UNIVERSIDADE DO EXTREMO SUL CATARINENSE – UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

LUANA GOMES SILVA

PROTÓTIPO DE APLICATIVO ANDROID PARA EXTRAÇÃO DE TEXTO EM IMAGENS PARA BUSCA SEMÂNTICA SOBRE RÓTULOS DE CERVEJAS

CRICIÚMA

2015

LUANA GOMES SILVA

PROTÓTIPO DE APLICATIVO ANDROID PARA EXTRAÇÃO DE TEXTO EM IMAGENS PARA BUSCA SEMÂNTICA SOBRE RÓTULOS DE CERVEJAS

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

Orientador: Prof. MSc. Gustavo Bisognin

Coorientador: Prof. Esp. William Bertan da Silva

CRICIÚMA

2015

LUANA GOMES DA SILVA

PROTÓTIPO DE APLICATIVO ANDROID PARA EXTRAÇÃO DE TEXTO EM IMAGENS PARA BUSCA SEMÂNTICA SOBRE RÓTULOS DE CERVEJA

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 Projeto de Software.

Criciúma, 26 de novembro de 2015.

BANCA EXAMINADORA

Prof. Gustavo Bisognin - Mestre - (UNESC) – Orientador

Prof. William Bertan da Silva- Especialista - (UNESC) – Coorientador

Prof. Evânio Ramos Nicoleit - Mestre - (UNESC)

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

Dedico este projeto para minha Família que tanto me motivou e investiu em mim, e também a meus amigos que me inspiravam e deram forças para continuar.

AGRADECIMENTOS

Agradeço primeiramente e principalmente aos meus pais, Luciene e Enéias, por não medirem esforços para a minha jornada acadêmica desde o inicio, pela motivação e força que me passaram ao longo desse período, por me guiar nesse caminho e me ensinar a cada momento o valor do esforço e dos estudos.

Agradeço também ao meu namorado e sua família, pela compreensão dos momentos que não pude estar presente e também dos momentos de mau humor, além do carinho, motivação e ajuda prestada.

Meu agradecimento a todos os professores que tive durante o curso, em especial ao professor, orientado, por aceitar o desafio, me guiar e por não me fazer desistir nos momentos mais difíceis que tive durante este período, meu muito obrigado. Agradeço muito ao meu co-orientador Wiliam Bertam por acreditar que seria possível o desenvolvimento deste projeto em sua área, por me guiar e também por aceitar o desafio de auxiliar este trabalho.

“Cada sonho que você deixa pra trás, é um pedaço do seu futuro que deixa de existir.”

Steve Jobs

RESUMO

O mercado das cervejas artesanais está crescendo tanto em nível nacional quanto a nível internacional, e tem se destacado também em Santa Catarina, fazendo com que surjam novos produtos a todo instante, nos quais os consumidores pouco sabem a respeito. Para auxiliar o crescimento deste ramo e fornecer informações relevantes ao consumidor, este trabalho propõe o uso da tecnologia móvel, por ampliar o compartilhamento instantâneo de informações, ajudando a difundi-las. Devido a popularização no uso de dispositivos móveis, surgiram diversos sistemas operacionais, dentre eles o Android, da Google. Líder no mercado e de código aberto, o Android oferece diversas funcionalidades práticas, inclusive acesso à câmera, tornando possível realizar a leitura de texto em uma imagem, utilizando-se de bibliotecas que realizam o Reconhecimento Óptico de Caracteres (OCR). Através do reconhecimento do texto extraído do rótulo de cervejas, torna-se possível a formação de um contexto para realizar a busca semântica das informações da cerveja no Google, utilizando-se de uma requisição realizada a um serviço fornecido pelo próprio Google, conhecido como Custom Search. Baseado neste contexto, este trabalho apresenta um protótipo que realiza a comunicação entre plataformas diferentes, possuindo aplicações cliente e servidor. A aplicação servidora é responsável por realizar a busca de informações através do nome do produto reconhecido utilizando OCR, e disponibilizando ao usuário o acesso a estas informações. Ao obter as informações do produto, é possível o compartilhamento em redes sociais das informações obtidas, utilizando-se do protótipo cliente em um *smartphone*. Esta comunicação entre cliente e servidor, é realizada por requisições a um web service RESTful disponível no servidor. Para isto, foi realizado um estudo das tecnologias e bibliotecas, a fim de entender o funcionamento destas. Em testes efetuados no protótipo foi analisado que é possível obter informações relevantes do produto desejado, e auxiliar em sua popularidade, fazendo o uso do compartilhamento em redes sociais populares, tais como Facebook e mensageiros instantâneos, tais como o WhatsApp.

