UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC
CURSO DE CIÊNCIA DA COMPUTAÇÃO

MARIA EDUARDA LAVINA

MONITORAMENTO DE PREÇOS DE PRODUTOS DE LOJAS VIRTUAIS UTILIZANDO MASHUPS

CRICIÚMA
2015

MARIA EDUARDA LAVINA

MONITORAMENTO DE PREÇOS DE PRODUTOS DE LOJAS VIRTUAIS UTILIZANDO MASHUPS

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. Esp. Gilberto Vieira da Silva

CRICIÚMA

2015

MARIA EDUARDA LAVINA

MONITORAMENTO DE PREÇOS DE PRODUTOS DE LOJAS VIRTUAIS UTILIZANDO MASHUPS

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia e Gestão do Conhecimento.

Criciúma, 26 de junho de 2015.

BANCA EXAMINADORA

Prof. Gilberto Vieira da Silva - Especialista - (UNESC) - Orientador

Prof. Gustavo Bisognin - Mestre - (UNESC)

Prof. Luciano Antunes - Mestre - (UNESC)

À minha família.

AGRADECIMENTOS

Aos meus pais, que me apoiaram em todos os momentos de dúvida, de desânimo, de alegrias e tristezas. A eles, que me ensinaram todos os valores que uma pessoa deve ter e praticar ao longo da vida e, com certeza, não existem pais melhores no mundo.

Ao Gilberto, que aceitou o convite para ser meu orientador e foi tão paciente e me ajudou muito na caminhada até aqui.

Ao professor Luciano Antunes, que me proporcionou a oportunidade de ganhar a bolsa PIC 170, sendo de grande auxílio financeiro para a minha graduação.

Ao George Lucca, meu sobrinho e afilhado, que me alegrava nos momentos de cansaço e estresse, ao mostrar seus dentinhos crescendo, ao dizer suas primeiras palavras, ao dar seus primeiros passos e ao rir para mim.

“Se você tem uma maçã e eu tenho uma maçã e nós trocamos essas maçãs, então eu e você ainda teremos uma maçã cada. Mas se você tiver uma idéia e eu tiver uma idéia e nós trocamos idéias, então cada um de nós terá duas idéias.”

(George Bernard Shaw)

RESUMO

Em um cenário onde existem diversas empresas varejistas oferecendo produtos similares, as lojas procuram cada vez mais disponibilizar ofertas com preços atrativos aos clientes, criando uma grande variação de preços no mercado. Os clientes, por sua vez, buscam as melhores ofertas dos produtos que são do seu interesse, porém, existe uma carência de ferramentas que fazem o acompanhamento destas ofertas, sendo que grande parte apenas compara preços na *web*. Este fato foi identificado por meio da pesquisa e auxiliou na definição da arquitetura e no desenvolvimento do protótipo que objetiva auxiliar na resolução do mesmo. O protótipo foi criado à base de *mashups,* onde a busca das ofertas na *web* foi feita por meio de uma API de terceiros, que disponibiliza ofertas de diversas lojas varejistas do *e-commerce,* e a validação do protótipo foi feita monitorando-se alguns produtos durante um determinado período de tempo, com isso, foi possível obter diversas informações. Estes resultados apontaram que existe uma grande variação de preços de produtos similares entre as diversas lojas e em uma mesma loja, dentro de um período de tempo, evidenciando a viabilidade do protótipo. A partir desta validação e dos resultados obtidos, constatou-se que o protótipo é uma ferramenta eficiente no acompanhamento de preços de produtos de lojas virtuais.

Palavras-chave: Preço. Mobile. Mashups. REST.

ABSTRACT

In a scenario where several retail companies offer similar products, the shops are increasingly looking for offering attractive prices to customers, creating a wide range of market prices. Customers, in turn, seek the best deals on products that are of their interests, however, there is a lack of tools that help tracking these offers, much of which only compares prices on the web. This fact was identified through research and it assisted in the architecture definition and development of the prototype that aims to help in the resolution thereof. The prototype was created based on mashups, and the search for deals on the web was made by a third-party API, that provides deals of several retail e-commerce stores, and the prototype validation was performed by monitoring up some products during a given period of time, with this, it was possible to obtain various information. These results indicate that there are a wide range of similar products prices between different shops and at the same store, within a period of time, showing the viability of the prototype. From this validation and the results obtained, it was found that the prototype is an effective tool in monitoring products prices at web shops.

Key words: Prices. Mobile. Mashups. REST.

LISTA DE ILUSTRAÇÕES

Figura 1 - Modelo com os conceitos e noções da WEB 2.0 27
Figura 2 - Representação da camada mashup 33
Figura 3 - Arquitetura server-based.....35
Figura 4 - Arquitetura client-based 36
Figura 5 - Arquitetura mobile-based.....37
Figura 6 - Diversas representações endereçadas por uma única URI 43
Figura 7 - Gráfico referente à utilização das versões Android.....51
Figura 8 - Modelo relacional.....61
Figura 9 – Comunicação entre protótipo e API.....64
Figura 10 – Mashup do protótipo.....66
Figura 11 – Comunicação entre protótipo e AppService 69
Figura 12 – Home.....70
Figura 13 – Pesquisa de produtos.....70
Figura 14 – Ofertas selecionadas para monitoramento.....71
Figura 15 – Ofertas monitoradas.....72
Figura 16 – Mensagem de confirmação enviada ao usuário.....72
Figura 17 – Indicadores de variação de preço 73
Figura 18 – Notificação de variação de preço 74
Figura 19 - Configurações.....75
Figura 20 – Gráfico da variação de preços do Smartphone Motorola Novo Moto G Android 4.4 Tela 5" 16GB em R$ 76
Figura 21 – Gráfico da variação de preços do livro O Mundo de Gelo e Fogo em R$ 77
Figura 22 – Gráfico da variação de preços da Fritadeira Air Fry Saúde Britânia Branca em R$ 78
Figura 23 – Gráfico de variação de preços da Smart TV 3D LED 50' Philips Ultra HD 4K em R$ 79

Quadro 1 - Resumo dos elementos de REST 42
Quadro 2 – Requisitos funcionais.....58

Quadro 3 – Requisitos não funcionais.....60
Quadro 4 – Produtos monitorados 75

LISTA DE TABELAS

Tabela 1 – Preços do Smartphone Motorola Novo Moto G Android 4.4 Tela 5" 16GB em R$.....76
Tabela 2 – Preços do livro O Mundo de Gelo e Fogo em R$.....77
Tabela 3 – Preços da Fritadeira Air Fry Saúde Britânia Branca em R$ 78
Tabela 4 – Preços do livro O Mundo de Gelo e Fogo em R$.....79

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
B2B	Bussines-To-Bussines
B2C	Bussines-To-Consumers
DOM	Document Object Model
FTP	File Transfer Protocol
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
REST	Representational State Transfer
RPC	Remote procedure call
RSS	Rich Site Summary
SDK	Software Development Kit
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SPAM	Sending and Posting Advertisement in Mass
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	eXtensible Markup Language

SUMÁRIO

1 INTRODUÇÃO 14
1.1 OBJETIVO GERAL 16
1.2 OBJETIVOS ESPECÍFICOS 16
1.3 JUSTIFICATIVA 16
1.4 ESTRUTURA DO TRABALHO 18
2 COMÉRCIO ELETRÔNICO 20
2.1 M-COMMERCE 21
2.2 O CONSUMIDOR ONLINE 22
2.3 MARKETING DIGITAL 23
2.3.1 Estratégias de marketing digital 23
2.4 DISPONIBILIDADE DE INFORMAÇÕES PESSOAIS 25
3 WEB 2.0 27
3.1 EXTRAÇÃO DE DADOS 28
3.1.1 API´s 28
3.1.2 Mashups 29
3.1.2.1 Tipos de mashups 32
3.1.2.2 Camadas do mashup 33
3.1.2.3 Arquitetura do mashup 35
4 RECURSOS TECNOLÓGICOS 38
4.2 WEB SERVICES 38
4.2.1 SOAP 40
4.2.2 REST 41
4.2.2.1 Identificador Uniforme de Recursos (URI) 43
4.2.3 Recursos e representações 43
4.2.3.1 XML 45
4.2.3.2 JSON 46
4.3 SISTEMAS OPERACIONAIS MOBILE 47
4.3.1 iOs 47
4.3.2 Windows Phone 48
4.3.3 Android 49

5 TRABALHOS CORRELATOS 52
5.1 MOBILE WEB MASHUPS 52
5.2 APLICATIVOS AGREGADORES DE OFERTAS DE COMPRA COLETIVA UTILIZANDO A PLATAFORMA ANDROID 53
5.3 SOFTWARES ANDROID PARA CORRETORES DE IMÓVEIS 54
5.4 APLICAÇÕES DE MASHUPS NO GERENCIAMENTO DE REDES 55
6 MEU PREÇO - PROTÓTIPO MÓVEL PARA MONITORAMENTO DE PREÇOS DE PRODUTOS DO E-COMMERCE 56
5.1 METODOLOGIA 56
5.1.1 Estudos das ferramentas 57
5.1.2 Requisitos funcionais 57
5.1.3 Requisitos não funcionais 59
5.1.4 Modelo conceitual 61
5.1.5 Desenvolvimento do protótipo 61
5.1.5.1 API Buscapé 62
5.1.5.2 Arquitetura 64
5.1.5.3 O protótipo 65
5.1.6 Testes 67
5.1.6.1 AppService 68
5.1.7 Funcionamento do protótipo 69
5.2 RESULTADOS DO MONITORAMENTO 75
7 CONCLUSÃO 81
REFERÊNCIAS 84
ANEXOS 92
APÊNDICE A - Artigo 97

1 INTRODUÇÃO

Devido à grande concorrência do mercado, é possível observar que existe uma grande dispersão de preços entre uma mesma marca e produto no universo de lojas do varejo (SHANKAR; BOLTON, 2004, tradução nossa). O comércio tipo *Business to Consumers* (B2C) é caracterizado pela sua alta volatilidade, pois a disponibilidade de *sites* e a oferta de produtos e serviços variam bastante (NOVAES, 2001). Consequentemente, os preços também oscilam fortemente em função da concorrência e das ofertas especiais.

Motivados pelas mais diversas razões, atualmente, a maioria das empresas já pratica alguma forma de comércio eletrônico, sendo que em 2014 foram feitos 103,4 milhões de pedidos via internet (WEBSHOPPERS, 2015). Segundo uma pesquisa realizada pelo Ibope Mídia (2012), 80% dos internautas brasileiros utilizam a *web* para comparar preços. Com base nos dados apresentados no relatório WebShoppers (2015), pode-se dizer que existem diversos *sites* que oferecem produtos e serviços, tornando trabalhosa a pesquisa de preços individualmente em cada *site*.

Existem diversos serviços de comparação de preços, como por exemplo, *sites* como o Buscapé, Compare Preços e Já Cotei, que fazem comparações de preços em diversas lojas virtuais, mostrando os produtos destas lojas e os seus respectivos preços, sendo que para os usuários terem acesso a estas comparações, estes devem acessar o *site* e consultar os produtos para obter o resultado. Entretanto, o usuário deve estar sempre realizando consultas e acompanhando os preços manualmente.

Os recursos citados acima e que são utilizados pelos usuários e lojas virtuais se caracterizam como *e-commerce*, porém, segundo Souza (2011), 14% dos usuários dizem já ter realizado algum tipo de compra via dispositivos móveis, o *m-commerce*, sendo esta uma das últimas tendências no mercado brasileiro (SOUZA, 2011). Os dispositivos móveis, como *tablets* e *smartphones* necessitam de sistemas operacionais para o seu funcionamento, sendo que, segundo uma pesquisa da Canalys (2012), 79% dos *smartphones* operam com o sistema operacional Android, desenvolvido e mantido pela empresa Google, que também disponibiliza uma loja

virtual (Google Play) onde é possível fazer *download* de aplicativos para as mais diversas finalidades.

Dentro do universo de aplicativos contidos na Google Play (2014), existe uma carência de aplicativos voltados ao *m-commerce*, ainda mais quando se restringe à pesquisa e monitoramento de preços que não se utilizam de *e-mail* ou mensagens via celular para avisar o usuário da variação de preços de determinado produto, impossibilitando que um usuário seja informado da variação de preços somente de produtos que o mesmo tenha interesse, ou seja, não existe uma facilidade e/ou recurso para configuração da informação a ser recebida.

Existem algumas interfaces de programação de aplicação, do inglês *Application Programming Interface* (API), que permitem o acesso por terceiros a ofertas dos *sites*, como é o caso da API do Buscapé, que disponibiliza uma lista de ofertas contidas em sua base dados. Este tipo de serviço, onde uma aplicação busca informações em fontes de dados de terceiros é chamada de *mashup* e o acesso a essas fontes de dados pode ser feita de diversas formas, como *web scraping*, *web services*, *widgets*, entre outros. Segundo Nordström (2010, tradução nossa), os *mashups* são uma maneira eficiente de criar novos serviços sem precisar iniciar aplicações do zero, além de serem reutilizáveis, leves, baseados na *web* e de rápida implementação (LIU et al, 2007, tradução nossa). Existem diversos recursos que são construídos a partir de *mashups*, como por exemplo o site Feedly, que agrupa notícias dos sites selecionados pelo usuário.

Desta forma, tendo como base o relatório WebShoppers (2015), a pesquisa encomendada pela empresa Neoconsumidor (2009) e a visível expansão do comércio *mobile*, foi possível perceber que o leque de aplicativos *mobile*, da plataforma Android, que monitoram preços de produtos previamente selecionados pelo usuário é de uma quantidade limitada, fazendo com que os usuários recorram a alternativas que nem sempre atendem completamente suas necessidades ou que demandam um certo esforço por parte do usuário.

1.1 OBJETIVO GERAL

Desenvolver um protótipo de um aplicativo para Android que monitore os preços de produtos previamente selecionados, fazendo uso de *mashups* por meio de *web services*.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos deste trabalho consistem em:

a) analisar as possibilidades de integração entre diferentes *sites* por meio de *mashups*;
b) utilizar a tecnologia Android;
c) pesquisar sobre o mercado *mobile*;
d) desenvolver o protótipo do software;
e) realizar testes a fim de obter conclusões sobre o modelo proposto.

1.3 JUSTIFICATIVA

O comércio eletrônico faturou R$ 28,8 bilhões em 2013 e dentre os fatores que contribuíram para esse resultado está o *m-commerce*. É o que afirma um artigo publicado no *site* E-commerceBrasil (2014), que ressalta ainda que a oferta de modelos mais simples de *smartphones* e a popularização da internet móvel fizeram com que muitas pessoas de classes mais baixas tivessem acesso à internet e, muitas vezes, se tornassem consumidoras *online*.

Neste cenário, o *m-commerce* se torna uma vantajosa aposta para desenvolvimento de aplicativos, pois, segundo pesquisas da Gouvêa e Souza (2014), os dispositivos móveis serão a principal ferramenta de acesso à internet no mundo em 2025. As previsões apontam que em algum momento as buscas feitas pelos dispositivos móveis superarão as feitas pelo desktop (LIVINGSTON, 2011, tradução nossa).

A utilização de um aplicativo para pesquisa e monitoramento de preços de produtos pré selecionados em lojas virtuais é sustentada pela premissa da

economia, pois, segundo Motta (2009), uma das vantagens da pesquisa de preços é o potencial de economia que pode ser feito pelo consumidor ao pesquisar preços nos diversos varejistas do mercado *online*.

O monitoramento automático de preços diminui a necessidade do acompanhamento contínuo dos produtos desejados por parte do usuário, já que este receberá notificações quando o preço do produto for alterado. O usuário irá acompanhar apenas o preço dos produtos que são de seu interesse, evitando o recebimento de conteúdos sem seu consentimento ou conteúdos que nada dizem respeito ao seu perfil ou desejo de compra. O recebimento de *e-mail´s* não desejados, a falta de relevância dos conteúdos destes *e-mail´s* e a grande quantidade de disparos que eventualmente acontecem desmotiva os consumidores (OLIVEIRA, 2010).

Utilizando um aplicativo para realizar o monitoramento de preços em lojas virtuais *online*, ocorre uma inversão de interesse em relação aos produtos dessas lojas, sendo é o usuário que irá definir o que deseja analisar a e monitorar, não sendo este um método intrusivo, como outros métodos utilizados pelas empresas, podendo citar o envio de publicidade em massa*, do inglês Sending and Posting Advertisement in Mass* (SPAM), e publicidade abusiva. Com a utilização de um aplicativo, a atividade de monitoramento de preços se torne mais prática, uma vez que o mesmo irá notificar o usuário da queda de preços de produtos específicos, que foram previamente escolhidos pelo usuário e, por isso, lhe é do seu interesse. Assim, o acesso aos *sites* de compras para que haja o acompanhamento de preços de produtos têm sua frequência reduzida, evitando assim que esses *sites* recolham dados de navegação que não foram autorizados pelos usuários, além de evitar também a publicidade abusiva, que pode vir a causar incômodos a muitos usuários.

Um aplicativo desenvolvido para o sistema operacional Android é sustentado pela pesquisa feita pela Canalys, onde 79% dos usuários possuem em seus *smartphones* o referido sistema operacional, e, segundo um relatório da Gartner (2014), os *tablets* com o Android são dominantes no mercado, respondendo por 62% das vendas.

Conforme o Google Android (2014), o sistema operacional Android permite que desenvolvedores utilizem diversos serviços e ferramentas de uso livres

e/ou proprietárias, aumentando as possibilidades de um aplicativo tanto no quesito desenvolvimento quanto em seu uso. Neste sentido, um aplicativo para monitorar a variação de preços em diversas lojas virtuais pode ser proporcionado pelo uso de *mashups*.

O *mashup* é uma tecnologia de composição, possibilitando a criação de novos serviços e aplicações a partir da integração de recursos obtidos de diferentes fontes (LIU et al, 2007). Eles propõem um maior dinamismo, através da possibilidade de um processo mais ágil de composição e mais amigável ao usuário (ALVES et al, 2007). Com base em Alves et al (2007) e Liu et al (2007), percebe-se que a viabilidade do projeto aumenta consideravelmente, já que os dados necessários para a construção do protótipo são previamente disponibilizados por uma API já existente.

A API em questão é a API do Buscapé (BUSCAPÉ, 2014), que disponibiliza informações sobre ofertas e produtos contidos em sua base de dados por meio de *web services*.

1.4 ESTRUTURA DO TRABALHO

O trabalho é composto por seis capítulos, sendo iniciado por uma breve introdução, abordando o problema e o conteúdo do trabalho, seguido pelos seus objetivos e justificativa.

No capítulo 2, são descritos os conceitos do *e-commerce*, do *mobile* commerce e de algumas formas existentes de propaganda e anúncio de ofertas no mundo digital. Este capítulo também descreve o comportamento do consumidor e qual a importância do preço no processo de compra de um produto ou serviço.

O estudo sobre a *Web* 2.0 e todos os componentes que ela proporcionou, inclusive a técnica de *mashups* está contido no capítulo 3.

Já no capitulo 4, são abordadas as ferramentas e recursos existentes que podem ser utilizados para construir aplicativos para dispositivos móveis, bem como as ferramentas e recursos que serão utilizados no desenvolvimento do protótipo proposto.

O capítulo 5 abrange os trabalhos correlatos desta pesquisa.

O desenvolvimento do protótipo, incluindo a descrição do seu funcionamento, está descrita no capítulo 6.

2 COMÉRCIO ELETRÔNICO

O comércio eletrônico consiste na realização de negócios por meio da internet, incluindo a venda de produtos e serviços físicos, entregues *off-line*, e de produtos que podem ser designados e entregues *online*, nos segmentos de mercado consumidor, empresarial e governamental (LIMEIRA, 2003).

Gordon e Gordon (2006) dizem que o comércio eletrônico ou *e-commerce* é toda a atividade desenvolvida na internet, com a utilização de ferramentas eletrônicas e tecnologias emergentes, que tem como principal objetivo a negociação de compra e venda de bens e serviços. Em 2006, Gates previa que em alguns anos vão existir dois tipos de empresas: as que fazem negócios pela internet e as que estão fora dos negócios.

O relatório WebShoppers (2015), divulgado pela empresa E-Bit, que analisa a evolução do comércio eletrônico, as mudanças de comportamento e preferências dos consumidores *online*, mostra que no ano de 2014 o comércio eletrônico brasileiro faturou R$ 35,08 bilhões, superando o mesmo período em 2013 (quando registrou R$ 28,08 bilhões), e com crescimento nominal de 24% no setor. A empresa E-Bit, responsável pelo relatório, prevê um crescimento no setor de *e-commerce* próximo a 20% para o ano de 2015 e o faturamento deve chegar a R$ 43 bilhões. Além disso, o relatório mostra que o *e-commerce* ganhou 10,2 milhões de novos consumidores no ano de 2014.

O *e-commerce* permite que uma empresa, mesmo sendo pequena, tenha presença global, beneficiando os clientes, que podem escolher onde comprar, independentemente da sua localização geográfica, sendo este um processo válido tanto na relação empresa-empresa como na relação empresa-consumidor (BERTAGLIA, 2003).

De acordo com Potter, Turban e Rainer (2005), existem vários tipos de comércio eletrônico, sendo que os mais comuns são:

a) *Bussines-To-Bussines* (B2B): é a negociação eletrônica entre empresas;

b) *Bussines-To-Consumers* (B2C): é a negociação eletrônica entre empresas e consumidores. Esta modalidade representa a virtualização da compra e venda.

Além dos modelos descritos acima, outros são citados em Catalani (2008), como, por exemplo, o comércio móvel ou *m-commerce*, que se refere ao comércio eletrônico realizado inteiramente em um ambiente sem fio.

2.1 M-COMMERCE

M-commerce é a compra e venda de produtos e serviços por meio de dispositivos móveis, sendo um subgrupo do *e-commerce* e se diferenciando pela sua mobilidade, a qual possibilita a conexão a qualquer hora em tempo real (TURBAN; WETHERBE; MCLEAN, 2002).

Sistemas computacionais móveis são sistemas que podem facilmente ser movidos fisicamente ou cujas capacidades podem ser utilizadas enquanto eles estão sendo movidos (B'FAR, 2005, tradução nossa). Um dispositivo móvel deve ser portável, pessoal, de fácil e rápida utilização e deve possuir conectividade (FIRTMAN, 2010, tradução nossa).

Podem ser considerados dispositivos móveis *smartphones*, *tablets*, laptops, entre outros, e, devido a esta grande diversidade de dispositivos móveis nas mãos das pessoas, existe uma grande quantidade de usuários em potencial que irão pagar para fazer *download* de aplicativos (BANGA; WEINHOLD, 2014, tradução nossa).

Segundo o IDC (2014), em 2013 foram vendidos 8,4 milhões de *tablets*, o que representou um crescimento de 57% se comparado ao volume comercializado em 2012, superando as vendas de notebooks em mais de 800 mil unidades. Em 2013, as vendas de *smartphones* somam 35,6 milhões de unidades, número recorde e que representa um crescimento de 123% em relação ao ano anterior (IDC, 2014).

O relatório WebShoppers (2015), mostrou que o faturamento das transações realizadas por dispositivos móveis no Brasil em 2014 cresceu significativamente, em comparação com o ano de 2013, apresentando R$ 3,4 bilhões – diante dos R$ 1,13 bilhão no ano de 2013. Das compras realizadas em

aparelhos móveis, 65% são originadas por *smartphones*, enquanto os 35% restantes são de *tablets* (via *sites* sem uso de aplicativos).

De acordo com o WebShoppers (2015), com o crescimento do mercado móvel, vê-se a adoção do uso de telefones celulares e *tablets* pelos consumidores para consulta de informações de produtos, comparação de preços e compra usando esses dispositivos móveis.

2.2 O CONSUMIDOR ONLINE

O consumidor do século XXI tem mais acesso à informação e por isso consegue conhecer, questionar, comparar e divulgar sua opinião sobre produtos, serviços, marcas e empresas, exigindo assim, uma mudança na proposta de valor das empresas (TAPSCOTT; WILLIAMS, 2007).

Os atributos inerentes a um serviço ou produto são aspectos de grande importância na decisão de compra do consumidor, pois, na maioria das vezes, os consumidores baseiam suas compras nos atributos apresentados pelo objeto de compra (FRANCISCHELLI, 2009). Estes atributos podem ser especificados como as características, funções, componentes e preços do produto. Levando em consideração a afirmação de Francischelli, percebe-se que as particularidades de serviço ou produto são fatores decisivos na opção de compra do consumidor.

Dentre os vários fatores que influenciam a compra feita pelo consumidor *online*, Kotler e Armstrong (2003) citam que estes buscam produtos e serviços de qualidade, entrega dentro do prazo, comportamento corporativo ético, comunicações honestas e preços competitivos.

