, cidades híbridas e cercas virtuais.

O capítulo 7, intitulado CONCLUSÃO, apresenta o fechamento deste trabalho e próximos trabalhos relacionados.

Nas referências bibliográficas, enumeram-se todas as referências utilizadas para o desenvolvimento do projeto e do documento.
Nos apêndices, encontram-se conteúdos para maior detalhamento de algumas partes específicas do projeto.

2. Aspectos Conceituais

As definições apresentadas neste capítulo ajudam a entender os conceitos envolvidas na ferramenta proposta no que diz respeito a:

- cidades inteligentes e a inserção de objetos virtuais no mundo real
- cidades híbridas
- serviços baseados em localização para associar relevância contextual a uma informação e aspectos de privacidade relacionado a esse tipo de serviço.]

Em seguida são apresentadas as tecnologias envolvidas na implementação do sistema.

2.1. Cidades inteligentes

O conceito de Cidades inteligentes (*Smart Cities)* relaciona-se ao uso de tecnologias digitais ou tecnologias de comunicação e informação para empoderar tanto o espaço urbano quanto os cidadãos. Este conceito é ainda cercado de alguma confusão e usado de maneira nem sempre consistente, não existindo um único modelo nem uma definição formal (NAAM e PARDO,2011).

Uma das primeiras visões nomeia cidades inteligentes àquelas que monitoram e integram condições de elementos de infraestrutura como estradas, pontes, túneis, metrô, aeroportos, portos, comunicação, água, energia, até grande construções, para otimizar seus recursos, planejar melhor as atividades de manutenção e aspectos de segurança enquanto otimizam os serviços para seus moradores (HALL, 2000).

Isso persiste quando se diz que a inteligência de uma cidade vem da utilização de tecnologias de computação para tornar componentes críticos de infraestrutura e serviços, incluindo administração, educação, saúde, segurança pública, moradia,

transporte e utilidade mais inteligentes, interconectados e eficientes (WASHBURN et al., 2010).

Ambas as descrições anteriores apresentam um foco voltado para questões de planejamento e infraestrutura. Uma perspectiva diferente propõe seis categorias que devem ser exploradas numa cidade inteligente - economia, pessoas, governança, mobilidade, meio-ambiente e vivência - por cidadãos envolvidos, independentes e com autonomia de decisão (GIFFINGER, 2007). Voltado ainda para o envolvimento dos cidadãos temos a definição de que uma cidade inteligente é aquela que dá inspiração, compartilha cultura, conhecimento e vivência, uma cidade que motiva seus habitantes a criar e prosperar em suas próprias vidas sendo um lugar de inteligência, acima de tudo, uma incubadora de espaços empoderados.

2.1.1. O papel da indústria

No desenvolvimento do conceito e aplicações envolvendo cidades inteligentes estão envolvidas também grandes companhias como Cisco e IBM. Os interesses em pesquisa nessa área buscam solucionar casos que possam gerar produtos que aplicáveis em outros clientes, praticamente todos na área de governo. Cada um atua numa área de *expertise* diferente, a Cisco com projetos que promovem a integração dos sistemas de diversas entidades associadas a uma mesma cidade enquanto a IBM na parte de gerência de informação e análise, fazendo uso de dados de sensores além de informações de entidades governamentais para auxiliar na tomada de decisões.(SWABEY, 2012)

Uma descrição mais detalhada de sistemas criados por essas empresas segue abaixo.

2.1.1.1. Cisco - Smart+Connected Communities

De acordo com Swabey (2012), no ano de 2005 a fundação Clinton lançou um desafio para a fabricante de equipamentos de rede Cisco de utilizar seu conhecimento tecnológico para tornar as cidades mais sustentáveis. Assim a empresa investiu cinco anos e $25 milhões no programa Desenvolvimento Urbano Conectado, com projetos-piloto envolvendo as cidades de São Francisco, Amsterdam e Seul para demonstrar o potencial da tecnologia. Os resultados deste projeto foi a criação, em 2010, da divisão *Smart+Connected Communities* para tornar em produtos e aplicar em outras cidades as experiências adquiridas no programa.

O portfólio da Cisco divide as soluções entre *Smart+Conneted City* e *Smart+Connected Real Estate*.
Smart+Connected City oferece soluções de acesso remoto para serviços governamentais assim como infraestrutura para gerenciamento da cidade:

- Wi-fi: produtos de infraestrutura para conectividade sem fio na cidade, interligando sensores e serviços e permitindo o acesso à internet para os cidadãos. É a base para outros produtos.
    - Cenário de uso de tecnologia: Internet sem fio pública, serviços baseados em localização, turista virtual, comércio, gerência de serviços de infraestrutura.
- Segurança: serviço de detecção de automática de incidentes com resposta rápida, melhora eficiência das operações.
    - Cenário de uso de tecnologia: monitoramento de locais, detecção de incidentes, administração e estatísticas.
- Soluções remotas para serviços do governo: faz uso de Pontos de Entrega equipados com um software Cisco, tela sensível a toque e um leitor de documentos para prover acesso remoto a serviços do governo.
- Tráfego: combina câmeras IP, sensores, aplicações e a infraestrutura do *Smart+Connected City Wi-Fi* para oferecer as autoridades de controle de trânsito maior visibilidade, poder e velocidade de ação sobre incidentes de trânsito.
    - Cenário de uso de tecnologia: detecção e gerenciamento automático de incidentes de trânsito, monitoramento do tráfego, administração de recursos e análise estatística.

- Estacionamento: utiliza a mesma infraestrutura do tráfego para oferecer informações sobre estacionamento para os cidadãos.
    - Cenário de uso de tecnologia: localizar onde estacionar baseado em pontos de interesse, monitoramento de regiões de estacionamento (permitidas e não permitidas), administrar recursos (agentes de trânsito, locais de estacionamento, arrecadação com zona azul).

Smart+Connected City Real Estate oferece produtos para gerenciar ambientes de trabalho e domésticos:

- Espaços de reunião: integra recursos físicos e digitais para otimizar a gestão de salas e o processo de gerenciamento de recursos.
- Espaços personalizados: permite alocação dinâmica de espaços de trabalho em um escritório, aumentado a utilização do imóvel e reduzindo o custo por empregado com locação.
- Residencial: permite que desenvolvedores imobiliários entreguem experiências de moradias conectadas oferecendo controle doméstico e gerenciamento.

Ainda no âmbito de cidades inteligentes, a Cisco oferece acesso em seu site ao caso da cidade de Mississauga em Ontario no Canadá onde foi construída uma rede de fibra utilizando a tecnologia da empresa em roteamento e segurança juntamente com uma rede sem fio com controladores e pontos de acesso que permitiu as equipe de operações e serviços emergenciais atuarem mais rápido, oferta de novos serviços sem grandes impactos no orçamento de tecnologia da informação e prover mais segurança pública com informação.

2.1.1.2. IBM Smater Cities

No ano de 2008 a empresa lançou a iniciativa Smarter Planet, um programa amplo para investigar a aplicação de instrumentação, interconexão e inteligência (ex.: sensores, redes e análise estatística) em alguns dos principais problemas do mundo.

No ano seguinte o programa *Smarter Cities* focou em utilizar essa combinação de tecnologia no cenário urbano. A empresa oferece soluções para cidades utilizando padrões e experiências adquiridas baseando-se em três pilares: Infraestrutura, Operações e Pessoas. Essa visão permite observar como a empresa conceitua cidades inteligentes.

Figura 1 - Pilares de cidades inteligentes

O produto oferecido para gerenciar as cidades é o Centro Inteligente de Operações IBM para monitorar e gerenciar recursos, eventos e incidentes, otimizar o crescimento e operações, estabelecer conexão com os cidadãos atento a suas preocupações, oferecer segurança e integrar dados de diversos departamentos e agências.

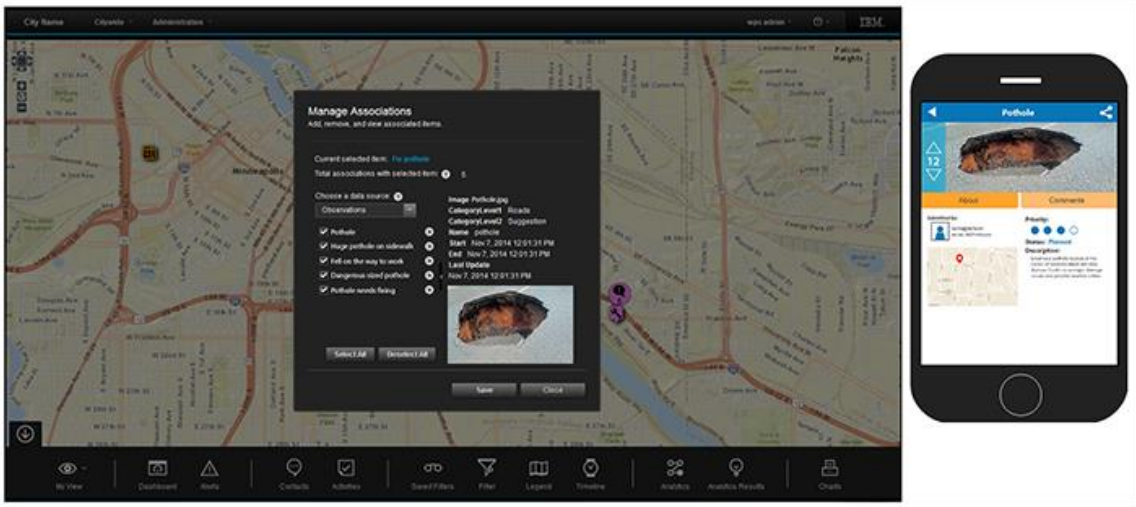


Figura 2 - Interface IBM de interação com cidadão para reportar problemas

As soluções da empresa já foram aplicadas em diversas cidades sendo algumas delas:

- Área metropolitana de Barcelona: com limitação de recursos, orçamentos caindo e infraestrutura envelhecendo era necessária uma solução eficiente para gerenciar as operações como controle de tráfego, coleta de resíduos e policiamento.
    - Solução: plataforma que agrega e centraliza dados, fornecendo análise de informação sobre gerência de espaços públicos, parques e praias.
- Madrid: melhorar a qualidade dos serviços de manutenção oferecidos por empresas terceirizadas.
    - Solução: Entrega de uma plataforma para gerência de serviços que agrega dados de sistemas da cidade, fornecedores além de informações da população coletada por meio de um portal. Oferecia ainda modelagem histórica e análise preditiva para identificar gargalos.
- Metrópole Nice Côte d'Azur: administração regional buscou tornar a região um lugar mais atraente para viver e realizar negócios. Isso se daria através da facilitação de comunicação e troca de dados para melhor gerenciar serviços.
    - Solução: Implementação de uma solução de *Big Data e Analytics* para prover gerenciamento integrado e centralizado de informação utilizado para criar modelos preditivos.

- São Paulo: a Agência de Transporte do Estado de São Paulo buscava uma maneira de gerenciar proativamente os problemas de trânsito.
    - Solução: Desenvolvimento de um sistema de monitoramento de estradas que usa análise estatística para identificar condições que causam os problemas de segurança mais significantes. Auxilia ainda no processo de mudança de rotas em caso de obstruções nas vias.

Enquanto a estrátegia da IBM para cidades inteligentes se baseia no gerenciamento de informação e análise de dados, os projetos da Cisco focam em conectar a infraestrutra existente com novas tecnologias.(SWABEY, 2012) Ambas as abordagens são voltadas para oferecer aplicações que, por meio da coleta de dados relacionados ao espaço urbano, auxiliarão na gestão de recursos da cidade e aumentarão o acesso das pessoas aos serviços públicos.

2.2. Cidades Híbridas

Um conceito que surge simultaneamente ao conceito de cidade inteligente é o de cidade híbrida. Uma cidade híbrida, como definida por Streitz (2009), é uma cidade real com suas entidades físicas e habitantes paralelos a uma cidade virtual com representações digitais das entidades e pessoas reais. Lange e Waal (2012) afirmam que por muito tempo, o domínio de mídias digitais foi visto como virtual, como algo separado da realidade física, mas agora estes dois mundos estão fortemente interligados.

Sendo assim, a cidade contemporânea é uma cidade híbrida com infraestrutura, serviços e processos físicos e digitais em todos os níveis.

Algumas aplicações que foram desenvolvidas nesse âmbito de cidades híbridas estão descritas a seguir:

O aplicativo *RouteMate*(Saridaki and Roinioti, 2013) oferece a um usuário a opção de criar uma rota, ou carregar e modificar uma já existente com o auxílio de um parente, responsável ou um treinador. Para isso, ele utiliza a posição via GPS adquirida do dispositivo móvel do usuário, que deve possuir o sistema operacional Android. Apesar de ser um aplicativo que pode ser utilizado por qualquer pessoa, ele é dedicado a pessoas com alguma deficiência. Para auxiliá-los em rotas, o aplicativo associa uma rota virtual que está diretamente conectada a uma real.

PIGS(Roinioti et al., 2013) é uma aplicação utilizada como instrumento de pesquisa, a fim de examinar e questionar, através de um jogo situado no espaço urbano, as normas sociais de uma cidade e se tornar um veículo de expressão política. Ele parte do princípio de que a cidade possui uma jogabilidade associada a ela e que pode ser explorada tanto no mundo físico quanto no mundo digital. Para tornar o espaço híbrido, o aplicativo utiliza o GPS, a câmera do celular e um QR Code que fica fixado em cada participante do jogo.

The Experimedia Blue Project(Antoniou et al., 2013) tem como propósito melhorar a experiência do usuário antes, durante e depois da visitação a um museu. O projeto utiliza uma plataforma online para realização de um teste de personalidade feito por meio de um questionário. Os dados do usuário também podem ser obtidos através de um jogo disponível na rede social Facebook. Utilizando os dados obtidos, o programa fornece informação personalizada sobre o museu, o comportamento do usuário e elementos de sua personalidade. Além dos atributos pessoais do usuário, o programa utiliza informações espaciais do museu, como tráfego de pessoas, para criar uma rota apropriada para o usuário.

O aplicativo *Green Seeker*(Ivkovic, Piepgras and van Emden, 2013) utiliza a posição atual do usuário e o *Google Maps* para disponibilizar informações sobre a presença de áreas verdes no contexto urbano. O intuito é despertar no usuário a consciência

da falta de áreas verdes em sua cidade e depois motivá-lo a apresentar soluções para toda a comunidade através de canais de mídias sociais.

Todos esses aplicativos utilizam os recursos virtuais para facilitar, ajudar ou contribuir de alguma maneira com o mundo físico, utilizando informações fornecidas por qualquer usuário. Elas podem ser ativamente enviadas à plataforma virtual, por interesse do usuário ou respondendo uma pergunta feita pelo programa, e podem ser enviadas de maneira passiva, como a localização por GPS, por exemplo.

Semelhanças tecnológicas entre Cidade Inteligente e Cidade Híbrida são explicadas por Nam e Pardo (2011), que definem Cidade Inteligente como a união de três fatores: Institucional, Humano e Tecnológico. Dentro deste último fator, definido como infraestrutura de hardware e software, encontramos Cidade Híbrida. Ambos os conceitos estão conectados, porém o escopo de uma Cidade Inteligente é muito mais amplo que o de Cidade Híbrida, atingindo áreas não exploradas nesse projeto.

2.3. Serviços Sensíveis ao Contexto e Baseados em localização

Na criação de ambientes híbridos, produzir e inserir conteúdos de forma relevante requer que eles estejam relacionados ao contexto em que o usuário está inserido.

Contexto é definido por Dey (2000) como qualquer informação que pode ser utilizada para caracterizar a situação de qualquer pessoa, lugar ou objeto que é considerada relevante para interação entre um usuário e uma aplicação, incluídos os próprios usuários e aplicação. Para um sistema sensível ao contexto o objetivo é fornecer informações e/ou serviços a um usuário baseado em seu contexto. São exemplos de informação contextual localização, momento, proximidade, situação do usuário e capacidades de rede.

Segundo Junglas e Watson (2008) serviços baseados em localização são aqueles que levam em consideração a localização geográfica de uma entidade. Sistemas baseados em localização são um tipo especial de sistemas sensíveis ao contexto. Os autores utilizam o termo entidade pois o elemento que aciona a informação geográfica pode ser humano ou não humano como, por exemplo, uma carga de mercadorias que se deseja rastrear. (ELAMIN, 2007).

Em uma requisição padrão de um serviço baseado em localização sempre existem duas entidades, onde o foco da análise é a posição relativa entre estas entidades. Em um cenário onde existem duas entidades A e B, uma é escolhida para ser o objeto do serviço, ou seja, a entidade da qual a localização será monitorada. A outra entidade serve como ponto de referência para o serviço, auxiliando na determinação da distância ou posição do primeiro objeto, isso torna a segunda entidade um ponto de interesse do sistema. (JUNGLAS e WATSON, 2008).

A posição das duas entidades pode ser estática ou dinâmica, dependendo da aplicação ou utilização do sistema. Um exemplo é o funcionamento de um sistema de navegação para automóvel auxiliado por GPS, neste caso as entidades são: entidade A, o carro em movimento; entidade B, o destino desejado (estático). As requisições então são compostas da posição atual do carro em relação a posição destino desejado, desta forma rotas são fornecidas em um mapa para que a posição destas duas entidades eventualmente se igualem.

Considerando a utilização da posição geográfica, os serviços baseados em localização se dividem em dois tipos, os que rastreiam a localização (*location-tracking services*) e os cientes da posição (*position-aware services*).

Esses serviços fornecem a posição geográfica de um usuário relativa a outras entidades que não são o usuário, por exemplo, em um sistema de rastreamento de caminhões de uma transportadora, a localização de cada caminhão é enviada a uma central de controle remota e utilizada para auxiliar a gestão de frotas.

Os serviços baseados em localização são utilizados para endereçar o problema de aumentar a relevância de um conteúdo dentro de um dado contexto buscando agregar valor como verificado no trabalho de Beldona, Kunwei e Yoo (2012).