Palavras-chave: Dispositivos móveis. Indústria cervejeira. Web service RESTful. Busca semântica.

ABSTRACT

The market for craft beers is growing both nationally and internationally, and has also excelled in Santa Catarina, causing the emergence of new products all the time, in which consumers know little about. To assist the growth of this sector and provide relevant information to the consumer, this paper proposes the use of mobile technology by expanding the instant sharing of information, helping to disseminate them. Due to the popularity in the use of mobile devices, there have been multiple operating systems, including Android, from Google. Leader in the market and open source, Android offers several practical features, including access to the camera, making it possible to perform the reading of text in an image through the use of libraries that perform Optical Character Recognition (OCR). By recognizing the extracted text from beer labels, it becomes possible to build a context to perform the semantic search of the beer information t in Google, through a request made to a service provided by Google itself, known as Custom Search. Based on this context, this paper presents a prototype that performs communication between different platforms, having client and server applications. The server application is responsible for performing the search for information through the name of the recognized product using OCR, and providing the user access to this information. When product information is obtained, it is possible the sharing of this information on social networks, by using the client prototype on a smartphone. This communication between client and server is performed by requests to a RESTful web service available on the server. For this, a study of technologies and libraries was conducted in order to understand how they work. In performed tests on the prototype, it was analyzed that it is possible to obtain relevant information from the desired product, and assist in its popularity, making use of the sharing on popular social networks such as Facebook and instant messaging such as WhatsApp.

Keywords: Mobile Devices. Brewing industry. RESTful web service. Search semantics.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DTDs	Document Type Definitions
HAL	Hardware abstraction layer
IDC	International Data Corporation
JSON	JavaScript Object Notation
OCR	Optical Character Recognition
OHA	Open Handset Alliance
RDF	Resource Description Framework
REST	Representational State Transfer
RPC	Remote Procedure Call
SGML	Standard Generalized Markup Language
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
URI	Uniform Resource Identifiers
URLs	uniform Resource Locators
URNs	uniform Resource Names
WADL	Web Application Description Language
WS	Web Services
WSDL	Web Services Description Language
XML	Extensible Markup Language

SUMÁRIO

1 INTRODUÇÃO

A indústria cervejeira é apresentada nos seus processos produtivos, matérias-primas e tipos de cerveja. Além disso, a estrutura do mercado mundial e nacional é descrita, destacando o dinamismo da indústria e o aumento, nos últimos anos, do número de microcervejarias no Brasil e em países como os Estado Unidos, sendo essas pequenas cervejarias pertencentes a um crescente segmento dentro da indústria cervejeira. Além disso, o mercado brasileiro de cerveja vem crescendo nos últimos anos, e, dentro desse mercado, as cervejas sofisticadas começam a ganhar importância, com destaque pelas produzidas por cervejarias artesanais. Santa Catarina se destaca nacionalmente por esse “movimento microcervejeiro”, sendo que algumas de suas microcervejarias têm reconhecimento nacional e mundial, por produzirem cervejas de qualidade superior (CUNHA, 2011).

Dentro desse contexto e dado o alto número de diferentes tipos de produtos com o qual o consumidor tem que lidar, surge a necessidade de obter informações sobre estes produtos, que podem ser retiradas muitas vezes de seus próprios rótulos.