Em meio a diversos fatores presentes no processo de decisão de compra, o preço tem forte influência nas intenções e satisfações de compra do cliente (STEFFEN, 2009). Kotler e Keller (2006) afirmam que, apesar da qualidade dos produtos ser um fator de grande influência na decisão de compra de um consumidor, a grande maioria ainda prefere optar por produtos com o menor preço, sendo essa informação corroborada por Rojo (2003), que afirma que o preço é o atributo principal para alguns clientes na escolha da empresa onde fará suas compras.

Korgaonkar e Wolin (1999) também analisaram o uso da *web* sob a perspectiva de usos e gratificações e concluíram que um dos fatores para a realização de compras *online* é por motivações econômicas, onde o uso da internet é feito para pesquisar empresas ou ações e ainda para poupar dinheiro buscando ofertas.

Por meio de elementos como produto, preço, ponto e promoção, as atividades de *marketing*, incluindo o marketing digital, também são responsáveis por influenciar o consumidor no momento da compra (HONORATO, 2004).

2.3 MARKETING DIGITAL

Marketing digital, também chamado de publicidade *online*, *marketing web*, publicidade na internet, é a utilização da internet como uma ferramenta de *marketing* que envolve comunicação, publicidade, propaganda e todo o tipo de estratégias e conceitos da teoria do marketing (TORRES, 2009), que tem como objetivo utilizar o poder da internet para realizar um novo tipo de comunicação e relacionamento com os consumidores, em um momento em que o consumidor se tornou parte ativa do processo de comunicação (OGDEN; CRESCITELLI, 2007).

Para Kotler e Armstrong (2003), o objetivo do *marketing* digital é usar o poder da internet para realizar um novo tipo de comunicação e de relacionamento com os consumidores, o chamado *marketing* interativo.

Dentre as estratégias utilizadas no *marketing* digital, pode-se citar a publicidade em *sites*, o *e-mail marketing* e o *mobile marketing* (TORRES, 2009).

2.3.1 Estratégias de marketing digital

Erenberg (2003) cita algumas das modalidades de publicidade mais utilizadas em *sites* da internet, como, por exemplo, a janela *pop-up*, que se trata de um artifício publicitário que aparece na tela do computador do usuário, invadindo seu campo visual sem permissão ou aviso anterior, e o *banner*, que é um anúncio pequeno que ocupa parte da tela do computador, sendo que, muitas vezes, os internautas que utilizam equipamentos lentos ou possuem baixa velocidade de

conexão à rede o consideram uma espécie de SPAM, já que o carregamento de um *banner* torna a navegação mais lenta.

O *e-mail* é um canal praticamente sem limites para empresas e profissionais do marketing fazerem ofertas de seus serviços ou produtos chegarem aos consumidores – atuais ou potenciais (ERENBERG, 2003). Para o autor, tal prática é extremamente invasiva, já que ocupa tempo de conexão, espaço no disco rígido do computador e toma a atenção do internauta.

Existem três importantes conceitos a respeito do *e-mail marketing*, que devem ser levados em consideração na hora de aplicar esta técnica de *marketing*, sendo que o primeiro deles diz que o destinatário precisa concordar em receber as informações por *e-mail*; o segundo ponto diz que o conteúdo deve ser relevante para o receptor e o terceiro ponto diz que mensagem precisa ser personalizada, respeitando os interesses individuais de cada consumidor (FREITAS, 2009). Essas regras cumprem as exigências de não abusividade, genericamente estabelecidas pelo artigo 37, § 2º, do Código de Defesa do Consumidor (ERENBERG, 2003).

Os *e-mail´s* que são direcionados aos consumidores sem o consentimento dos mesmos, são conhecidos como SPAM, ou mensagens comerciais não solicitadas, que são mensagens que desmotivam os consumidores (ASSIS, 2003). Os SPAMS geralmente oferecem produtos e serviços, tanto na forma individual como na coletiva, ou seja, mala-direta, lista de notícias, correntes de dinheiro ou simpatias (RIBEIRO; SILVA; WAISBERG, 2001). Para Lorenzetti (2004), o envio de *e-mail´s* não desejados é uma flagrante ofensa à privacidade, que é invadida com tal prática.

Entre outras várias práticas condenáveis do *e-mail* marketing pode-se citar ainda a falta de relevância dos conteúdos, a emissão de grandes quantidades de *e-mail´s* e acompanhamento falho da comunicação instaurada entre empresas e consumidores por meio do *e-mail marketing* (DITOLVO, 2010).

Outra forma de *marketing* é o *mobile marketing*, que se caracteriza pela utilização de dispositivos móveis para a transmissão de mensagens publicitárias e comercialização de produtos e serviços (LAUDON; LAUDON, 2007).

Independentemente da forma de *marketing* digital utilizada, essa mídia tem a capacidade de gerar um banco de dados de consumo como nunca houve,

com base em dados da navegação e preferências dos usuários, o que levanta a questão da disponibilidade e privacidade das informações dos internautas (PINHEIRO, 2007).

2.4 DISPONIBILIDADE DE INFORMAÇÕES PESSOAIS

Pereira e Blum (2001) afirmam que a utilização e o abuso de informações pessoais sem autorização e, na maioria das vezes, sem o conhecimento do internauta, lesam tanto a moral, por conta de razões éticas, quanto o Direito, devido a violações legais.

O Marco Civil da Internet busca instituir princípios, garantias, direitos e deveres para usuários, provedores de serviço e de conteúdo e demais agentes envolvidos com o uso da internet (BRASIL, 2014). O Marco Civil é uma iniciativa legislativa que nasceu em 2009 para regular o uso da internet no Brasil, por meio da previsão de princípios, garantias, direitos e deveres de quem usa a rede, e da determinação de diretrizes para a atuação do Estado (BRASIL, 2014).

É comum que os usuários da internet recebam propagandas personalizadas, com ofertas de produtos e serviços selecionados pelos *sites*, considerados pela lei servidores de aplicação, de acordo com o seu histórico de navegação (OLIVEIRA, 2009). Por exemplo: se um usuário realiza uma pesquisa referente à uma televisão, com o intuito de comprá-la, essa informação de navegação na internet, que segundo a lei é um registro de acesso a aplicações, pode ser utilizada pelo *site* para atacar os usuários com propagandas de televisões divulgadas em outros acessos do internauta. De acordo com o art. 7º, incisos VII e X, do Marco Civil da Internet, a utilização desses dados pessoais só vai poder ocorrer se os internautas mostrarem consentimento livre, expresso e informado, o qual poderá ser revogado a qualquer momento pelo próprio usuário, que tem direito à exclusão definitiva de todos os dados pessoais que tiver fornecido ao *site* (OLIVEIRA, 2009).

Desta forma, os provedores de aplicação deverão possibilitar aos seus usuários, de modo claro, compreensível e sem ambiguidades que induzam a resposta, o direito de consentir ou não com a disponibilização de seus dados

pessoais, incluindo o histórico de navegação, para terceiros (OLIVEIRA, 2009). Para o autor, esta prática evitará diversas situações, como por exemplo, que os usuários sejam bombardeados com propagandas e publicidade inconvenientes de produtos e serviços, muitas vezes baseados em um histórico de navegação decorrente de um erro de percurso ou de uma utilização do computador por outra pessoa. Estes recursos, como propagandas e emails direcionados ao usuário, foram possíveis graças a chegada da Web 2.0, onde os sites passaram a ser dinâmicos e colaborativos.

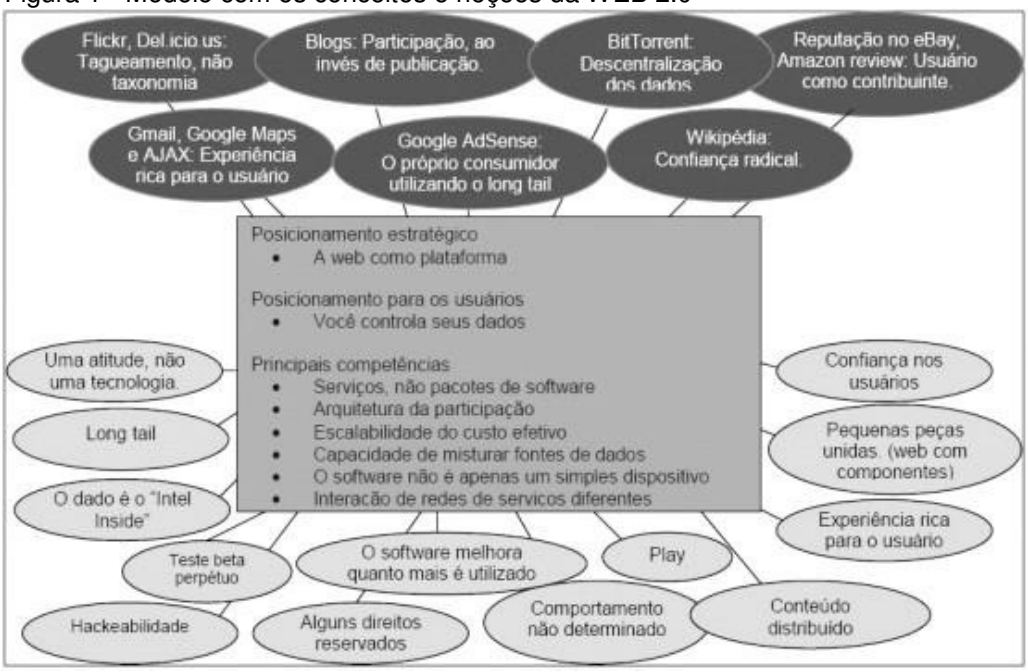
3 WEB 2.0

Em 2004, a empresa O'Reilly Media definiu o termo *Web* 2.0 para designar uma geração de comunidades e serviços *web*, onde o ambiente *online* fosse mais dinâmico e os usuários colaborassem para a criação e organização de conteúdo (O'REILLY, 2005, tradução nossa).

O'Reilly (2005, tradução nossa) diz que a *Web* 2.0 vem como uma mudança para uma internet como plataforma e um entendimento das regras para obter sucesso nesta plataforma, sendo que dentre essas regras, a mais importante é que se tenha a possibilidade de desenvolver aplicativos que aproveitem a inteligência coletiva. Podem ser citados cinco princípios fundamentais da *Web* 2.0, sendo esses princípios a simplicidade, o foco no conteúdo, a colaboração, o compartilhamento e a *web* como plataforma (BRANDT, 2006, tradução nossa).

O foco principal da *Web* 2.0 é o conteúdo, podendo ser apresentado em forma de texto, imagem, áudio, vídeo, sendo que este conteúdo não seja estático, mas sim um conteúdo participativo e democrático, onde há sempre uma realimentação de informação (MOREIRA; DIAS, 2009). A figura 1 representa o modelo com os conceitos e noções da *Web* 2.0 proposta por O´Reilley.

Figura 1 - Modelo com os conceitos e noções da WEB 2.0



Fonte: Adaptado de O`Reilley (2005)

A *Web* 2.0 permitiu que usuários comuns publicassem e consumissem informações de forma rápida e constante, pois a maioria dos *sites* e ferramentas destinados a este propósito é gratuita, além do fato de que foram criadas e disponibilizadas API's que permitem a comunicação com outros *sites* (MOREIRA; DIAS, 2009).

Esta nova *web* vem ganhando cada vez mais visibilidade, tendo como consequência o desenvolvimento e evolução das comunidades de cultura *web* e hospedagem de serviços, simplificando ainda mais as contribuições dos usuários para a internet e popularizando conceitos como *mashups* e tecnologias como *sites* ricos, do inglês *Rich Site Summary* (RSS), e Atom (VRIEZE et al, 2010, tradução nossa).

Dentre os diversos componentes presentes na *Web* 2.0, podem ser citados os *sites* de redes sociais, as *wikis*, *blogs* e os *mashups* (SHUEN, 2008, tradução nossa), que podem ser construídos por meio da extração de dados de diversas fontes de dados disponíveis na *web*.

3.1 EXTRAÇÃO DE DADOS

Existem diversas formas de extração e coleta de dados para recuperação de informações existentes na *web*, onde as mais comuns são as API´s, disponibilizadas por empresas e *sites*, para que outras pessoas possam ter acesso ao seu conteúdo, e técnicas de *web scraping*, que recupera informações que foram originalmente criadas apenas para o consumo do internauta.

3.1.1 API´s

API´s *online*, ou API´s *web*, são uma forma recente de distribuir conteúdo, funcionalidade e serviços entre diferentes sistemas da *WEB* 2.0 (MAXIMILIEN; RANABAHU; GOMADAM, 2008, tradução nossa).

As API´s são projetadas como o canal oficial de acesso aos dados e serviços do *web site*, fornecendo acesso aos dados do *site* quase como em um banco de dados local (YEE, 2008, tradução nossa).

Essas API´s normalmente são implementadas como bibliotecas em linguagens de *scripting* para *web*, como Javascript, ou por meio de serviços RESTful e é por meio dessas API´s que sistemas da *Web* 2.0 expõem funcionalidades que podem ser empregadas por outras aplicações da *web*.

A eclosão das API's permitiu consideravelmente o incremento da disponibilidade pública de dados na rede, possibilitando aos usuários combiná-los da forma que desejarem, seja por meio direto através de acesso ao código, ou ferramentas desenvolvidas e aperfeiçoadas dia a dia para este propósito, a fim de expandir a massa *mashup* presente na rede por meio da criação de *mashups* de modo cada vez mais simples, criativos e interessantes (SOUZA, 2009).

A API do Buscapé é uma solução oferecida pela empresa Buscapé para disponibilizar a sua base de produtos e seus serviços publicitários para uso em aplicações, *sites*, lojas, *banners* e *plugins* (BUSCAPÉ, 2014).

Todos os serviços oferecidos pela API utilizam a tecnologia de *web services* de transferência de estado representativo, do inglês *Representational State Transfer* (REST), no tratamento de pedidos e o formato de resposta padrão é por linguagem de marcação extensível, do inglês *eXtensible Markup Language* (XML), porém há a opção de usar a notação de objetos Javascript, do inglês *JavaScript Object Notation* (JSON) (BUSCAPÉ, 2014). Segundo informações contidas no *site*, para realizar a integração com a API do Buscapé, é necessário se registar como um desenvolvedor, obtendo assim um identificador para a aplicação a ser desenvolvida, que deve ser usado em todas as solicitações, sendo que as aplicações que fazem requisição de serviços para a API do Buscapé têm diferentes níveis de acesso à informação em seus retornos.

Utilizando esta API, é possível criar novos serviços com os dados disponibilizados pela empresa Buscapé, recurso este que pode ser caracterizado como um *mashup*.

3.1.2 Mashups

Um *web mashup* é uma página *web* que combina informações e serviços de múltiplas fontes disponíveis na *web*, de modo a criar novos serviços

(MURUGESAN, 2007, tradução nossa). Um *mashup* é uma derivação do conceito de aplicação composta, que é uma aplicação que consome informações disponibilizadas por meio de *web services* (SAMPAIO, 2007), e se popularizou devido a *Web* 2.0, pois um de seus objetivos é reutilizar os dados, *web services* e micro aplicativos para criar aplicações híbridas (YEE, 2008, tradução nossa).

Para Nordström (2010, tradução nossa), *mashup* é um termo usado para designar aplicações *web* híbridas, que combinam serviços já existentes para fornecer um novo serviço que nenhum dos serviços originais prestava isoladamente, podendo também adicionar novos conteúdos para serem combinados com os serviços existentes. Os *mashups* podem ser criados para uso comercial, *data mining* ou utilização pelo consumidor e são uma ótima maneira de criar novos serviços sem ter que começar aplicações do zero (NORDSTRÖM, 2010, tradução nossa). Vrieze et al (2010, tradução nossa) dizem que *mashups* devem ter uma interface amigável e devem oferecer a possibilidade de reutilização de recursos e adaptabilidade.

Segundo Liu et al (2007, tradução nossa), os *mashups* possuem importantes características, descritas a seguir.

a) são reutilizáveis: cada bloco de construção *mashup* tem seu próprio contexto e lógica de negócios, sendo que, geralmente, são as API´s de *mashups* que contém a maior parte da lógica comercial, tornando os serviços *mashup* mais reutilizáveis;
b) baseado na *web*: isso significa que *mashups* utilizam Javascript, *web services* e *feeds* baseados diretamente HTTP, usando JSON e XML para recuperação de dados;
c) aplicações leves: um *mashup* é a integração de múltiplas fontes de dados e conteúdo em um único local e, pelo fato de *mashups* utilizarem dados e serviços de *sites* e aplicativos da *web*, eles são de leve e fácil implementação e são construídos com uma quantidade mínima de código, fazendo com que haja redução dos custos de desenvolvimento e maior satisfação do usuário.

Tecnologias emergentes como *web services*, bibliotecas de *widgets* e ferramentas específicas para modelar *mashups* têm simplificado significativamente o acesso e a reutilização de recursos, facilitando, especialmente, o desenvolvimento

das chamadas aplicações situacionais, ou seja, aplicações que tem um propósito específico, onde o desenvolvedor é também o usuário final e são destinadas a serem utilizadas dentro de um horizonte temporal limitado, sendo que essas aplicações normalmente visam atender a necessidade de uma consulta sobre um limitado, mas heterogêneo, espaço de dados (CAPPIELLO et al, 2010, tradução nossa).

Os *mashups* tem a capacidade de combinar dados/informações de diversas fontes para criar e produzir aplicações que agreguem valor aos usuários, já que o termo *mashup* implica fácil e rápida integração, possibilitada pelo acesso a API´s gratuitas, *widgets* e fontes de dados, para produzir resultados que vão além dos recursos inicialmente disponibilizados (VRIEZE et al, 2010, tradução nossa). Os mesmos autores afirmam que um *mashup* é o resultado do rápido desenvolvimento em pequena escala e, devido à sua capacidade ágil, um *mashup* reage muito melhor à alterações provenientes de mudanças de requisitos, mudanças de processos de recursos relacionados, etc.

Mashups devem possuir um modelo de componentes bem definido, que encapsula os dados de várias fontes e manipula os recursos existentes na *web*, por meio de um padrão de serviços como, por exemplo REST, Atom / RSS, entre outros (LIU et al, 2007, tradução nossa). Os autores dizem que um *mashup* é composto por alguns componentes, que consistem em componentes de interface e um conjunto de serviços de *back-end*, local ou remoto, de ligação para o componentes de interface do usuário. Os mesmos classificaram o modelo de componentes em três elementos:

a) componentes de interface de usuário: representações de componentes de interface, como um conjunto de *widgets* no navegador, por exemplo, uma janela, um botão, uma lista *drop-down*, entre outros;
b) componentes de serviços: componentes de serviço representam a interface de manipulação de dados, que pode ser acessado por protocolos simples de acesso a objetos, do *inglês Simple Object Access Protocol* (SOAP), e REST ou pode ser uma interface de um database, recuperando e armazenando dados em um banco de dados local ou remoto;

c) componentes de ação: componentes de ação agem como o conectores entre os componentes de interface do usuário e componentes de serviço. Eles definem, por exemplo, uma ação impulsionada por eventos, como *onClick* ou *onMouseOver*.

Mashups podem ajudar a resolver desafios de negócios e de tecnologia de informação, especialmente para pequenas e médias empresas virtuais que têm menos recursos para criar soluções tradicionais e que buscam uma maior agilidade, maior configurabilidade, aplicações multi plataforma e que tem a necessidade de responder mais rapidamente a um aumento no ritmo dos negócios (VRIEZE et al, 2010, tradução nossa). Zeng e Zhang (2010, tradução nossa) afirmam que a combinação de dados ou serviços disponíveis na *web* e recursos de redes móveis permite que os usuários de dispositivos móveis possam desfrutar de uma ótima experiência quando se trata da busca de informações no *e-commerce*.

3.1.2.1 Tipos de mashups

Os *mashups* podem ser classificados em três tipos: *mashups* de dados, *mashups* de consumo, que é a categoria mais comum de *mashups*, e *mashups* de negócios, que vem se tornando muito populares (VIEDMA, 2010, tradução nossa).

Mashups de dados combinam múltiplas fontes de dados, que devem ser de tipos similares, em uma única representação, que irá fornecer um novo serviço que não estava disponível originalmente (VIEDMA, 2010, tradução nossa). Esses *mashups* integram dados em uma lógica definida pelo usuário para atribuir ou derivar deles novos significados, podendo servir também para codificar esses dados em outros formatos, mais adequados ao consumo humano ou de máquina, dependendo da intenção do seu criador (BOUILLET et al, 2009, tradução nossa).

Já os *mashups* de consumo são projetados para o público em geral e, normalmente, combinam múltiplas fontes de diferentes tipos de dados em uma representação visual, provando ser um meio muito eficaz de personalização de dados, de acordo com as necessidades do cliente (VIEDMA, 2010, tradução nossa). Estes *mashups* têm como foco a visualização de conteúdo, onde um usuário pode

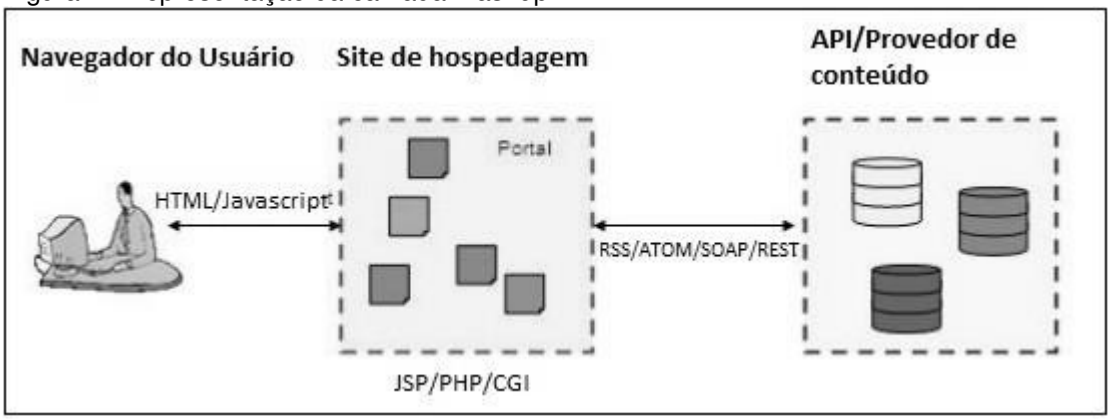
utilizar diversas API´s *online* para a criação de conteúdos *web* (BOUILLET et al, 2009, tradução nossa).

O terceiro tipo de *mashup* é o de negócios, que são semelhantes aos *mashups* de consumo, porém, com o objetivo de resolver um problema de negócio. Os *mashups* desta categoria diferem dos de consumo também em outros aspectos, tais como o nível de segurança necessário, o nível de sofisticação, entre outros, e geralmente combinam informações internas e serviços de uma corporação com recursos externos para criar uma aplicação *web* visualmente rica. Muitas empresas estão adotando este tipo de *mashup* para poder acompanhar o ritmo acelerado das mudanças nas empresas ou quando eles não têm os recursos ou competências necessárias para desenvolver determinados serviços (VIEDMA, 2010, tradução nossa).

3.1.2.2 Camadas do mashup

Segundo Merril (2009, tradução nossa), são três as camadas que compõem um *mashup*, que estão lógica e fisicamente separadas: o provedor de conteúdo, o *site mashup* e o navegador do cliente, sendo que estas camadas estão representadas na figura 2.

Figura 2 - Representação da camada mashup



Fonte: Adaptado de Liu et al (2007)

O *site mashup* é o lugar onde o *mashup* está hospedado, mas não é necessariamente onde ele é executado (MERRIL, 2009, tradução nossa). O mesmo autor afirma que *mashups* podem ser implementados de forma semelhante a

aplicações *web* tradicionais, utilizando tecnologias de geração de conteúdo dinâmico como *servlets* Java, entre outras linguagens *web*, ou o conteúdo do *mashup* pode ser gerado diretamente no navegador do cliente por meios de *scripts*, sendo que esta lógica de geração de conteúdo no cliente é muitas vezes a combinação de código diretamente incluído na página de *mashup web*, assim como bibliotecas ou *applets* de API´s referenciados por essas páginas da *web*.

Os benefícios da maceração do lado do cliente incluem menos sobrecarga em nome do servidor *mashup*, onde os dados podem ser recuperados diretamente do provedor de conteúdo, e uma experiência de usuário mais uniforme, onde as páginas podem pedir autorização para partes de seu conteúdo sem a necessidade de atualizar a página inteira (MERRIL, 2009, tradução nossa). Muitas vezes, os *mashups* utilizam uma combinação de lógicas do lado servidor com o lado do cliente para alcançar sua agregação de dados, pois realizar consultas complexas em dados provenientes de múltiplas fontes requer processamento, que seria impossível de realizar no navegador *web* do cliente (MERRIL, 2009, tradução nossa).