2.3.1. Serviços cientes da posição

Neste cenário, a informação associada com a localização é fornecida diretamente ao usuário, por exemplo, o sistema de navegação de automóveis auxiliado por GPS, mencionado anteriormente, no qual a posição atual do automóvel reflete em tempo real quais informações são fornecidas ao usuário em relação a direção, problemas na pista, regulamentação de velocidade, postos de gasolina e outras informações de interesse do usuário. Outro exemplo é um sistema de pagamento de pedágios automático, em que um dispositivo no automóvel possui um sensor de localização que é reconhecido por receptores no pedágio e autorizam a passagem do automóvel, eliminando a necessidade de parada total do veículo. Um terceiro exemplo é no âmbito de publicidade, no qual anúncios específicos são enviados ao celular de um consumidor quando o usuário se aproxima de uma loja. Unni e Harmon (2007) caracterizam os dois primeiros exemplos como serviços baseados em localização e o último como marketing baseado em localização.

2.4. Cerca virtual (*geofence*)

Segundo Namiot e Sneps-Sneppe (2013), os serviços baseados em localização modernos, principalmente envolvendo mobilidade, baseiam-se em áreas ou lugares ao invés de uma localização precisa. Ele argumenta que a detecção de uma posição exata é ineficiente considerando que o *chip* de GPS em dispositivos móveis nem sempre fornece uma posição exata. Logo, a representação de um espaço físico para um serviço baseado em localização é feita através de um círculo sobre as coordenadas do ponto desejado ou um polígono composto de várias coordenadas. A

esta forma geométrica denomina-se cerca virtual. (NAMIOT e SNEPS-SNEPPE, 2013).

2.5. Hyperlink geográfico

Conceito criado em trabalho de um dos autores deste projeto com seu então orientador Ohene-Djan, *hyperlink geográfico* é análogo a um *hyperlink*. Também chamado de *geolink*, são compostos de:

- um elemento de origem, referenciável pela aplicação, que no caso é a coordenada geográfica e no caso do hyperlink é o elemento (texto ou imagem) que comanda a navegação;
- um elemento de destino que é, exatamente como no *hyperlink*, um endereço contendo um recurso digital.

Para *geolinks*, o comportamento é ativado quando o usuário fisicamente alcança a localização que foi previamente definida por meio de uma cerca virtual. O comportamento básico é apresentar o conteúdo ao usuário.

2.6. Tecnologias

Nesta seção são apresentadas as tecnologias escolhidas para implementação do projeto e uma breve descrição do que motivou sua escolha.

2.6.1. Node.js

Node.js foi concebido em 2009 por Ryan Dahl com intuito de criar um framework baseado em eventos para desenvolver aplicações de rede escaláveis que conseguem suportar múltiplas conexões concorrentes, em contraste com o modelo comum de concorrência de *threads*. Esse framework utiliza o interpretador Google V8 para executar o código escrito em JavaScript, fator que confere altas velocidades de execução ao servidor. Somado a estes fatores escrever o código do servidor e do cliente em uma mesma linguagem facilita a comunicação através de objetos JSON.

Outras opções de *framework* consideradas para o desenvolvimento do servidor foram Ruby On Rails que utiliza linguagem de programação Ruby e Django que utiliza linguagem de programaçao Python. Apesar de ambos os *frameworks* também utilizarem linguagens intepretadas, tendo em vista que as rotinas do cliente seriam escritas em JavaScript, escrever o código do servidor e do cliente em uma mesma linguagem facilita a comunicação através da utilização da mesma estrutura básica de objetos descritos em notação JSON.

2.6.2. Arquitetura MVC

A arquitetura MVC é dividida em três partes, separadas por responsabilidade. O modelo é uma abstração do banco de dados, são criados objetos que representam as entidades salvas no banco de dados e desta forma possibilitam um fácil meio de busca e manipulação das entidades. A visão representa a interface com o usuário, aceita entradas e apresenta saídas, normalmente representadas por uma página web. O controlador representa as regras de negócio, ele analisa as entradas do sistema, rotas e invoca métodos dos modelos para modificar e devolver informações para a visão.

Sua escolha foi para organizar a estrutura do servidor uma vez que o padrão define uma série de políticas de como estabelecer a hierarquia do software.

2.6.3. Express

Express é um *framework* escrito sobre Node.js que implementa e facilita operações comuns em servidores *web*, especificamente em gerenciar o protocolo HTTP de comunicação.
Com o auxílio deste *framework*, é possível definir os parâmetros do servidor para hospedagem na nuvem, definição e controle de rotas, segurança na comunicação, facilitando desta forma a definição de uma API.

2.6.4. Application Program Interface

API significa uma Interface de Programação de Aplicativos, consiste em uma série de rotinas, métodos e protocolos que auxiliam o desenvolvimento de um software.

Neste projeto a API será a interface que permitirá a comunicação entre cliente e servidor, estabelecendo rotas, mensagens e *WebSockets*. A utilização dos padrões de requisição REST (Representational State Transfer) e dos *WebSockets* definidos pela biblioteca SocketIO são apresentados a seguir.

2.6.5. REST

REST define um estilo de arquitetura para o *design* de requisições utilizando um padrão semelhante ao de requisições HTTP (GET, POST, PUT, DELETE) tradicionais. Tal estilo de comunicação foi inicialmente proposto por Fielding (2000) e funciona através da definição de rotas identificadas pela URI, por exemplo para obter uma cor poderia se utilizar a requisição /cores/azul, bem como para deletar tal entrada com uma requisição DELETE /cores/azul.

Por conseguinte, cada método definido pelo protocolo HTTP tem uma função análoga nesta arquitetura. O método GET funciona para listar entradas de uma coleção ou

obter uma entidade particular de uma coleção, o método PUT serve para atualizar coleções ou entidades particulares, o método POST é utilizado para criar uma nova entidade em uma coleção e o método DELETE é utilizado para deletar coleções ou entidades particulares.

2.6.6. SocketIO e WebSocket

SocketIO é uma biblioteca escrita em Javascript que implementa o protocolo de *WebSockets*, através de uma API fornece uma comunicação bidirecional em tempo real entre clientes e servidores por mensagens, possui uma integração com Node.js que foi aproveitada no desenvolvimento deste projeto.

WebSocket é um protocolo de comunicação *full-duplex* sobre uma única conexão TCP, criado para facilitar a comunicação em tempo real entre cliente e servidor a conexão entre ambos permanece aberta uma vez estabelecida, fator esse que permite que tanto o cliente quanto o servidor ativem uns aos outros. Desta forma o cliente pode ser notificado ou ativado mesmo que não tenha realizado nenhuma requisição ao servidor.

2.6.7. Sequelize.js

Para gerenciar a conexão com o banco de dados, utilizamos o ORM para fazer uma interface entre o banco de dados relacional MySQL para objetos nativos do NodeJS, transformando entidades relacionais em objetos de uma linguagem orientada a objetos, no caso JavaScript.

2.6.8. Android

Sistema operacional, atualmente desenvolvido pela Google, baseado no kernel do Linux e com foco em dispositivos sensíveis ao toque. Por seu código ser disponível como um projeto *open source*, existe uma alta personalização do sistema para ser

embarcado em diversas soluções distintas, de *smartphones* à sistemas para automóveis. Essa flexibilidade do sistema conferiu um grande sucesso do sistema operacional que também é o sistema padrão de grandes empresas como Samsumg, Motorola, entre outros.

Aplicativos para este sistema operacional podem ser desenvolvidos utilizando a linguagem Java e não necessitam de licença especial, logo qualquer desenvolvedor é livre para publicar uma aplicação na plataforma de aplicativos do Google, chamada Google Play, desde que tal aplicação esteja dentro dos padrões definidos pela empresa.

2.6.9. Android Studio

Para desenvolver aplicações para o sistema operacional Android, é possível utilizar a IDE do Android Studio, esta IDE facilita o gerenciamento do código fonte, da compilação, da implementação do código em dispositivos reais ou em simuladores e dos testes de uma aplicação.

2.6.10. Apache Cordova

Apache Cordova é um *framework* que permite o desenvolvimento de aplicações móveis utilizando exatamente essas tecnologias web, como HTML, CSS e JavaScript, ao invés de utilizar as APIs especificas de cada sistema operacional móvel. Utilizando esse *framework*, é possível invocar funcionalidades nativas de um dispositivo móvel, como GPS, câmera, acelerômetro utilizando apenas com funções invocadas em JavaScript.

Ao invés de desenvolver o aplicativo do cliente em código nativo para o Android, a abordagem escolhida foi criar uma aplicação híbrida que utiliza páginas e tecnologias web incorporadas em um aplicativo nativo.

Além de manter a consistência entre servidor e cliente devido à escolha de tecnologias, a escolha do framework também aumenta a portabilidade. Se necessário a implementação do projeto em outros sistemas operacionais como iOS da Apple ou Windows Phone da Microsoft o mesmo desenvolvimento pode ser aproveitado tornando mais simples a transição.

2.6.11. Pixate

Pixate é um software criado para facilitar a prototipagem de aplicações móveis. Utilizando um sistema de camadas é possível criar protótipos funcionais para o teste de interfaces e de usabilidade. Permite também a colaboração de projetos entre membros e a implementação em dispositivos reais, além de fornecer a funcionalidade de gravar a utilização do protótipo a fim de gerar *feedbacks* de uso.

2.6.12. HTML 5

HTML5 (*Hypertext Markup Language 5*) é a quinta versão de uma linguagem de estruturação e apresentação de conteúdos para a *web* desenvolvido pela Opera Software em 1993, se tornou o padrão para a WWW e é interpretada por todos os navegadores modernos.

2.6.13. CSS3

CSS3 é uma linguagem de estilização de *markups*, no caso deste projeto para estilizar a página da *web* descrita em HTML5. O CSS permite ajustar o posicionamento dos elementos a fim de criar um *layout* personalizado e com fatores de usabilidade. Com ele também é possível estabelecer o estado dos elementos que reagem as interações do usuário para criar efeitos e transições específicas.

2.6.14. JavaScript

JavaScript é uma linguagem de programação de alto nível, não tipada, dinâmica e interpretada. Em conjunto com HTML e CSS formam os blocos fundamentais da WWW, o JavaScript neste caso é o que fornece interatividade e dinamismo para uma página da *web*. Por ser interpretada diretamente no navegador do cliente é possível estabelecer conexões com servidores através de métodos assíncronos ou estabelecendo *websockets*, essas conexões posteriormente permitem a troca de dados e a atualização do conteúdo uma vez estático de páginas HTML padrão.

2.6.15. JSON

JSON é uma notação enxuta e altamente legível para a estrutura de dados de par chave-valor para JavaScript, utilizada principalmente por aplicações *web* em comunicações assíncronas e transferências de informações.

2.6.16. AngularJS

AngularJS é um *framework open-source* para o desenvolvimento de aplicações web mantido pelo Google, o *framework* auxilia a implementação da arquitetura MVC, modelo visão controlador, em aplicações web. A sua estrutura baseada totalmente em JavaScript permite o desenho de SPAs dentro de um ambiente integrado da web.

Em conjunto com HTML5, CSS3, JavaSript e o Apache Cordova, o sistema do cliente fornece o ferramentário necessário para a implementação de uma aplicação móvel híbrida de alta qualidade.

2.6.17. BitBucket

O BitBucket é um serviço que permite hospedar e controlar versões de projetos, facilitando assim o acesso aos códigos e resolução de conflitos de código. Ele utiliza uma ferramenta chamada Mercurial, mas nas versões mais recentes possibilita a criação de repositórios com o Git, um sistema que foi desenvolvido para suprir problemas de velocidade. Além disso, permite criar repositórios privados sem custo adicional, uma vez que o custo do serviço provém do número de pessoas que acessam esses repositórios.

2.6.18. Trello

O Trello é uma ferramenta de controle de tarefas, com ela é possível criar diferentes listas de tarefas e controlar suas determinadas responsabilidades, como a pessoa designada a tal tarefa, que área a tarefa se encontra e determinar datas para conclusão destas. Essa ferramenta é gratuita e foi utilizada para gerenciar as atividades para o projeto de desenvolvimento.

2.6.19. Heroku

Heroku é um PaaS que auxilia a implementação de projetos na nuvem, sua alta integração com Git transforma o projeto na nuvem em um repositório, permitindo que atualizações e modificações sejam facilmente feitas com atualização do repositório remoto.

3. Aspectos da interação em ambientes híbridos

Com base nos conceitos expostos na seção anterior, destacamos nesse capítulo os itens abordados no escopo desse projeto. São eles: a utilização de informações de contextos para oferecer conteúdo com maior relevância ao usuário, o modelo de interação onde a ferramenta atua proativamente enviando notificações para o usuário, as questões referentes a tráfego de dados e privacidade envolvendo aplicações baseadas em localização e a utilização de objetos virtuais no mundo real para expandir o ambiente e fornecer conteúdo ciente de contexto.

3.1. Conteúdo com relevância de contexto envolvendo localização

A principal maneira que os usuários de computadores e dispositivos móveis hoje consomem informação é de maneira ativa. Entidades provedoras de conteúdo como revistas, jornais, governo e pessoas autônomas por meio de redes sociais como Facebook, Twitter e ferramentas como Youtube disponibilizam seu material na internet que então é buscado e acessado.

Com a facilidade de acesso e a alta velocidade com que o conteúdo é produzido o consumo de informação muitas vezes ocorre em momento e local irrelevantes. Uma alternativa para isso é utilizar informações de contexto para fornecer conteúdo mais relevante ao usuário. Van Setten, Pokraeve e Koolwaaij (2004) definem informações de contexto através de fatores como localização, horário, situação do usuário e capacidades de redes das quais ele dispõem. Destes os autores citam localização como o fator mais explorado.

3.2. Modelo de interação proativa

A utilização dos serviços baseados em localização esta associada a presença massiva de dispositivos móveis, como *smartphones* e *tablets* com capacidade de fornecer a informação de sua localização e processar o conteúdo recebido através do serviço, no cotidiano da maioria da população. Garzon e Deva (2014) afirmam que apesar disso, esses dispositivos atuam como um sistema inteligente somente de maneira reativa necessitando da iniciativa e intervenção manual do usuário para utilizar a funcionalidade apropriada. Garzon e Deva (2014) propõem que os serviços baseados em localização passem a atuar de maneira proativa.

O conceito de *pull-based* que consiste do usuário engajar com serviço de acordo com sua demanda está associado a aplicações atuando reativamente. Numa atuação proativa do sistema na qual ele inicia o diálogo, dessa maneira ativando o usuário, o provedor de conteúdo ou a aplicação controla o contexto em que serão ativados os serviços caracterizando o *modelo push-based.*

3.3. Tráfego de dados e privacidade

Um dos desafios relacionados a utilização de serviços baseados em localização é referente ao tráfego de dados (Munson e Gupta, 2002). Isto decorre do grande número de requisições trocadas entre o dispositivo móvel e o servidor provedor do serviço baseado em localização para que a posição do usuário seja a mais acurada possível.

Outro fator relacionado ao grande número de requisições trocados diz respeito a privacidade de informações do usuário. Rust, Kannan e Peng (2002) definem privacidade de informação como a extensão na qual uma informação pessoal é desconhecida por outros. O rastreio constante decorrente da grande troca de requisições é um aspecto que preocupa os usuários. Pesquisas anteriores relacionadas a serviços baseados em localização apontam que a preocupação com privacidade se torna ainda maior se associada ao uso de métodos intrusivos nos quais o consumidor experiência perda de controle (UNNI e HARMON, 2007).

3.4. Locais expandidos por objetos digitais

O uso de aplicações baseadas em localização permite agregar conteúdo a uma localização e expandir, por meio de objetos virtuais, as maneiras com que se interage com esse espaço. Dentro do espaço urbano diversas aplicações podem ser exploradas muitas delas associadas a oferecer informações sobre comércio e serviços locais próximos ao usuário.

Além do uso em espaços genéricos, outros ambientes mais específicos são beneficiados com o uso dessas aplicações. Parques, jardins, museus e outras áreas públicas abertas que apresentam um grande espaço de circulação muitas vezes carecem de informação não só para orientar o visitante dentro do local como aumentar a experiência interativa da visita.

4. Especificação de Requisitos

"Boxes" é um conceito criado por Ohene-Djan e Lazaro como parte de um projeto de verão titulado "Aumentando Espaços Físicos com Objetos Digitais Criando Ambientes Inteligentes". Boxes foram concebidas como containers de informação digital associadas a uma coordenada geográfica. O objetivo inicial deste projeto era aperfeiçoar a comunicação de serviços baseados em localização do ponto de vista do tráfego de dados, utilizando a troca de identificadores ao invés do recurso digital. Essa ideia será explicada com mais detalhes na seção 4.6. Modelo de Localização.

4.1. Definindo uma "Box"

"Boxes" é um conceito criado por Ohene-Djan e Lazaro como parte de um projeto de verão titulado "Aumentando Espaços Físicos com Objetos Digitais Criando Ambientes Inteligentes". Boxes foram concebidas como *containers* de informação digital associadas a uma coordenada geográfica. O objetivo inicial deste projeto era aperfeiçoar a comunicação de serviços baseados em localização do ponto de vista do tráfego de dados, utilizando a troca de identificadores ao invés do recurso digital. Essa ideia será explicada com mais detalhes na seção 4.6. Modelo de Localização.

Baseados na aplicação de *hyperlinks* geográficos, "boxes" servem para permitir o acesso a conteúdos digitais baseados na localização de um dispositivo. Como esse conceito é simples e flexível, é possível utilizá-lo em vários contextos associados a fornecimento de informação baseado em localização e ambientes inteligentes.

A estrutura de dados deste recurso digital utiliza os seguintes metadados, caracterizando as informações de contexto:

- Onde: a posição da "box", logo a cerca virtual que define o elemento de origem para o *hyperlink* geográfico;
- Quando: informações sobre o tempo de vida da box, quando foi criada, quando pode ser acessada, até quando será válida e quantas vezes pode ser aberta;

- O que: recurso digital que será o destino do *hyperlink* geográfico, pode assumir qualquer tipo popular de dados digitais como audio, vídeo, texto, *hyperlink* para outras páginas ou aplicações;
- Quem: informações a respeito do criador do recurso e elementos de autorização para o acesso ao conteúdo digital.

Essa estrutura de dados é acessada através de uma aplicação móvel em um dispositivo que possui sistema de posicionamento global (GPS) e uma conexão com a internet. Uma box pode ser obtida por aqueles usuários do sistema que estiverem próximos ao ponto onde ela foi cadastrada, dentro dos limites da cerca virtual e atenderem as restrições de acesso. Após obter uma box, ela pode ser aberta pelo usuário no momento que ele desejar sendo a primeira vez o momento em que ele obtém o conteúdo em seu dispositivo móvel.