Considerando o crescente uso de dispositivos como “smartphones”, surgiram diversos sistema operacionais para estes, dentre eles o Android, que é líder no mercado e desenvolvida pelo Google. Com algumas funcionalidades fornecidas pelo Android, como a câmera, é possível a automatização da pesquisa do rótulo desconhecido.

No entanto, muitos dos recursos de dispositivos móveis ainda exigem uma intervenção manual por parte de seus utilizadores. Um bom exemplo disso é a realização de pesquisas que muitas vezes requerem a descrição em formato de texto do que será pesquisado.

Para automatizar a entrada de dados nestes dispositivos, este projeto propõe utilizar softwares de reconhecimento óptico de caracteres em imagens, também chamados de OCR, que serão responsáveis por extrair as informações contidas em um rótulo, permitindo que seja realizada correção manual ou adição de informações.

O processo de *Optical Character Recognition* (OCR) que consiste em separar os caracteres das outras informações que estão contidas na imagem para obter somente o texto desta (SOUZA, 2011).

Normalmente essas informações são compostas por figuras (gráficos, diagramas, fotos, gravuras, e outros) e texto. É importante identificar e classificar essas informações para que o OCR melhore seu desempenho, pois assim não perderá tempo tentando interpretar áreas sem informação textual (ALVES, 2003).

Dado a importância do compartilhamento de informações referente a determinados segmentos ou produtos, este projeto permitirá compartilhar através de um aplicativo os dados extraídos por OCR, uma mensagem, a própria foto do produto, e informações adicionais trazidas através de busca semântica na Web, permitindo assim a divulgação e valorização da indústria nacional de cervejas especiais.

1.1 OBJETIVO GERAL

Desenvolver um protótipo para busca semântica de informações através da extração de texto do rótulo da cerveja com o uso de reconhecimento óptico de caracteres para o sistema operacional Android.

1.2 OBJETIVOS ESPECÍFICOS

O trabalho proposto baseia-se em tais objetivos específicos:

a) realizar um estudo do sistema operacional Android;
b) analisar API’s e aplicativos do framework do Android;
c) realizar um estudo da tecnologia OCR;
d) conceituar web semântica, e explicar sua estrutura;
e) analisar a implementação de busca semântica;
f) desenvolver um protótipo na plataforma Android;
g) efetuar busca semântica das informações da cerveja;
h) testar protótipo desenvolvido.

1.3 JUSTIFICATIVA

Com o crescimento do mercado de cervejas no Brasil, cresce também sua diversidade, assim como o número de cervejarias e microcervejarias que a produzem.

Mesmo com uma pequena produção em comparação a uma grande cervejaria, o seu produto se torna competitivo, com uma boa aceitação no mercado nacional (CUNHA, 2011).

As microcervejarias colocam no mercado cervejas diferenciadas e acabam conquistando um número de consumidores atraídos pela qualidade e criatividade dessas novas empresas (CUNHA, 2011).

Com a injeção de tantas cervejas importadas e artesanais no mercado brasileiro, os consumidores acabam comprando cervejas que não conhecem, ou que dispõem de pouca informação a respeito.

Para ajudar e dar praticidade ao processo de busca de informações sobre tais produtos, a digitalização do texto contido no rótulo com a utilização do OCR torna-se muito viável.

A utilização de OCR ajuda de modo geral, na digitalização e a extração de informações de um documento de forma pratica e possibilita o acesso ao seu conteúdo de maneira rápida e objetiva, mantendo fidelidade aos documentos originais (SILVA; THOMÉ, 2007).

A busca semântica ajudara no melhor refinamento das informações relevantes sobre a cerveja.

Com o desenvolvimento deste protótipo será possível extrair alguns dados dos rótulos de forma automática, permitindo aos consumidores o acesso a estas informações, onde possam indicar suas preferências e buscar informações relacionadas, incentivando este tipo de indústria e dando espaço e oportunidade a pequenas produtoras que possuam um produto de qualidade.

1.4 ESTRUTURA DO TRABALHO

Este projeto de pesquisa tem sua estrutura dividia em 8 capítulos, no primeiro capítulo tem-se uma breve descrição na introdução sobre o tema da pesquisa, assim como demonstra o objetivo geral e os objetivos específicos do projeto, por fim apresenta a justificativa para o qual o projeto foi realizado.