O navegador do cliente é o lugar onde a aplicação é criada graficamente e a interação do usuário ocorre (MERRIL, 2009, tradução nossa).

O provedor de conteúdo é a fonte que disponibiliza o conteúdo utilizado pelo *mashup* (MERRIL, 2009, tradução nossa). O autor diz que, para facilitar a recuperação de dados, provedores frequentemente expõem o seu conteúdo por meio de protocolos *web* tais como REST, *web services* e RSS / Atom. *Mashups* que extraem o conteúdo de *sites* como Wikipedia, Guia TV, e praticamente todos os *sites* do governo e de domínio público fazem isto por meio de uma técnica conhecida como *screen scraping*, que é o processo pelo qual uma ferramenta tenta extrair informações do provedor de conteúdo, tentando analisar as páginas da *web* do provedor, que foram originalmente destinadas ao consumo humano (MERRIL, 2009, tradução nossa).

O método mais comum para se obter dados a partir de um *web site* é por meio de uma interface de programação de aplicação (API), que é projetada especificamente para facilitar a comunicação entre os programas, porém, existem outras alternativas para recuperar informações de *web sites* como, por exemplo, o

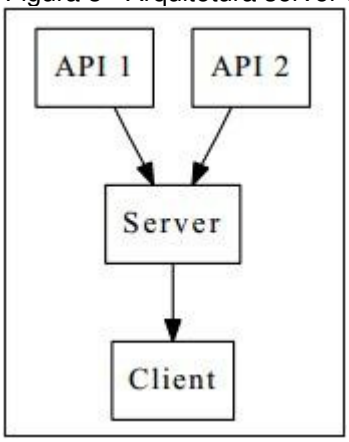
screen scraping (YEE, 2008, tradução nossa), programação HTTP, análise de modelos de objetos de documento, do inglês *Document Object Model* (DOM) e *parsers* de linguagem de marcação de hipertexto, do inglês *HyperText Markup Language* (HTML) (VARGIU; URRU, 2012, tradução nossa).

3.1.2.3 Arquitetura do mashup

De um ponto de vista arquitetônico, existem dois estilos de *mashups*: os baseados nos clientes, do inglês *client-based*, e os baseados em servidores, do inglês *server-based*, sendo que é possível ter ainda um terceiro estilo, que é a arquitetura móvel, onde ambos os estilos, *client-based* e *server-based*, são combinados aproveitando as vantagens de cada um (VIEDMA, 2010, tradução nossa).

Como o nome indica, *mashups* baseados em servidor realizam a integração de dados e serviços no servidor, sendo que este medeia todas as mensagens entre o cliente e o *web service*, agindo como um *proxy*, sendo também responsável por recolher, transformar e combinar os dados de serviços de terceiros antes de enviá-lo para o cliente (ORT; BRYDON; BASLER, 2007, tradução nossa), podendo essa arquitetura ser visualizada na figura 3.

Figura 3 - Arquitetura server-based



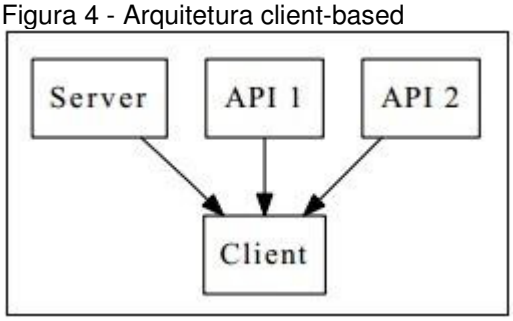
Fonte: Viedma (2010)

Como apontado pelos autores, existem várias razões para usar esse tipo de arquitetura. Em primeiro lugar, o número de bibliotecas e protocolos disponíveis é

consideravelmente extenso e o servidor pode simplificar muitas das tarefas necessárias para a construção de um *mashup*. Além de poder armazenar em *cache* informações importantes para a aplicação, o servidor também pode pré processar todas as informações coletadas, reduzindo o tamanho dos dados, mudando o formato ou combinando-o com os outros dados, sendo também mais fácil lidar com os requisitos de segurança. Também é possível realizar chamadas simultâneas e assíncronas para vários *web services*, algo que nem sempre é possível a partir de um navegador, devendo-se também considerar a potência computacional limitada disponível na maioria dos dispositivos móveis e a possibilidade de transferir toda essa carga para o lado do servidor.

Em contraste com uma arquitetura baseada em servidor, em uma arquitetura baseada em cliente a integração de serviços e conteúdos é feita no lado do cliente, onde o navegador do cliente é o encarregado de reunir o conteúdo, transformando-o para o formato correto antes de mostrá-lo na tela, como mostra a figura 4.

Figura 4 - Arquitetura client-based



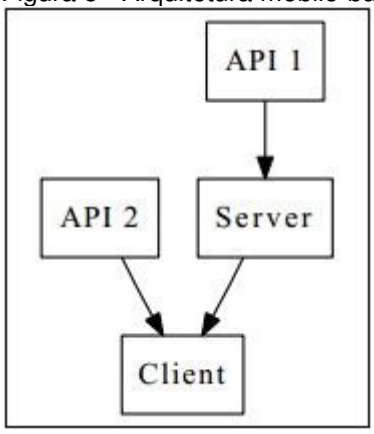
Fonte: Viedma (2010)

Existem várias razões para usar o estilo baseado no cliente (ORT; BRYDON; BASLER, 2007, tradução nossa), pois, segundo os autores, nesta arquitetura a implementação dos *mashups* pode se tornar mais simples do que na arquitetura baseada em servidor. Atualmente, muitas empresas estão fornecendo API´s Javascript, que estão prontas para serem usadas no código do aplicativo da *web*, não havendo necessidade de um componente do lado do servidor (VIEDMA, 2010, tradução nossa). No momento em que o serviço é carregado no cliente, o servidor torna-se secundário uma vez que todas as chamadas são feitas diretamente entre o cliente e o os diferentes serviços da *web*, evitando o gargalo apresentado em

uma arquitetura baseada em servidor (ORT; BRYDON; BASLER, 2007, tradução nossa).

O terceiro tipo de arquitetura é o de *mashups* móveis, que aproveita as vantagens das outras arquiteturas (VIEDMA, 2010, tradução nossa), conforme é possível visualizar na figura 5.

Figura 5 - Arquitetura mobile-based



Fonte: Viedma (2010)

O autor cita como exemplo deste tipo de arquitetura o *site* HousingMaps. Este *mashup* carrega os mapas, usando a biblioteca Javascript fornecida pelo Google Maps, diretamente no navegador do cliente, evitando sobrecarregar o servidor com informações desnecessárias. Por outro lado, a informação do Craiglist não é disponibilizada por uma API, sendo que é necessário usar uma técnica chamada *web scrapping*, que recebe a informação diretamente do código HTML do *site*, requerendo o carregamento de todo o *site* Craiglist.org para extrair a informação desejada. Este processo pode ser consideravelmente lento para ser realizado em um dispositivo móvel, devido ao poder computacional restrito, à largura de banda limitada, e ao aumento significativo do consumo da bateria, optando-se por realizar este procedimento no servidor (VIEDMA, 2010, tradução nossa).

4 RECURSOS TECNOLÓGICOS

A construção de um *mashup* integra elementos heterogêneos disponíveis na *web*, tais como *feeds* RSS / Atom, *web services*, conteúdos de *sites* de terceiros, ou *widgets* como o Google Maps (CAPPIELLO et al, 2010, tradução nossa). A API do Buscapé, que é a API que será utilizada na construção do protótipo, disponibiliza seus dados por meio de *web services* REST.

4.2 WEB SERVICES

Com os avanços da internet e dos protocolos de comunicação baseados em XML, surgiram os *web services*, tendo como principal objetivo integrar sistemas heterogêneos (GOMES, 2010).

Um *web service* pode ser descrito com uma parte da lógica de negócio alocada em algum lugar na internet, podendo ser acessada por meio de diversos padrões como, por exemplo, HTTP e pelo protocolo simples de transferência de correio, do inglês *Simple Mail Transfer Protocol* (SMTP) (CHAPPELL; JEWELL, 2002, tradução nossa). Segundo a definição do W3C, os *web services* são aplicações com interfaces baseadas em XML e que descrevem uma coleção de operações acessíveis por meio da rede, independente da tecnologia usada na implementação do serviço (W3C, 2004).

Atualmente, é possível observar que várias empresas utilizam os *web services* para automatizar e agilizar seus processos de negócio, como, por exemplo, a criação e disponibilização de portais que combinam ofertas de vários produtos e serviços de empresas distintas, apresentando estes dados com uma aparência unificada para o consumidor final (CHAPPELL; JEWELL, 2002, tradução nossa).

Chappell e Jewell (2002, tradução nossa) apontam que um *web service* deve ter baixo acoplamento, onde um consumidor de um recurso não está vinculado a esse serviço *web* diretamente, sendo que a interface do *web service* pode mudar ao longo do tempo, sem comprometer a capacidade do cliente de interagir com o serviço. Essa adoção de uma arquitetura de baixo acoplamento tende a tornar os sistemas de software mais flexíveis e permitir uma integração simples entre sistemas

diferentes (CHAPPELL; JEWELL, 2002, tradução nossa). Os mesmos autores afirmam que os *web services* devem ter suporte a chamada a procedimentos remotos, do inglês *Remote Procedure Calls* (RPC´s), permitindo que os clientes invoquem remotamente procedimentos, funções e métodos usando um protocolo baseado em XML.

Complementando os autores anteriores, Kalin (2013, tradução nossa) afirma que existem diversas características que distinguem os *web services* de outros tipos de sistema, sendo que algumas são:

a) infra estrutura aberta: *web services* são implementados usando protocolos independentes de fornecedor, tais como o protocolo de transferência de hipertexto, do inglês *hypertext transfer protocol* (HTTP) e XML;
b) linguagens transparentes: *web services* e seus clientes podem interoperar, mesmo que escritos em linguagens de programação distintas;
c) design modular: *web services* são destinados a ser um projeto modular, para que novos serviços possam ser gerados por meio da integração de camadas de serviços já existentes.

Com o desenvolvimento da tecnologia *web*, cada vez mais empresas e organizações estão encapsulando seus produtos por meio de *web services* publicados na internet, fazendo com que este paradigma de desenvolvimento permita que aplicações *web* reutilizem os recursos e funcionalidades já anteriormente implementados e disponibilizados na *web* (LI; SHI; ZU, 2014, tradução nossa).

Todos os *web services* usam HTTP, porém de maneiras diferentes, e todos os *web services* são orientados a mensagens, porque o próprio HTTP é orientado a mensagens (RICHARDSON; RUBY, 2007, tradução nossa).

Um *web service* é, portanto, uma aplicação distribuída, cujos componentes podem ser implementados e executados em dispositivos distintos e consumidos em computadores pessoais, *handhelds*, entre outros dispositivos, e podem ser divididos basicamente em dois grupos, os baseados em SOAP e os de estilo REST (KALIN, 2013, tradução nossa).

4.2.1 SOAP

É um protocolo projetado para invocar aplicações remotas por meio de RPC ou trocas de mensagens, em um ambiente independente de plataforma e linguagem de programação (CHAPPELL; JEWELL, 2002, tradução nossa).

Segundo a especificação contida em Fielding e Reschke (2014), o protocolo simples de acesso a objetos, do inglês *Simple Object Access Protocol* (SOAP) é um protocolo leve cujo objetivo é realizar o intercâmbio de informações estruturadas em ambientes distribuídos e descentralizados, de modo que utiliza tecnologias XML para definir um *framework* de mensagens extensíveis provendo a construção de mensagens que podem ser trocados através de uma variedade de protocolos.

A tecnologia SOAP fornece uma estrutura de empacotamento para o transporte de documentos XML por meio de diversas tecnologias padrão da internet, incluindo o protocolo SMTP, HTTP e o protocolo de transferência de arquivos, do inglês *File Transfer Protocol* (FTP) (CHAPPELL; JEWELL, 2002, tradução nossa).

As solicitações e respostas dos *web services* são transmitidas como mensagens, por meio do protocolo HTTP, permitindo uma troca completamente interoperável entre clientes e serviços *web*, todos rodando em diferentes plataformas e em vários locais na internet. HTTP é um padrão de solicitação e resposta familiar para envio de mensagens pela internet e SOAP é um protocolo baseado em XML que segue o modelo de solicitação e resposta HTTP (JENDROCK et al, 2011).

Embora SOAP e HTTP tenham relacionamento, muitos aspectos de SOAP estão em desacordo com a arquitetura da *web*. Ao invés de focar em identificadores uniformes de recursos, do inglês *Uniform Resource Identifier* (URI), (que é a forma da *Web*), o SOAP se concentra em ações, sendo que os serviços baseados em SOAP quase sempre têm apenas uma URI e muitas ações diferentes (FLANDERS, 2009, tradução nossa). O mesmo autor comenta que muitos adeptos da tecnologia SOAP apontam que, pelo fato de o SOAP não seguir a arquitetura da *web*, a mesma pode ser utilizada por vários protocolos diferentes e não apenas pelo protocolo HTTP, trazendo a desvantagem de não aproveitar efetivamente todos os recursos fornecidos pelo HTTP.

4.2.2 REST

Representational State Transfer (REST) é um estilo arquitetural para sistemas distribuídos que prima pela generalização de interfaces, escalabilidade da integração entre os componentes e instalação independente dos mesmos (FIELDING, 2000, tradução nossa).

Em sua tese, Fielding (2000, tradução nossa) diz que a principal característica que distingue o estilo arquitetônico REST de outros estilos baseados em rede é sua ênfase em uma interface uniforme entre os componentes. Ele diz também que a interface REST é projetada para ser eficiente para uma grande transferência de dados hipermídia e é definida por quatro limitações de interface: identificação de recursos, manipulação de recursos através de representações, mensagens auto-descritivas e hipermídia como o motor do estado do aplicativo.

Burke (2009, tradução nossa) cita alguns dos princípios arquiteturais da tecnologia REST, descritos na tese de PhD de Roy Fielding, sendo eles:

a) endereçabilidade: cada objeto e recurso do sistema pode ser acessado por meio de um identificador exclusivo, que, em REST, é gerido por meio do uso de URI´s;
b) restrição de interface: deve se utilizar apenas os métodos HTTP para manipular os serviços *web*. Isto possibilita que outras pessoas possam usar o serviço sem quaisquer requisitos adicionais além de atender os requisitos do serviço;
c) orientada a representação: cada serviço é endereçável através de uma URI;
d) comunicação *statelessly*: não há dados de sessão do cliente armazenados no servidor.

Portanto REST ganha a separação de interesses do estilo cliente-servidor, sem o problema de escalabilidade do servidor, permite a ocultação de informações por meio de uma interface genérica para permitir o encapsulamento e evolução dos serviços, e prevê um conjunto diversificado de funcionalidades por meio de recursos motores de *download* (FIELDING, 2000, tradução nossa).

No estilo arquitetural da tecnologia REST, os dados e as funcionalidades são considerados recursos e são acessados por meio de uma URI, ou seja, *links* na *web* (KAMALELDIN; DUMINDDA, 2012, tradução nossa). Fielding (2000) resume os elementos de dados presentes na arquitetura REST com o quadro 1.

Quadro 1 - Resumo dos elementos de REST

Elemento	**Exemplo web**
recurso	alvo conceitual pretendido
identificador do recurso	URL, URI
representação	documento HTML, imagem JPEG
metadado da representação	arquivos de mídia, hora da última modificação
metadado do recurso	link da fonte de dados
dados de controle	controle de cache

Fonte: Adaptado de Fielding (2000)

Os recursos que serão utilizados pelas aplicações são identificados nas requisições realizadas ao *web service* (KALALI; MEHTA, 2013, tradução nossa), sendo que, como resposta, o servidor retorna para o cliente um documento que é uma representação do recurso (ALLAMARAJU, 2010, tradução nossa).

Estes recursos são manipulados com a utilização de um conjunto composto por quatro comandos: PUT, GET, POST e DELETE, que significam, respectivamente, criação, leitura, atualização, exclusão de operações (JENDROCK et al, 2011).

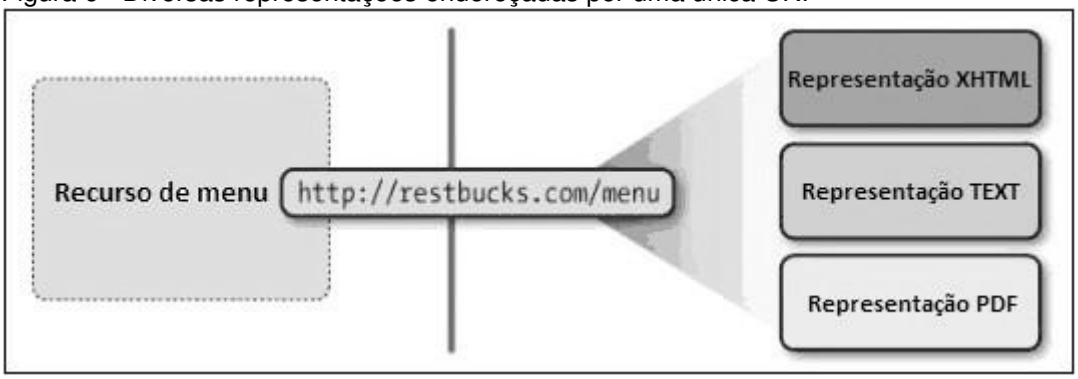
Em REST, um cliente faz a requisição de um recurso, que está hospedado em um servidor, sendo que este retorna uma resposta na forma de uma representação (ABEYSINGHE, 2009, tradução nossa). Conectores REST darão uma interface genérica para acessar e manipular o valor de um recurso, independentemente de como a função de pertinência é definida (FIELDING, 2000, tradução nossa).

4.2.2.1 Identificador Uniforme de Recursos (URI)

URI é o nome e endereço de um recurso (RICHARDSON; RUBY, 2007, tradução nossa). Na arquitetura REST, um recurso é obrigatoriamente referenciado por, no mínimo, uma URI, que identifica e localiza este recurso, sendo que o mesmo pode ser referenciado por um número ilimitado de URI´s (O'REILLY, 2007).

As URI´s tem o papel de relacionar as representações com os seus recursos na *web* (WEBBER; PARASTATIDIS; ROBINSON, 2010, tradução nossa), sendo que ao projetar um serviço REST, devem ser identificados os recursos que o serviço irá expor e usar, para, posteriormente, mapeá-los para URI´s (FLANDERS, 2009, tradução nossa). Os recursos devem ter pelo menos um identificador para ser endereçável na *web*, e cada identificador deve estar associado com uma ou mais representações, como XML, JSON, entre outros (WEBBER; PARASTATIDIS; ROBINSON, 2010, tradução nossa), como pode ser visualizado na figura 6.

Figura 6 - Diversas representações endereçadas por uma única URI



Fonte: Adaptado de Webber, Parastatidis e Robinson (2010)

A URI tem a forma <esquema>: <regime específico da estrutura>, onde o esquema define como o resto do identificador deve ser interpretado, ou seja, a parte de uma URI HTTP como <http://example.org/reports/book.tar> diz que o resto da URI deve ser interpretado de acordo com o esquema de HTTP.

4.2.3 Recursos e representações

Um recurso é qualquer coisa que é importante o suficiente para ser referenciada como uma coisa em si, que pode ser armazenado em um computador,

como por exemplo documentos eletrônicos, linhas de um banco de dados, ou o resultado da execução de um algoritmo, tendo como única restrição a utilização de um localizador padrão de recursos, do inglês *Uniform Resource Locator* (URL), para referenciar o recurso (RICHARDSON; AMUNDSEN; RUBY, 2013, tradução nossa).

Flanders (2009, tradução nossa) diz que um recurso é um mapeamento conceitual a uma entidade ou conjunto de entidades em particular em que se deseja que outras aplicações possam interagir.

Recursos são os blocos fundamentais de sistemas baseados em *web*, na medida que a *web* é muitas vezes referida como sendo orientada a recursos, onde o alvo de uma solicitação HTTP é o recurso, sendo que este protocolo define a interface que pode ser usada para interagir com os recursos (FIELDING; RESCHKE, 2014, tradução nossa).

Os recursos não precisam ser arquivos estáticos, podendo ser também programas de software que geram conteúdo sob demanda com base na identidade do requisitante (GOURLEY et al, 2002, tradução nossa).

Considerando-se que o recurso pode ser qualquer coisa, e que a interface uniforme fornecida pelo HTTP é semelhante a uma janela por onde se pode observar e se comunicar por meio de mensagens, é necessário que haja uma abstração para representar o estado atual ou desejado do objeto da comunicação, sendo que esta abstração é chamada de representação (FIELDING; RESCHKE, 2014, tradução nossa). Uma representação é a informação que reflete o passado, o presente ou o estado desejado de um determinado recurso, num formato que pode ser facilmente transmitido via protocolo, e que consiste de um conjunto de metadados e uma potencialmente ilimitada representação de dados (FIELDING; RESCHKE, 2014, tradução nossa).

Representações são encapsulamentos codificados das informações (estado, dados ou marcação) do recurso, podendo um recurso ter uma ou mais representações (ALLAMARAJU, 2010, tradução nossa), Atom, XML, JSON, texto simples, valores separados por vírgulas, MPEG-1/2 Audio Layer 3 (MP3) ou imagens no formato criado pelo grupo *Joint Photographics Experts Group* (JPEG) (WEBBER; PARASTATIDIS; ROBINSON, 2010, tradução nossa).

O acesso a um recurso é sempre mediado por meio de suas representações, sendo que esta separação entre um recurso e sua representação promove o acoplamento fraco entre os sistemas de *back-end* e aplicações de consumo, contribuindo com a escalabilidade da aplicação (WEBBER; PARASTATIDIS; ROBINSON, 2010, tradução nossa).

4.2.3.1 XML

XML é uma linguagem de marcação baseada em texto, utilizada para a troca de dados entre aplicações distintas, sendo baseada em regras simples e independentes de plataforma para a representação de estruturas de informação textual (VOHRA; VOHRA, 2006, tradução nossa).

O formato XML segue alguns conceitos, como o da estrutura, também chamado de hierarquia e o conceito da semântica, que significa aplicar nomes às coisas para que elas se tornem significativas para as pessoas (HOSKINS, 2013, tradução nossa). É um formato bastante detalhado, possibilitando informar atributos, dados de caracteres e *tags* aninhadas, sendo uma maneira poderosa e descritiva de representação de dados (MITCHELL, 2013, tradução nossa).

Essas características tornam o formato XML um pouco mais pesado do que outros formatos, o que não se torna uma grande preocupação quando se trabalha com máquinas potentes e conexões de rede rápidas, tornando o XML a escolha popular quando se trata de troca de dados entre computadores ou servidores (MITCHELL, 2013, tradução nossa).

Tidwell, Snell e Kulchenko (2001, tradução nossa) destacam que existem várias regras que é preciso conhecer para criar documentos XML:

a) elemento *root*: o primeiro elemento no documento XML é chamado de elemento do documento ou o elemento raiz e deve abranger todo o documento. Não pode haver mais de um elemento raiz no documento e se isso acontecer, o analisador XML lança uma exceção;
b) todos os elementos devem ser aninhados: se um elemento for inicializado, o mesmo deve ser finalizado;

c) todos os atributos devem possuir valores: o formato XML não permite que atributos sejam informados sem possuírem valores;
d) as *tags* XML são *case sensitive*: elas diferenciam letras maiúsculas de minúsculas.

Um documento XML deve conter informações separadas por *tags*, que descrevem o que são todos os itens do documento, bem como a relação entre elas (TIDWELL; SNELL; KULCHENKO, 2001, tradução nossa).

4.2.3.2 JSON

JSON é um formato bastante simples, leve e que representa dados estruturados (MITCHELL, 2013, tradução nossa), podendo representá-los utilizando objetos e *arrays* literais (STEFANOV, 2008, tradução nossa). É um formato de intercâmbio de dados baseado em um subconjunto da linguagem de programação JavaScript, de fácil leitura e escrita para os seres humanos e de fácil análise e geração para as máquinas, sendo completamente independente de linguagem, porém, utilizando convenções que são familiares aos programadores das linguagens da família C, incluindo C, C ++, C#, Java, JavaScript, Perl, Python, e muitos outros (LOTT, 2014, tradução nossa).

O JSON foi introduzido como um substituto para o formato XML, quando se percebeu que a extensibilidade e detalhamento do formato XML não eram necessários em determinadas trocas de informação (MEHTA; KALALI, 2013, tradução nossa).