Um sistema intermediará a criação das “boxes” e o consumo desses conteúdos, desta forma existem dois contextos de uso: de produtor de conteúdo e de consumidor de conteúdo a serem detalhados na seção 4.5 Contextos de Uso.

Análoga a uma caixa do mundo real, uma box pode ser atribuído a uma localização e possuir um conteúdo dentro. Dessa maneira os usuários podem colocar as caixas no mapa, endereçadas para uma ou mais pessoas oferecendo um conteúdo digital de sua escolha. Esse conteúdo fica disponível uma vez que a pessoa esteja dentro do raio de alcance da caixa.

Considerando sobre o que poderia ser oferecido como conteúdo foi destacado que a maioria das informações relevantes baseada em localização pode ser classificada em quatro categorias para eventual filtragem pelo usuário considerando a revisão dos autores a respeito das aplicações listadas por Van Setten, Pokraev e Koolwaaij (2004):

- eventos temporários: Informações sobre atividades que estejam ocorrendo em um período específico num determinado local como feiras, exposições itinerantes, espetáculos e eventos esportivos.

- promoções locais: Divulgação de material promocional sobre o comércio local podendo envolver distribuição de cupons ou somente informes dos produtos.
- informações turísticas: Material turístico e cultural associado a locais ou monumentos da cidade.
- alertas da cidade: Informes sobre atividades de manutenção em vias públicas, sistemas elétrico ou de água.

Como explicado anteriormente, a utilização de *Geolinks*, agora expostas como caixas virtuais busca a criação de um ambiente propício para cidades híbridas onde os objetos digitais, oferecendo informação com relevância de contexto, empodera não só as pessoas mais também o ambiente.

Para o desenvolvimento da aplicação visando expandir o espaço físico e a criação da camada de objetos digitais para aumentar a interação pessoa ambiente estão envolvidos a elaboração dos seguintes elementos:

- Estruturas de dados e banco de dados;
- *Website* responsivo;
- Design do link server;
- Design da aplicação mobile.

4.2. Cenários de uso e lógica de operação

Como mencionado anteriormente, existem dois contextos de uso previstos para essa aplicação, são eles: o de produtor de conteúdo e de consumidor.

O produtor de conteúdo é responsável por criar uma ou mais boxes definindo os metadados explicados na seção anterior. O processo de criação de boxes segue um *wizard* de três passos: no primeiro passo o produtor cria uma cerca virtual simples para a box, ou seja, um ponto de latitude e longitude com um raio; o produtor pode

criar esta cerca virtual em volta de sua localização atual, com auxílio de um mapa ou com uma busca inteligente de endereços ou coordenadas. Após selecionar o endereço ou ponto desejado o produtor define então o raio de alcance daquela cerca virtual.

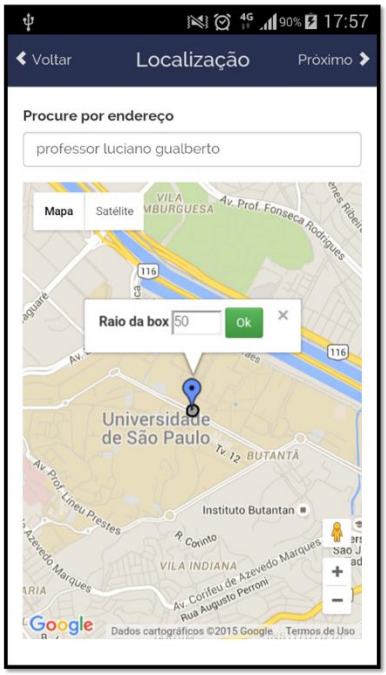
O segundo passo é a seleção de conteúdo para a box, como definido anteriormente qualquer tipo de conteúdo digital poderia ser atrelado a uma box, a decisão de qual tipo de conteúdo fica inteiramente em responsabilidade do produtor. Em aplicações para o espaço urbano, é esperado que o conteúdo fosse algo relevante em relação com o local que está sendo inserido, por exemplo, informações sobre o local, fotos do local, vídeos ou áudios, *hyperlinks* para outros recursos digitais que potencialmente tenham conteúdo relevante para a localização da box.

No último passo, o produtor ajusta as permissões de acesso da box. As opções variam entre deixar o recurso publicamente disponível ou selecionar um grupo de pessoas através de seus *emails* de cadastro. Durante este passo também é possível definir etiquetas para a box a fim de organizá-las em categorias, as definições de quantas vezes uma caixa pode ser aberta, qual data de expiração deste conteúdo também são definidos nesta terceira etapa.

Ao terminar de criar uma box o produtor revisa os parâmetros inseridos antes que a box fique ativa, para garantir a integridade e validade dos dados inseridos. Isso se deve ao fato de que a partir do momento que uma box é criada ela não pode ser editada, apenas desativada. Esta condição é definida, pois, quando uma box é criada, usuários potencialmente já receberam seus conteúdos, logo editá-la após seu envio implicaria em notificar novamente consumidores que já receberam os conteúdos ou distribuir conteúdos desiguais para usuários diferentes. É preferida a primeira abordagem onde uma vez criada a box não é possível editá-la, de forma a manter a consistência entre os potenciais consumidores das boxes.

Telas do processo de criação de uma box

1 - Definição do local



2 - Definição do conteúdo

3 - Permissão de acesso e validade

4 - Criação concluída

Figura 3 - Criando uma BOX

Consumidores se beneficiam do conhecimento armazenado nas boxes e são capazes de compartilhá-lo em redes sociais e contribuir ativamente para a construção desse conhecimento.

No contexto de consumidor, o usuário é capaz de selecionar sobre quais categorias de conteúdo gostaria de ser notificado. Um filtro é aplicado nas boxes disponíveis de maneira que o usuário só será notificado para aquelas categorias que tiver interesse.

Ilustração do processo de obtenção de uma box

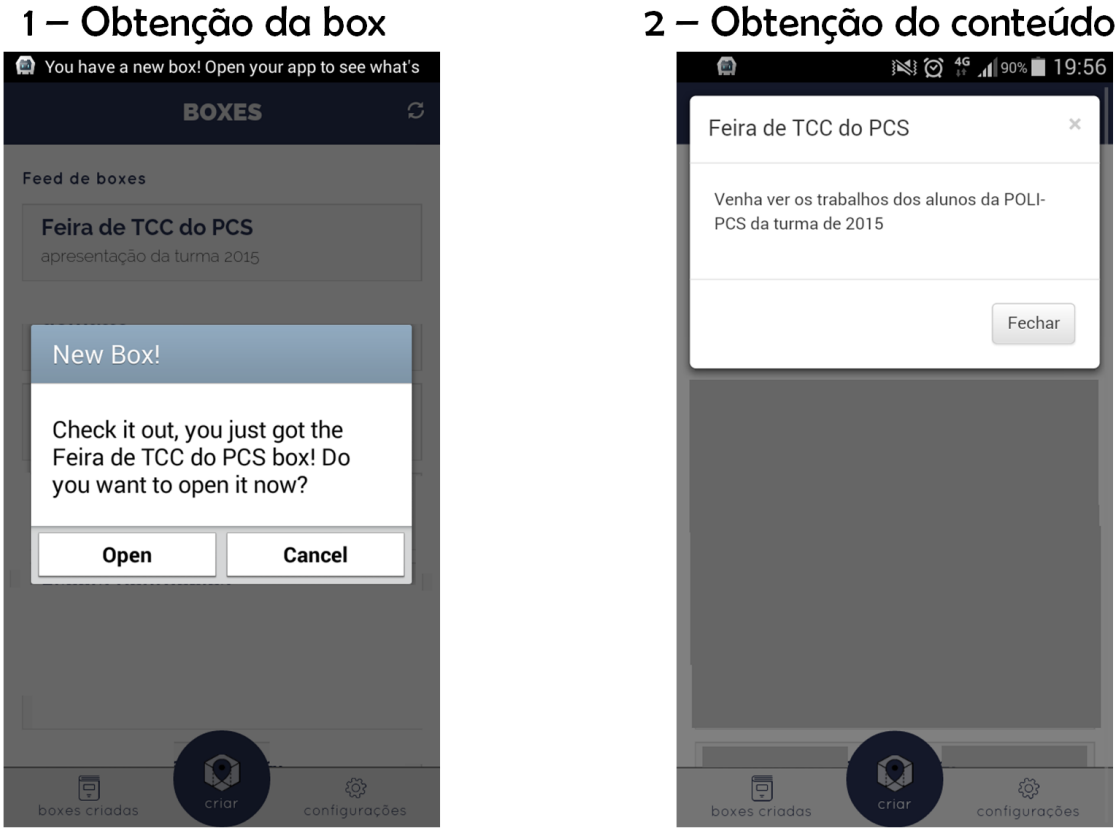


Figura 4 - Obtendo uma BOX

A aplicação possui também dois modos de operação para o recebimento das notificações a respeito das boxes ativas, podendo ser definido quantas notificações num determinado período serão recebidas, independente da frequência com que elas foram armazenadas na caixa de entrada. Há também um modo onde todas as boxes

encontradas são automaticamente notificadas. O primeiro modo de operação é um modo padrão no qual os conteúdos ativam moderadamente o usuário de forma a manter uma interação pouco intrusivae o segundo modo garante um estilo de exploração para ambientes onde o usuário quer receber todos os conteúdos disponíveis.Ambos os modos são selecionados pelo consumidor e podem ser alterados a qualquer momento.

Todas as boxes recebidas ficam na caixa de entrada do consumidor, mesmo aquelas que não notificaram o usuário, e podem ser acessados posteriormente ao serem obtidas.

4.3. Requisitos do sistema

Neste item serão descritos os requisitos funcionais e não funcionais que definiram a implementação do sistema.

4.3.1. Requisitos Funcionais e Não-Funcionais

O BOXES possui seguintes requisitos:

No: 00	(X) Funcional ( ) Não funcional
Requisito: Criar conta de Usuário	
Descrição: O sistema deve permitir o cadastro do usuário utilizando um endereço de e-mail e senha definidos pelo usuário.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	

Estabilidade: (X) Alta ( ) Média ( ) Baixa
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna
Rationale: Ter um controle de quem utiliza Boxes, com suas informações pessoais.
Requisitos associados: R01

No: 01	(X) Funcional ( ) Não funcional
Requisito: Alterar cadastro de Usuário	
Descrição: O sistema deve permitir que usuário altere seu próprio cadastro.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Caso o usuário queira mudar alguma informação pessoal.	
Requisitos associados: R00	

No: 02	(X) Funcional ( ) Não funcional
Requisito: Criar a Box	
Descrição: O usuário deve poder criar uma box e atribuir um nome para ela.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	

Estabilidade: (X) Alta ( ) Média ( ) Baixa
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna
Rationale: Representar um armazenador de conteúdo digital, que possa ser criado por qualquer usuário.
Requisitos associados: R03, R04, R05, R06, R07, R08, R09

No: 03	(X) Funcional ( ) Não funcional
Requisito: Associar uma localização à Box	
Descrição: Para cada box criada deve ser possível associar uma localização a ela baseada num mapa exibido para o usuário.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Tornar a Box georeferenciada.	
Requisitos associados: R02	

No: 04	(X) Funcional ( ) Não funcional
Requisito: Atribuir uma distância limite para a Box	
Descrição: Para cada box criada deve ser delimitada pelo usuário uma distância em metros ao redor do local onde ela foi posicionada a partir da qual ela será acessível.	

Prioridade: (X) Alta ( ) Média ( ) Baixa
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado
Estabilidade: (X) Alta ( ) Média ( ) Baixa
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna
Rationale: A necessidade de tornar um ponto no mapa em uma área circular.
Requisitos associados: R02

No: 05	(X) Funcional ( ) Não funcional
Requisito: Atribuir uma validade à Box	
Descrição: O usuário deve poder determinar uma data até quando uma Box estará disponível para ser obtida.	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Permitir que o usuário programe uma vida útil para a Box, caso o conteúdo seja temporal, por exemplo.	
Requisitos associados: R02	

No: 06	(X) Funcional ( ) Não funcional
Requisito: Atribuir número limite de acesso à Box	
Descrição: O usuário deve poder determinar um número máximo de acesso à Box. Uma vez que esse número é atingido, a Box ficará indisponível.	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Permitir que o usuário programe um número máximo de acesso à Box, caso o conteúdo seja público, porém limitado, por exemplo.	
Requisitos associados: R02	

No: 07	(X) Funcional ( ) Não funcional
Requisito: Restringir acesso à Box	
Descrição: O acesso a uma Box deve poder ser limitado utilizando o email das pessoas permitidas, que serão incluídos pelo produtor.	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: ( ) Alta (X) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Criar conteúdos restritos à pessoas específicas.	

Requisitos associados: R02

No: 08	(X) Funcional ( ) Não funcional
Requisito: Incluir categoria à Box	
Descrição: Permitir que o usuário atribua uma categoria à Box.	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: ( ) Alta (X) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Poder criar um filtro usando essas categorias.	
Requisitos associados: R02	

No: 09	(X) Funcional ( ) Não funcional
Requisito: Incluir conteúdo na Box	
Descrição: O usuário deve poder incluir conteúdo em uma caixa no momento de sua criação sendo texto, foto, áudio ou vídeo.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: ( ) Alta (X) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	

Rationale: Armazenar conteúdo digital dentro da Box.
Requisitos associados: R02

No: 10	(X) Funcional ( ) Não funcional
Requisito: Selecionar categoria desejável	
Descrição: O usuário deve poder selecionar filtrar as boxes que recebe utilizando as categorias disponíveis.	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Fazer com que o usuário receba apenas conteúdo que ele deseja.	
Requisitos associados: R08	

No: 11	(X) Funcional ( ) Não funcional
Requisito: Receber notificação de Boxes	
Descrição: O usuário deve ser notificado quando receber uma Box.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	

Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna
Rationale: Alertar o usuário que ele obteve uma Box, para que ele não precise checar o celular a todo momento.
Requisitos associados: R12

No: 12	(X) Funcional ( ) Não funcional
Requisito: Alterar frequência de notificação	
Descrição: O usuário deve poder alterar o quanto gostaria de ser notificado sobre recebimento de Box.	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Receber muita notificação pode causar incômodo ao usuário.	
Requisitos associados: R11	

No: 13	(X) Funcional ( ) Não funcional
Requisito: Colher e armazenar informações de *feedback*	
Descrição: O sistema deve permitir ao administrador coletar informações sobre o uso da plataforma como: número de usuário ativos, número de box criadas e número de box abertas.	
Prioridade: ( ) Alta (X) Média ( ) Baixa	

Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado
Estabilidade: (X) Alta ( ) Média ( ) Baixa
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna
Rationale: Tornar automática a coleta de informação útil para o administrador.
Requisitos associados: R00, R02

No: 14	( ) Funcional (X) Não funcional
Requisito: Usabilidade	
Descrição: O sistema deve possuir uma interface minimalista composta de poucas telas.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Devido à variedade cultural, etária e social dos usuários, a simplicidade do uso é essencial.	
Requisitos associados:	

No: 15	( ) Funcional (X) Não funcional
Requisito: Desempenho	
Descrição: O sistema deve garantir um tempo de resposta aceitável para o usuário.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Usuários tendem a desistir de utilizar caso ele demore para executar as ações desejadas.	
Requisitos associados:	

No: 16	( ) Funcional (X) Não funcional
Requisito: Disponibilidade	
Descrição: O sistema deverá estar disponível para o usuário, exceto quando houver uma falha na conexão com a internet ou manutenção programada dos servidores.	
Prioridade: (X) Alta ( ) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Serviço não disponível pode fazer com que o usuário desista de utilizá-lo.	

Requisitos associados:

No: 17	( ) Funcional (X) Não funcional
Requisito: Segurança	
Descrição: O sistema deve manter em sigilo informações dos dados fornecidos pelos usuários, assim como o conteúdo que for categorizado como privado;	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Cada um acessar apenas suas próprias informações. Acessos indesejados geram desconforto no usuário.	
Requisitos associados:	

No: 18	( ) Funcional (X) Não funcional
Requisito: Privacidade	
Descrição: Manter a privacidade a respeito da localização de um usuário	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	

Rationale: Quanto maior a privacidade do usuário, maior sua vontade de utilizar BOXES.
Requisitos associados:

No: 19	( ) Funcional (X) Não funcional
Requisito: Economia de dados	
Descrição: Trabalhar com uma quantidade reduzida de requisições	
Prioridade: ( ) Alta (X) Média ( ) Baixa	
Status: ( ) Proposto ( ) Aprovado ( ) Incorporado (X) Validado	
Estabilidade: (X) Alta ( ) Média ( ) Baixa	
Origem: ( ) Usuário ( ) Cliente ( ) Competidor (X) Interna	
Rationale: Menos requisições geram menos fluxo de dados, que permite o usuário utilizar a aplicação por mais tempo.	
Requisitos associados:	

O requisito de desempenho a utilização de estruturas de dados simples e leves para garantir que a localização e a transferência de conteúdos aconteçam com baixo tempo de resposta dentro da cerca virtual da box.

A disponibilidade faz com que seja feita a escolha por um serviço de *cloud* para o servidor, de forma que o serviço fique responsável por restabelecer o servidor em caso de algum problema, desta forma sempre que um usuário entrar em uma cerca virtual o servidor consegue responder as requisições.

Para manter a privacidade do usuário sua localização exata não é armazenada no banco de dados ao longo do tempo, pois a responsabilidade do serviço de localização

se encontra do dispositivo móvel, sem conhecimento do servidor. Ao mesmo tempo minimizando tráfego de dados tentando minimizar requisições desnecessárias ao servidor fazendo processamento local no dispositivo.

4.4. Modelo de Entidades Relacionamento

Nesta seção é descrito o modelo entidade relacionamento do sistema, apresentando o diagrama entidade relacionamento e a seguir a descrição das entidades e seus atributos.

O diagrama entidade relacionamento do sistema pode ser ilustrado pelo diagrama abaixo.