No segundo capítulo traz o estudo levantado sobre os mecanismos de busca e sua evolução e as características deste, por fim traz uma breve história do motor de busca Google, sua evolução e o quanto o mesmo está revolucionando em suas tecnologias, utilizando de semântica para melhorar seus resultados de pesquisa.

No terceiro capítulo traz uma breve introdução do que é semântica, para logo após definir a web semântica e explicar a sua estrutura, bem como as tecnologias que a mesma aborda.

No quarto capítulo foi fundamentada a tecnologia mobile, sua importância, devido ao grande uso dos *smartphones,* e explicado a estrutura de como funciona o desenvolvimento na plataforma Android, líder no mercado deste ramo.

O capítulo 5 aborda a comunicação entre sistemas híbridos e explica como a mesma pode ser realizada através de *Web Service* (WS). Depois explica os padrões e as tecnologias que os WS utilizam.

No capitulo 6 foi definido a tecnologia de Reconhecimento óptico de Caracteres (OCR), demonstrando o seu processo e as etapas que a mesma utiliza para se extrair um texto de uma imagem.

O capítulo 7 trata dos trabalhos relacionados aos assuntos deste projeto, mostrando o que os mesmos abordam direta ou indiretamente de alguns objetivos semelhantes a este, e como tais projetos de diferentes objetivos utilizam das mesmas tecnologias que este projeto para alcançar seus objetivos.

Para finalizar, o capítulo 8 apresenta o resultado obtido com a presente pesquisa levantada, e explica a solução para o problema justificado, que se trata do protótipo que foi desenvolvido, além de apresentar como o protótipo foi construído, com quais tecnologias e como as mesmas foram utilizadas para chegar até o protótipo final.

2 MECANISMOS DE BUSCA

Cada vez mais a Web é utilizada como fonte de pesquisa, por possuir informações para praticamente qualquer área, seja de pesquisa, entretenimento ou notícias, estando estas facilmente disponíveis.

De acordo com Cedón (2001), existem praticamente dois tipos de ferramenta de busca na WEB, os diretórios e os motores de busca, esses deram origem ao outros que surgiram.