A simplicidade é o ponto forte do JSON, pois não é preciso muito espaço de armazenamento em comparação com o XML, e não é um formato demasiado grande para transferir dados, se tornando uma tecnologia barata em termos de processador, que o torna ideal para dispositivos menos potentes, como telefones, por exemplo (MITCHELL, 2013, tradução nossa).

Neste formato, os objetos são informados entre chaves "{}" e contêm pares de chave/valor, onde o valor pode ser representado por booleanos (verdadeiro ou falso), valores numéricos ou matrizes de tipos simples; pares de chave e valor

são separados por caracteres e delimitados por vírgulas (BURKE, 2010, tradução nossa).

4.3 SISTEMAS OPERACIONAIS MOBILE

Para a utilização de todo o potencial proporcionado pelos dispositivos móveis, existe a necessidade de sistemas operacionais, que neste caso são chamados de plataformas, já que com raras exceções podem ser substituídos por outras plataformas e são programados com adaptações específicas para cada modelo de dispositivo (MORIMOTO, 2009).

Banga e Weinhold (2014, tradução nossa) afirmam que dentre os vários atributos comumente utilizados para categorizar os dispositivos móveis, pode-se citar o sistema operacional que o mesmo executa. O autor diz que a maioria dos dispositivos móveis atuais são baseados em sistemas operacionais pós computadores pessoais, que se diferenciam das interfaces dos computadores *desktop* e dos *laptops*.

Um sistema operacional móvel é um conjunto de programas com a função de gerenciar os recursos de hardware e software para dispositivos móveis, além de fornecer uma interface ao usuário final (SILBERSCHATZ et al, 2004). Todos os aplicativos e serviços que estarão rodando em dispositivos móveis necessitam possuir uma plataforma *mobile*, que tem como principal objetivo proporcionar ao usuário o acesso aos recursos de seu dispositivo móvel (FLING, 2009, tradução nossa).

Como explica Martins (2009), há vários tipos de sistema operacional para dispositivos móveis, tais como o iOS, da Apple, Microsoft Windows Mobile, Nokia Symbian e Android, sendo que os principais sistemas operacionais serão melhor descritos nos tópicos seguintes.

4.3.1 iOs

iOS é o sistema operacional que roda nos dispositivos iPad, iPhone, e iPod Touch (APPLE, 2014, tradução nossa).

O iOS *Software Development Kit* (SDK) contém as ferramentas e interfaces necessárias para desenvolver, instalar, executar e testar aplicativos em dispositivos que utilizam iOS, sendo que estes aplicativos são construídos usando linguagens SWIFT (APPLE, 2014, tradução nossa).

No nível mais alto, iOS atua como um intermediário entre o hardware e os aplicativos que criados, sendo que os aplicativos não conversam diretamente com o hardware, se comunicando por meio de um conjunto de interfaces de sistemas bem definidos, interfaces essas que tornam mais fácil a escrita de aplicativos que funcionam consistentemente em dispositivos com diferentes capacidades de hardware (APPLE, 2014, tradução nossa).

4.3.2 Windows Phone

Windows Phone é um sistema operacional móvel, desenvolvido pela Microsoft que tem seu foco no mercado consumidor.

O Windows Phone oferece capacidade de tela *multitouch*, uma interface de usuário que implementa o *design* chamado Metro, serviços de redes sociais, como o Facebook, e suporte para serviços populares de *e-mail*, como Yahoo, Hotmail e Gmail (LEE; CHUVYROV, 2012, tradução nossa).

Conforme descrito por (LEE; CHUVYROV, 2012, tradução nossa), o design Metro segue alguns princípios:

a) a ênfase no *design* e limpo e tipografia simples de ler;
b) foco em conteúdo, onde a premissa do projeto é voltada para a forma como o conteúdo é apresentado;
c) foco em fazer aplicações onde as informações que mais importam para o usuário sejam apresentadas de uma forma que seja facilmente acessível por um toque simples.

A Microsoft adaptou suas estruturas existentes para que fosse possível programar usando linguagens como C# e Visual Basic com o framework .NET, que fornece uma biblioteca comum com a qual programadores Microsoft NET terão familiaridade, que inclui suporte para *multithreading*, XML, coleções, eventos, dados, exceções, entrada / saída, modelo de serviço, rede, texto, localização, reflexão, a

globalização, recursos, tempo de execução, segurança e diagnósticos, entre muitas outras funcionalidades (LEE; CHUVYROV, 2012, tradução nossa).

4.3.3 Android

Android é um esforço da empresa Google para construir um ambiente de software integrado para dispositivos móveis (ABLESON et al, 2011, tradução nossa) que está revolucionando o espaço móvel, pois, pela primeira vez, foi disponibilizada uma plataforma aberta que separa o hardware do software, permitindo que um número muito maior de dispositivos executem as mesmas aplicações, criando um grande e rico ecossistema para desenvolvedores e consumidores (GARGENTA; NAKAMURA, 2011, tradução nossa).

O ambiente Android inclui um sistema operacional *open source* com o *kernel* baseado em Linux, uma rica interface de usuário, do inglês *user* interface (UI), aplicativos para usuário final, bibliotecas de código, *frameworks* de aplicação, suporte multimídia, entre outros (ABLESON et al, 2011, tradução nossa). O autor ainda diz que o *kernel* baseado em Linux do Android promete agilidade e portabilidade para aproveitar as inúmeras opções de hardware para futuros aparelhos com tecnologia Android e fornece uma camada de abstração de hardware, bem como serviços essenciais, como gerenciamento de processos, de memória e de sistema de arquivos (ABLESON et al, 2011, tradução nossa).

A plataforma Android tem algumas características que o faz ser uma plataforma especialmente interessante para o desenvolvimento integrado: por ser *open source*, muitos de seus componentes podem ser substituídos de imediato, como por exemplo, a tela inicial do sistema (YAGHMOUR, 2013, tradução nossa.).

O sistema operacional Android é projetado para minimizar o consumo de energia dos dispositivos, onde os aplicativos que não estão sendo utilizados em determinado momento são suspensos, sendo mantidos em segundo plano até que sejam necessários novamente (SANDBERG; ROLLINS, 2013, tradução nossa). O Android foi construído para ser flexível, tendo muitos componentes opcionais, como toque telas, câmeras, receptores de sistemas de posicionamento global, do inglês *Global Positioning System* (GPS) e acelerômetros, que dependem da disponibilidade

de hardware em um determinado dispositivo (ABLESON et al, 2011, tradução nossa).

Yaghmour (2013, tradução nossa) cita diversos outros componentes presentes no sistema operacional Android:

a) *framework* de aplicação: é a estrutura usada por desenvolvedores de aplicativos para criar os aplicativos Android;
b) Dalvik Virtual Machine: a implementação do interpretador de *byte-code* usado em Android é como uma recolocação para a máquina virtual Java da Sun;
c) navegador integrado: Android inclui um navegador baseado em WebKit, como parte de sua lista padrão de aplicações;
d) SQLite: é o banco de dados padrão disponibilizado para o desenvolvimento de aplicativos;
e) suporte para telefonia com a tecnologia de sistemas globais para comunicações móveis, do inglês *Global System for Mobile Communications* (GSM): o apoio de telefonia é dependente do hardware e as fabricantes de dispositivos móveis devem fornecer um módulo para permitir a interação do Android com o seu hardware;
f) *bluetooth*, 3G e Wi-Fi: Android inclui suporte para a maioria das tecnologias de conexão sem fio;
g) câmera, GPS, bússola, acelerômetro: interagir com o ambiente do usuário é fundamental para Android;
h) rico ambiente de desenvolvimento: a SDK completa é disponibilizada gratuitamente para *download*, junto com um emulador, um *plug-in* do Eclipse, e um grande número de ferramentas de depuração e *profiling*.

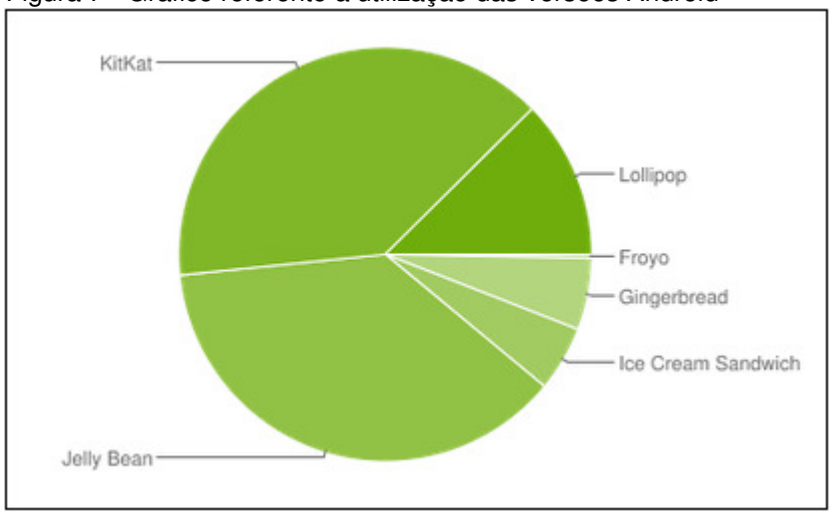
Depois que um aplicativo é baixado para o dispositivo, o Android tem um modelo de segurança que protege ainda mais os usuários de aplicativos maliciosos, pois cada aplicativo é executado em sua própria *sandbox*, o que significa que uma aplicação tem acesso limitado aos recursos do sistema (SANDBERG; ROLLINS, 2013, tradução nossa).

O Android é a plataforma mais popular do mundo móvel e está instalada em 75% do mercado de *smartphones* (OLHAR DIGITAL, 2013) e possibilita uma

integridade com mais de 600.000 aplicativos e jogos disponíveis no Google Play, sua loja virtual (ANDROID, 2013).

Dentre as suas versões, as mais utilizadas são a versão Jelly Bean e a versão Kit Kat, alcançando um mercado de 78,3% de dispositivos, como pode ser observado na figura 7.

Figura 7 - Gráfico referente à utilização das versões Android



Fonte: Android (2015)

Para a realização do projeto, foi escolhida a plataforma Android, pois, além de ser o sistema operacional mais utilizado em *smartphones* e *tablets*, é uma plataforma *open source* tendo um grande e popular mercado de aplicativos.

5 TRABALHOS CORRELATOS

Ao longo do tempo, foram e estão sendo desenvolvidos diversos trabalhos e pesquisas voltados para a integração de dados de diversas fontes, os *mashups*.

Dentre esses trabalhos, destacam-se os artigos, teses e dissertações, que abordam o conceito e utilização de *mashups*, os meios pelos quais se pode trabalhar com essa tecnologia, os parâmetros de qualidade que devem ser aplicados às informações recuperadas por estes *mashups*, entre outros tantos temas que fazem parte deste universo.

Neste capítulo, estão relacionados alguns destes trabalhos, sendo que os seus conteúdos vão de encontro ao conteúdo do projeto proposto.

5.1 MOBILE WEB MASHUPS

A dissertação de Viedma (2010, tradução nossa), para obtenção do grau de Mestre em Sistemas de Comunicação pela universidade ICT, em Estocolmo, tem como objetivo construir um conteúdo de referência para compreender e analisar *mashups* desenvolvidos no ambiente móvel, sendo que o mesmo analisa e descreve as possíveis arquiteturas para o *mobile web*, as ferramentas para construção de interfaces *mobile* e as melhores práticas para a construção de *mashups* para *web* móvel com uma boa experiência do usuário.

Para alcançar estes objetivos, o autor realizou uma pesquisa em literaturas, visando o aprofundamento do seu conhecimento em relação às diferentes arquiteturas e *frameworks* anteriormente desenvolvidos e, subsequentemente, foi realizado um estudo de caso, analisando primeiramente, um *mashup mobile* já existente - o TELAR - e após, quatro exemplos de *mashups* móveis, de modo a proporcionar diretrizes para o seu desenvolvimento. Por fim, o autor implementou dois exemplos de *web mashups* móveis, Sound-Square e Antipodes, para compreendê-los a partir de ponto de vista dos desenvolvedores e testar o quadro de referência criado.

Após realizar as pesquisas e analisar o estudo de caso, Viedma (2010, tradução nossa) chegou à conclusão que não é recomendável utilizar o protocolo SOAP ou o formato de dados XML diretamente no dispositivo móvel, sendo que a comunicação com *web services* deve ser feito usando uma API RESTful e o formato de dados JSON. O autor também afirma que combinar o conceito de *mashups* com os dispositivos móveis podem revelar um mundo de novos *mashups*, satisfazendo as necessidades de nichos e possibilitando a criação de negócios de valor extraordinário. Em relação a arquitetura que deve ser utilizada, Viedma (2010, tradução nossa) encontrou desvantagens nos três tipos abordados. Na arquitetura baseada em servidor, pode ocorrer gargalos, além de ser mais difícil de desenvolver; a arquitetura baseada no cliente pode não ser a melhor opção quando é necessário alto poder computacional e a arquitetura *mobile* compartilha as desvantagens das duas outras arquiteturas. Desta forma, é necessário analisar a necessidade da aplicação para escolher a arquitetura que melhor atende a necessidade da aplicação.

5.2 APLICATIVOS AGREGADORES DE OFERTAS DE COMPRA COLETIVA UTILIZANDO A PLATAFORMA ANDROID

Na dissertação de Patricio e Zanivan (2011), para obtenção do título de Bacharel em Sistemas de Informação na Esucri em Criciúma, o objetivo foi realizar o desenvolvimento de um aplicativo para dispositivos móveis utilizando a plataforma Android para agregar ofertas de *sites* de compra coletiva.

A metodologia aplicada na realização do trabalho consiste no desenvolvimento de um servidor que coleta e organiza os dados de ofertas contidas nos *sites* de compra coletiva, sendo que este servidor é também responsável pela disponibilização das ofertas ao aplicativo móvel. Os *sites* de compras coletivas onde as informações serão coletadas são pré cadastrados e o mapeamento e extração destas informações é realizado por meio das *meta-tags* (ou metadados) e estrutura de *layout* contidas no *site*, sendo que estas *meta-tags* possuem informações sobre a descrição da oferta, categoria e preço. Desta forma, nos *sites* não estruturados, ou seja, que não possuem *tags* que classificam a informação, este tipo de busca não é

possível. As ofertas são devolvidas ao cliente do aplicativo no formato JSON e armazenadas no banco de dados SQLite, nativo do Android.

Segundo Patricio e Zanivan (2011), houve interesse pelo aplicativo, sendo que a taxa de instalações ficou acima da média para a modalidade de compras, que não é tão popular quanto os jogos, e a quantidade de avaliações positivas foram na sua maioria de nota máxima. Com base nas estatísticas, foi possível perceber que a maior parte das instalações foi realizada em dispositivos da linha Galaxy da fabricante Samsung, que é um dispositivo que apenas uma pequena parcela dos usuários de *smartphones* tem acesso, mostrando que apenas os usuários que possuem um poder aquisitivo mais elevado têm condições de acesso a essa tecnologia.

5.3 SOFTWARES ANDROID PARA CORRETORES DE IMÓVEIS

A dissertação de Bitencourt (2013), para obtenção do grau de Bacharel na Universidade do Extremo Sul Catarinense em Criciúma, tem como objetivo o desenvolvimento de uma aplicação na plataforma Android para auxiliar corretores de imóveis no gerenciamento de imóveis e na apresentação dos mesmo aos clientes. O autor utilizou o conceito *Model View Controller* (MVC) no desenvolvimento do aplicativo, que foi criado na plataforma Android, utilizando as API´s Google Maps e o serviço de Street View, também da empresa Google, para localização e visualização de ruas e imóveis.

A sua proposta é disponibilizar um aplicativo onde o corretor de imóveis possa cadastrar os imóveis que estão disponíveis para venda e locação, juntamente com fotos e localização, para que este possa mostrar aos seus clientes a localização exata dos imóveis, por meio do Google Maps e Google Street View, facilitando o processo de locação e venda.

Para Bitencourt (2013), o uso dos mapas torna a apresentação de imóveis muito mais fácil, e o aplicativo se destaca dos já existentes no mercado devido a possibilidade de cadastro com localização e fotos, e a utilização do Google Street View. Ele ainda afirma que os recursos de localização e fotos do imóvel são informações indispensáveis para a venda de um imóvel.

5.4 APLICAÇÕES DE MASHUPS NO GERENCIAMENTO DE REDES

Em sua dissertação para obter o grau de Mestre em Ciência da Computação pela Universidade federal do Rio Grande do Sul, em Porto Alegre, Bezerra (2012) propõe uma solução para gerenciamento de redes utilizando a tecnologia de *mashups*, para que um administrador de rede possa lidar com diversos recursos distintos e problemas situacionais de gerenciamento, fazendo com que o mesmo possa criar suas próprias ferramentas de gerência.

O autor criou um protótipo de sistema de *mashups* de gerenciamento, que possibilita a criação e execução de *mashups* de gerenciamento pelos administradores da rede, por meio de uma interface visual de composição. Este protótipo é um sistema *web* baseado em Ajax que utiliza JSON para intercâmbio de dados. O lado cliente do protótipo é baseado em HTML e Javascript e o lado servidor foi implementado na tecnologia Java Server Pages, rodando em um servidor Apache Tomcat. O repositório do sistema consiste em um banco de dados MySQL, sendo que a comunicação entre o repositório é feita por meio de servlets. Os *mashups* são acessados por meio de *web services* RESTful.

Segundo Bezerra (2012), *mashups* podem sim ser aplicados no gerenciamento de redes, sendo, inclusive, um recurso muito vantajoso, pois um administrador leigo em desenvolvimento de software pode desenvolver ferramentas de gerenciamento de redes. O autor afirma que a aplicação de *mashups* otimiza a distribuição de responsabilidades no processo de desenvolvimento do sistema, onde os administradores cuidam da lógica de gerenciamento de redes enquanto que os desenvolvedores de software lidam com problemas de desenvolvimento envolvendo a criação dos *wrappers*.

6 MEU PREÇO - PROTÓTIPO MÓVEL PARA MONITORAMENTO DE PREÇOS DE PRODUTOS DO E-COMMERCE

O projeto constituiu-se do desenvolvimento de um protótipo de aplicativo para o sistema operacional Android, que permite selecionar e monitorar os preços de produtos de lojas do *e-commerce,* por meio de dispositivos móveis.

Foram utilizados conceitos bastante atuais, como os *mashups*, e diversas tecnologias, como *web services* REST e o formato de representação JSON, aliando-se à plataforma *mobile*, com o sistema operacional Android. Buscou-se utilizar recursos nativos da plataforma Google Android, incorporando outros recursos pertinentes ao desenvolvimento das funcionalidades do protótipo.

5.1 METODOLOGIA

A parte inicial do projeto constituiu-se pelo levantamento bibliográfico de informações sobre as variações de preços dos produtos de lojas virtuais e de recursos existentes para acompanhamento e monitoramento de preços destes produtos. Essas informações serviram de base para que fossem pesquisadas API´s que fornecessem dados referentes aos produtos das lojas virtuais e, neste momento, optou-se por utilizar a API da empresa Buscapé. A partir desta escolha, foi feito um levantamento bibliográfico sobre a tecnologia necessária para utilizar esta API.

Para a execução do projeto, foi escolhida a plataforma Google Android, pois este é uma plataforma muito popular no mundo móvel. Desta forma, o protótipo pode alcançar um número bastante grande de usuários em potencial. A documentação disponibilizada pela Google foi a base para construção do protótipo, sendo que é uma documentação muito completa e de fácil entendimento.

Foram realizados testes em todas as rotinas do protótipo, sendo estes testes feitos em um ambiente fictício, em um ambiente de testes disponibilizado pela Buscapé e no ambiente de produção.

5.1.1 Estudos das ferramentas

No desenvolvimento do protótipo, foi utilizada a SDK do Android referente à API 14, que diz respeito ao Android 4.0. Juntamente com o Android, utilizou-se a linguagem de programação Java, sendo que a SDK já está inclusa na SDK do Android. e para o armazenamento dos dados necessários ao protótipo, foi utilizado o banco de dados SQLite, já incluso no Android.

Foi utilizada a interface ambiente de desenvolvimento integrado, do inglês *Integrated Development Environment* (IDE), Eclipse Luna para desenvolvimento do protótipo, onde também estava incluso a ferramenta LogCat, que auxiliou na depuração do código fonte, mostrando eventuais erros que ocorriam no protótipo, durante o seu desenvolvimento.

Para simular o protótipo dentro de um dispositivo móvel, e assim poder realizar os testes, foi utilizado o emulador que é incorporado ao eclipse no momento da instalação do *plugin*, sendo utilizada a ferramenta *Android Virtual Device Manager* (AVD), que gerencia todas as imagens de emuladores. A imagem selecionada para testes foi a ARM EABI v7a *System Image*.

5.1.2 Requisitos funcionais

Antes de iniciar o desenvolvimento do protótipo, foram definidos os requisitos que o mesmo deveria atender, estando em conformidade com os objetivos propostos. Estes requisitos direcionaram a construção da ferramenta, sendo que as funcionalidades foram criadas e melhoradas a partir do que já havia sido definido. Estes requisitos podem ser visualizados no quadro 2.

Quadro 2 – Requisitos funcionais

Regra de negócio	Descrição
RF-01	O protótipo deve conter uma tela inicial com algumas das opções do aplicativo, sendo a pesquisa de produtos, ofertas monitoradas e uma opção que permite atualizar de imediato as ofertas já selecionadas.
RF-02	Deve existir uma tela onde o usuário pesquisar os produtos que deseja monitorar, que deve conter um onde o usuário poderá informar a descrição do produto que deseja pesquisar e os resultados da pesquisa devem ser mostrados em forma de lista. Em cada oferta, será mostrada ao usuário uma imagem ilustrativa, descrição e preço do produto, a loja onde o produto está sendo ofertado, e deve ser possível selecionar o produto para que o mesmo seja monitorado.
RF-03	Também deve existir uma tela onde seja possível visualizar todos os produtos que estão sendo acompanhados, sendo que cada oferta mostrada deve possuir uma imagem ilustrativa e descrição do produto, a loja de onde o produto está sendo ofertado e o preço do produto. Em cada oferta, deve haver um indicador, mostrando se o preço do produto aumentou ou diminuiu desde a última verificação.
RF-04	O usuário deve poder parar de acompanhar um produto e consultar o histórico de preços de um produto selecionado.
RF-05	Deve ser possível visualizar um histórico de preços do produto, que deve ser mostrado em forma de gráfico, sendo que os preços serão mostrados por data de verificação, no formato mês/ano.
RF-06	A verificação de preços deve ser periódica, sendo que o usuário pode configurar o intervalo de tempo entre cada verificação. Ao identificar uma queda no preço de um produto acompanhado, o protótipo deve enviar uma notificação para o dispositivo do usuário.

Fonte: Do autor

5.1.3 Requisitos não funcionais

Durante a análise das ferramentas utilizadas na construção do protótipo, bem como na definição dos requisitos funcionais, foram constatadas algumas características em relação a API utilizada e também foram definidos alguns processos que visavam o bom funcionamento da ferramenta. Estas características e processos são descritos como requisitos não funcionais, que auxiliam no conhecimento sobre o funcionamento do protótipo. Os requisitos não funcionais podem ser visualizados no quadro 3.