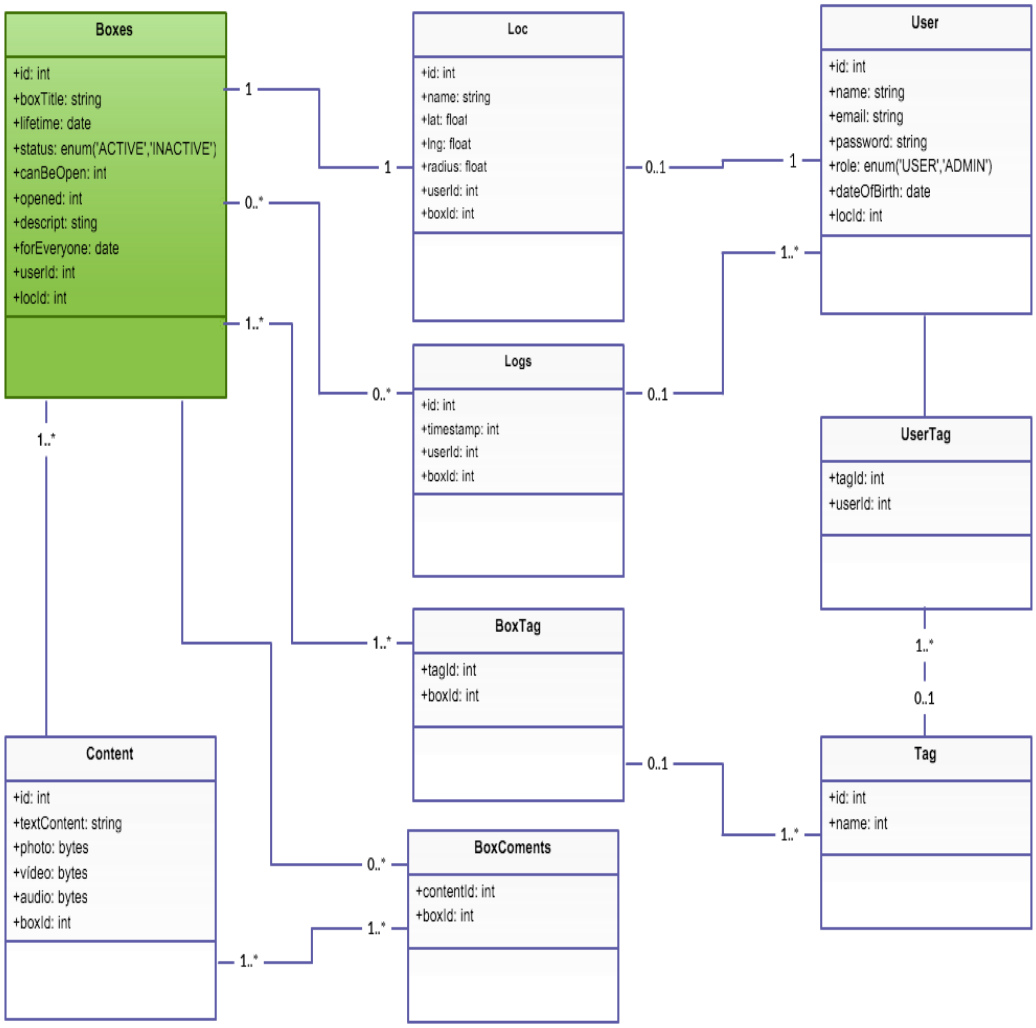


Figura 5 - Diagrama Entidade-Relacionamento

Todos possuem dois atributos que são adicionados sempre que uma entidade é criada, eles são o *created_at* e *updated_at* ambos são *timestamps* que representam a data de criação e atualização, respectivamente.

“Boxes” é a entidade mais importante do sistema e sua estrutura é composta por:

- Id: identificador único de cada box;
- boxTitle: título da box;
- lifetime: data limite na qual a box permanece ativa;
- status: status de uma box, pode assumir dois valores *ACTIVE* (ativo) ou *INACTIVE* (inativo). Nenhuma box é deletada permanentemente do sistema, apenas desativada;
- canBeOpen: número de vezes que pode ser aberta;
- opened: número de vezes que foi aberta;
- descript: descrição curta da box;
- forEveryone: caso a box não seja atribuída a nenhum grupo de usuários esse atributo permanece verdadeiro;
- userId: identificador do usuário criador da box;
- locId: identificador da entidade *Loc* que possui a cerca virtual da box.

User é a entidade responsável por armazenar os dados de um usuário e possui:

- id: identificador único do usuário;
- name: nome do usuário
- email: endereço de email único do usuário, utilizado para entrar no sistema;
- password: senha do usuário, deve possuir mais de seis caracteres e ser alfanumérica;
- role: nível de autorização do usuário, pode assumir dois valores *USER* (usuário) ou *ADMIN* (abreviação de administrador);
- dateOfBirth: data de nascimento do usuário;
- locId: identificador da localização do usuário, apenas para registrar o inicio de sessão.

Content é a entidade responsável por guardar o conteúdo da box, e possui:

- id: identificador único do conteúdo;
- textContent: suporte a qualquer tipo de HTML, desde hyperlink para fotos, vídeos, outros aplicativos ou simplesmente texto;
- photo: foto tirada na hora com *smartphone* ou armazenada em outro local;
- vídeo: vídeo gravado na hora com *smartphone* ou armazenado em outro local.
- audio: aúdio gravado na hora com *smartphone* ou armazenado em outro local;
- boxId: identificador de qual box o conteúdo pertence.

BoxComents é a entidade responsável por guardar comentários feitos a uma box, possui:

- boxId: identificador único de um box;
- contentId: identificador único de um conteúdo.

Loc é a entidade responsável por armazenar a especificação da cerca virtual, possui:

- id: identificador único da localização;
- name: nome da localização, por exemplo São Paulo ou Rua Brigadeiro;
- lat: latitude da cerca virtual;
- lng: longitude da cerca virtual;
- radius: raio de alcance da cerca virtual;
- userId: identificador de usuário, utilizado para guardar a posição da primeira requisição do usuário;
- boxId: identificador de uma box.

Log **é a** entidade responsável por manter registro da abertura de boxes, cada vez que um usuário abre uma box um log é criado, e possui os seguintes atributos:

- id: identificador único do *log*;
- timestamp: representação em segundos desde primeiro de janeiro de 1970 da data em que foi criado o log, a data é mantida na representação de segundos

para facilitar a consulta de uma data, já que é possível utilizar comparativos de maior e menor para determinar um período de tempo;

- userId: identificador de um usuário;
- boxId: identificador de uma box.

Tag é a entidade responsável por categorizar as boxes e guardar as categorias que usuários gostariam de ser notificados e possui:

- id: identificador da tag;
- name: nome dado a categoria, deve ser único entre todos os nomes de *Tags*;

BoxTag é a entidade que relaciona boxes e *tags*:

- tagId: identificador da tag;
- boxId: identificador da box.

UserTag é a entidade que relaciona *users* e *tags*:

- tagId: identificador da tag;
- userId: identificador do *user*.

4.5. Modelo de localização

O método de requisição entre cliente e servidor de dados de localização criado para esse projeto envolve três motivações distintas: preservar a privacidade de localização de um usuário, aproveitar o processamento latente de seu dispositivo móvel a fim de reduzir a carga de decisão do servidor e criar um sistema que considere a densidade de pontos de interesse para regular a quantidade de requisições que são feitas a um servidor.

Para explicar o método primeiramente serão definidos os componentes utilizados para descrevê-lo. Os componentes são:

- *User*: possui relacionamento com a entidade Loc que armazena a informação de latitude e longitude mais atual;
- *Box*: elemento criado para representação de um recurso digital que é acessado por meio de um *hyperlink* geográfico, logo possui dois componentes fundamentais:
- Conteúdo: uma *box* pode guardar todos os tipos de conteúdos digitais, como vídeo, áudio, texto, imagem, *hyperlinks*, entre outros;
- Localização: uma *box* também possui a posição de acesso ao conteúdo anteriormente descrito, a localização define uma cerca virtual e neste projeto será composta de latitude, longitude e um raio;
- Vetor de Locs: vetor unidimensional de entidades Loc que representam cercas virtuais e possuem o identificador de *boxes.* A criação deste vetor é ordenada de acordo com a distância de um determinado ponto referencial, ou seja, a primeira cerca virtual será a mais próxima do referencial e a última a mais distante. Este componente é importante pois só possui identificadores das *boxes*, logo é mais leve para a transição na rede, para facilitar a denominação deste componente o mesmo será referenciado como Grupo de Boxes nas seções a seguir;
- Densidade de *boxes*: número máximo de cercas virtuais de *boxes* em um grupo de *boxes*;
- Raio de procura: distância máxima do cliente em que uma *box* é de interesse;
- Servidor: possui acesso a todas as *boxes*, recebe e fornece dados ao cliente.

O modelo de requisição possui duas funcionalidades fundamentais: a primeira é a obtenção de um grupo de *boxes* para o cliente e a segunda é a obtenção de um conteúdo de uma *box* especifica para o cliente. Para o cálculo das distâncias entre os pontos será utilizada a fórmula de Haversine que calcula a distância entre dois pontos em uma esfera através de longitudes e latitudes, esta formula é definida como:

$$haversin\left(\frac{d}{r}\right) = haversin\,(\varphi_1 - \varphi_2) - \cos(\varphi_1)\cos(\varphi_2)\,haversin(\omega_2 - \omega_1) \quad (1)$$

Onde haversin é definido como:

$$haversin(\theta) = sin^2\left(\frac{\theta}{2}\right) = \frac{1-\cos(\theta)}{2} \quad (2)$$

- d é a distância entre os dois pontos;
- r é o raio da esfera;
- φ_1, φ_2: latitude do ponto 1 e latitude do ponto 2;
- ω_1, ω_2: longitude do ponto 1 e longitude do ponto 2;`

Para facilitar os cálculos que realizaremos neste trabalho, será utilizada a seguinte variação:

$$d = 2r \arcsin\left(\sqrt{sin^2(\frac{\varphi_1 - \varphi_2}{2} + \cos\varphi_1 \cos\varphi_2 sin^2(\frac{\omega_2 - \omega_1}{2})}\right) \quad (3)$$

4.5.1. Obtenção de grupo de boxes

A obtenção de um grupo de *boxes* pelo Cliente funciona da seguinte maneira:

1) Como ilustrado na figura 5 , o Cliente(aplicativo no dispositivo móvel) obtém sua localização atual e faz uma requisição de um grupo de *boxes* para o Servidor; o servidor utiliza a localização do cliente e calcula quais as cinquenta (densidade de boxes) *boxes* mais próximas da localização deste cliente; o servidor coleta as cercas virtuais dos boxes e as agrupa em ordem de distância do cliente ;o servidor envia esse grupo de *boxes* para o cliente.

Figura 6 – Início de sessão com solicitando N identificados de BOX

2) Como ilustrado na figura 6 o Cliente salva o grupo de *boxes* e utiliza a distância da última box (box mais distante) enviado como o raio de procura.

Figura 7 - Início de sessão com solicitando N identificados de BOX

3) Como ilustrado na figura 7 a definição do raio de procura tem a função de informar ao cliente a respeito de quando realizar uma nova busca por outro

grupo de *boxes*. No primeiro passo ele salva a localização da primeira requisição e ao se mover metade do raio de procura o cliente fará outra requisição de obtenção de grupo de *boxes* para o servidor.

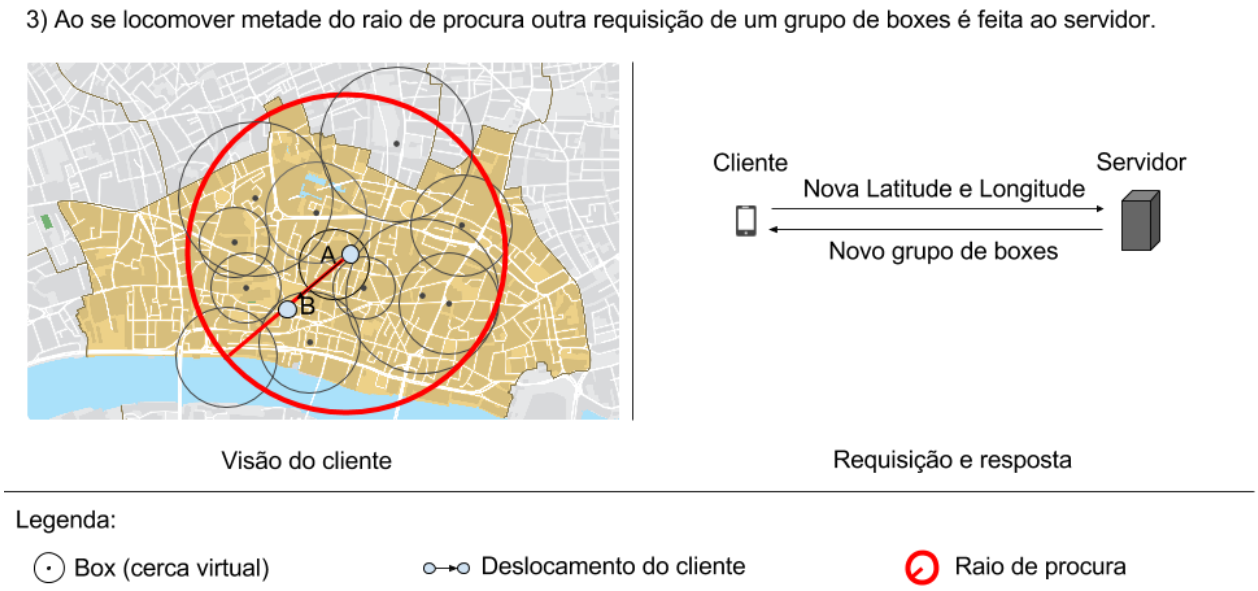


Figura 8 - Requisitando novo grupo de identificadores de box

Outra situação em que uma nova requisição de grupo de boxes é feita caso uma nova box seja criada. Nesse cenário o servidor informa ao cliente que requisite um novo grupo para o caso da nova box estar entre as n boxes mais próximas.

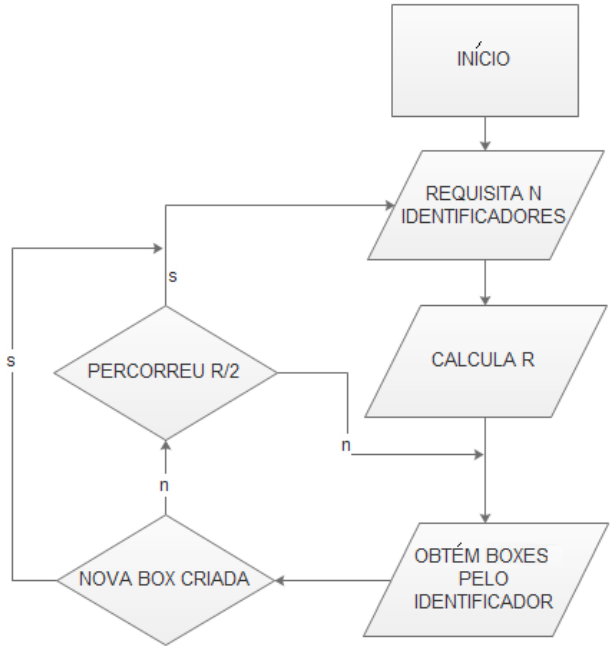


Figura 9 - Fluxo de obtenção de identificadores e BOX

4.5.2. Obtenção de box

Após a obtenção do grupo inicial de *boxes,* o próprio cliente é responsável por determinar se o mesmo se encontra dentro de uma cerca virtual.

Utilizando a fórmula de Haversine, que dá a distância entre dois pontos em uma esfera através de suas latitudes e longitude, o cliente realiza as seguintes ações: determina se sua distância a uma das *boxes* salvas é inferior àquela do raio da *box* que a distância foi calculada; em caso positivo isso configura que o usuário está dentro da cerca virtual. O cliente pode requisitar o conteúdo daquela *box* através do identificador; situação na qual o servidor devolve o conteúdo da *box* requisitado se as permissões de acesso forem atendidas; caso o cliente possa consumir o conteúdo, o conteúdo da *box* aparece para o cliente; o *box* que foi obtido é registrado como utilizado e portanto não entrará em buscas ou em grupos de *boxes* futuros.

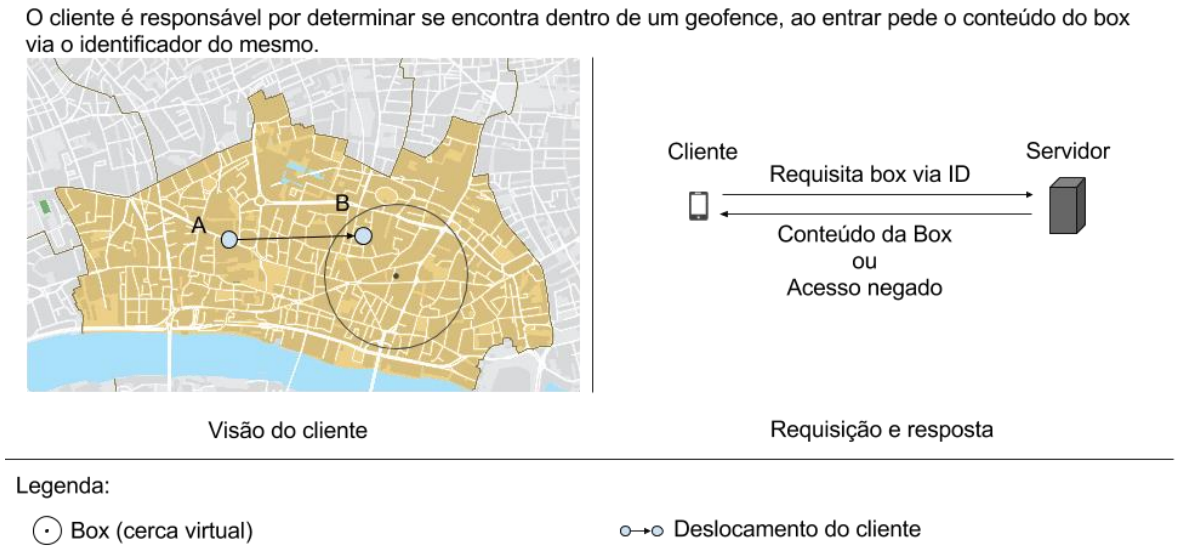
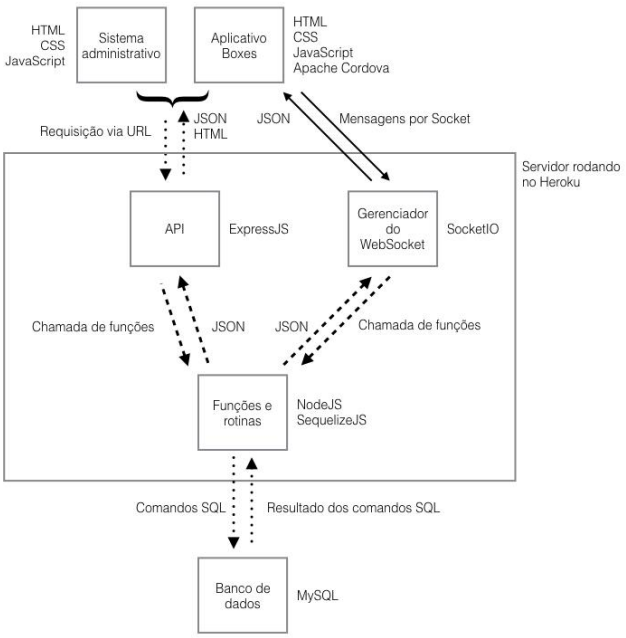


Figura 10 - Obtenção de BOX e seu contéudo

4.6. Arquitetura

Todos os componentes da arquitetura do Boxes é apresentada na imagem a seguir, seguido da descrição de cada componente individual que compõem o sistema



.