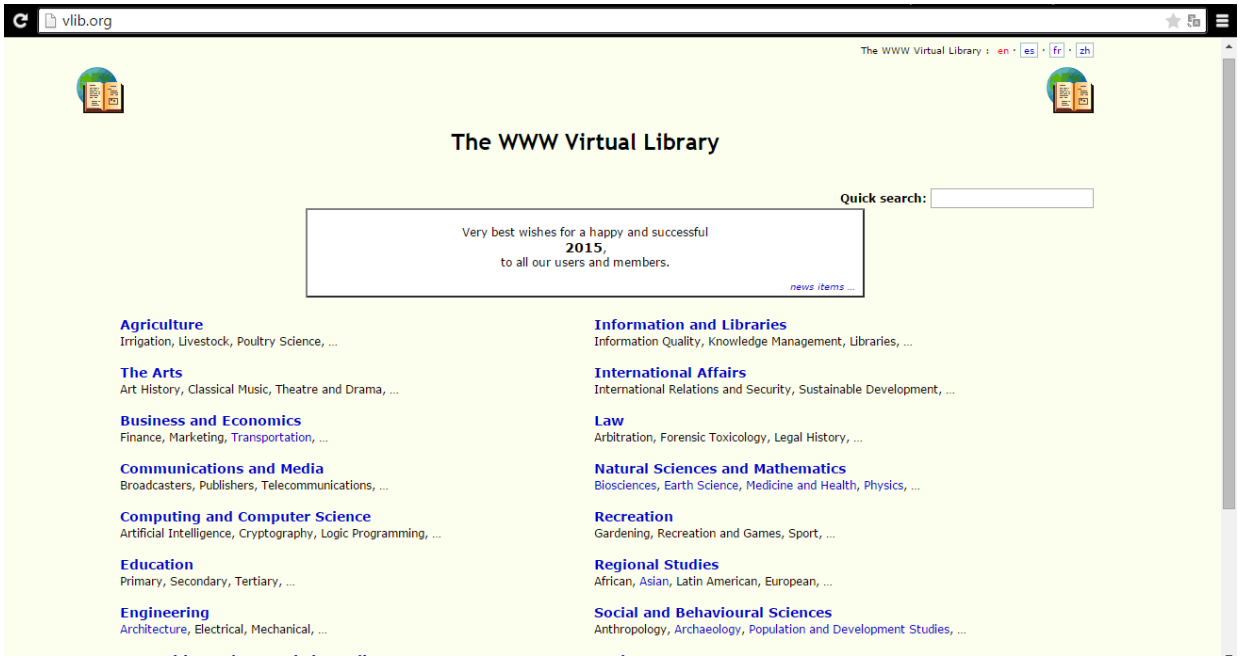
2.1 DIRETÓRIOS

Essas ferramentas foram iniciadas ainda quando havia pouco conteúdo na web, caracterizada por uma organização hierárquica dos sites de sua base em categorias ou subcategorias de seus assuntos (CENDÓN, 2001).

Segundo Branski (2001), para serem catalogados os assuntos dos sites, estes passam por uma seleção manual de pessoas, geralmente editores, que recebem os cadastros dos sites interessados com uma breve descrição de seu conteúdo, tais editores podem ou não aceitar esse cadastro, ao julgarem catalogam como acharem adequados.

De acordo com Cedón (2001), um dos pioneiros surgiu em novembro de 1992 a *The Word Wide Web Virtual Library*, na figura 1 é possível verificar a característica de diretórios nesta ferramenta.

Figura 1 - Exemplo de mecanismo de busca por Diretório.



Fonte: Virtual Library (2015).

2.2 MOTORES DE BUSCA

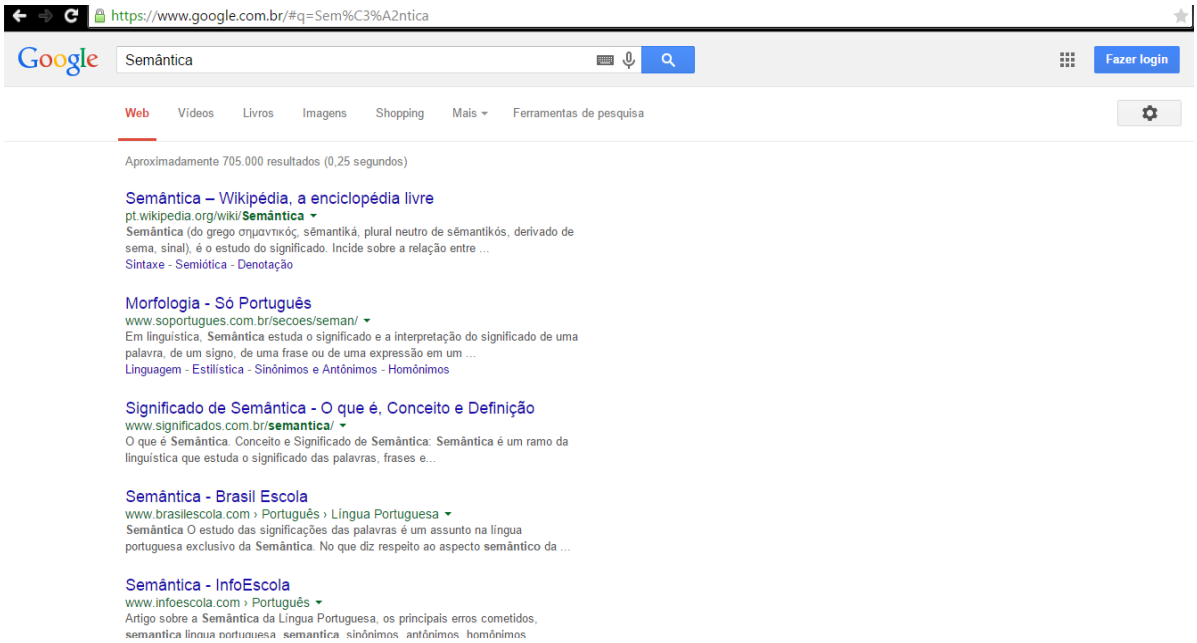
Devido ao aumento de documentos e dados na web, tornou-se difícil o método de diretórios por estes utilizarem métodos mais simples, sendo assim logo após surgiram os Motores de Busca que não organizam seus dados hierarquicamente, ou seja, não se preocupam com a seletividade e sim com a quantidade de recursos encontrados para sua base de dados (CENDÓN, 2001).