Quadro 3 – Requisitos não funcionais

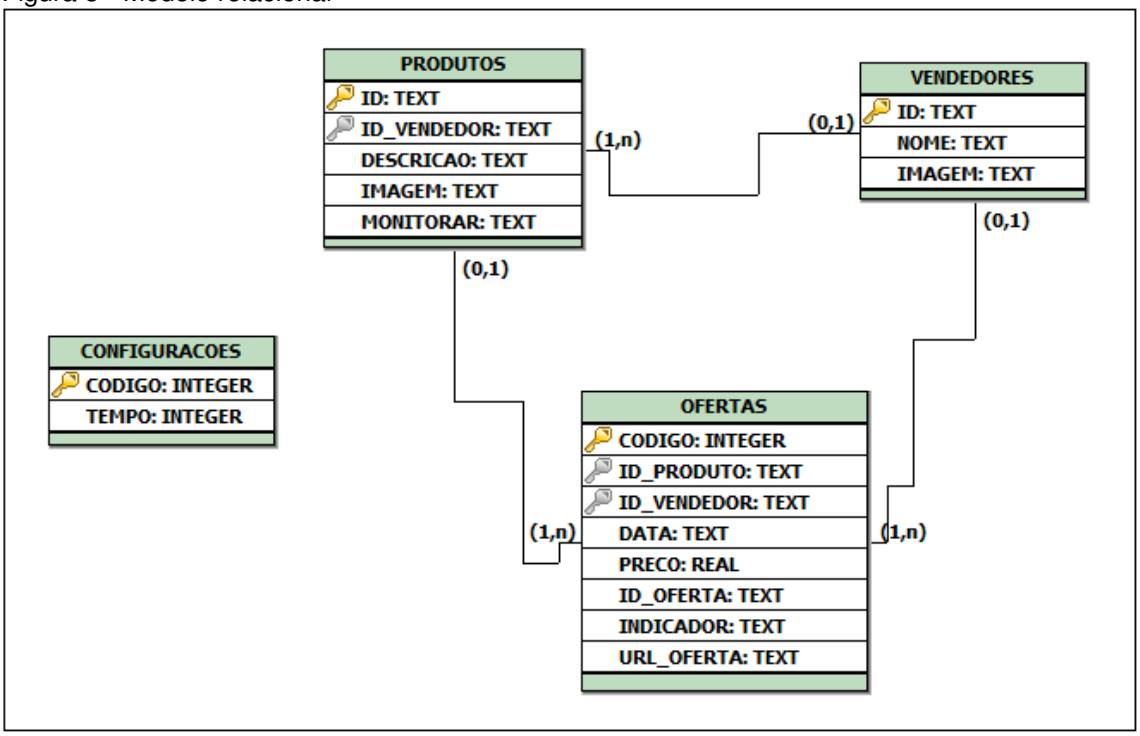
Regra	Descrição
RNF-01	A recuperação das ofertas no *e-commerce* deve ser feita por meio do uso de um *mashup,* criado a partir da API do Buscapé como fonte de dados das ofertas, sendo acessada pelo serviço REST disponibilizado, onde o recurso retorna os dados requisitados no formato JSON.
RNF-02	Manter as imagens do produto/vendedor em *cache*, após serem requisitadas pela primeira vez, aumentando o desempenho da aplicação, pois a imagem de um produto/vendedor não são alteradas constantemente. Deve ser apresentada uma imagem padrão de "ausência de imagem" quando a oferta/vendedor não possuir uma imagem ou a mesma não estiver disponível devido a ausência de conexão com a internet e ausência das imagens no *cache*.
RNF-03	Para serem monitorados, os produtos devem possuir um identificador junto à Buscapé, caso contrário não poderá ser acompanhado.
RNF-04	Caso, no momento da atualização das ofertas não exista conexão com a internet, não será mostrado qualquer erro, sendo que a verificação apenas não será realizada.
RNF-05	Os resultados da pesquisa de produtos devem ser paginados, sendo que cada página deve conter, no máximo, dez itens. Ao carregar os itens da próxima página, os itens da página anterior não serão eliminados, sendo que a lista conterá os itens de todas as páginas.
RNF-06	O protótipo não deve realizar a gravação de informações sobre preferências de consumo dos usuários e não deve disponibilizar qualquer informação do usuário para terceiros, garantindo a privacidade do mesmo na utilização do aplicativo. Também não deve mostrar publicidade baseada nas pesquisas de ofertas do usuário.
RNF-07	A versão mínima para a execução do protótipo é a 4.0 do Android.

Fonte: Do autor

5.1.4 Modelo relacional

No modelo relacional, pode ser visualizada a estrutura de tabelas utilizada para o armazenamento das informações do protótipo, conforme consta na figura 8.

Figura 8 - Modelo relacional



Fonte: Do autor

5.1.5 Desenvolvimento do protótipo

O desenvolvimento do protótipo constituiu-se de algumas etapas bem definidas, sendo que, inicialmente, foi feita uma análise da API utilizada no *mashup*, para levantamento dos dados retornados pela representação em JSON e priorização e definição dos dados que seriam utilizados pelo protótipo.

Após, foi definida a arquitetura do *mashup,* levando em consideração os estudos realizados na etapa de levantamento bibliográfico e, com estas informações, foi possível definir a arquitetura mais apropriada para o protótipo.

Depois de definir os dados a serem utilizados no *mashup* e a arquitetura do *mashup,* se iniciou, efetivamente, o desenvolvimento do protótipo, aplicando

todos os conceitos e tecnologias necessários, que foram descritos no levantamento bibliográfico.

A etapa de homologação incluiu todo o conjunto de testes do protótipo e homologação do mesmo pela empresa Buscapé, processo este que é indispensável para a utilização do ambiente oficial da API.

5.1.5.1 API Buscapé

A Buscapé, conhecida principalmente pelo seu buscador de ofertas, disponibiliza uma API que permite que seus usuários tenham acesso as informações contidas no site Buscapé. Esta API consiste em um *web service* REST, onde são disponibilizados recursos que permitem que o usuário requisite ofertas e produtos contidos na base de dados do Buscapé. As requisições ao *web service* são feitas por meio de uma URI, sendo que devem ser enviados alguns parâmetros obrigatórios, assim como também podem ser enviados parâmetros opcionais.

Para utilizar a API do Buscapé, é necessário criar uma conta de desenvolvedor no site http://developer.buscape.com/pt/. Assim que a conta for criada, é preciso cadastrar o serviço que irá utilizar a API, podendo ser um site, um aplicativo, entre outros, sendo que no caso deste projeto, foi criado um serviço *mobile*. Com este cadastro realizado, será criado um identificador para a aplicação, sendo que este identificador é único e deverá ser usado em todas as requisições realizadas aos recursos da API. O ID do protótipo Meu Preço é 564771466d477a4458664d3d.

Na fase de desenvolvimento do protótipo, o Buscapé disponibiliza uma base de dados de testes, sendo esta base de dados acessível por meio do endereço http://sandbox.buscape.com. Após a homologação do aplicativo, esta URL passa a ser http://bws.buscape.com. As requisições realizadas ao serviço são constituídas pelo endereço do serviço seguido por alguns *paths* e alguns parâmetros, sendo que os *paths* são:

a) *service*: é um *path* padrão da requisição, de uso obrigatório;
b) tipo de listagem: existem vários tipos de listagem das ofertas, como por exemplo, listagem de ofertas, listagem de categorias, entre outros;

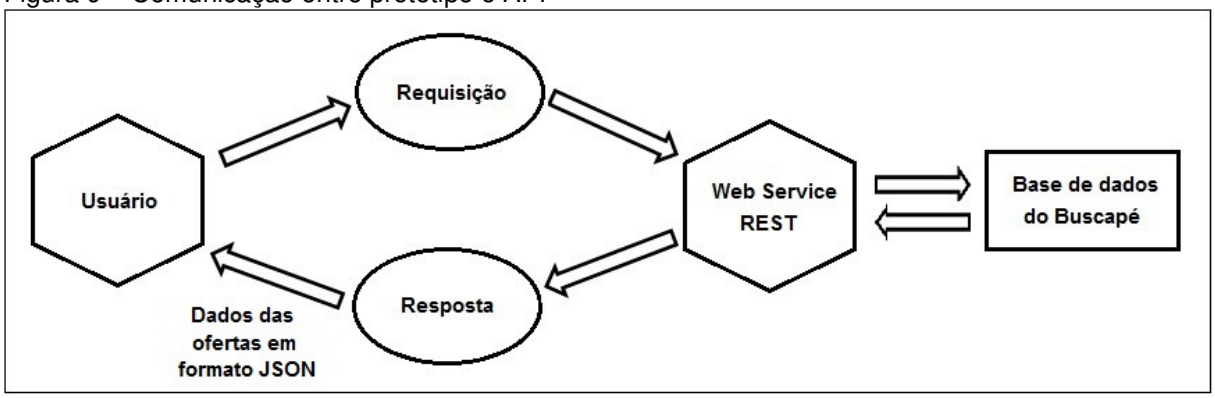
c) identificador da aplicação: se refere ao identificador da aplicação que irá utilizar os recursos da API. Deve ser informado, obrigatoriamente, o identificador gerado pelo site do Buscapé;
d) localidade: é utilizado quando se deseja trazer ofertas de produtos ou lojas que estão dentro das coordenadas passadas por parâmetro.

Existem também parâmetros que podem ser utilizados para filtrar os dados a serem retornados, assim como parâmetros para identificar o tipo de retorno, entre outros. Os parâmetros que são utilizados na construção do protótipo são os parâmetros *productId* e *keyword*, sendo que os resultados serão filtrados pelo identificador ou descrição do produto. O filtro por descrição do produto retorna apenas dez ofertas por requisição, sendo que para carregar mais ofertas, é preciso realizar novas requisições informando os números das páginas subsequentes. Para informar o número da página que se deseja obter as ofertas, utiliza-se o parâmetro *page*.

A API disponibiliza dois tipos de representação dos dados requisitados, sendo eles XML e JSON. O protótipo Meu Preço utiliza a representação em formato JSON, pois este formato é mais leve e se torna mais eficiente na troca de dados para dispositivos móveis. Para que a representação seja em JSON, é necessário informar um parâmetro na requisição ao recurso, sendo este parâmetro chamado *format*, tendo seu valor como JSON. O JSON retornado contém diversas informações referentes à oferta, como por exemplo, código do produto no Buscapé, preço, descrição do produto, loja anunciante, entre outras informações que podem ser visualizadas no Anexo A.

A figura 9 ilustra a forma como o protótipo se comunica com a API do Buscapé, realizando uma requisição ao *web service* REST, que faz uma consulta à base de dados do Buscapé e retorna uma representação no formato escolhido pelo usuário, com os dados solicitados, que no caso do protótipo, é o formato JSON.

Figura 9 – Comunicação entre protótipo e API



Fonte: Do autor

Ao final, tem-se uma URL parecida com http://bws.buscape.com/service/findOfferList/5a73794373456f6a4a6c633d/BR/?format=json&page=1&keyword="Smartphone", sendo que esta URL irá requisitar a lista de ofertas que possuam em sua descrição a palavra chave "Smartphone", onde devem ser retornadas as ofertas da página de número um e este resultado deve estar representado no formato JSON.

Após o desenvolvimento da aplicação, o Buscapé precisa homologar a aplicação, sendo que esta deve ser enviada à empresa e, com o protótipo homologado, o Buscapé libera a URL de acesso ao ambiente oficial da API.

5.1.5.2 Arquitetura

A API do Buscapé é um dos vários exemplos da internet colaborativa, ou *Web* 2.0, sendo que esta API é usada para que se possa criar e disponibilizar novos conteúdos na *web*. Este processo de utilizar recursos disponibilizados por terceiros para criar novos serviços na *web* é conhecido como *mashup*.

O protótipo é composto por um *mashup* baseado na *web*, já que utiliza o serviço fornecido pela Buscapé, por meio de *web services*, para recuperar informações de fontes de dados de terceiros. Ele apresenta as três principais camadas citadas por Liu et al (2007, tradução nossa), sendo que a camada de *interface* de usuário é representada pelos componentes disponibilizados pelo Android, podendo citar os botões, os campos de texto e as imagens; a camada de serviço é representada pela API da Buscapé, que é acessada por meio de *web*

services REST, onde se recupera as informações desejadas; e a camada de ação é representada pelas classes onde são feitos os relacionamentos entre a camada de serviço e a camada de *interface*.

O protótipo Meu Preço caracteriza um *mashup* de consumo, já que combina dados distintos de uma fonte de dados e tem foco na visualização do conteúdo pelo usuário final, utilizando-se de API´s de terceiros para compor o *mashup*. Neste protótipo não são utilizados os três componentes descritos por Merril (2009, tradução nossa), sendo que este possui apenas o provedor de conteúdo e o navegador *mashup*, não sendo necessário possuir o *site mashup*, pois o aplicativo em si já faz o papel deste componente. Desta forma, o navegador *mashup* é o aplicativo propriamente dito, onde ocorre a interação com o usuário e onde a lógica de negócio é executada. A API do Buscapé representa o provedor de conteúdo, sendo este o local de onde o protótipo irá buscar as informações para processar e mostrar ao usuário final.

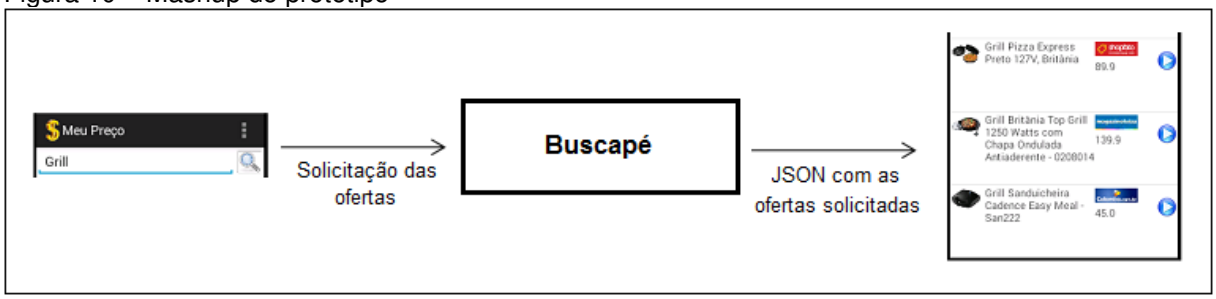
Dentre as arquiteturas de um *mashup*, optou-se por utilizar a arquitetura *client-based*, onde o processamento do conteúdo é feito no navegador do cliente, pois, segundo Ort, Brydon e Basler (2007, tradução nossa), a implementação dos *mashups* se torna muito mais simples quando utilizado este tipo de arquitetura.

5.1.5.3 O protótipo

O protótipo foi construído de forma que permita ao usuário pesquisar e monitorar os preços das ofertas selecionadas. Ao pesquisar ofertas para serem monitoradas, o protótipo faz uma requisição ao recurso disponibilizado pelo Buscapé, informando o filtro escolhido pelo usuário, o parâmetro referente ao número da página a ser retornada e a forma de representação dos dados, que deve ser JSON. Com estes parâmetros, forma-se uma URI parecida com http://bws.buscape.com/service/findOfferList/5a73794373456f6a4a6c633d/BR/?format=json&page=1&keyword="Smartphone". Já com o JSON, o protótipo monta uma lista com todas as ofertas retornadas pela API, mostrando informações que são pertinentes ao usuário, criando assim, um *mashup* com as informações solicitadas. A figura 10 ilustra a criação do *mashup,* sendo que o usuário pesquisa as ofertas de

seu interesse, o protótipo requisita ao Buscapé estas ofertas e a API retorna ao protótipo um JSON com as ofertas solicitadas, onde este irá listá-las para o usuário, mostrando como o protótipo utiliza dados de outra fonte para criar o seu próprio serviço.

Figura 10 – Mashup do protótipo



Fonte: Do autor

Neste momento, o usuário pode selecionar uma ou mais ofertas que deseja acompanhar. Em cada oferta, é mostrada uma imagem que indica se a oferta já está sendo monitorada e, se não estiver sendo monitorada, se torna passível de monitoramento. Ao selecionar uma oferta para ser monitorada, a mesma será gravada no banco de dados SQLite, sendo que as informações serão armazenadas nas tabelas de Produtos, Vendedores e Ofertas. Esta oferta irá aparecer automaticamente na lista de ofertas monitoradas, localizada na tela de ofertas.

Na tabela de Ofertas são gravadas todas as ofertas selecionadas e também todas as atualizações das ofertas. Desta forma, para mostrar uma oferta referente a um produto, é preciso buscar a última oferta gravada nesta tabela, identificada por um código sequencial único para cada registro de oferta, gerado pelo protótipo. Logo, a última oferta de um produto corresponde ao último código sequencial registrado na tabela.

As ofertas são monitoradas por meio dos identificadores dos produtos de cada oferta, que são únicos e provenientes do *mashup* com a API do Buscapé, não sendo alterados nas atualizações de preços ou de qualquer outra informação da oferta. Para fazer com que a oferta não seja mais monitorada, existe uma rotina que informa que a oferta correspondente ao identificador de determinado produto não será mais monitorada. Nenhuma informação é excluída, para que se possa manter históricos dos acompanhamentos.

A atualização das ofertas monitoradas é feita por meio de uma *thread* que é executada periodicamente, de acordo com a configuração feita pelo usuário na tela de Configurações. Esta rotina busca na base de dados do protótipo todas as ofertas que estão sendo monitoradas e, por meio do identificador do produto de cada oferta, realiza mais um *mashup* com a base de dados do Buscapé, fazendo uma requisição à API, enviando o identificador do produto como filtro e não mais a descrição do produto, como é feito na pesquisa. A API do Buscapé irá retornar uma única oferta em seu JSON, oferta esta que será armazenada na tabela de Ofertas do protótipo, independentemente se o preço da oferta foi alterado ou não.

Caso o preço tenha sido alterado na nova oferta, a rotina irá identificar se o novo preço é maior ou menor do que o último preço armazenado no banco de dados referente à oferta buscada. Se o preço for maior, ou seja, o produto ficou mais caro, o protótipo irá mostrar uma imagem indicando que o preço aumentou, na tela de Ofertas. Porém, se o preço for menor, ou seja, se o produto ficou mais barato, o protótipo irá mostrar uma imagem indicando que o preço diminuiu, na tela de Ofertas, e também enviará uma notificação ao dispositivo, avisando sobre a queda do preço, onde nesta notificação o usuário é informado sobre qual produto ficou mais barato, o preço antigo do produto e o novo preço. Ao clicar na notificação, o usuário será redirecionado diretamente para a tela de Ofertas do protótipo, onde poderá visualizar mais informações sobre a oferta. No caso de o preço não ter sido alterado, o protótipo mostra uma imagem indicando que o preço está constante.

O protótipo permite que o usuário visualize um gráfico com o histórico recente da oferta selecionada, sendo que para o desenvolvimento deste gráfico foi utilizada a biblioteca core do Android Plot, versão 0.6.1, onde é possível informar valores para o eixos X e Y.

5.1.6 Testes

Os testes foram feitos tanto em ambientes simulados, como no ambiente de testes do Buscapé e no próprio ambiente de produção. Foram realizados testes no emulador disponibilizado pela Google, ARM EABI v7a *System Image,* e em um dispositivo físico, tendo este um display de 4,7'' e Android 5.01 Lollipop.

Para os testes simulados foi criado um serviço denominado AppService, que simula a API do Buscapé, podendo realizar testes de naturezas distintas, incluindo diversas situações que podem ocorrer com o protótipo.

5.1.6.1 AppService

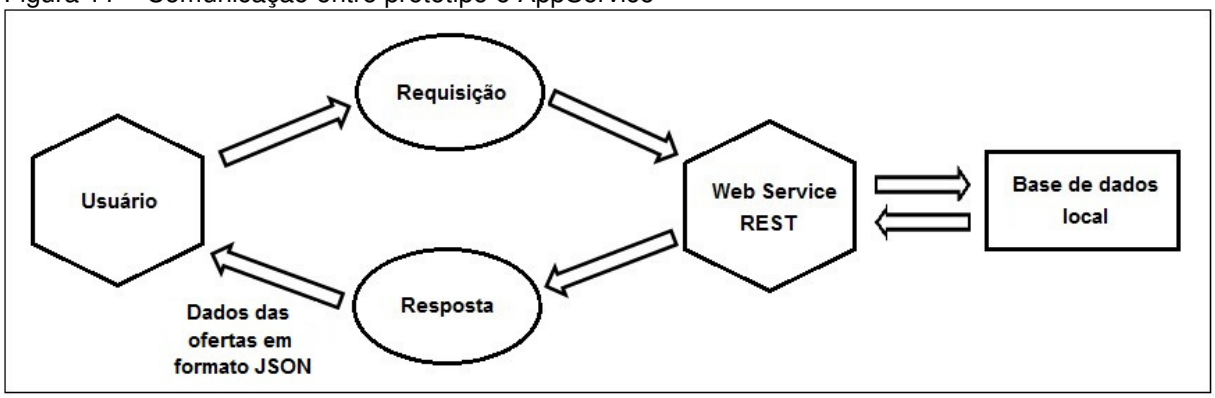
Para realizar determinados testes com o protótipo, foi criado um recurso chamado de AppService, que disponibiliza dados semelhantes ao *mashup* feito com a API do Buscapé, se comportando como um emulador da API. Este recurso simula o serviço disponibilizado pelo Buscapé, com o objetivo de testar certas funcionalidades do protótipo, o que não seria possível no ambiente de testes do Buscapé, como a alteração de preços de um produto em específico em um determinado momento.

Este serviço é composto por um *web service* que, quando requisitado, retorna um JSON com diversas ofertas que foram previamente cadastradas em uma base de dados. O *web service* desenvolvido é baseado na tecnologia REST, seguindo o mesmo princípio da API do Buscapé, sendo que para sua construção foi utilizada a biblioteca JAX-RS. Este *web service* aceita requisições para consultas por descrição e por código do produto, podendo ser informados os parâmetros *keyword*, *productId* e *page*.

Foi criada também, uma base de dados fictícia, onde são armazenadas diversas ofertas também fictícias, simulando a base de dados do Buscapé. Ao realizar uma requisição ao *web service*, o mesmo faz uma consulta nesta base de dados, onde são retornados todos os registros que atendem aos requisitos dos filtros aplicados. Após, estes registros são representados em um JSON, onde neste JSON estão contidas apenas as informações necessárias ao protótipo Meu Preço, sendo este muito mais resumido do que o JSON disponibilizado pela API do Buscapé.

A forma de comunicação do protótipo com este serviço é muito semelhante à forma com que ele se comunica com a API do Buscapé, porém este serviço é disponibilizado apenas localmente e busca os dados solicitados em uma base de dados também local, como mostra a figura 11.

Figura 11 – Comunicação entre protótipo e AppService



Fonte: Do autor

O comportamento do protótipo perante este JSON é o mesmo comportamento obtido com o JSON retornado pela API do Buscapé, sendo possível realizar testes mais realistas com o protótipo.

O serviço AppService foi construído na IDE Netbeans 8, pois esta IDE possui grande facilidade na utilização de servidores de aplicação. O servidor de aplicação onde o *web service* roda é o Glassfish. Como em todo *web service* REST, este possui um recurso e uma representação, sendo que a representação do conteúdo requisitado é no formato JSON, estando em conformidade com o formato de representação da API do Buscapé. O banco de dados escolhido para o armazenamento dos dados fictícios foi o Firebird 2.5, pois é um banco gratuito, leve e de fácil utilização, sendo manipulado por meio da ferramenta IBExpert. Para realizar a conexão entre a aplicação e o banco de dados, foi utilizada a biblioteca Jaybird 2.2.4, devido à sua simplicidade de utilização.

5.1.7 Funcionamento do protótipo

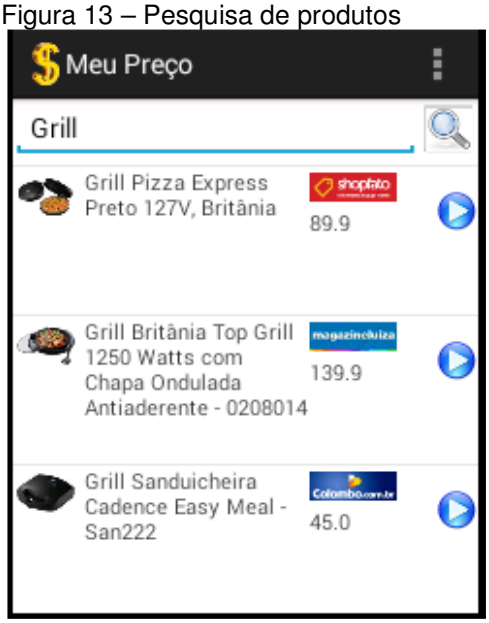
A proposta é que o protótipo tenha uma interface simplificada, minimalista e de fácil uso pelo usuário. Desta forma, ao executar o protótipo, serão mostradas três opções de atividades, além do menu de acesso, onde essas opções são a pesquisa de ofertas, a visualização das ofertas que estão sendo monitoradas e a atualização das ofertas, conforme pode ser visto na figura 12.

Figura 12 – Home

Fonte: Do autor

Na opção Pesquisar, é possível buscar uma lista de ofertas referente ao filtro aplicado, sendo este filtro a descrição do produto que se deseja encontrar. Nesta tela, informa-se o produto desejado e, pressionando o botão de pesquisa, será renderizada uma lista com várias ofertas referentes ao produto pesquisado, como pode ser visto na figura 13.

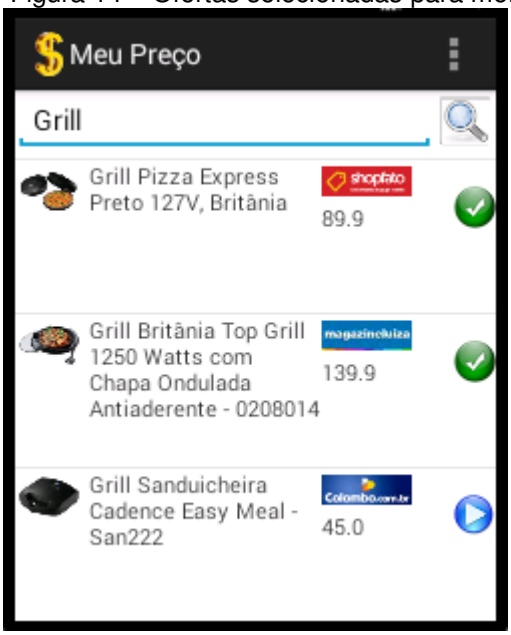
Figura 13 – Pesquisa de produtos



Fonte: Do autor

Para monitorar uma oferta, pressiona-se o botão azul, indicado por uma flecha para a direita, onde esta ação irá alterar a imagem do botão, indicando que a oferta está sendo monitorada. Quando um item é pesquisado e o mesmo já está sendo monitorado, a imagem é alterada automaticamente, onde estas ações podem ser visualizadas na figura 14.