Figura 11 - Diagrama de arquitetura

Ao visualizar o diagrama da arquitetura é possível distinguir três regiões com responsabilidades distintas, elas são os clientes, servidor e o banco de dados.

Os clientes são compostos pelo aplicativo móvel boxes foco deste trabalho e o sistema administrativo web que permite visualizar o sistemas como um todo, os usuários criados, as boxes criadas e métricas de utilização do aplicativo, como quantidade de usuários cadastrados, número de usuários online, quantidade de boxes ativas e quantidades de boxes abertas.

Ambos os clientes funcionam com tecnologias web e possuem comunicação REST com o servidor através da API definida pelo ExpressJS, essa comunição é utilizada para os métodos de CRUD (criar, ler, atualizar e deletar) para as classes de usuários e boxes. No aplicativo o modelo de localização funciona inteiramente através da conexão via *websocket* que é controlada pelo SocketIO, desta forma cliente e servidor podem trocar mensagens e dados sem depender de requisições de um dos lados.

O servidor agrupa a API e o Gerenciador de WebSocket mencionado anteriormente e controla as regras de negócio de cada requisição ou mensagem que for recebida por ambos esses componentes. O servidor também possui uma interface com o banco de dados que foi construída com SequelizeJS e abstrai as entidades do banco de dados para objetos nativos do JavaScript.

O banco de dados armazena todas as entidades definidas pelas classes e responde aos comandos que o SequelizeJS traduz para SQL e executa no banco de dados, como criação de usuário, busca por boxes, entre outros.

5. **Aplicação**

São diversos os tipos de aplicações envolvendo uma ferramenta que ofereça conteúdo baseado em localização. As finalidades das aplicações variam de suporte ao comércio, turismo, serviços urbanos até entretenimento. Van Setten, Pokraev e Koolwaaij (2004) listam algumas delas envolvendo as categorias citadas na seção 4.8:

- Notificar um consumidor ao entrar num shopping a respeito de promoções;
- Enviar multimídia para turistas a respeito de monumentos históricos;
- Comunicar visitantes de uma feira sobre atrações do evento;
- Alertar motoristas a respeito da aproximação de uma região com neblina;
- Informar sobre corte no fornecimento de água ou luz para moradores de uma determinada região;
- Notificar jogadores a respeito da entrada numa zona de perigo sob ataque alienígena;
- Informar apostadores para a proximidade de um pote de ouro no final do arco-íris e que devem procurar a pessoa vestida de duende.

Além de notificar, a possibilidade de associar mídias diversas torna mais ampla a aplicação do sistema, ressaltando o aspecto de expansão de ambientes reais por meio de objetos digitais que caracteriza as cidades híbridas.

os tópicos seguintes será abordado o uso da ferramenta em um parque de ciências, o Cientec, como meio para melhorar a experiência do visitante e aumentar o acesso aos recursos do local. O intuito desta aplicação é mostrar o potencial de Boxes como ferramenta para integrar o espaço físico e o espaço virtual, de forma colaborativa e sobretudo, simples. Esta aplicação foi publicada pelo grupo de projeto em evento sobre cidades híbridas. Aqui colocar referência.

Na seção 5.3, outros cenários de utilização são considerados, porém com menor detalhamento.

5.1. Parque de Ciências e Tecnologia da USP (Cientec)

Localizado na Zona Sul da cidade de São Paulo, o Parque do Estado (PEFI) abriga o Jardim Botânico, o Jardim Zoológico, o Observatório de São Paulo e o Cientec.

Figura 12 - Localização do PEFI

Essa região que pertence à USP corresponde a uma área de 141 hectares (aproximadamente 1/3 da área total do PEFI) que previamente era ocupado pelo IAG, até que, no dia 14 de Dezembro de 2001, o reitor Prof. Dr. Adolpho José Melfi oficializou a criação do parque e a então mudança do IAG para a Cidade Universitária. (MANTOVANI, GLEZER e MASSABKI, 2013)

“O Cientec tem como missão “promover o reconhecimento, a valorização e a preservação do patrimônio cultural científico da Universidade de São Paulo, por meio da articulação entre sociedade, cultura, ciência e tecnologia, garantindo acessibilidade e sustentabilidade ambiental.” e chega a receber em torno de 150 visitantes diariamente, principalmente por excursão escolar.” (Parque Cientec, 2015).

O parque possui atividades permanentes para qualquer visitante, como a Trilha dos Ecossistemas, Laboratório de Microbiologia, Planetário e a Estação Meteorológica mais antiga em atividade de São Paulo, e oferece alguns roteiros temáticos, como o Cosmos, Meio ambiente e Física no cotidiano, para escolas que agendam visitas. Esses roteiros são guiados por monitores do parque, que são estudantes da USP cumprindo estágio.

5.2. Modelo atual de visitação do parque

A visitação ao parque é aberta ao público em geral e o modelo de visitação utilizado é a visita monitorada. Os visitantes se organizam em grupos de até vinte pessoas que são acompanhadas por um monitor. Este atua como guia durante os deslocamentos no parque indicando a localização das exposições, fornecendo informações didáticas a respeito do material exposto e assistindo na execução de experimentos quando presentes.

Este modelo implica numa limitação do número de visitantes durante os dias de semana, pois o parque conta com um número pequeno de estagiários, já que muitos se encontram em aula no horário que o parque está aberto. O atendimento foca nos grupos que realizaram um agendamento prévio da visita, sendo geralmente escolas de ensino fundamental. Aos finais de semana, visitantes espontâneos podem ser recebidos, pois há mais monitores e o volume de pessoas visitando o parque é menor.

As informações obtidas a respeito do modelo de visitação do parque e necessidade do monitoramento durante a visita foram obtidos após visita ao Parque no dia 13/11/2015. As pessoas contactadas foram Ira Almeida, que trabalha com a equipe educacional e agendamento de grupos para visitação ao parque, e o monitor Ricarod. Uma demonstração das aréas externas do parque e de como funciona a visita guiada foi feita pelo monitor Cauê.

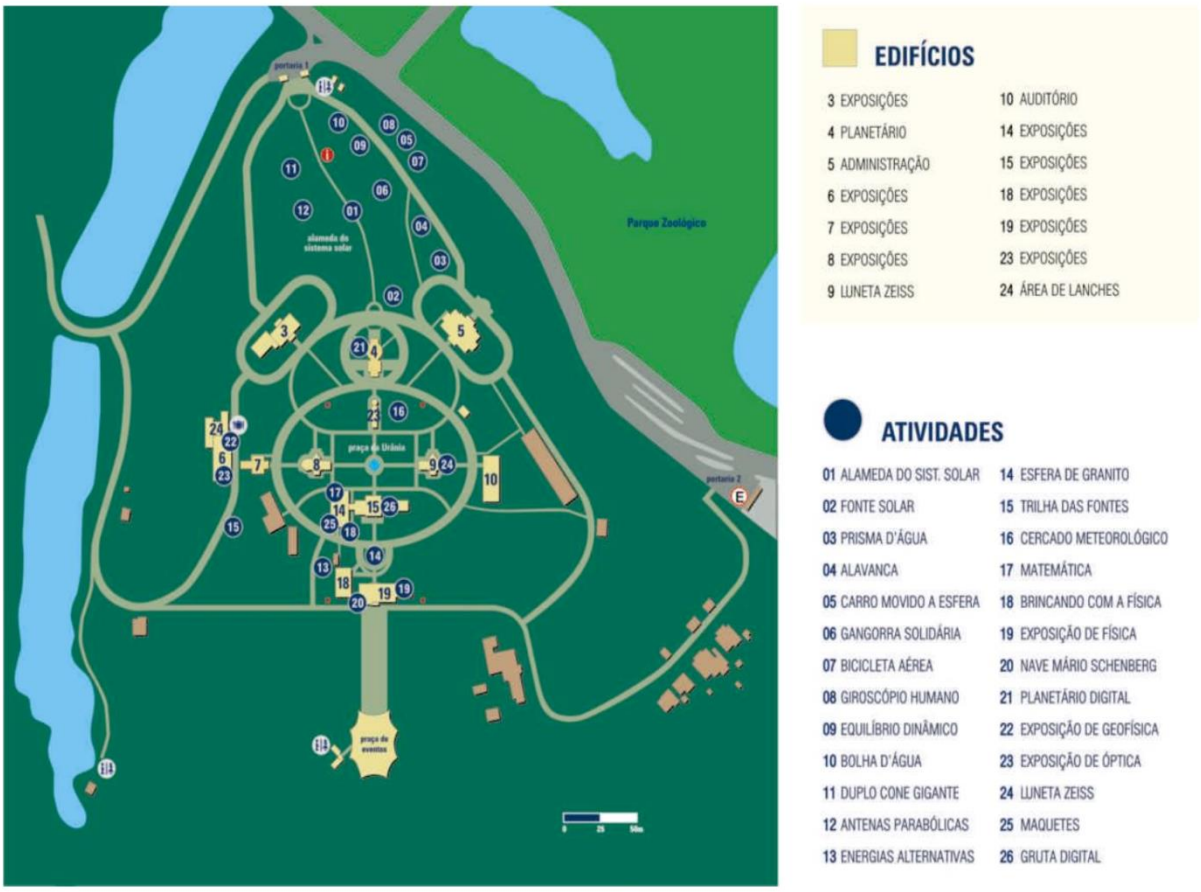


Figura 13 - Mapa do parque com atrações

Pudemos concluir da visitação, que a necessidade de a visitação ser guiada é decorrente da falta de sinalização que permita o visitante localizar com facilidade as exposições assim como identificar o elementos das exposições em áreas externas.

Outro fator que demanda a presença do monitor é a falta de orientação para execução de experimentos ou para atividades que, por motivos de segurança, precisam ser assistidas. Assim mesmo que um visitante explore o parque de forma autônoma e localize as exposições, o mesmo não poderá usufruir inteiramente das atividades disponíveis sem o apoio do monitor.

No contexto apresentado, propomos a aplicação de Boxes para prover conteúdo de orientação e de motivação aos visitantes dentro do parque, diminuindo a dependência dos monitores e criando um cenário lúdico de engajamento com as descobertas. Monitores podem então se dedicar a outras atividades de apoio ao visitante ou mesmo

ficar fixos em exposições onde existem experimentos ou atividades que necessitam de assistência.

Cercas virtuais foram colocadas ao redor dos prédios e de atrações externas assim notificando um visitante que se aproxima a respeito do que está ao seu redor. O raio das cercas foi definido de maneira a interceptar as rotas de circulação próximas ao ponto de interesse dentro da cerca.

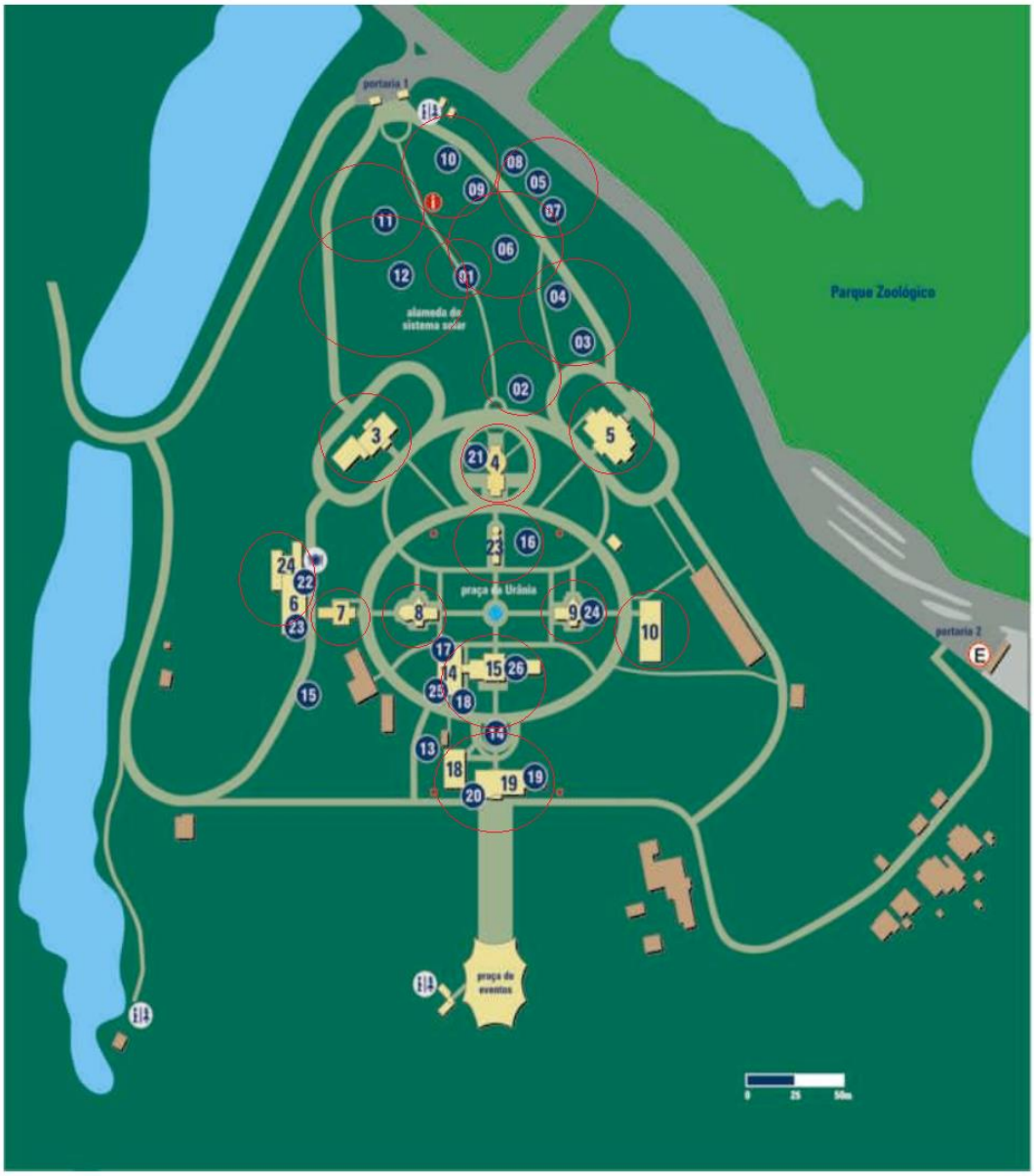


Figura 14 - Distribuição de cercas virtuais no parque

Por meio da associação de conteúdo a uma box referente a um prédio próximo o visitante pode ainda saber qual exposição existe dentro do prédio e quais atividades ele pode realizar.

Caso exista algo que necessite de assistência, o texto associado a box pode informar o horário que os monitores estarão naquele prédio. Peças de exposições externas podem ter conteúdo de áudio ou vídeo oferecendo conteúdo didático a respeito delas ou instruções para realização do experimento como o telefone sem fio que existe na Alameda Solar.

Devido à disposição dos elementos do parque algumas cercas teriam que ser referentes a mais de uma atração. A distinção dos elementos pertencentes a mesma cerca poderia ser feito por meio de foto contida na box ou mesmo uma descrição textual ou em áudio. Um vídeo poderia também mostrar e descrever os elementos. As cercas virtuais deverão ser posicionadas de maneira a interceptar os caminhos designados no mapa.

Além de contribuir com o Parque no quesito de experiência da visitação, BOXES permite ainda que os visitantes deixem uma box ou várias para outros visitantes ou mesmo a equipe do parque, compartilhando suas experiências ou conteúdos que ele mesmo gerou.

No caso desta aplicação os conteúdos associados a cada BOX foi produzido pela instituição e a medida que visitantes exploram o parque com BOXES, novos conteúdos serão deixados se misturando ao material já existente.

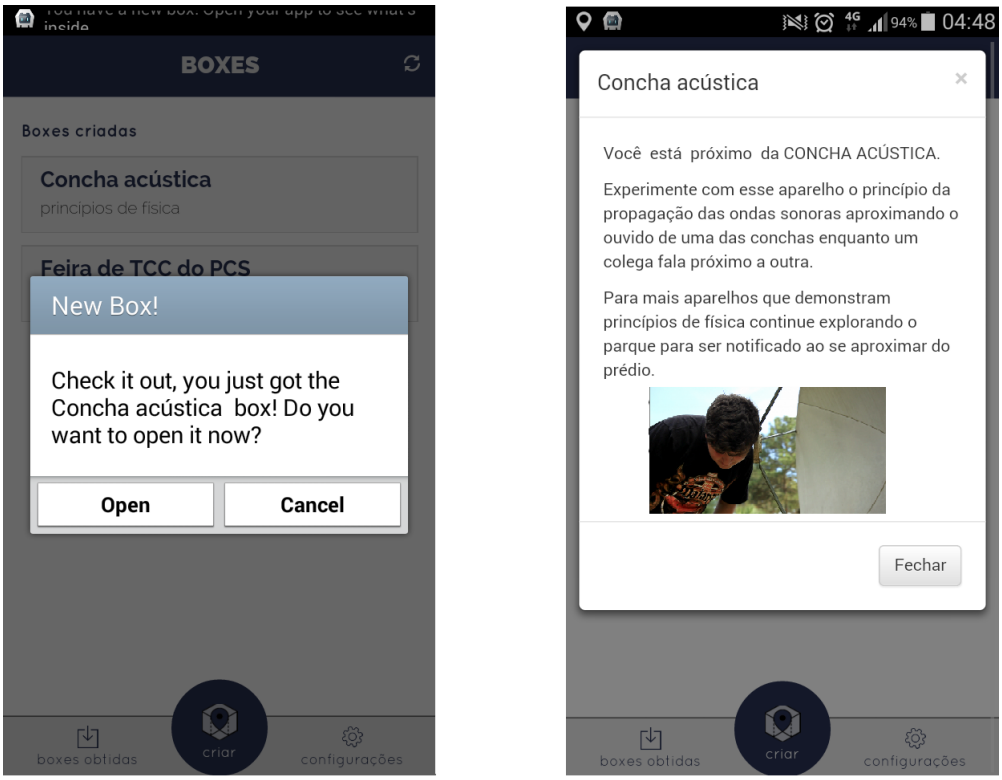


Figura 15 - Obtenção de uma BOX no parque

5.3. Outros possíveis cenários de aplicação.

Diversos outros espaços podem usufruir do sistema provendo conteúdo por localização na cidade de São Paulo. Na relação que se segue, propomos cenários para espaços relacionados a USP - como a Cidade Universitária e o Instituto Butantã e para espaços públicos como o parque Ibirapuera e o cemitério da Consolação.

O Instituto Butantã, como exposição científica, possui uma estrutura bem similar ao Cientec, permitindo que uma aplicação similar a que foi proposta na seção 5.2 seja desenvolvida, porém focada no Instituto.

A aplicação de Boxes à Cidade Universitária tem o caráter de orientar os visitantes dentro do campus. Propõe-se criar Boxes em pontos de interesse relacionados às rotas dos ônibus circulares.

A aplicação no Parque Ibirapuera teria um foco misto entre o Cientec e a USP. Por possuir uma área muito grande de circulação, conteúdos de orientação são muito úteis, porém o parque possui diversos espaços com mostras e exposições rotativas que podem usufruir de conteúdo educativo.

O Cemitério da Consolação abriu suas portas para visitação recentemente, podendo ser guiada ou independente. Nesta, o visitante tem acesso à um mapa e usando o celular acessa às informações através de QR Code, que consome muito dado da rede. O sistema Boxes poderia oferecer as informações aos visitantes assim que eles se aproximassem dos pontos de interesse sem a necessidade da leitura do QR Code alertando inclusive a respeito da aproximação de um ponto que possua conteúdo multimídia associado.

6. Discussão

Neste capítulo iremos apresentar uma discussão sobre a ferramenta e os conceitos utilizados em seu desenvolvimento.

6.1. Sobre a característica de aplicação baseada em localização

Munson e Gupta (2002) em seu trabalho abordam três aspectos principais de uma aplicação para fornecer serviços baseados em localização:

- Habilidade de localizar com precisão um usuário da aplicação;
- Definição precisa de áreas de notificação localizadas de maneira arbitrária;
- Habilidade de detectar a entrada em uma área de notificação em um curto período.

Revisando os aspectos propostos e a arquitetura utilizada no desenvolvimento da aplicação referente à **habilidade de localizar o usuário**, considera-se que:

- A habilidade de localizar o usuário é, em nosso projeto, um aspecto dependente das capacidades do dispositivo móvel. Dhar e Varshney (2011) listam os recursos tecnológicos necessários para localização sendo: disponibilidade de sistema de GPS, localização por meio da rede de telefonia móvel, localização baseada em rede local sem fio e posicionamento por meio de sensores e RFID. De todos os apresentados, apenas este último está fora do escopo do projeto desenvolvido, embora possa ser acrescentado na medida em que os dispositivos de uso comum incorporem efetivamente os leitores de RFID. A falta de informação proveniente de uma das tecnologias reduz a precisão com que é possível localizar o usuário.
- O modelo de requisições baseado no agrupamento de identificadores deixando a tarefa de notificação por parte do dispositivo móvel reduz a dependência da rede de telefonia móvel e rede local sem fio para rastrear o usuário em situações nas quais não é necessário requisitar um novo grupo de identificadores.

Referente à **definição precisa de áreas de notificação** em localizações arbitrárias:

- As áreas de notificação podem ser definidas em localizações arbitrárias pelo usuário.
- A definição precisa de áreas de notificação é possível pela inserção das Boxes por escolha de local a partir de um mapa, ao invés de ser “deixada” pelo usuário no local em que ele está. Nesta última situação, apesar de ser possível, vale o mesmo afirmado anteriormente: a falta de informação proveniente de uma das tecnologias reduz a precisão com a Box é colocada.
- A utilização de cercas virtuais permite que as áreas de notificação sejam definidas com precisão baseadas no valor de um raio ao redor de uma posição geográfica escolhida.
- A limitação desse método diz respeito ao fato das áreas poderem ser somente circulares na atual implementação.

Referente a **habilidade de detectar entrada** em uma área de notificação, em curto período, pode-se afirmar que:

- O modelo adotado no qual o processamento para verificação de entrada numa cerca virtual é feito por meio dos identificadores armazenados no dispositivo móvel reduz a dependência de comunicação com o servidor. Enquanto numa região da qual já se tenha obtido o agrupumaneto de identificadores as notifições ocorrem mesmo se houver falha na conexão porém a requisição de um novo agrupamento e do conteúdo requer que a comunicação com o servidor seja possível

6.2. Sobre o acesso à box

Alguns fatores, como o tamanho da cerca virtual definida pelo criador da box, localização da box, restrição de visualização por e-mail ou qualquer característica do usuário, influenciam diretamente no acesso ao conteúdo de uma box. Alguns desses fatores podem ser ajustados pelo criador da box e assim, trazer novas possibilidades para as boxes.

6.2.1. Tamanho da cerca virtual

A cerca virtual é a única restrição de acesso obrigatória, uma vez que, junto à localização, é necessária para criar a box. Ela influencia diretamente no acesso ao conteúdo, pois quanto maior for a cerca, mais fácil será o acesso a ela. Com essa delimitação geográfica, você pode associar uma box a uma construção, ou uma região delimitada, permitindo aplicações como mapeamento de parques de uma cidade, por exemplo. Cercas muito grandes, porém, podem tornar a notificação pouco relevante se a associação do conteúdo ao objeto real não for clara ou o objeto seja de difícil localização dentro da região da cerca.

6.2.2. Formato da cerca virtual

O formato da cerca virtual ser circular permite que a Box seja criada com maior facilidade pelos produtores, pois não é necessário delimitar um espaço em detalhes, apenas atribuir um raio.

6.2.3. Localização

Uma vez que o acesso ao conteúdo digital depende do usuário alcançar essa posição, a localização da box é uma característica essencial.

Quando um usuário cria uma box em lugar qualquer, ele corre o risco de essa box não ficar mais alcançável por conta de bloqueios, obras ou paralisações. Por exemplo, cria-se uma box no meio da rua, porém, a rua foi interditada para obras e ninguém pode passar por lá. Ninguém conseguirá acessar o conteúdo dessa box enquanto a rua não for liberada.

6.2.4. E-mail

O mecanismo escolhido para restringir o acesso a uma Box, permitindo que usuários criem conteúdos direcionados a uma certa pessoa ou grupo de pessoas conhecidas foi a associação à Box dos e-mails dos usuários permitidos. Essas boxes são as menos visíveis, porém têm maiores chances de agradar o usuário, uma vez que é gerado por alguém que conhece os gostos e interesses do consumidor desse conteúdo.

Esse fator permite que a ferramenta se desenvolva em uma rede social, uma vez que alguém pode deixar um conteúdo digital para outra pessoa específica através do seu e-mail, sem que mais ninguém veja, ou mandar para uma lista de pessoas, possivelmente e provavelmente amigos.

6.2.5. Características do Usuário

Uma restrição de acesso deve ser implementada de acordo com o desejo de quem fará a aplicação. Algumas características sobre o usuário pode permitir a criação de um conteúdo digital específico para o público.

Por exemplo, no zoológico, uma criança não tem os mesmos objetivos que um adulto, para isso você pode criar conteúdos específicos para um público jovem e outro conteúdo, com outra linguagem, para um público mais avançado. Para isso basta adicionar a restrição por idade, onde o usuário preenche em seu cadastro e você cria boxes utilizando a idade como fator de restrição.

Outros exemplos são a criação de conteúdos específicos baseados nas profissões ou temas de interesse.

6.3. Privacidade

Pesquisas relacionadas a serviços baseados em localização apontam que a preocupação com privacidade se torna maior com o uso de métodos intrusivos de entrega nos quais o consumidor experiencia perda de controle.(Unni e Harmon ,2007). Esse cenário se caracteriza com o envio constante de requisições informando a localização para o servidor provedor de conteúdo e a ativação do usuário por meio do método de *push-notification* que deixa o controle de envio das notificações por parte da aplicação.

6.4. Frequência de notificação

O modelo de requisições discutido em 4.5, baseado em agrupamento de identificadores, oferece uma redução no número de requisições enviadas e consequentemente menos registros no servidor da posição em tempo real do usuário. Outro registro que fica armazenado é referente à obtenção do conteúdo associado a uma box, este porém pode não representar a posição do usuário uma vez que a requisição é feita através do identificador e não requer que o usuário esteja dentro da cerca virtual da Box.

Um cenário no qual o agrupamento pode não refletir em redução no número de requisições num determinado período é aquele em que existe uma grande densidade de caixas, que faz com que a requisição por um novo agrupamento seja realizada com maior frequência. Essa frequência está atrelada também a velocidade de deslocamento do usuário.

O modo de uso passivo no qual o usuário não recebe todas as notificações referentes as boxes encontradas trata a questão de intrusividade relacionada à utilização de *push notification*. Isso permite manter o aspecto proativo da aplicação de acionar o usuário porém pode interferir com a interatividade que os conteúdos associados a localização podem oferecer uma vez que o usuário necessita verificar sempre o dispositivo móvel.

7. Conclusão

BOXES caracteriza-se como uma ferramenta para associar conteúdo digital a localizações do mundo real, notificando os usuários a respeito do conteúdo. Diversas são as aplicações que podem ser criadas com os recursos desta ferramenta.

Como a ferramenta não é direcionada a um determinado assunto, ela foi desenvolvida para que o usuário tivesse liberdade para explorar o espaço físico sob seu próprio ponto de vista. Essa simplicidade trouxe consigo infinitas possibilidades de aplicação da ferramenta, ou mesmo de sua expansão.

Para demonstrar este potencial, foi criada uma aplicação de BOXES a um parque científico, promovendo a hibridização do espaço físico e do espaço digital.

7.1. Benefícios do projeto ao grupo de trabalho

O estudo do conceito de cidades híbridas permitiu entender melhor os aspectos que envolvem a criação de uma camada de interação entre o mundo físico e o digital.

O trabalho trouxe também o entendimento de como o serviços de notificação baseado em localização podem trazer valor ao usuário pois buscam oferecer contéudo com maior relevância e como os conceitos de cercas virtuais são explorados para essa finalidade.

No âmbito da implementação da ferramenta foi possível entender melhor os desafios envolvendo o desenvolvimento de uma aplicação móvel que faça uso dos conceitos abordados explorando o uso de APIs de localização.

Foram aprendidos os conceitos de cidades inteligentes e híbridas, serviços sensíveis ao contexto e baseados em localização. A utilização de cercas virtuais para envio de notificações.

7.2. Dificuldades enfrentadas

Ao desenvolver o servidor, a maior dificuldade foi manter e manipular as diferentes conexões via *websocket*. Em decorrência de a troca de mensagens ser assíncrona, foi necessário criar um sistema de *callbacks* que espera a requisição assíncrona terminar para determinar a qual *socket* devolver a informação.

O aplicativo móvel foi o maior desafio no desenvolvimento deste projeto. Inicialmente, a escolha por um aplicativo híbrido deixava a responsabilidade de guardar as informações inteiramente por tecnologias *web*, logo, guardar o grupo de boxes que era recebido pelo servidor foi inicialmente feito utilizando o cache do *browser*. O problema surgiu quando as buscas ao cache não eram feitas de forma eficiente e fácil. O problema foi resolvido ao encontrar a biblioteca para JavaScript chamada TaffyDB, TaffyDB usa o *local storage* do *browser* nativo, que pode consumir até 5MB, como um banco de dados não relacional, desta forma é possível incluir todos os identificadores de boxes e buscá-los de forma trivial e eficiente.

A segunda dificuldade encontrada na implementação do aplicativo móvel foi a determinação da frequência de notificações a respeito de estar em uma cerca virtual (considerando o “modo exploração” descrito na seção 3.4.) uma vez que notificar constantemente um usuário pode ser uma prática negativa. Logo, a frequência com que o usuário é notificado é temporal. O usuário é notificado uma vez por dia sobre uma box nova, porém esse tempo pode ser modificado na parte de configurações do aplicativo.

7.3. TRABALHOS FUTUROS

No desenvolvimento da ferramenta, surgiram diversas questões envolvendo os conceitos de cidade híbridas e a notificação por localização utilizando cercas virtuais.

Fornecer a ferramenta para que os usuários definam qual será sua aplicação permite entender melhor o potencial da inserção de conteúdo digital com relevância de localização.

Outro aspecto a ser explorado com a utilização da ferramenta por usuários gerais é o quão intuitivo ocorre a interação com objetos virtuais espalhados no mundo real. Como os usuários consomem e geram esse objetos e como seria feita a gestão de conteúdo.

Definidas as aplicações, estudos envolvendo identificar as maneiras com que os usuários respondem as notificações e associam valor ao conteúdo oferecido poderiam ser feitos para definir outros modos de operação para a ferramenta.

REFERÊNCIAS

Antoniou,A.;Lepouras,G.; Lykourentzou,I; Naudet,Y. **Connecting physical space, human personalities and social networks: The Experimedia Blue Project.** 2nd International Hybrid City Conference, Atenas: 2013.

Beldona, S.; Lin,K.; Yoo, J. **The roles of personal innovativeness and push vs pull delivery methods in travel-oriented location-based marketing services.** Journal of Hospitality and Tourism Technology, Vol. 3 Iss 2. [Sl]: 2012, pp. 86 – 95

Blumenfeld, P. C. et all. **Motivating Project-Based Learning: Sustaining the Doing, Supporting and Learning**, The University of Michigan, USA, 1991.
de Lange, M.; de Waal, M. **Ownership in the Hybrid City**. Amsterdam: Virtueel Platform , 2012.

Dey, A. K. Providing **Architectural Support for Building Context-Aware Applications**. Ph.D. thesis, College of Computing, Georgia Institute of Technology. Georgia: 2000

Dhar , S.; Varshney, U. **Challenges and business models for mobile location-based services and advertising**. Commun. ACM 54. [Sl]:2011,121-128.
ElAmin, A. Metro Group completes Europe's largest RFID rollout. 2007. Disponível em: www.foodproductiondaily.com/news/ng.asp?n=81315-metrogroup-rfid-logistics. Acesso em: 15 de outubro de 2015

Fielding, R. T. **Principled design of the modern Web architecture**. Irvine: 2000 p. 407-416. ISBN: 1-58113-206-9. DOI: 10.1145/337180.337228.

Garzon, S. R.; Deva, B. **Geofencing 2.0**: taking location-based notifications to the next level. Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14). New York: ACM, 921-932. 2014.

Giffinger, C. et al. **Smart cities: Ranking of European medium-sized cities.** Vienna: oct 2007.

Hall, R. E. **The vision of a smart city.** 2nd International Life Extension Technology Workshop. Paris: sep 2000.

Ivkovic,M.; Piepgras,B.; van Emden,R. **Fun, games and collaborative plans: Benefits and shortcomings of including interactivity and gaming into the collaborative urban planning.** 2nd International Hybrid City Conference, Atenas: 2013.

Jagoe, A. **Mobile Location Services**:The Definitive Guide. Upper Saddle River: Prentice-Hall, 2003.

Junglas, I. A.; Watson, R.T. **Location-based services**. Commun. ACM 51, 3 (March 2008). [SI]: mar 2008, 65-69

Lazaro, L. H. M. et al. **Empowering science parks for disabled persons using Boxes**. 3nd International Hybrid City Conference. Greece: 2015.

Mantovani, M.S.M.; Glezer, R.; Massabki, P.H.B. **Preservar e proteger em um museu de ciências**. Franca : História. v. 32, n. 2, 2013, p. 64-86

Munson, J.P.; Gupta, V.K. **Location-based notification as a general-purpose service**. In Proceedings of the 2nd international workshop on Mobile commerce (WMC '02). New York: ACM, 2002, 40-44

Naam, T., e Pardo, T. **Conceptualizing smart city with dimensions of technology, people, and institutions.** The Proceedings of the 12th Annual International Conference on Digital Government Research. Maryland: jun 2011, pp. 282-291

Namiot, D.; Sneps-Sneppe, M. **Geofence and network proximity.** Internet of Things, Smart Spaces, and Next Generation, 13th International Conference, NEW2AN 2013 and 6th Conference, ruSMART 2013, St. Petersburg: 2013.

Rios, P. **Creating 'the smart city**, Detroit: 2008 University of Detroit Mercy, Michigan, USA.

Roinioti,E.; Saridaki,M.; Hiotis,G.; Arabatzis,D. **P.I.G.S. and the city: Playing with guilt and truth in the streets of Athens.** 2nd International Hybrid City Conference, Atenas: 2013

Rust, R.; Kannan, P.; Peng, N. **The customer economics of internet privacy**, Journal of the Academy of Marketing Science, Vol. 30 No. 4, [Sl]:2002, pp. 455-64
Streitz, N. **Ambient intelligence landscapes for realizing the cities of the future**: Introduction and overview. Proceedings of the 3rd European Conference on Ambient Intelligence. Salzburg: 2009

Saridaki, M.;Roinioti, E. **RouteMate, a location based route learning system for users with disabilities. A playful methodological experience in different urban European landscapes.** 2nd International Hybrid City Conference, Atenas: 2013.

Swabey, P. **IBM, Cisco and the business of smart cities:** how two of the IT industry's largest companies plan to rewire urban living**,** Information Age: Insight and analysis for IT leaders. 2012. Disponível em: www.information-age.com/industry/hardware/2087993/ibm-cisco-and-the-business-of-smart-cities. Acesso em: 20 de outubro de 2015

Unni, R.; Harmon, R. **Perceived effectiveness of push vs pull mobile location-based advertising**, [Sl]: Journal of Interactive Marketing, Vol. 7 No. 2, 2007, pp. 28-40.