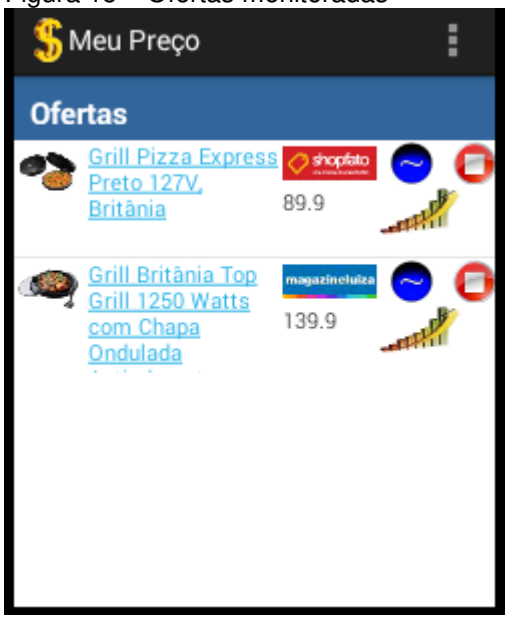
Figura 14 – Ofertas selecionadas para monitoramento



Fonte: Do autor

Quando as ofertas são selecionadas para serem monitoradas, elas são mostradas na tela de Ofertas, acessível pela opção Ofertas da tela Home ou pelo menu de acesso. Esta tela lista todas as ofertas que estão sendo monitoradas, mostrando o preço mais atual da oferta. A descrição do produto possui um *link* para a oferta no site do Buscapé. Dentre as informações mostradas em cada oferta, é possível visualizar uma imagem que indica se o preço da oferta está maior ou menor do que o preço anterior. Porém, se no momento da seleção da oferta ainda não houve verificações de variação de preço, a imagem irá indicar que o preço está constante, como pode ser visualizado na figura 15.

Figura 15 – Ofertas monitoradas



Fonte: Do autor

O botão sinalizado pelo símbolo de *stop* permite que uma oferta deixe de ser monitorada. Quando este botão é pressionado, o usuário é questionado sobre o real desejo de não acompanhar mais a oferta, e se a resposta for afirmativa, a oferta não será mais monitorada e também não será mais mostrada na lista de ofertas. A mensagem de confirmação enviada ao usuário pode ser vista na figura 16.

Figura 16 – Mensagem de confirmação enviada ao usuário

Fonte: Do autor

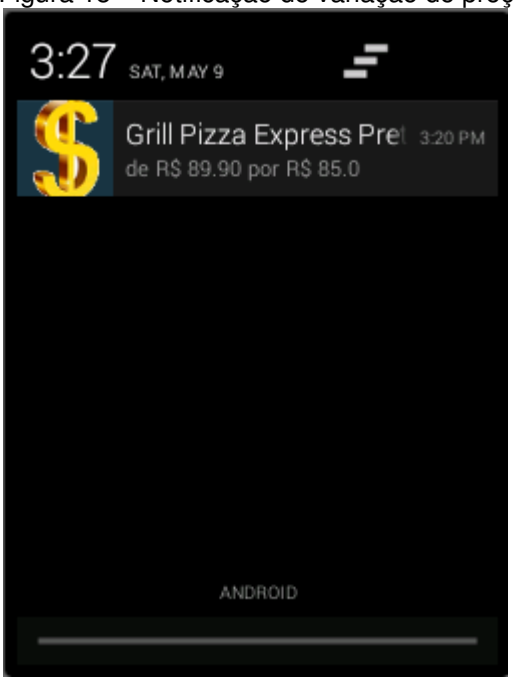
Quando as ofertas são atualizadas, com a finalidade de verificar alterações no preço das ofertas, os indicadores mudam de acordo com a variação do preço. Se o produto de uma oferta ficou mais caro, será mostrada a imagem de uma seta vermelha apontada para cima, que significa que o preço subiu. Porém, se o produto ficou mais barato, será mostrada a imagem de uma seta verde apontada para baixo, indicando que o preço diminuiu, e é lançada uma notificação para o dispositivo, conforme é mostrado na figura 17.

Figura 17 – Indicadores de variação de preço

Fonte: Do autor

A notificação mostra o nome do produto referente à oferta em que o preço diminuiu, o preço antigo da oferta e o novo preço, sendo que ao selecionar a notificação, o usuário é redirecionado para a tela de Ofertas do protótipo, conforme é mostrado na figura 18.

Figura 18 – Notificação de variação de preço



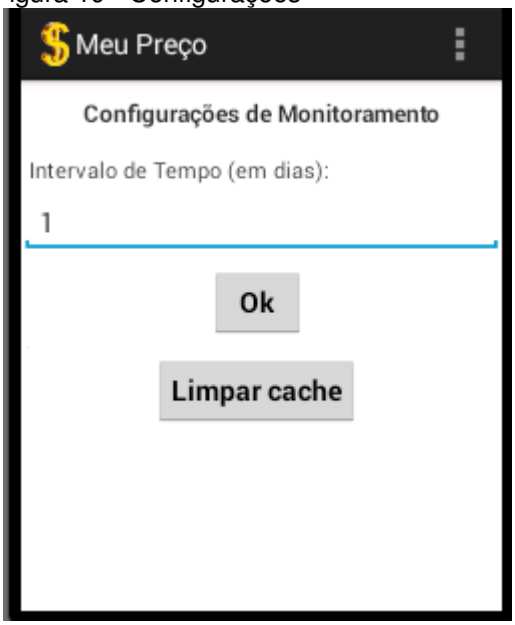
Fonte: Do autor

As ofertas são atualizadas automaticamente, levando em consideração a periodicidade configurada na tela de Configurações. Porém, este processo pode ser feito manualmente, por meio da opção de Atualizar Ofertas, contida na Home do protótipo.

O botão referente ao gráfico mostra um histórico, em forma de gráfico, das variações dos preços das ofertas monitoradas, sendo acessível somente pela tela de Ofertas. Neste gráfico são mostradas as últimas dez ofertas de um mesmo produto, sendo possível visualizar a variação de preços de um produto.

A tela de Configurações é acessível somente pelo menu de acesso, e possui uma opção para definir qual a periodicidade da atualização das ofertas no protótipo, além de uma opção para limpar o cache de imagens, referentes ao protótipo, do dispositivo, sendo esta tela ilustrada na figura 19.

Figura 19 - Configurações



Fonte: Do autor

5.2 RESULTADOS DO MONITORAMENTO

Para verificar a efetividade do protótipo, foram monitorados, durante dez dias, quatro produtos distintos em diferentes lojas virtuais, totalizando vinte ofertas acompanhadas, conforme pode ser visto no quadro 4.

Quadro 4 – Produtos monitorados

	Loja A	Loja B	Loja C	Loja D	Loja E	Loja F
Smartphone Motorola Novo Moto G Android 4.4 Tela 5" 16GB	X	X	X	X	X	
O Mundo de Gelo e Fogo	X	X	X	X		X
Fritadeira Air Fry Saúde Britânia Branca	X		X	X		X
Smart TV 3D LED 50' Philips Ultra HD 4K	X	X	X	X	X	X

Fonte: Do autor

Durante este período, o aplicativo atualizou as ofertas diariamente, sendo que, a partir dos dados obtidos, foi possível constatar algumas informações. O produto Smartphone Motorola Novo Moto G Android 4.4 Tela 5" 16GB teve alterações graduais na maioria das lojas, com preços que variaram entre R$699,00 a

R$898,99, sendo que para o consumidor que deseja pagar menos pelo produto, o valor que pode ser economizado é bastante grande, chegando a quase R$200,00. Os preços praticados pelas lojas durante os dez dias de acompanhamento, referentes ao Novo Moto G, podem ser visualizados na tabela 1.

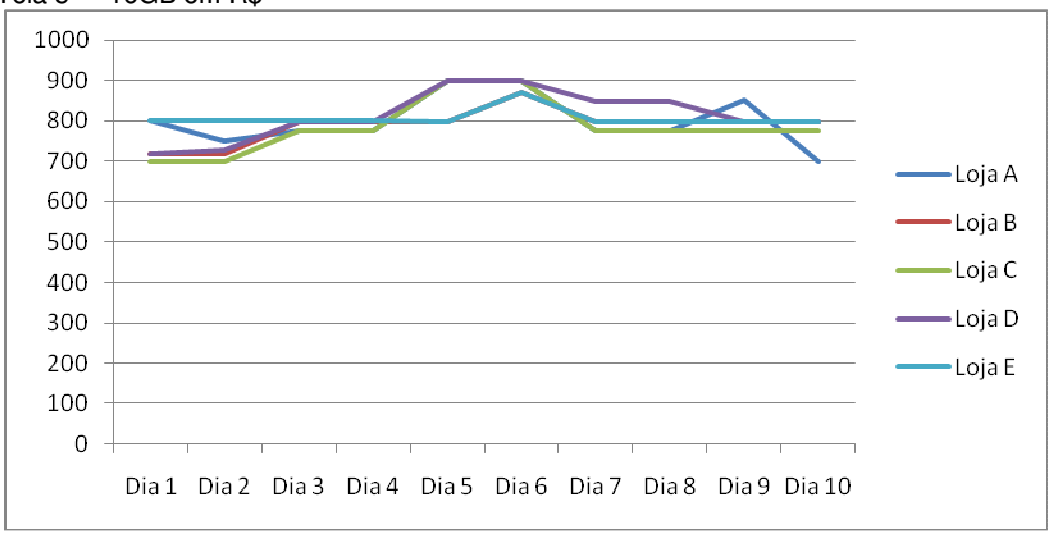
Tabela 1 – Preços do Smartphone Motorola Novo Moto G Android 4.4 Tela 5" 16GB em R$

	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9	Dia 10
Loja A	799,00	749,00	776,67	776,67	898,99	898,99	776,67	849,00	699,00	699,00
Loja B	719,10	719,10	798,00	798,00	796,99	868,99	797,00	796,00	795,79	795,79
Loja C	699,00	699,00	776,77	776,77	898,99	898,99	776,67	776,67	776,67	776,67
Loja D	719,10	729,00	799,00	799,00	898,99	898,99	849,00	849,00	799,00	799,00
Loja E	798,00	798,00	798,00	798,00	797,00	868,99	796,99	796,99	796,99	796,99

Fonte: Do autor

A variação de preços do Novo Moto G 16Gb é mostrada no gráfico contido na figura 20.

Figura 20 – Gráfico da variação de preços do Smartphone Motorola Novo Moto G Android 4.4 Tela 5" 16GB em R$



Fonte: Do autor

O livro O Mundo de Gelo e Fogo, que foi monitorado em cinco lojas diferentes, também apresentou preços bastante distintos em certos momentos, como por exemplo, no primeiro dia de acompanhamento, onde o preço na loja D era de R$118,66 e nas lojas B e F era de R$62,01. Com o acompanhamento diário, foi

observado que no oitavo dia o preço caiu para R$49,90, proporcionando ao usuário do protótipo a possibilidade de economia de aproximadamente R$68,00. Todos os preços do livro, que foram obtidos nos dez dias de monitoramento estão expressos na tabela 2.

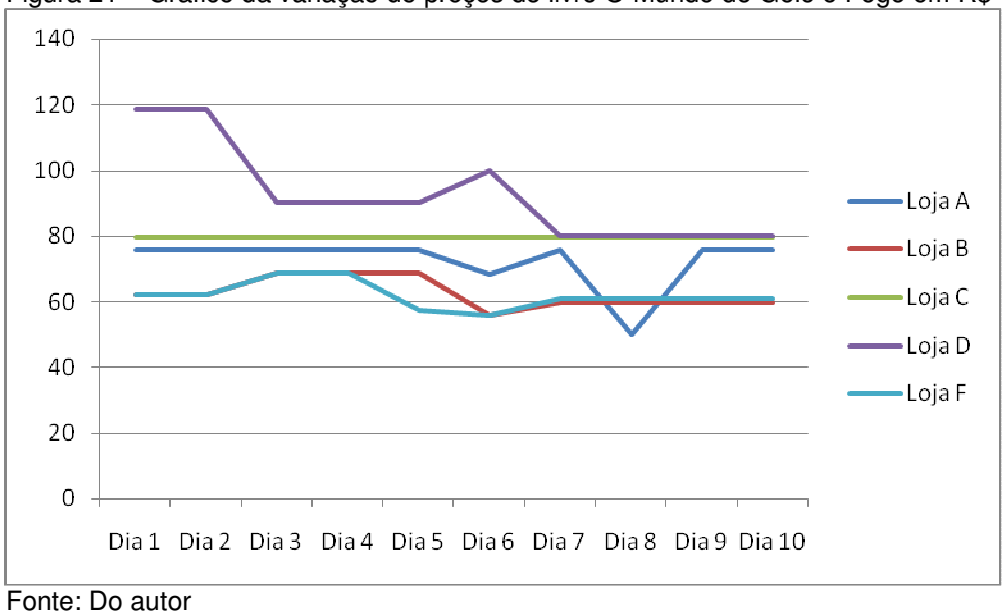
Tabela 2 – Preços do livro O Mundo de Gelo e Fogo em R$

	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9	Dia 10
Loja A	75,90	75,90	75,90	75,90	75,90	68,31	75,90	49,90	75,90	75,90
Loja B	62,01	62,01	68,90	68,90	68,90	55,81	59,90	59,90	59,90	59,90
Loja C	79,80	79,80	79,80	79,80	79,80	79,80	79,80	79,80	79,80	79,80
Loja D	118,66	118,66	89,91	89,91	89,91	99,90	79,92	79,92	79,92	79,92
Loja F	62,01	62,01	68,90	68,90	57,57	55,81	60,90	60,90	60,90	60,90

Fonte: Do autor

A figura 21 ilustra a variação de preços do livro O Mundo de Gelo e Fogo nas lojas monitoradas, onde é possível observar algumas curvas bastante acentuadas, principalmente nas lojas F e A.

Figura 21 – Gráfico da variação de preços do livro O Mundo de Gelo e Fogo em R$



Fonte: Do autor

A Fritadeira Air Fry Saúde Britânia Branca teve preços mais similares entre as lojas, se mantendo, na maior parte do tempo, na mesma faixa de preço, salvo algumas exceções. Porém também, analisando os dez dias de

acompanhamento deste produto nas quatro lojas selecionadas, observa-se que o usuário do protótipo teve a possibilidade de economia de aproximadamente R$159,00. Verificou-se que a loja F, após o produto ficar indisponível por um dia, aumentou substancialmente o preço da fritadeira. A tabela 3 permite que sejam visualizados os preços deste produto no decorrer dos dez dias de monitoramento.

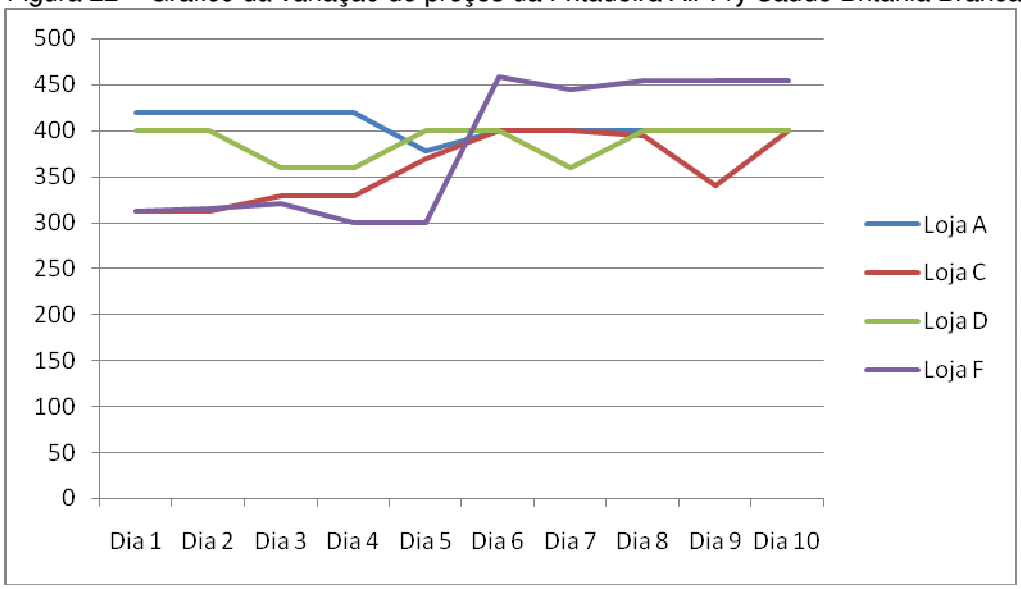
Tabela 3 – Preços da Fritadeira Air Fry Saúde Britânia Branca em R$

	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9	Dia 10
Loja A	419,90	419,90	419,90	419,90	377,91	399,90	399,90	399,90	399,90	399,90
Loja C	312,55	312,55	329,00	329,00	369,90	399,90	399,90	393,90	339,90	399,90
Loja D	399,90	399,90	359,91	359,91	399,90	399,90	359,91	399,90	399,90	399,90
Loja F	311,90	314,90	320,89	299,90	-	459,00	444,90	454,90	454,90	454,90

Fonte: Do autor

A figura 22 permite que seja visualizada a variação de preços da Fritadeira Air Fry Saúde Britânia Branca, onde é possível observar que a loja F teve uma grande alteração de preços entre o quinto e sétimo dia de acompanhamento, sendo mais viável comprar o produto nas outras lojas.

Figura 22 – Gráfico da variação de preços da Fritadeira Air Fry Saúde Britânia Branca em R$



Fonte: Do autor

A Smart TV 3D LED 50' Philips Ultra HD 4K foi a que teve maior possibilidade de economia por parte do usuário do protótipo, sendo que nos

primeiros dias de monitoramento, o preço na loja C chegou a R$2.348,19, sendo que o preço deste produto girou em torno de R$2.899,00 na maior parte do tempo. Houve indisponibilidade em algumas lojas entre o sétimo e oitavo dia, porém, ao se tornar disponível novamente, observou-se que o preço não sofreu grandes alterações. Todos os preços obtidos durante o monitoramento estão contidos na tabela 4.

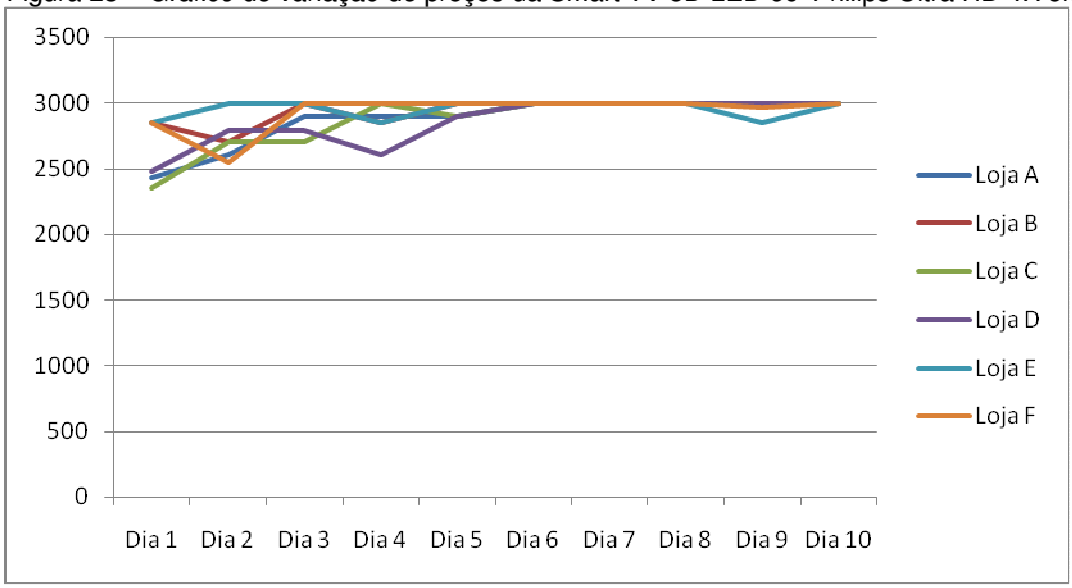
Tabela 4 – Preços do livro O Mundo de Gelo e Fogo em R$

	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9	Dia 10
Loja A	2428,10	2609,10	2899,00	2899,00	2899,00	2999,00	2999,00	2999,00	2999,00	2999,00
Loja B	2849,05	2699,10	2999,00	2849,05	2999,00	2999,00	-	-	2999,00	2999,00
Loja C	2348,19	2699,10	2699,10	2999,00	2899,00	2999,00	2999,00	2999,00	2999,00	2999,00
Loja D	2478,65	2792,70	2792,70	2609,10	2899,00	2999,00	2999,00	2999,00	2999,00	2999,00
Loja E	2849,05	2999,00	2999,00	2849,05	2999,00	2999,00	-	-	2849,05	2999,00
Loja F	2849,05	2549,15	2999,00	2999,00	2999,00	2999,00	-	-	2968,90	2999,00

Fonte: Do autor

Com a figura 23 é possível visualizar que a 3D LED 50' Philips Ultra HD 4K se manteve com preços estáveis na maior parte do tempo, sendo estes preços similares inclusive entre as seis lojas monitoradas.

Figura 23 – Gráfico de variação de preços da Smart TV 3D LED 50' Philips Ultra HD 4K em R$



Fonte: Do autor

Em alguns dos produtos monitorados foram observadas alterações sutis de preço em alguns momentos, sendo que mesmo que eles fiquem mais baratos do que a oferta do dia anterior, este valor é mínimo, algumas vezes se tratando de apenas R$1,00. Também foi observado que alguns produtos tiveram seu preço aumentado substancialmente em um dia e no dia seguinte este preço foi diminuído novamente, não caracterizando, muitas vezes uma boa oportunidade de compra, se o histórico de preços for levado em consideração.

7 CONCLUSÃO

Atualmente, o mercado apresenta um cenário de grande concorrência entre lojas do varejo, sendo que os clientes estão sempre em busca de qualidade e melhores preços. Movidos pela demanda do mercado, os lojistas procuram sempre oferecer produtos com preços atrativos ao cliente, gerando uma grande variação de preços entre lojas e produtos. Para encontrar o melhor preço, os clientes utilizam diversos artifícios, como, por exemplo, pesquisas na internet, que são feitas muitas vezes por dispositivos móveis. O universo móvel está em constante expansão, tendo cada vez mais usuários e recebendo fortes investimentos por parte das empresas.

A procura do cliente por melhores preços, constatada com a fundamentação teórica, foi o que motivou o desenvolvimento deste trabalho, onde foi proposto um protótipo que realizasse o monitoramento de preços de produtos selecionados pelo usuário. Para alcançar este objetivo, propôs-se que o protótipo utilizasse *mashups*, onde conteúdos já existentes na *web* fossem utilizados na construção do mesmo. O conceito estudado sobre *mashups* foi aplicado por meio da definição da arquitetura do protótipo e da API utilizada para busca das ofertas na *web.*

A utilização de *mashups* simplificou significativamente o desenvolvimento do protótipo, pois este teve uma base de dados de ofertas já existente e ampla à disposição para que fosse feito o tratamento dos dados. A construção do *mashup* mostrou que existem diversas formas de desenvolver aplicativos e recursos utilizando conteúdos já existentes, tornando a implementação mais ágil e com diversas possibilidades de reutilização, sendo que esta técnica é, inclusive, usada em diversos serviços já existentes.

A mobilidade do protótipo também foi um dos objetivos do trabalho, alcançado por meio do uso da plataforma *mobile* Google Android, que fez com que o protótipo possa ser usado em *smartphones*, *tablets*, entre outros, já que foi constatado que o mercado *mobile* está em constante expansão. Com o protótipo, o usuário pode selecionar os produtos que forem do seu interesse e monitorar os preços apenas destes produtos, evitando propagandas abusivas e gravação de dados pessoais da sua navegação pela internet.

Os resultados do monitoramento das ofertas selecionadas mostraram que existe uma grande variação de preços de produtos, não só entre as lojas varejistas, como também na mesma loja, onde o preço de um determinado produto era, eventualmente, alterado diversas vezes em uma mesma semana. Verificou-se também que nem sempre uma queda no preço indica que o produto ficou mais barato, pois houveram casos onde o preço de um produto variava entre maior e menor, sendo que este histórico é mostrado no gráfico disponibilizado pelo protótipo. Como o protótipo mantém um histórico das ofertas acompanhadas, foi observado que ele pode auxiliar também na identificação de boas ofertas em grandes promoções, como por exemplo a Black Friday, onde o usuário possa ter certeza de que está realmente pagando menos pelo produto desejado.

Com o desenvolvimento deste trabalho, foi possível entender melhor o mercado *mobile* e como as pessoas procuram sempre o melhor preço, buscando cada vez mais a economia. Houve um grande aprendizado sobre técnicas de extração de dados da *web* e desenvolvimento *mobile* em Android. Desta forma, foi possível observar que *mashups* são soluções bastante simples e muito úteis, podendo ser criados por meio de mais de uma técnica de extração de dados.

Observa-se que, caso existisse uma padronização na forma de representação dos dados disponibilizados pelas API´s, poderiam ser utilizadas, de forma transparente, diversas fontes de dados para a pesquisa de ofertas na *web*. Com base na análise dos resultados, constatou-se que o protótipo é uma ferramenta eficiente no acompanhamento de preços de produtos de lojas virtuais, auxiliando o usuário na tomada de decisão de compra de um produto do seu interesse, alcançando, desta forma, os objetivos propostos. O protótipo não grava informações de preferências de ofertas dos usuários e não disponibiliza nenhuma informação do usuário para terceiros, garantindo a privacidade do mesmo na utilização do aplicativo. Dentre as dificuldades encontradas, pode-se citar a criação dos layouts do protótipo e a fundamentação teórica sobre *mashups,* sendo que não existem muitos livros que falam sobre o assunto.

Para trabalhos futuros sugere-se:

a) desenvolver uma forma de controle de atualizações das ofertas, para que as ofertas sejam atualizadas dentro do período configurado pelo

usuário, levando em consideração a disponibilidade de internet;

b) criar um servidor *mashup,* onde todo o processamento do protótipo seja feito neste servidor, retornando ao usuário as ofertas já atualizadas;
c) utilizar *web* semântica para realizar a consulta de ofertas nas lojas virtuais, fazendo uso de mais uma técnica de *mashup*;
d) sugerir para o usuário, caso existam, ofertas que estejam com um preço menor do que aquelas que ele já está acompanhando, sendo que estas ofertas devem se referir ao mesmo produto acompanhado.

REFERÊNCIAS

ABEYSINGHE, S. **PHP Team Development**. Birmingham: Packt Publishing, 2009.

ABLESON, F. W.; KING, C.; ORTIZ, E.; SEN, R. **Android in Action**. 3. ed. Nova York: Manning Publications, 2011.

ALLAMARAJU, S**. RESTful Web Services Cookbook**. Sebastopol: O'Reilly Media, 2010.

ALVES, A.; ARKIN, A.; ASKARY, S.; BARRETO, C. **Web services business process execu tion language version 2.0**. OASIS standard. 2007. Disponível em: <http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html>. Acesso em: 30 ago. 2014.

ANDROID. **Welcome to Android and About**. Android. 2015. Disponível em: <http://source.android.com>. Acesso em: 23 mai. 2015.

APPLE. **iOS**. Apple. 2014. Disponível em: <https://developer.apple.com/devcenter/ios/index.action>. Acesso em: 20 out. 2014.

ASSIS, G. **Guia de e-mail marketing**. São Paulo: Ibrasa, 2003.

BANGA C.; WEINHOLD, J**. Essential mobile interaction design**: Perfecting Interface Design in mobile Apps. Crawffordsville: Pearson Education, 2014.

BERTAGLIA, P.R. **Logística e gerenciamento da cadeia de abastecimento**. São Paulo: Saraiva. 2003.

B'FAR, R. **Principles Designing and Developing Mobile Applications with UML and XML**. 1. ed. Cambridge: Cambridge University Press, 2005.

BOUILLET, E. M. Middleware for assembly and deployment of multi-platform flow-based applications. In: **ACM/IFIP/USENIX INTERNATIONAL CONFE-RENCE ON MIDDLEWARE**, n. 10, 2009, New York. Springer-Verlag... New York: Inc., 2009. p.26–26.

BRANDT, B. **Web 2.0**. Berg Brandt. 2006. Disponível em <http://bergbrandt.com.br/v1/arquivos/artigos/20061006_web2.pdf>. Acesso em: 23 ago. 2014.

BURKE, B. **RESTful Java with JAX-RS**. Sebastopol: O'Reilly, 2009.

BUSCAPE. **API para afiliados**. Dev Buscapé Company. 2014. Disponível em: < http://developer.buscape.com/pt/product/buscape/manual-api.html>. Acesso em: 26 out. 2014.

BRASIL. **Marco Civil da Internet**. Câmara dos Deputados. 2014. Disponível em: <http://www2.camara.leg.br/documentos-e-pesquisa/fiquePorDentro/temas/marco-civil>. Acesso em: 26 out. 2014.

CANALYS. **Canalys**: Insight. Innovation. Impact. Palo alto, 2012. Disponível em: <http://www.canalys.com/newsroom/smart-phones-overtake-client-pcs-2011>. Acesso em: 29 mar. 2014.

CAPPIELLO, C.; DANIEL, F.; MATERA, M.; PAUTASSO, C. Information Quality in Mashups. **Internet Computing, IEEE**, Salt Lake City, v. 14, n. 4, p. 14-22, ago. 2010.

CATALANI, L. **E-commerce**. 2. ed. Rio de Janeiro: FGV, 2008.

CHAPPELL, D.; JEWELL, T. **Java Web Services**. New York: O'Reilly, 2002.

DITOLVO, Mariana. Redes Sociais crescem em importância estratégica. **Meio&mensagem,** São Paulo, n. 1415, p. 38 – 39, jun. 2010.

E-COMMERCE. **E-commerce Brasil**: Excelência em e-commerce. São Paulo, 2014. Disponível em: <http://www.ecommercebrasil.com.br/artigos/comercio-eletronico-2013-bons-motivos-para-ser-otimista-em-2014/>. Acesso em: 30 mar. 2014.

ERENBERG, J. J.. ***Publicidade patológica na internet à luz da legislação brasileira***. São Paulo: Juarez de Oliveira, 2003.

FIELDING, R. **Architectural Styles and the Design of Network-based Software Architectures**. 2000. 180 f. Tese (Doutorado Filosofia da Informação e Ciência de Computação) – University of California, Universidade Estadual Paulista, Irvane, 2000.

FIELDING, R.; RESCHKE, J. **Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content**. IETF Tools. 2014. Disponível em: <http://tools.ietf.org/html/rfc7231>. Acesso em 14 set. 2014.

FIRTMAN, M. **Programming the Mobile Web.** Sebastopol: O'Reilly, 2010.

FLANDERS, J. **RESTful .NET**. 1. ed. Sebastopol : O'Reilly, 2009.

FLING, B. **Mobile Design and Development**. Sebastopol: O'Reilly, 2009.

FRANCISCHELLI, P. **A importância da marca no processo de decisão de compra de calçados esportivos para a população de Baixa Renda**. 2009. Dissertação (Mestrado Executivo em Gestão Empresarial) - Fundação Getúlio Vargas do Rio de Janeiro. Rio de Janeiro, 2009.

FREITAS, J. P. **E-mail Marketing**: Errando no Basicão. Proxxima. São Paulo, n. 13, p.18-25, out. 2009.

GARGENTA, M.; NAKAMURA, M. **Learning Android**. 2. ed. Sebastopol: O'Reilly, 2014.

GARTNER. Gartner says worldwide tablet sales grew 68 percent in 2013, with android capturing 62 percent of the market**. GARTNER. 2014. DISPONÍVEL EM: <HTTPS://WWW.GARTNER.COM/NEWSROOM/ID/2674215>. ACESSO EM: 15 MAR. 2014.**

GLEZ-PEÑA, D.; LOURENÇO, A.; LOPEZ-FERNANDEZ, H.; REBOIRO-JATO, M.; FDEZ-RIVEROLA, F. Web scraping technologies in an API world. **Briefings in Bioinformatics**, Oxford, v. 15. n. 5, p. 788-797, abr. 2013.

GOMES, D. A. Web **Services SOAP em Java**. São Paulo: Novatec, 2010.

GOOGLE ANDROID. **Android.** Android. 2014. Disponível em: <http://source.android.com/ >. Acesso em: 10 out. 2014.

GOOGLE PLAY. **Loja**. Google Play. 2014. Disponível em: <https://play.google.com/store?hl=pt_BR>. Acesso em: 05 out. 2014.

GORDON, S. R.; GORDON, J. R. **Sistemas de informação**: uma abordagem gerencial. Rio de Janeiro: LTC, 2006.

GOURLEY, D.; SAYER, M.; REDDY, S.; AGGARWAL, A. **HTTP**: The Definitive Guide. Sebastopol: O'Reilly, 2002.

HONORATO, G. **Conhecendo o Marketing** .São Paulo: Manole, 2004.

HOSKINS, D. J. **XML and InDesign**. Sebastopol: O'Reilly, 2013.

IBOPE MÍDIA. **80% dos internautas usam web pra comparar preços**. Bahia Econômica. 2012. Disponível em < http://www.bahiaeconomica.com.br/noticia/16100,80-dos-internautas-usam-web-para-comparar-precos-mostra-pesquisa.html>. Acesso em 29 jun. 2015.

IDC. **Worldwide Smartphone Shipments Top One Billion Units for the First Time.** IDC. 2014. Disponível em: <http://www.idc.com/getdoc.jsp?containerId=prUS24645514>. Acesso em: 23 mar. 2014.

JENDROCK, E.; EVANS, I.; GOLLAPUDI, D.; HAASE, K.; SRIVATHSA, C. **The Java EE 6 Tutorial**: Basic Concepts. 4. ed. Michigan: Addison-Wesley, 2011.

JSOUP. **JSOUP**: Java HTML Parser. JSOUP. 2014. Disponível em: <http://jsoup.org/>. Acesso em 30 out. 2014.

KALALI, M.; MEHTA, B. **Developing RESTful Services with JAX-RS 2.0, WebSockets, and JSON**. Birmingham: Packt Publishing, 2013.

KALIN, M. **Java Web Services**: Up and Running. 2. ed. Sebastopol: O´Reilley, 2013.

KAMALELDIN , M.; DUMINDA, W. Performance Analysis of Web Services on Mobile Devices. **Procedia Computer Science**, v. 10, p. 744-751, 2012.

KORGAONKAR, P. K.; WOLIN, L. D.; A Multivariate Analysis of Web Usage. **Journal of Advertising Research**, p. 53-68, 1999.

KOTLER, P.; ARMSTRONG, G. **Princípios de marketing**. 9. Ed. São Paulo: Pearson Prentice Hall, 2003.

KOTLER, P.; KELLER, K. L. **Administração de marketing**. 12. ed. São Paulo: Pearson Prentice Hall, 2006.

LAUDON, K,C.; LAUDON, J.P. **Sistemas de informações gerencias**. São Paulo: Pearson Prentice Hall, 2007.

LEE, H.; CHUVYROV, E. **Beginning Windows Phone App Development**. Nova York: Apress, 2012.

LI. Q.; SHI, J.; ZHU, H. A formal framework for service mashups with dynamic service selection. Innovations Syst Softw Eng, Londres, v. 10, p. 219–234 , jul. 2014.

LIMEIRA, T. **Fundamentos de marketing**. São Paulo: Saraiva, 2003.

LIU, X.; HUI, Y.; SUN, W.; LIANG, H. Towards service composition based on mashup. In: **IEEE Congress on Services,** Salt Lake City, p. 332–339, jul. 2007.

LIVINGSTON, C. **In mobile search, there can be only Google (for now)**. TECHi. 2011. Disponível em: <http://www.techi.com/2011/08/google-mobile-search/>. Acesso em: 15 mar. 2014.

LORENZETTI, R. L. **Comércio eletrônico.** São Paulo: Revista dos Tribunais, 2004.

LOTT, S. F. **Mastering Object-oriented Python**. Birmingham: Packt Publishing, 2014.

MARTINS, R. J. W. de A. **Desenvolvimento de Aplicativo para Smartphone com a Plataforma Android**. 2009. 50 f. Graduação em Engenharia de Computação - Pontifica Universidade Católica do Rio de Janeiro, Rio de Janeiro. 2009.

MAXIMILIEN, E.; RANABAHU, A.; GOMADAM, K. An Online Platform for Web APIs and Service Mashups. **IEEE Internet Computing**, v.12, n.5, p.32-43, out. 2008.

MERRILL, D. Mashups: T**he new breed of Web app - An introduction to mashups**. IBM. 2009. Disponível em: <http://www.ibm.com/developerworks/library/x-mashups/>. Acesso em: 15 set. 2014.

MITCHELL, L. J. **PHP Web Services**. Sebastopol: O'Reilly, 2013.

MOREIRA, D. R.; DIAS, M. S. **Web 2.0 – A Web Social**. CEPPG, n. 20, p. 196-208. 2009.

MORIMOTO, C. E. **Smartphones guia prático**. Porto Alegre: Sul Editores, 2009.

MOTTA, A. M. **Análise da Relação entre o Comportamento de Preço dos Produtos e a Participação Relativa de Mercado**. São Paulo: 2009.

MURUGESAN, S. Understanding Web 2.0. **Revista IT Professional**, v. 9, n. 4, p. 34-38, ago. 2007.

NOVAES, A. G. **Logística e Gerenciamento da Cadeia de Distribuição**: Estratégia, operação e avaliação. Rio de Janeiro: Campus, 2001.

NORDSTRÖM, M. A case study in social media mashup concept validation. 2010. 111 f. Dissertação (Mestrado em Engenharia) - School of Science and Technology, Alto University, Helsinki, 2010.

OGDEN, J. R.; CRESCITELLI, E. **Comunicação integrada de marketing**: conceitos, técnicas e práticas. 2. ed. São Paulo: Pearson Prentice Hall, 2007.

OLHAR DIGITAL. **Retrospectiva Android**: Relembre a história do sistema operacional do Google. 2013. Disponível em: <http://olhardigital.uol.com.br/produtos/central_de_videos/retrospectiva-android-relembre-a-historia-do-sistema-operacional-do-google>. Acesso em: 17 set. 2014.

OLIVEIRA, C. C. **Marketing digital**: um estudo exploratório sobre a utilização das mídias digitais como canal de comunicação. 2010. 127 f. (Graduação em Administração) - Faculdade Alvorada de Tecnologia e Educação de Maringá, Maringá.

OLIVEIRA, D. dos S. de. Publicidade abusiva na internet. **Jus Navigandi**, Teresina, v. 14, n. 2071, mar. 2009.

ORT, E.; BRYDON, S.; BASLER, M.. **Mashup Styles, Part 1: Server-Side Mashups**. 2007. Disponível em: <http://www.oracle.com/technetwork/articles/javaee/mashup-1-142202.html#>. Acesso em: 30 out 2014.

O'REILLY , T. **What is Web 2.0**. O´Reilly Media. 2005. Disponível em <http://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html>. Acesso em: 21 set. 2014.

O'REILLY, T. **Web 2.0 - Principles and Best Practices**. 2007. Disponível em <http://repo.mynooblife.org/.priv8/Ebook/Web%202.0%20Principles%20and%20Best%20Practices.pdf >. Acesso em: 16 ago. 2014.

PEREIRA, R. A.; BLUM, R. O. ***Direito eletrônico:*** **a internet e os tribunais**. Bauru: Edipro, 2001.

PINHEIRO, P. P. **Direito digital**. 2. ed. São Paulo: Saraiva, 2007.

POTTER, R.; TURBAN, E.; RAINER, K. **Administração de Tecnologia da Informação**. 3. ed. São Paulo: Campus, 2005.

RHODES, B.; GOERZEN, J. **Foundations of Python Network Programming**: The comprehensive guide to building network applications with Python. Nova York: Apress: 2012.

RIBEIRO, J. H. H. R.; SILVA JUNIOR, R. L.; WAISBERG, I. ***Comércio eletrônico***. São Paulo: Revista dos Tribunais, 2001.
RICHARDSON, L.; RUBY, S. **RESTful Web Services**. Sebastopol: O'Reilly, 2007.

ROJO, R. D. **Gestão de Marketing**. São Paulo: Saraiva, 2003.

SAMPAIO, C. **Web 2.0 e Mashups**: reiventando a internet. Rio de Janeiro: Brasport, 2007.

SANDBERG, R.; ROLLINS, M. **The Business of Android Apps Development**. 2. ed. Nova York: Apress, 2013.

SCHEMA.ORG. **What is Schema.org?.** schema.org. 2014. Disponível em: < http://schema.org/>. Acesso em: 22 out. 2014.

SHANKAR, V.; BOLTON, R. **An Empirical Analysis of determinants of Retailer Pricing Strategy.** Marketing Science, v. 23, n. 1, p. 28-49. 2004.

SHUEN, A. **Web 2.0**: A Strategy Guide. Sebastopol: O'Reilly, 2008.

SILBERSCHATZ, A.; GALVIN, P. B.; GAGNE, G. **Sistemas Operacionais com Java**. 6. ed. Rio de Janeiro: Campus, 2004.

SOUZA. J. N. **As Diferenças entre o desenvolvimento tradicional e o mobile**. 2011. 87 f. (Graduação em Processamento de Dados) - Faculdade de tecnologia de São Paulo, São Paulo.

SOUZA, R. A. A estética do mashup. 2009. 108 f. Dissertação (Mestrado em Tecnologias da Inteligência e Design Digital) – Processos Cognitivos e Ambientes Digitais, Pontíficia Universidade Católica de São Paulo, São Paulo, 2009.

STEFANOV, S. **Object-Oriented JavaScript**: Create scalable, reusable high-quality JavaScript applications, and libraries. Birmingham: Packt Publishing, 2008.

STEFFEN, R. **A influência do mix de marketing e dos fatores comportamentais nas decisões do consumidor:** O caso Sayuri produtos orientais. 2009. 104 f. Monografia (Bacharelado em Administração) - Universidade Federal de Santa Catarina. Florianópolis. 2009.

TAPSCOTT, D.; WILLIAMS, A. D. **Wikinomics – como a colaboração em massa pode mudar o seu negócio**. Rio de Janeiro: Nova Fronteira, 2007.

TIDWELL, D.; SNELL, J.; KULCHENKO, P. **Programing Web Services with SOAP**. Sebastopol: O'Reilly, 2001.

TORRES, C. **A Bíblia do Marketing Digital**. São Paulo: Novatec, 2009.

TURBAN, E.; WETHERBE, J.; MCLEAN, E. **Tecnologia da Informação para Gestão**. 3. ed. São Paulo: Bookman, 2002.

VARGIU, E.; URRU, M. Exploiting web scraping in a collaborative filtering-based approach to web advertising. **Artificial Intelligence Research,** Cagliari, v. 3, n. 2, p. 44-54, dez. 2012.

VIEDMA, C. Mobile Web Mashups. 2010. 80 f. Dissertação (Mestrado em Sistemas de Comunicação) – Sistemas de Comunicação, School of ICT, Estocolmo, 2010.

VOHRA, A.; VOHRA, D. **Pro XML development with Java Technology**. Nova York: Apress, 2006.

VRIEZE, P. de; XUA, L.; BOUGUETTAYA, A.; YANGC, J.; JINJUN, C. Building enterprise mashups. Future Generation Computer Systems, v. 27, n. 5, p. 637-642, mai, 2011.

W3C. Web Services Architeture. W3C. 2004. Disponível em: <http://www.w3.org/TR/ws-arch/>. Acesso em: 15 set. 2014.

WEBBER, J.; PARASTATIDIS, S.; ROBINSON, I. **REST in Practice**. Sebastopol: O'Reilly Media, 2010.

WEBSHOPPERS. **Evolução do comércio eletrônico**. E-bit. 2015. Disponível em: <http://www.ebit.com.br/webshoppers>. Acesso em: 01 jul. 2015.

YAGHMOUR, Karim. **Embedded Android**. Sebastopol: O'Reilly. 2013.

YEE, R. **Pro Web 2.0 Mashups:** Remixing Data and Web Services. Nova York: Apress, 2008.

ZENG, Z.; ZHANG, X. Research on Mobile E-commerce Information Search Approach Based on Mashup Technology. **International Journal of Business and Management,** Wuhan, v. 5, n.5, maio 2010.

ANEXOS

ANEXO A


```
{ "category" : { "concatenatecategoryname" : false,
      "hasoffer" : false,
      "id" : 77,
      "isfinal" : true,
      "links" : [ { "link" : { "type" : "category",
                "url" : "http://www.buscape.com.br/celular-e-
smartphone.html?mdsrc=9262544&mdapp=6598&mddtn=51922611"
              } },
           { "link" : { "type" : "xml",
                "url" :
"http://sandbox.buscape.com.br/service/findProductList/5a73794373456f6a4a6c633d
/br/?categoryId=77&keyword=smartphone&sourceId=9262544"
              } }
         ],
      "name" : "Celular e Smartphone",
      "parentcategoryid" : 0,
      "thumbnail" : { "url" : "http://imagem.buscape.com.br/bp5/categorias/77.jpg" }
    },
  "details" : { "applicationid" : "5a73794373456f6a4a6c633d",
      "code" : 0,
      "date" : { "day" : 8,
          "eonandyear" : { "lowestsetbit" : 0 },
          "hour" : 20,
          "millisecond" : 585,
          "minute" : 1,
          "month" : 5,
          "second" : 30,
          "timezone" : -180,
          "valid" : true,
          "xmlschematype" : { "localpart" : "dateTime",
```

```
            "namespaceuri" : "http://www.w3.org/2001/XMLSchema",
            "prefix" : ""
          },
        "year" : 2015
      },
    "elapsedtime" : 5,
    "message" : "success",
    "status" : "success"
  },
 "idpg" : "17",
 "offer" : [ { "offer" : { "categoryid" : 77,
         "id" : 149749027,
         "links" : [ { "link" : { "type" : "offer",
                 "url" :
"http://origimgbm.buscape.com.br/sandbox/developer/offer/?app=5a73794373456f6a
4a6c633d%20()&src=9262544&ptr=null"
               } } ],
         "offername" : "Smartphone Samsung Galaxy Young Plus TV S6293 Branco,
TV Digital, Android 4.1 - CD",
         "offershortname" : "Samsung Galaxy ",
         "price" : { "currency" : { "abbreviation" : "BRL" },
           "parcel" : { "interest" : 0,
             "number" : 10,
             "value" : "31.90"
            },
           "value" : "319.00"
          },
         "productid" : 583760,
         "seller" : { "advertiserid" : 0,
           "contacts" : [ { "contact" : { "label" : "Televendas",
                 "value" : "11-40035009"
               } } ],
```

```
            "cpcdifferentiated" : false,
            "id" : 174509,
            "istrustedstore" : true,
            "links" : [ { "link" : { "type" : "seller",
                  "url" : "http://www.commcenter.com.br"
                } } ],
            "oneclickbuy" : false,
            "oneclickbuyvalue" : 0,
            "pagamentodigital" : false,
            "rating" : { "ebitrating" : { "numcomments" : 1033,
                  "rating" : "Prata",
                  "ratingid" : "20",
                  "ratingnew" : "e-bit Boa"
                },
               "useraveragerating" : { "numcomments" : 1033,
                  "rating" : "4.0"
                }
              },
            "sellername" : "COMMCENTER",
            "thumbnail" : { "url" : "http://imagem.buscape.com.br/vitrine/logo174509.gif"
}
          },
         "thumbnail" : { "url" : "http://thumbs.buscape.com.br/T100x100/__2.174509-
8ecfd23.jpg" }
        } } ],
  "schk" : true,
  "totallooseoffers" : 0,
  "totalresultsavailable" : 1,
  "totalresultsreturned" : 1
}
```

APÊNDICES

APÊNDICE A – Artigo

MONITORAMENTO DE PREÇOS DE PRODUTOS DE LOJAS VIRTUAIS UTILIZANDO MASHUPS

Maria Eduarda Lavina[1]

[1]Universidade do Extremo Sul Catarinense - (UNESC)
Caixa Postal 3167 – 88806-000 – Criciúma – SC – Brazil

lavina.mariae@gmail.com

Abstract. *In a scenario where several retail companies offer similar products, the shops are increasingly looking for offering attractive prices to customers, creating a wide range of market prices, however, there is a lack of tools that help tracking these offers, much of which only compares prices on the web. The prototype was created based on mashups, and the validation was performed by monitoring up some products during a given period of time. The results indicate that there are a wide range of similar products prices between different shops and at the same store, within a period of time, showing the viability of the prototype, and it was found that the prototype is an effective tool in monitoring products prices at web shops.*

Resumo. *Em um cenário onde existem diversas empresas varejistas oferecendo produtos similares, as lojas procuram disponibilizar ofertas com preços atrativos aos clientes, criando uma grande variação de preços no mercado, porém, existe uma carência de ferramentas que fazem o acompanhamento destas ofertas. O protótipo foi criado à base de mashups e a validação do protótipo foi feita monitorando-se alguns produtos durante um período de tempo. Os resultados apontaram uma grande variação de preços de produtos similares entre as diversas lojas e em uma mesma loja, evidenciando a viabilidade do protótipo, constatando-se que o protótipo é uma ferramenta eficiente no acompanhamento de preços de produtos de lojas virtuais.*

1 Introdução

Devido à grande concorrência do mercado, é possível observar que existe uma grande dispersão de preços entre uma mesma marca e produto no universo de lojas do varejo (SHANKAR; BOLTON, 2004, tradução nossa). O comércio tipo *Business to Consumers* (B2C) é caracterizado pela sua alta volatilidade, pois a disponibilidade de sites e a oferta de produtos e serviços variam bastante (NOVAES, 2001). Consequentemente, os preços também oscilam fortemente em função da concorrência e das ofertas especiais.