Van Setten, M.; Pokraev, S.; Koolwaaik,J. **Context-Aware Recommendations in the Mobile Tourist Application COMPASS.** Volume 3137 of the series Lecture Notes in Computer Science [Sl]: 2004. pp 235-244

Virrantaus, K.;Markkula, J.; Garmash, A.; Terziyan, Y. **Developing GIS-supported location based services.** Proceedings of WGIS'2001 – First International Workshop on Web Geographical Information Systems. Kyoto: 2001, pp. 423-32.

Washburn, D. et al. **Helping CIOs understand 'smart city'initiatives.** Massachusetts: feb 2010.

ANEXO A – Detalhamento dos casos de uso

Neste apêndice são especificados os atores do sistema e uma descrição sobre os casos de uso.

Atores

Existem três atores do sistema: o administrador do sistema, o produtor e o consumidor.. Como o nome sugere o produtor é responsável por criar conteúdos para boxes na plataforma e o consumidor é responsável por receber os boxes geolocalizados. O administrador além de poder produzir e consumir boxes é responsável por auditar as boxes criadas e gerenciar usuários. O Administrador do sistema possui acesso a todas as funcionalidades do sistema.

Lista de casos de uso

O Produtor pode:

1. Cadastrar usuário
2. Editar usuário
3. Redefinir senha
4. Desativar conta
5. Criar uma box

O consumidor pode:

1. Cadastrar usuário
2. Editar usuário
3. Redefinir senha
4. Desativar conta

5. Alterar frequência de notificação
6. Receber uma box
7. Filtrar categorias de Box

O Administrador do sistema pode:

1. Consultar usuários
2. Cadastrar usuário
3. Editar usuário
4. Redefinir senha
5. Deletar usuário
6. Consultar boxes
7. Criar uma box
8. Deletar box

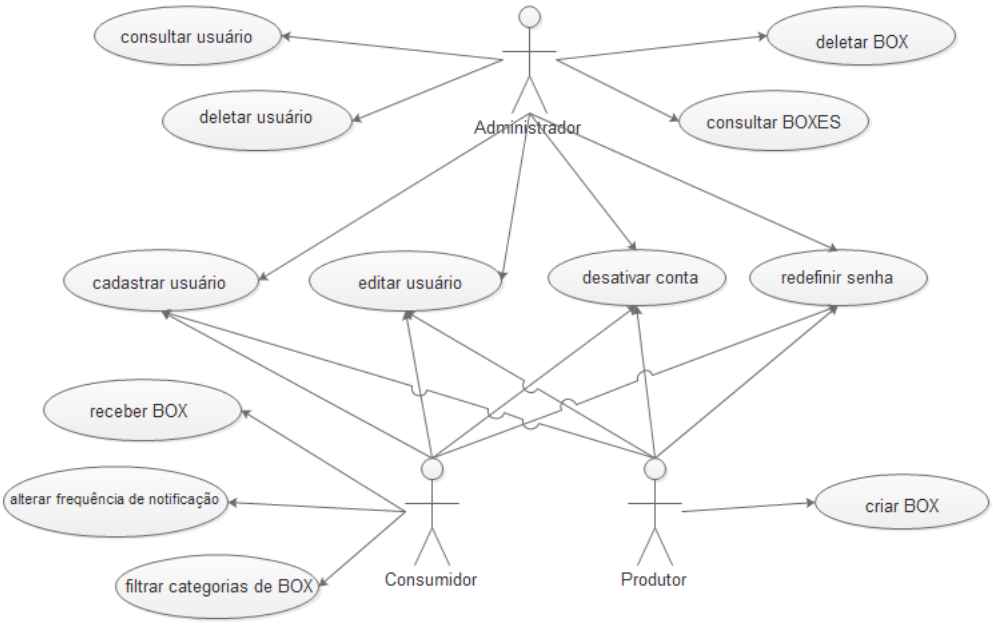


Figura 16 - Diagrama de casos de uso

Caso de uso 1: Cadastrar usuário

Descrição: Criar uma conta de usuário, com nome, *e-mail*, senha e data de nascimento.

Evento iniciador: Usuário solicita registrar novo usuário.
Atores: Produtor, Consumidor e Administrador
Pré-condição: Não há.
Sequência de eventos:
1. Sistema fornece formulário com nome, *e-mail*, senha e data de nascimento para registro.
2. Usuário preenche formulário e envia.
Pós-condição: Registro de novo usuário no sistema.
Extensões: UC002: Editar Usuário; UC003: Redefinir Senha; UC004: Desativar Conta
Inclusões: Não há.

Caso de uso 2: Editar usuário
Descrição: Editar nome, senha ou data de nascimento de um usuário.
Evento iniciador: Usuário solicita edição de usuário.
Atores: Produtor, Consumidor e Administrador
Pré-condição: Estar logado no sistema.
Sequência de eventos:
1. Sistema fornece formulário com nome, senha e data de nascimento para edição.
2. Usuário/Administrador preenche formulário e envia.
Pós-condição: Atualização de registro no sistema.
Extensões: Não há.
Inclusões: UC003: Redefinir Senha

Caso de uso 3: Redefinir senha
Descrição: Alterar senha de usuário para uma conhecida e a envia para o *e-mail* do usuário.
Evento iniciador: Usuário solicita edição de senha.
Atores: Produtor, Consumidor e Administrador
Pré-condição: Estar logado no sistema.
Sequência de eventos:
1. Sistema fornece formulário para redefinir senha.
2. Usuário preenche formulário e envia.
Pós-condição: Atualização de senha no sistema.

Extensões: Não há.
Inclusões: Não há.

Caso de uso 4: Desativar conta
Descrição: Desativa sua conta e todos os conteúdos que foram criados pela mesma.
Evento iniciador: Usuário solicita desativação do usuário.
Atores: Produtor e Consumidor
Pré-condição: Estar logado no sistema.
Sequência de eventos:
1. Sistema pede confirmação.
2. Usuário confirma.
Pós-condição: Conta desativada e conteúdos criados pelo usuário são deletados.
Extensões: Não há.
Inclusões: Não há.

Caso de uso 5: Criar uma box
Descrição: Cria uma box utilizando informações como nome, local, conteúdo e regras de acesso para uma box.
Evento iniciador: Usuário solicita criação de box.
Atores: Produtor
Pré-condição: Estar logado no sistema.
Sequência de eventos:
1. Sistema adquire localização do usuário através do GPS.
2. Sistema fornece formulário com nome, conteúdo e regras de acesso da box.
3. Usuário preenche formulário e envia.
Pós-condição: Box é criada e adicionada na lista de Boxes Criadas do usuário.
Extensões: UC005: Receber uma Box; UC013: Deletar Box;
Inclusões: Não há.

Caso de uso 6: Receber uma box
Descrição: Permite usuário receber conteúdo de uma box, desde que esteja na localização de ativação da mesma.
Evento iniciador: Usuário deve estar dentro da cerca virtual de uma box.

Atores: Consumidor
Pré-condição: Sistema estar rodando.
Sequência de eventos:
1. Sistema notifica usuário sobre ele ter encontrado uma box.
2. Sistema salva a box no registro do usuário.
Pós-condição: Box adicionada na lista de Boxes Recebidas
Extensões: UC007: Alterar frequência de notificação; UC011: Consultar Boxes;
Inclusões: Não há.

Caso de uso 7: Alterar frequência de notificação
Descrição: Permite usuário alterar a frequência que deseja receber notificação de recebimento de Box.
Evento iniciador: Usuário clica para alterar frequência.
Atores: Consumidor
Pré-condição: Sistema estar rodando.
Sequência de eventos:
1. Consumidor seleciona a frequência desejada.
Pós-condição: Sistema salva a frequência no registro do usuário.
Extensões: Não há.
Inclusões: Não há.

Caso de uso 8: Filtrar categorias de Box
Descrição: Permite usuário filtrar o tipo de conteúdo que deseja receber a partir das categorias da Box.
Evento iniciador: Usuário clica para escolher categoria.
Atores: Consumidor
Pré-condição: Sistema estar rodando.
Sequência de eventos:
1. Consumidor seleciona a categoria desejada.
Pós-condição: Sistema salva a categoria no registro do usuário.
Extensões: Não há.
Inclusões: Não há.

Caso de uso 9: Consultar usuários
Descrição: Permite visualizar uma lista com todos os usuários do sistema.
Evento iniciador: Administrador seleciona a opção Users no dashboard
Atores: Administrador
Pré-condição: Estar no *dashboard* do sistema
Sequência de eventos:
1. Sistema exibe lista com todos os usuários
Pós-condição: Não há
Extensões: Não há.
Inclusões: Não há.

Caso de uso 10: Deletar usuário
Descrição: Permite deletar usuário do sistema
Evento iniciador: Acionar o botão delete ao lado do nome do usuário
Atores: Administrador
Pré-condição: Estar na tela de visualização da lista de usuários
Sequência de eventos:
1. Sistema remove usuário do banco de dados
Pós-condição: Usuário removido do sistema
Extensões: Não há.
Inclusões: Não há.

Caso de uso 11: Consultar Boxes
Descrição: Permite fazer uma busca em todas as Boxes do sistema.
Evento iniciador: Administrador seleciona a opção Boxes no dashboard
Atores: Administrador
Pré-condição: Estar no *dashboard* do sistema
Sequência de eventos:
1. Sistema exibe lista com todos os usuários
Pós-condição:
Extensões: Não há.
Inclusões: Não há.

Caso de uso 12: Deletar box

Descrição: Permite deletar box do sistema.

Evento iniciador: Acionar o botão delete ao lado do nome da box

Atores: Administrador

Pré-condição: Estar na tela de visualização da lista de box

Sequência de eventos:

1. Sistema remove box do banco de dados

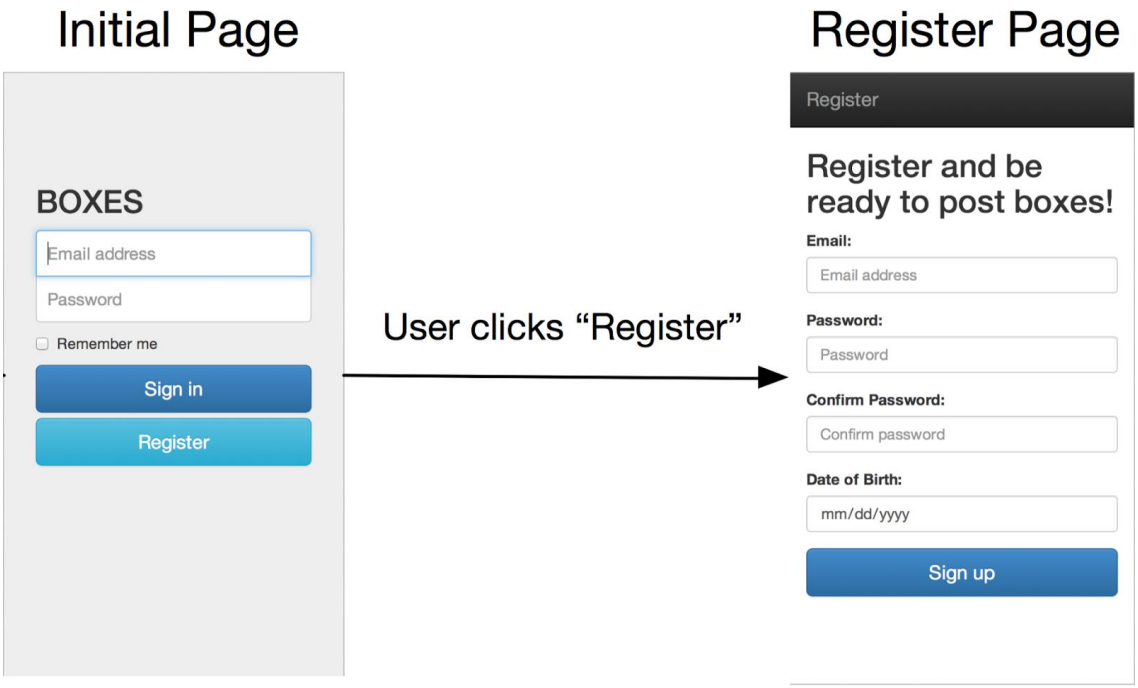
Pós-condição: Box removida do sistema

Extensões: Não há.

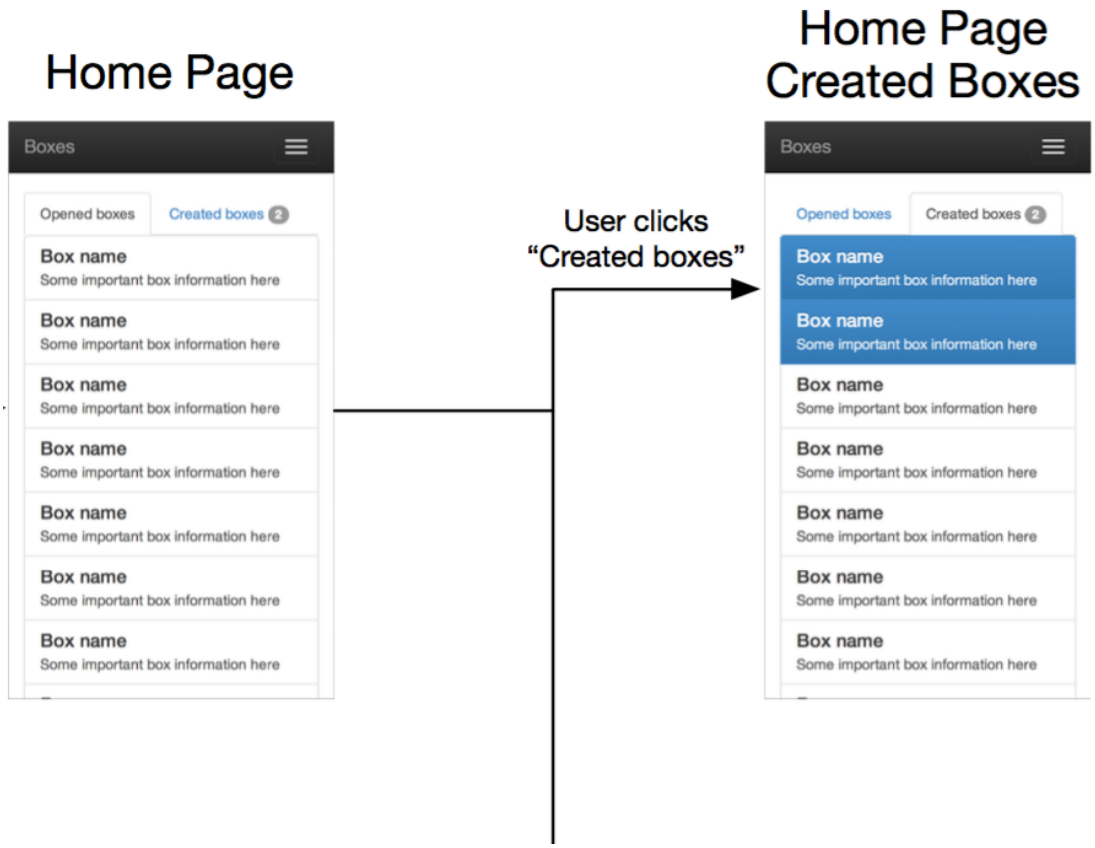
Inclusões: Não há.

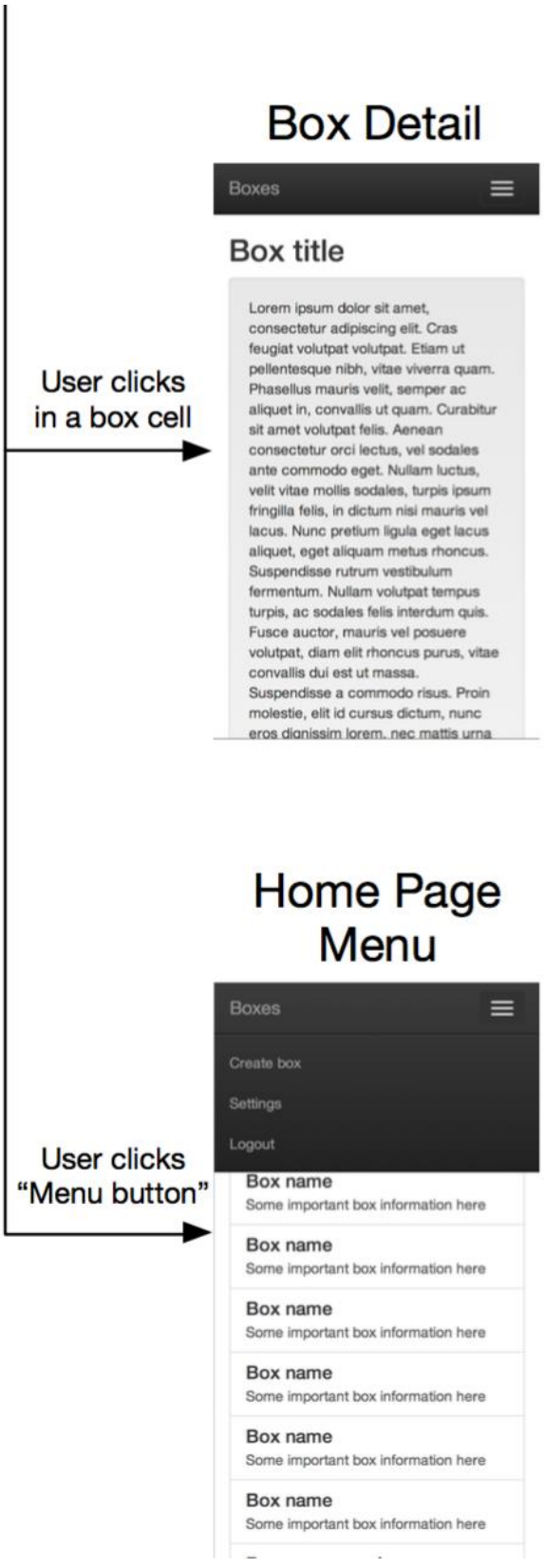
ANEXO B - Protótipos de navegação de tela

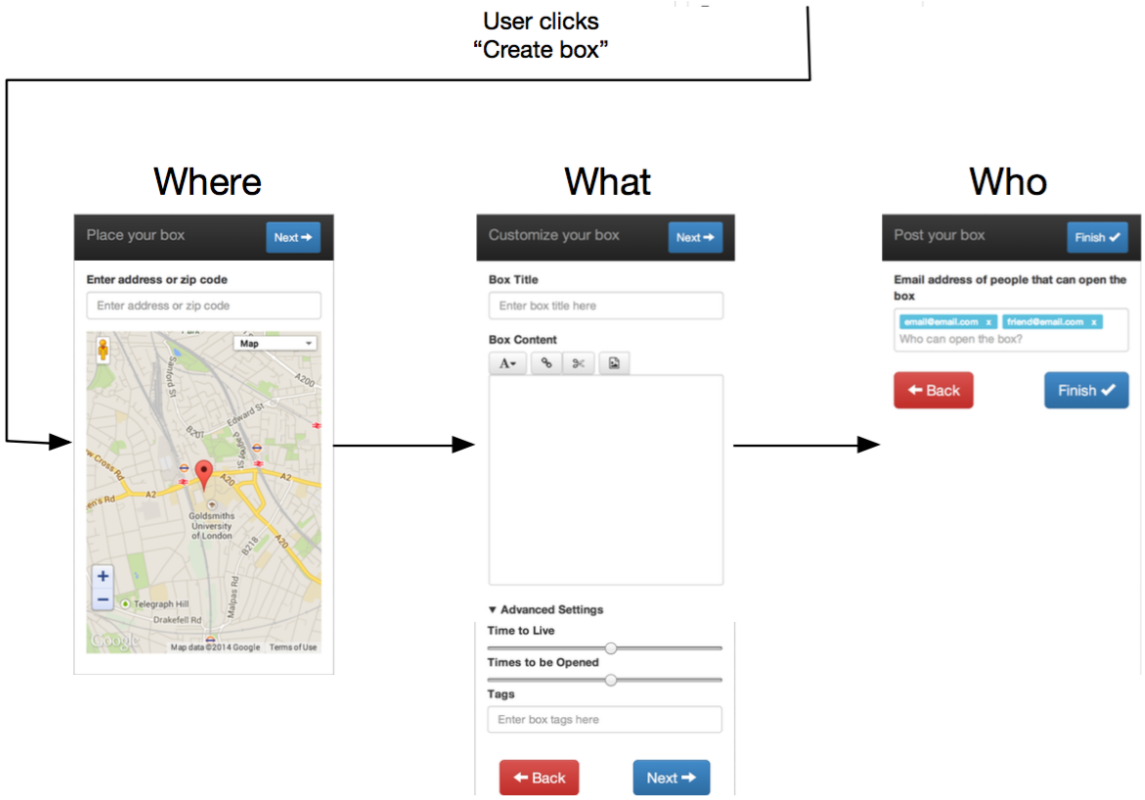
Tela de login



Tela inicial quando logado


Home Page
Boxes
Opened boxes
Created boxes 2
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
User clicks
“Created boxes”
Home Page
Created Boxes
Boxes
Opened boxes
Created boxes 2
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here


Box Detail
Boxes
Box title
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras feugiat volutpat volutpat. Etiam ut pellentesque nibh, vitae viverra quam. Phasellus mauris velit, semper ac aliquet in, convallis ut quam. Curabitur sit amet volutpat felis. Aenean consectetur orci lectus, vel sodales ante commodo eget. Nullam luctus, velit vitae mollis sodales, turpis ipsum fringilla felis, in dictum nisi mauris vel lacus. Nunc pretium ligula eget lacus aliquet, eget aliquam metus rhoncus. Suspendisse rutrum vestibulum fermentum. Nullam volutpat tempus turpis, ac sodales felis interdum quis. Fusce auctor, mauris vel posuere volutpat, diam elit rhoncus purus, vitae convallis dui est ut massa. Suspendisse a commodo risus. Proin molestie, elit id cursus dictum, nunc
User clicks in a box cell
Home Page Menu
Boxes
Create box
Settings
Logout
User clicks “Menu button”
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here
Box name
Some important box information here


User clicks
“Create box”
Where
Place your box
Next
Enter address or zip code
Enter address or zip code
Map
What
Customize your box
Next
Box Title
Enter box title here
Box Content
Advanced Settings
Time to Live
Times to be Opened
Tags
Enter box tags here
Back
Next
Who
Post your box
Finish
Email address of people that can open the box
Who can open the box?
Back
Finish

APÊNDICE A - Códigos do Servidor

Obtenção de grupo de boxes

```
// Users API responder
var db = require('../../models');
var helper = require('../../helpers/application_helpers');
var Q = require('q');

// Get Current Radius Event
// param data has:
// - lat (client latitude)
// - lng (client longitude)
// - userID
exports.getCurrentRadius = function(data, callback) {
 co
 var currentRadius = []; // Current Radius
 var searchLimit = 20; // Search limit determines how many boxes will
be returned in order of distance

 // First filter
 // Get all the boxes that are active
 db.Box.findAll({
   where: {
     status: 'ACTIVE'
   },
   include: [db.Loc, { model: db.User, as: 'canOpen' }]
 }).success(function(boxes) {
   // For each box add temp attribute distanceToClient with the distance
between box and client
   boxes.forEach(function(box) {
     var isBoxValid = true;

     // Chech if box is for the active client, if not so remove from
the array
     var searchFlag = false;
     if(!box.forEveryone) {
       box.canOpen.forEach(function(user) {
```

```
            if(user.id == data.clientID){
              searchFlag = true;
            }
          });

          if(!searchFlag) {
            var isBoxValid = false;
          }
        }

        // Evaluate the validity of the box
        if(isBoxValid) {
          // Add distance to client attribute
          var boxLat = box.loc.lat;
          var boxLng = box.loc.lng;

          box.distanceToClient                                                    =
helper.getDistanceBetween2Points(data.lat, data.lng, boxLat, boxLng);
        } else {
          // Remove box from array
          boxes.splice(boxes.indexOf(box), 1);
        }
      });

      // Ordenate the boxes array so the ones closer to the client stand
on top
      //   http://stackoverflow.com/questions/1129216/sorting-objects-in-
an-array-by-a-field-value-in-javascript
      function compare(a,b) {
        if (a.distanceToClient < b.distanceToClient)
           return -1;
        if (a.distanceToClient > b.distanceToClient)
          return 1;
        return 0;
      }

      boxes.sort(compare);
```

```
    // Slice the array so we get only the searchLimit ones
    boxes = boxes.slice(0, searchLimit);

    // Create smaller box Objects so we only transmit the essencial
information over to the client
    boxes.forEach(function(box) {
      var optimizedBox = {
        id: box.id,
        lat: box.loc.lat,
        lng: box.loc.lng,
        radius: box.loc.radius
      }

      currentRadius.push(optimizedBox);

    return callback(currentRadius);
  });
};
```

Obtenção do conteúdo de um box

```
// Get Box Event - It already updates the log
// param data has:
// - clientID
// - boxID
exports.getBox = function(data, callback) {
 db.Log.find({
    where: {
      UserId  : data.clientID,
      BoxId   : data.boxID
    }
 }).success(function(l
    if(log == null) {
      // Update Log entry for client openning the box at this time using
promises
      // http://www.html5rocks.com/en/tutorials/es6/promises/
```

```
        Q.all([
          db.User.find(data.clientID),
          db.Box.find({
            where: {
              id: data.boxID,
              status: 'ACTIVE'
            },
            include: [db.Content]
          })
        ]).then(function(response){
          // Response[0] is for the User find response and Response[1] for
    the Box find response
          var user = response[0];
          var box = response[1];

          // Check validity of the box
          var unableBox = false;
          // Check if box was open more times than could be opened
          if(box.canBeOpen <= box.opened) {
            isBoxValid = false;
            unableBox = true;
          }

          // Check if date is expired
          var boxDate = new Date(box.lifetime);
          if(boxDate.getTime() <= Date.now()) {
            isBoxValid = false;
            unableBox = true;
          }

          // If box also needs to be deactivate take this extra step
          if(unableBox) {
            box.status = 'NOTACTIVE';
            box.save();
          } else {
            // update number of times opened
            ++box.opened;
            box.save();
```

```
            db.Log.create({ timestamp: new Date().getTime()/10000 })
            .success(function(log) {
              log.setUser(user);
              log.setBox(box);
            })

            // Prepare content object to send to client
            var boxContent = {
              id: box.id,
              title: box.title,
              content: box.contents[0].textContent
            };

            callback(boxContent);
          }

        }).catch(function (err) {
          console.log("There was an error " + err + " while resolving the
promise");
        });
      }
    });
};
```

Registra nova sessão de usuário

```
// Users API responder
var db = require('../../models');

// New Session Event
// param data has:
// - deviceID
// - clientID
// - lat (client latitude)
// - lng (client longitude)
exports.registerNewSession = function(deviceID, clientID, lat, lng) {
 db.User.find({ where: { id: clientID},
```

```
      include: [db.Loc]
    })
    .success(function (user){
      // Update the device ID
      user.deviceID = deviceID;

      // Store the user's current location
      if (typeof user.loc === 'undefined' || user.loc === null) {
        db.Loc.create({
          lat : lat,
          lng : lng
        })
        .success(function (loc){
          user.setLoc(loc);
        });
      } else {
        user.loc.lat = lat;
        user.loc.lng = lng;
      }

      // Save updates to user
      user.save();
      console.log("Register New Session with deviceID = " + deviceID + "
and clientID:" + clientID);
    });
}
```

Código auxiliar para calcular distâncias em um mapa utilizando função de haversine

```
// Application Helper
// Helper methods that don't regard the controller or are specific
should go here

// Get distance between two points in a map
// #=> Return in KM
exports.getDistanceBetween2Points = function(lat1,lon1,lat2,lon2) {
  function deg2rad(deg) {
```

```
    return deg * (Math.PI/180)
  }
  var R = 6371; // Radius of the earth in km
  var dLat = deg2rad(lat2-lat1);  // deg2rad below
  var dLon = deg2rad(lon2-lon1);
  var a =
    Math.sin(dLat/2) * Math.sin(dLat/2) +
    Math.cos(deg2rad(lat1)) * Math.cos(deg2rad(lat2)) *
    Math.sin(dLon/2) * Math.sin(dLon/2)
    ;
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
  var d = R * c * 1000; // Distance in m
  return d;
}
```

APÊNDICE B - Códigos da aplicação móvel

Estabelece e controla conexão entre cliente e servidor

```
/*
* Socket handler && Daemon
*/

// Connection string
// -> Development = 158.223.169.110:3000
// -> Production = http://doc.gold.ac.uk:65045
// var connectionString = "http://10.170.174.56:3000";
var connectionString = "http://doc.gold.ac.uk:65045";

var socket = null;  // Socket that connects to the server

// We will have four local databases
// incomingBoxesDB  -> For the itenerary searches based on location
// openedBoxesDB    -> For the User's opened Boxes
// createdBoxesDB   -> For the boxes created by this user
var incomingBoxesDB = TAFFY();
var openedBoxesDB = TAFFY();
var createdBoxesDB = TAFFY();

// Logged user ID
var currentUserID = localStorage.getItem('userID');

var geolocationWatchID = null; // Watch Position ID
var geolocationOptions = { enableHighAccuracy: true };

// The object that states the current radius limit
// it has:
// lat
// lng
// distance
var firstLocation = {lat: null, lng: null};
var currentRadiusLimit = null;

var connection = {
 // connection Constructor
 initialize: function() {
   socket = io(connectionString);
   this.bindMessages();
   this.newSession();
   this.fetchData();
 },
 // Bind messages
 bindMessages: function () {
   console.log("Bind Messages");
   // When receiving a new radius of boxes update entry
   // data response consists of an array with boxes that have
   // id
   // lat
   // lng
```

```
    // radius
    socket.on('currentRadius', function(data){
      // Remove all past records
      incomingBoxesDB().remove();
      // Interate over response and add to Local Storage
      data.forEach(function(box) {
        var boxEntry = new Box(box.id, box.lat, box.lng,
box.radius);
        incomingBoxesDB.insert(boxEntry);
      });

      // Get the location and distance from last box and store it so
we can get the next current radius
      var lastBox = incomingBoxesDB({id: data[data.length -
1].id}).first();
      currentRadiusLimit = new Box(0, firstLocation.lat,
firstLocation.lng, lastBox.distanceFrom(firstLocation.lat,
firstLocation.lng));

      // Throw an error if no update is received every 30 seconds
      geolocationWatchID =
navigator.geolocation.watchPosition(checkBoxes,
watchPositionFailed, geolocationOptions);
    });

    // When receiving a new box content
    // data response consists of an object that have
    // id
    // title
    // content
    socket.on('boxContent', function(data) {
      console.log("Box Content")
      console.log(data);
      // Adjust the receivebox in the localstorage database
      var receivedBox = incomingBoxesDB({id: data.id}).first();
      receivedBox.setTitle(data.title);
      receivedBox.setContent(data.content);

      // Display a native confirm to check if user wants to proceed
with box
      var onConfirm = function () {
        openModal(receivedBox);
      }

      navigator.notification.vibrate(1000);
      window.plugin.notification.local.add({ title: 'New box!' ,
message: 'You have a new box! Open your app to see what\'s
inside.', badge: 1, autoCancel: true });
      navigator.notification.confirm(
        'Check it out, you just got the '  + receivedBox.title + '
box! Do you want to open it now?', // message
         onConfirm,            // callback to invoke with index of
button pressed
        'New Box!',           // title
        'Open,Cancel'         // buttonLabels
      );

      // Fetch new data
```

```
        connection.fetchData();
      });

      // When receiving the opened boxes
      // data response consists of an array of objects that have
      // id
      // title
      // description
      // content
      socket.on('openedBoxes', function(data) {
        console.log("Box Content")
        console.log(data);

        // Remove all past records
        openedBoxesDB().remove();
        // Interate over response and add to Local Storage
        data.forEach(function(box) {
          var boxEntry = new Box(box.id, null, null, null);
          boxEntry.setTitle(box.title);
          boxEntry.setContent(box.content);
          boxEntry.setDescription(box.description);

          openedBoxesDB.insert(boxEntry);
        });

        renderOpenedBoxes();
      });

      // When receiving the created boxes
      // data response consists of an array of objects that have
      // id
      // title
      // description
      // content
      socket.on('createdBoxes', function(data) {
        console.log("Box Content")
        console.log(data);

        // Remove all past records
        createdBoxesDB().remove();
        // Interate over response and add to Local Storage
        data.forEach(function(box) {
          var boxEntry = new Box(box.id, null, null, null);
          boxEntry.setTitle(box.title);
          boxEntry.setContent(box.content);
          boxEntry.setDescription(box.description);

          createdBoxesDB.insert(boxEntry);
        });

        renderCreatedBoxes();
      });

      // When a new box is added to the database the app needs to be
    refreshed
      // TODO: Simply ask for a new current radius
      socket.on('refreshRadius', function(){
        console.log("Refresh Request");
```

```
        location.reload();
      })
    },
    // Starts new session
    newSession: function () {
      console.log("New Session")
      navigator.geolocation.getCurrentPosition(function (position) {
        firstLocation.lat = position.coords.latitude;
        firstLocation.lng = position.coords.longitude;

        console.log("Lat" + firstLocation.lat);
        console.log("Lng" + firstLocation.lng);
        console.log("User ID " + currentUserID);

        socket.emit('newSession', { deviceID: device.uuid, clientID:
currentUserID, lat: firstLocation.lat, lng: firstLocation.lng });
      }, function(error) {
        console.log('code: '    + error.code    + '\n' +
                    'message: ' + error.message + '\n');
      }, geolocationOptions);
    },
    // Get Current Radius
    getCurrentRadius: function (clientLat, clientLng) {
      firstLocation.lat = clientLat;
      firstLocation.lng = clientLng;

      socket.emit('getCurrentRadius', { lat: clientLat, lng:
clientLng, clientID: currentUserID});
    },
    // Get Box
    getBox: function(confirmedBox) {
      console.log("Get Box")
      socket.emit('getBox', { clientID: currentUserID, boxID:
confirmedBox.id });
    },
    // Fetch data
    fetchData: function () {
      socket.emit('fetchData', { clientID: currentUserID });
    },
    // If application offline cancel running connection operations
    destroy: function () {

    }
};
```

Rotina que checa se esta dentro de um box

```
// Check Boxes
var checkBoxes = function (position){
 console.log("Check Boxes");
 var confirmedBoxes = []; // array with the boxes that the user is
inside

 // Position of the client
 var clientLat = position.coords.latitude;
 var clientLng = position.coords.longitude;
```

```
 // Check if I'm in the current radius
 if(currentRadiusLimit.distanceFrom(clientLat, clientLng) <=
currentRadiusLimit.radius/2){
   console.log("User is inside Current Radius.");
   // Go over the Database and check if the user is inside each box
   incomingBoxesDB().each(function(box) {
     console.log("Box Check " + box.id);
     if (box.isUserInside(clientLat, clientLng)) {
       console.log("User is inside " + box.id);
       connection.getBox(box);
     }
   });
 } else {
   connection.getCurrentRadius(clientLat, clientLng);
 }
};
```

Abstração do modelo de box utilizada no cliente

```
// Box model
var Box = function (id, lat, lng, radius) {
 this.id           = id;
 this.title        = null;
 this.lat          = lat;
 this.lng          = lng;
 this.radius       = radius;
 this.content      = null;
 this.description  = null;

 this.setTitle = function (title) { this.title = title; };
 this.setContent = function (content) { this.content = content; };
 this.setDescription = function (description) { this.description =
description; };

 this.isUserInside = function (clientLat, clientLng) {
   var distance = getDistanceBetween2Points(this.lat, this.lng,
clientLat, clientLng);
   return ( distance < this.radius ? true : false );
 };

 this.distanceFrom = function(lat, lng) {
   return getDistanceBetween2Points(this.lat, this.lng, lat, lng);
 }
}

// Helper methods that don't regard the controller or are specific
should go here

// Get distance between two points in a map
// #=> Return in M
var getDistanceBetween2Points = function(lat1,lon1,lat2,lon2) {
 function deg2rad(deg) {
   return deg * (Math.PI/180)
 }
 var R = 6371; // Radius of the earth in km
 var dLat = deg2rad(lat2-lat1);  // deg2rad below
```

```
    var dLon = deg2rad(lon2-lon1);
    var a =
      Math.sin(dLat/2) * Math.sin(dLat/2) +
      Math.cos(deg2rad(lat1)) * Math.cos(deg2rad(lat2)) *
      Math.sin(dLon/2) * Math.sin(dLon/2)
      ;
    var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
    var d = R * c * 1000; // Distance in m
    return d;
  }
```