Existem diversos serviços de comparação de preços, como por exemplo, sites como o Buscapé, Compare Preços e Já Cotei, que fazem comparações de preços em diversas lojas virtuais, sendo os usuários devem acessar o site e consultar os produtos para obter o resultado. Dentro do universo de aplicativos contidos na Google Play (2014), existe uma carência de aplicativos voltados ao m-commerce, ainda mais quando se restringe à pesquisa e

monitoramento de preços que não se utilizam de e-mail ou mensagens via celular para avisar o usuário da variação de preços de determinado produto.

Neste contexto, é de grande valia uma ferramenta que faça esse acompanhamento automaticamente, sendo que o usuário possa selecionar os produtos de seu interesse e a ferramenta faça o monitoramento destes produtos, avisando o usuário da queda do preço do produto.

Para construir uma ferramenta desta natureza, existem facilitadores como interfaces de programação de aplicação, do inglês Application Programming Interface (API), que permitem o acesso por terceiros a ofertas dos sites, como é o caso da API do Buscapé, que disponibiliza uma lista de ofertas contidas em sua base dados. Este tipo de serviço, onde uma aplicação busca informações em fontes de dados de terceiros é chamada de mashup e o acesso a essas fontes de dados pode ser feita de diversas formas, como web scraping, web services, widgets, entre outros, sendo o mashup uma maneira eficiente de criar novos serviços sem precisar iniciar aplicações do zero, além de serem reutilizáveis, leves, baseados na web e de rápida implementação (LIU et al, 2007, tradução nossa).

Esta ferramenta faz o acompanhamento de preços de produtos do *e-commerce,* auxiliando, inclusive, no acompanhamento de preços de grandes promoções, dando suporte ao usuário no momento da compra de um produto.

2 Comércio eletrônico

O comércio eletrônico ou *e-commerce* é toda a atividade desenvolvida na internet, com a utilização de ferramentas eletrônicas, que tem como principal objetivo a negociação de compra e venda de bens e serviços (GORDON; GORDON, 2006). O comércio voltado para o consumidor é caracterizado pela sua alta volatilidade, pois existe uma grande diversidade de *sites* e a oferta de produtos e serviços variam bastante (NOVAES, 2001), havendo uma grande variação de preços no mercado.

Como um subgrupo do *e-commerce* pode-se citar o *m-commerce,* que é a compra e venda de produtos e serviços por meio de dispositivos móveis (TURBAN; WETHERBE; MCLEAN, 2002). O relatório WebShoppers (2015), mostrou que o faturamento das transações realizadas por dispositivos móveis no Brasil em 2014 cresceu significativamente, em comparação com o ano de 2013, apresentando R$ 3,4 bilhões – diante dos R$ 1,13 bilhão no ano de 2013. De acordo com o WebShoppers (2015), com o crescimento do mercado móvel, vê-se a adoção do uso de telefones celulares e *tablets* pelos consumidores para consulta de informações de produtos, comparação de preços e compra usando esses dispositivos móveis.

Em meio a diversos fatores presentes no processo de decisão de compra, o preço tem forte influência nas intenções e satisfações de compra do cliente (STEFFEN, 2009). Segundo uma pesquisa realizada pelo Ibope Mídia (2012), 80% dos internautas brasileiros utilizam a *web* para comparar preços, sendo que estes utilizam de diversos artifícios, como agregadores de ofertas, entre outros.

Neste cenário, foi constatada uma carência de aplicativos para dispositivos móveis que fizessem o acompanhamento de ofertas previamente selecionadas pelo usuário, onde o mesmo seria avisado sobre a alteração de preço de um produto que seja do seu interesse.

3 Mashups

Um *web mashup* é uma página *web* que combina informações e serviços de múltiplas fontes disponíveis na *web*, de modo a criar novos serviços (MURUGESAN, 2007, tradução nossa) e se popularizou devido a *Web* 2.0, pois um de seus objetivos é reutilizar os dados, *web services* e micro aplicativos para criar aplicações híbridas (YEE, 2008, tradução nossa.

Mashups devem possuir um modelo de componentes bem definido, que encapsula os dados de várias fontes e manipula os recursos existentes na *web*, por meio de um padrão de serviços como, por exemplo REST, Atom / RSS, entre outros (LIU et al, 2007, tradução nossa).

Segundo Liu et al (2007, tradução nossa), os *mashups* possuem importantes características, descritas a seguir.

a) são reutilizáveis: cada bloco de construção *mashup* tem seu próprio contexto e lógica de negócios, sendo que, geralmente, são as API´s de *mashups* que contém a maior parte da lógica comercial, tornando os serviços *mashup* mais reutilizáveis;
b) baseado na *web*: *mashups* utilizam Javascript, *web services* e *feeds* baseados diretamente HTTP, usando JSON e XML para recuperação de dados;
c) aplicações leves: um *mashup* é a integração de múltiplas fontes de dados e conteúdo em um único local e, pelo fato de *mashups* utilizarem dados e serviços de *sites* e aplicativos da *web*, eles são de leve e fácil implementação, reduzindo custos de desenvolvimento e proporcionando maior satisfação do usuário.

Zeng e Zhang (2010, tradução nossa) afirmam que a combinação de dados ou serviços disponíveis na *web* e recursos de redes móveis permite que os usuários de dispositivos móveis possam desfrutar de uma ótima experiência quando se trata da busca de informações no *e-commerce*.

3.1 Tipos de mashup

Os *mashups* podem ser classificados em três tipos: *mashups* de dados, *mashups* de consumo, que é a categoria mais comum de *mashups*, e *mashups* de negócios, que vem se tornando muito populares (VIEDMA, 2010, tradução nossa).

Mashups de dados combinam múltiplas fontes de dados, que devem ser de tipos similares, em uma única representação, que irá fornecer um novo serviço que não estava disponível originalmente (VIEDMA, 2010, tradução nossa).

Já os *mashups* de consumo são projetados para o público em geral e, normalmente, combinam múltiplas fontes de diferentes tipos de dados em uma representação visual, provando ser um meio muito eficaz de personalização de dados, de acordo com as necessidades do cliente (VIEDMA, 2010, tradução nossa).

O terceiro tipo de *mashup* é o de negócios, que são semelhantes aos *mashups* de consumo, porém, com o objetivo de resolver um problema de negócio.

3.2 Camadas do mashup

Segundo Merril (2009, tradução nossa), são três as camadas que compõem um *mashup*, que estão lógica e fisicamente separadas: o provedor de conteúdo, o *site mashup* e o navegador do cliente.

O *site mashup* é o lugar onde o *mashup* está hospedado, mas não é necessariamente onde ele é executado; o navegador do cliente é o lugar onde a aplicação é criada graficamente e a interação do usuário ocorre; o provedor de conteúdo é a fonte que disponibiliza o conteúdo utilizado pelo *mashup* (MERRIL, 2009, tradução nossa).

O método mais comum para se obter dados a partir de um *web site* é por meio de uma interface de programação de aplicação (API), que é projetada especificamente para facilitar a comunicação entre os programas, porém, existem outras alternativas para recuperar informações de *web sites* como, por exemplo, o *screen scraping* (YEE, 2008, tradução nossa), programação HTTP, análise de modelos de objetos de documento, do inglês *Document Object Model* (DOM) e *parsers* de linguagem de marcação de hipertexto, do inglês *HyperText Markup Language* (HTML) (VARGIU; URRU, 2012, tradução nossa).

3.3 Arquitetura do mashup

De um ponto de vista arquitetônico, existem dois estilos de *mashups*: os baseados nos clientes, do inglês *client-based*, e os baseados em servidores, do inglês *server-based*, sendo que é possível ter ainda um terceiro estilo, que é a arquitetura móvel, onde ambos os estilos, *client-based* e *server-based*, são combinados aproveitando as vantagens de cada um (VIEDMA, 2010, tradução nossa).

Como o nome indica, *mashups* baseados em servidor realizam a integração de dados e serviços no servidor (ORT; BRYDON; BASLER, 2007, tradução nossa), onde o servidor pode pré processar todas as informações coletadas, reduzindo o tamanho dos dados, mudando o formato ou combinando-o com os outros dados, sendo também mais fácil lidar com os requisitos de segurança.

Em uma arquitetura baseada em cliente a integração de serviços e conteúdos é feita no lado do cliente, onde o navegador do cliente é o encarregado de reunir o conteúdo, transformando-o para o formato correto antes de mostrá-lo na tela. Existem várias razões para usar o estilo baseado no cliente (ORT; BRYDON; BASLER, 2007, tradução nossa), pois, segundo os autores, nesta arquitetura a implementação dos *mashups* pode se tornar mais simples do que na arquitetura baseada em servidor.

4 Protótipo de monitoramento

O projeto constituiu-se do desenvolvimento de um protótipo de aplicativo para o sistema operacional Android, que permite selecionar e monitorar os preços de produtos de lojas do *e-commerce,* por meio de dispositivos móveis.

Foram utilizados conceitos bastante atuais, como os *mashups*, e diversas tecnologias, como *web services* REST e o formato de representação JSON, aliando-se à plataforma *mobile*, com o sistema operacional Android. Buscou-se utilizar recursos nativos da plataforma Google Android, incorporando outros recursos pertinentes ao desenvolvimento das funcionalidades do protótipo.

O desenvolvimento do protótipo constituiu-se de algumas etapas bem definidas, sendo que, inicialmente, foi feita uma análise da API utilizada no *mashup*, para levantamento dos dados retornados pela representação em JSON e priorização e definição dos dados que seriam utilizados pelo protótipo.

Após, foi definida a arquitetura do *mashup,* levando em consideração os estudos realizados na etapa de levantamento bibliográfico e, com estas informações, foi possível definir a arquitetura mais apropriada para o protótipo. Depois de definir os dados a serem

utilizados no *mashup* e a arquitetura do *mashup,* se iniciou, efetivamente, o desenvolvimento do protótipo, aplicando todos os conceitos e tecnologias necessários, que foram descritos no levantamento bibliográfico.

A etapa de homologação incluiu todo o conjunto de testes do protótipo e homologação do mesmo pela empresa Buscapé, processo este que é indispensável para a utilização do ambiente oficial da API.

4.1 API do Buscapé

A Buscapé, conhecida principalmente pelo seu buscador de ofertas, disponibiliza uma API que permite que seus usuários tenham acesso as informações contidas no site Buscapé. Esta API consiste em um *web service* REST, onde são disponibilizados recursos que permitem que o usuário requisite ofertas e produtos contidos na base de dados do Buscapé. As requisições ao *web service* são feitas por meio de uma URI, sendo que devem ser enviados alguns parâmetros obrigatórios, assim como também podem ser enviados parâmetros opcionais.

Para utilizar a API do Buscapé, é necessário criar uma conta de desenvolvedor no site http://developer.buscape.com/pt/, e cadastrar o serviço que irá utilizar a API, sendo que no caso deste projeto, foi criado um serviço *mobile*. Com este cadastro realizado, será criado um identificador para a aplicação, sendo que este identificador é único e deverá ser usado em todas as requisições realizadas aos recursos da API.

Na fase de desenvolvimento do protótipo, o Buscapé disponibiliza uma base de dados de testes, sendo esta base de dados acessível por meio do endereço http://sandbox.buscape.com. Após a homologação do aplicativo, esta URL passa a ser http://bws.buscape.com.

Existem parâmetros que podem ser utilizados para filtrar os dados a serem retornados, assim como parâmetros para identificar o tipo de retorno, entre outros. A API disponibiliza dois tipos de representação dos dados requisitados, sendo eles XML e JSON. O protótipo utiliza a representação em formato JSON, pois este formato é mais leve e se torna mais eficiente na troca de dados para dispositivos móveis. O JSON retornado contém diversas informações referentes à oferta, como por exemplo, código do produto no Buscapé, preço, descrição do produto, loja anunciante, entre outras informações

4.2 Arquitetura

A API do Buscapé é um dos vários exemplos da internet colaborativa, ou *Web* 2.0, sendo que esta API é usada para que se possa criar e disponibilizar novos conteúdos na *web*. Este processo de utilizar recursos disponibilizados por terceiros para criar novos serviços na *web* é conhecido como *mashup*.

O protótipo é composto por um *mashup* baseado na *web*, já que utiliza o serviço fornecido pela Buscapé, por meio de *web services*, para recuperar informações de fontes de dados de terceiros. Ele apresenta as três principais camadas citadas por Liu et al (2007, tradução nossa), sendo que a camada de *interface* de usuário é representada pelos componentes disponibilizados pelo Android, podendo citar os botões, os campos de texto e as imagens; a camada de serviço é representada pela API da Buscapé, que é acessada por meio de *web services* REST, onde se recupera as informações desejadas; e a camada de ação é representada pelas classes onde são feitos os relacionamentos entre a camada de serviço e a camada de *interface*.

O protótipo Meu Preço caracteriza um *mashup* de consumo, já que combina dados distintos de uma fonte de dados e tem foco na visualização do conteúdo pelo usuário final, utilizando-se de API´s de terceiros para compor o *mashup*.

Dentre as arquiteturas de um *mashup*, optou-se por utilizar a arquitetura *client-based*, onde o processamento do conteúdo é feito no navegador do cliente, pois, segundo Ort, Brydon e Basler (2007, tradução nossa), a implementação dos *mashups* se torna muito mais simples quando utilizado este tipo de arquitetura.

4.3 O protótipo

O protótipo foi construído de forma que permita ao usuário pesquisar e monitorar os preços das ofertas selecionadas. Ao pesquisar ofertas para serem monitoradas, o protótipo faz uma requisição ao recurso disponibilizado pelo Buscapé, que retorna um JSON utilizado para montar uma lista com todas as ofertas retornadas pela API, mostrando informações que são pertinentes ao usuário, criando assim, um *mashup* com as informações solicitadas.

As ofertas são monitoradas por meio dos identificadores dos produtos de cada oferta, que são únicos e provenientes do *mashup* com a API do Buscapé, não sendo alterados nas atualizações de preços ou de qualquer outra informação da oferta.

Caso o preço da oferta tenha sido alterado, uma rotina irá identificar se o o produto ficou mais caro ou mais barato. Se ficou mais caro, o protótipo irá mostrar uma imagem indicando que o preço aumentou, na tela de Ofertas. Porém, se ficou mais barato, o protótipo irá mostrar uma imagem indicando que o preço diminuiu, na tela de Ofertas, e também enviará uma notificação ao dispositivo, avisando sobre a queda do preço, onde nesta notificação o usuário é informado sobre qual produto ficou mais barato, o preço antigo do produto e o novo preço.

O protótipo permite que o usuário visualize um gráfico com o histórico recente da oferta selecionada, sendo que para o desenvolvimento deste gráfico foi utilizada a biblioteca core do Android Plot, versão 0.6.1, onde é possível informar valores para o eixos X e Y.

4.4 Testes

Os testes foram feitos tanto em ambientes simulados, como no ambiente de testes do Buscapé e no próprio ambiente de produção. Para os testes simulados foi criado um serviço denominado AppService, que simula a API do Buscapé, podendo realizar testes de naturezas distintas, incluindo diversas situações que podem ocorrer com o protótipo.

O AppService disponibiliza dados semelhantes ao *mashup* feito com a API do Buscapé, se comportando como um emulador da API. Este recurso simula o serviço disponibilizado pelo Buscapé, com o objetivo de testar certas funcionalidades do protótipo que não seriam possíveis no ambiente de testes do Buscapé, como a alteração de preços de um produto em específico em um determinado momento.

Este serviço é composto por um *web service* que, quando requisitado, retorna um JSON com diversas ofertas que foram previamente cadastradas em uma base de dados. Esta base de dados é fictícia, onde são armazenadas diversas ofertas também fictícias, simulando a base de dados do Buscapé.

4.5 Resultados

Foram monitorados, durante dez dias, quatro produtos distintos em diferentes lojas virtuais, totalizando vinte ofertas acompanhadas. Estas ofertas tiveram grande variação de preços durante o período monitorado e em alguns dos produtos foram observadas alterações sutis de preço em alguns momentos, sendo que mesmo que eles fiquem mais baratos do que a oferta do dia anterior, este valor é mínimo, algumas vezes se tratando de apenas R$1,00. Também foi observado que alguns produtos tiveram seu preço aumentado substancialmente em um dia e no dia seguinte este preço foi diminuído novamente, não caracterizando, muitas vezes uma boa oportunidade de compra, se o histórico de preços for levado em consideração.

5 Conclusão

Atualmente, o mercado apresenta um cenário de grande concorrência entre lojas do varejo, sendo que os clientes estão sempre em busca de qualidade e melhores preços. Movidos pela demanda do mercado, os lojistas procuram sempre oferecer produtos com preços atrativos ao cliente, gerando uma grande variação de preços entre lojas e produtos. Para encontrar o melhor preço, os clientes utilizam diversos artifícios, como, por exemplo, pesquisas na internet, que são feitas muitas vezes por dispositivos móveis. O universo móvel está em constante expansão, tendo cada vez mais usuários e recebendo fortes investimentos por parte das empresas.

A procura do cliente por melhores preços, constatada com a fundamentação teórica, foi o que motivou o desenvolvimento deste trabalho, onde foi proposto um protótipo que realizasse o monitoramento de preços de produtos selecionados pelo usuário. Para alcançar este objetivo, propôs-se que o protótipo utilizasse *mashups*, onde conteúdos já existentes na *web* fossem utilizados na construção do mesmo. O conceito estudado sobre *mashups* foi aplicado por meio da definição da arquitetura do protótipo e da API utilizada para busca das ofertas na *web.*A utilização de *mashups* simplificou significativamente o desenvolvimento do protótipo, pois este teve uma base de dados de ofertas já existente e ampla à disposição para que fosse feito o tratamento dos dados. A construção do *mashup* mostrou que existem diversas formas de desenvolver aplicativos e recursos utilizando conteúdos já existentes, tornando a implementação mais ágil e com diversas possibilidades de reutilização, sendo que esta técnica é, inclusive, usada em diversos serviços já existentes.

A mobilidade do protótipo também foi um dos objetivos do trabalho, alcançado por meio do uso da plataforma *mobile* Google Android, que fez com que o protótipo possa ser usado em *smartphones*, *tablets*, entre outros, já que foi constatado que o mercado *mobile* está em constante expansão. Com o protótipo, o usuário pode selecionar os produtos que forem do seu interesse e monitorar os preços apenas destes produtos, evitando propagandas abusivas e gravação de dados pessoais da sua navegação pela internet.

Os resultados do monitoramento das ofertas selecionadas mostraram que existe uma grande variação de preços de produtos, não só entre as lojas varejistas, como também na mesma loja, onde o preço de um determinado produto era, eventualmente, alterado diversas vezes em uma mesma semana. Verificou-se também que nem sempre uma queda no preço indica que o produto ficou mais barato, pois houveram casos onde o preço de um produto variava entre maior e menor, sendo que este histórico é mostrado no gráfico disponibilizado pelo protótipo. Como o protótipo mantém um histórico das ofertas acompanhadas, foi observado que ele pode auxiliar também na identificação de boas ofertas em grandes promoções, como por exemplo a Black Friday, onde o usuário possa ter certeza de que está

realmente pagando menos pelo produto desejado.

Com o desenvolvimento deste trabalho, foi possível entender melhor o mercado *mobile* e como as pessoas procuram sempre o melhor preço, buscando cada vez mais a economia. Houve um grande aprendizado sobre técnicas de extração de dados da *web* e desenvolvimento *mobile* em Android. Desta forma, foi possível observar que *mashups* são soluções bastante simples e muito úteis, podendo ser criados por meio de mais de uma técnica de extração de dados.

Observa-se que, caso existisse uma padronização na forma de representação dos dados disponibilizados pelas API´s, poderiam ser utilizadas, de forma transparente, diversas fontes de dados para a pesquisa de ofertas na *web*. Com base na análise dos resultados, constatou-se que o protótipo é uma ferramenta eficiente no acompanhamento de preços de produtos de lojas virtuais, auxiliando o usuário na tomada de decisão de compra de um produto do seu interesse, alcançando, desta forma, os objetivos propostos. O protótipo não grava informações de preferências de ofertas dos usuários e não disponibiliza nenhuma informação do usuário para terceiros, garantindo a privacidade do mesmo na utilização do aplicativo. Dentre as dificuldades encontradas, pode-se citar a criação dos layouts do protótipo e a fundamentação teórica sobre *mashups,* sendo que não existem muitos livros que falam sobre o assunto.

Referências

GOOGLE PLAY. Loja. Google Play. 2014. Disponível em: <https://play.google.com/store?hl=pt_BR>. Acesso em: 05 out. 2014.

GORDON, S. R.; GORDON, J. R. Sistemas de informação: uma abordagem gerencial. Rio de Janeiro: LTC, 2006.

IBOPE MÍDIA. 80% dos internautas usam web pra comparar preços. Bahia Econômica. 2012. Disponível em < http://www.bahiaeconomica.com.br/noticia/16100,80-dos-internautas-usam-web-para-comparar-precos-mostra-pesquisa.html>. Acesso em 29 jun. 2015.

LIU, X.; HUI, Y.; SUN, W.; LIANG, H. Towards service composition based on mashup. In: IEEE Congress on Services, Salt Lake City, p. 332–339, jul. 2007.

MERRILL, D. Mashups: The new breed of Web app - An introduction to mashups. IBM. 2009. Disponível em: <http://www.ibm.com/developerworks/library/x-mashups/>. Acesso em: 15 set. 2014.

MURUGESAN, S. Understanding Web 2.0. Revista IT Professional, v. 9, n. 4, p. 34-38, ago. 2007.

NOVAES, A. G. Logística e Gerenciamento da Cadeia de Distribuição: Estratégia, operação e avaliação. Rio de Janeiro: Campus, 2001.

ORT, E.; BRYDON, S.; BASLER, M.. Mashup Styles, Part 1: Server-Side Mashups. 2007. Disponível em: <http://www.oracle.com/technetwork/articles/javaee/mashup-1-142202.html#>. Acesso em: 30 out 2014.

SHANKAR, V.; BOLTON, R. An Empirical Analysis of determinants of Retailer Pricing Strategy. Marketing Science, v. 23, n. 1, p. 28-49. 2004.

STEFFEN, R. A influência do mix de marketing e dos fatores comportamentais nas decisões do consumidor: O caso Sayuri produtos orientais. 2009. 104 f. Monografia (Bacharelado em Administração) - Universidade Federal de Santa Catarina. Florianópolis. 2009.

TURBAN, E.; WETHERBE, J.; MCLEAN, E. Tecnologia da Informação para Gestão. 3. ed. São Paulo: Bookman, 2002.

VARGIU, E.; URRU, M. Exploiting web scraping in a collaborative filtering-based approach to web advertising. Artificial Intelligence Research**,** Cagliari, v. 3, n. 2, p. 44-54, dez. 2012.

VIEDMA, C. Mobile Web Mashups. 2010. 80 f. Dissertação (Mestrado em Sistemas de Comunicação) – Sistemas de Comunicação, School of ICT, Estocolmo, 2010.

WEBSHOPPERS. Evolução do comércio eletrônico. E-bit. 2015. Disponível em: <http://www.ebit.com.br/webshoppers>. Acesso em: 01 jul. 2015.

YEE, R. Pro Web 2.0 Mashups: Remixing Data and Web Services. Nova York: Apress, 2008.

ZENG, Z.; ZHANG, X. Research on Mobile E-commerce Information Search Approach Based on Mashup Technology. International Journal of Business and Management, Wuhan, v. 5, n.5, maio 2010.