Esse tipo de busca se utiliza de software robôs que também pode ser chamados de aranhas (*spiders),* agentes viajantes (*wanderers*), ou vermes (*worms*) (CENDÓN, 2001).

Os *spiders* vasculham as paginas e geralmente começam pelas mais populares. Através de links encontrados nas mesmas vão seguindo e assim visitando mais páginas guardando o que julgam necessário em sua base de dados, o resultado é mostrado em forma de lista com os links guardados da base, onde o usuário pode seguir e encontrar a informação necessária (BRANSKI, 2001).

Na figura 2 podemos observar um site que utiliza desta técnica.

Figura 2 - Exemplo de mecanismo de busca que utiliza Motor de busca



Fonte: Do autor.

Os Sistemas de mecanismos de busca trabalham de acordo com um conjunto de palavras-chave fornecidas pelo usuário, onde estes retornam uma página exibindo uma lista de documentos que possuem estas palavras chaves. (ZAINA, 2005).

2.2.1 Motor de Busca da Google

Com o aumento de fluxo de dados que surgiam na web e o interesse de empresários em destacar seus sites nos buscadores da web que acabavam afastando o publico e comprometendo a confiança dos usuários. A disputa no mercado girava em torno de quem possuía a maior quantidade de dados, o que dificultou a sobrevivência de pequenas ferramentas de busca (BORTOLIN, 2009).

O Google nasceu em 1996, originalmente chamado BackRub, foi a criação dos estudantes da universidade de Stanford, Larry Page e Sergey Brin como uma melhor maneira de organizar e pesquisar a crescente Web (BURNS; SAUERS, 2014).

Burns e Sauers (2014) explica que ao invés de ranquear os resultados contando quantas vezes os termos pesquisados apareciam em uma página, como os outros mecanismos de buscas faziam naquele tempo, eles criaram um motor de busca que determinava a relevância de um website contando o número de páginas e a importância dessas páginas que se relacionavam diretamente a pagina original.

O nome foi mudado para Google em 1998, um erro de ortografia da palavra googol, um termo matemático para descrever o número 1 seguido por cem 0s. Este foi então escolhido como o novo nome para refletir o desejo de indexar a imensa quantidade de dados na internet (BURNS; SAUERS, 2014).

Segundo Google (2015) a pesquisa ocorre basicamente em três etapas:

1) antes de pesquisar: São utilizados softwares robôs, também chamados de indexadores ou rastreadores, que anexam paginas para futuramente serem retornadas como resultado das pesquisas caso estiverem relacionadas de acordo com a palavra chave, além de incluírem também informações dessas palavras chaves;
2) durante a pesquisa: Existem algoritmos que usam de mais de 200 mil sinais para decidir o resultado mais relevante ao que foi pesquisado, um desses sinais, ajuda a interpretar a palavra digitada procurando seu sinônimo e até mesmo corrigindo erros de ortografia;
3) resultado final: Quando o algoritmo encontra as paginas relevantes, as pesquisas se organizam em relevância e popularidade, permitindo ainda uma pré-visualização da pagina.

A Google mudou o seu famoso algoritmo para se focar na fase de entendimento das pesquisas. Isto facilitaria as fases subsequentes, limitando o número de documentos no índice que eram consultados para mostrar o melhor resultado possível (REDONDO, 2015, tradução nossa).

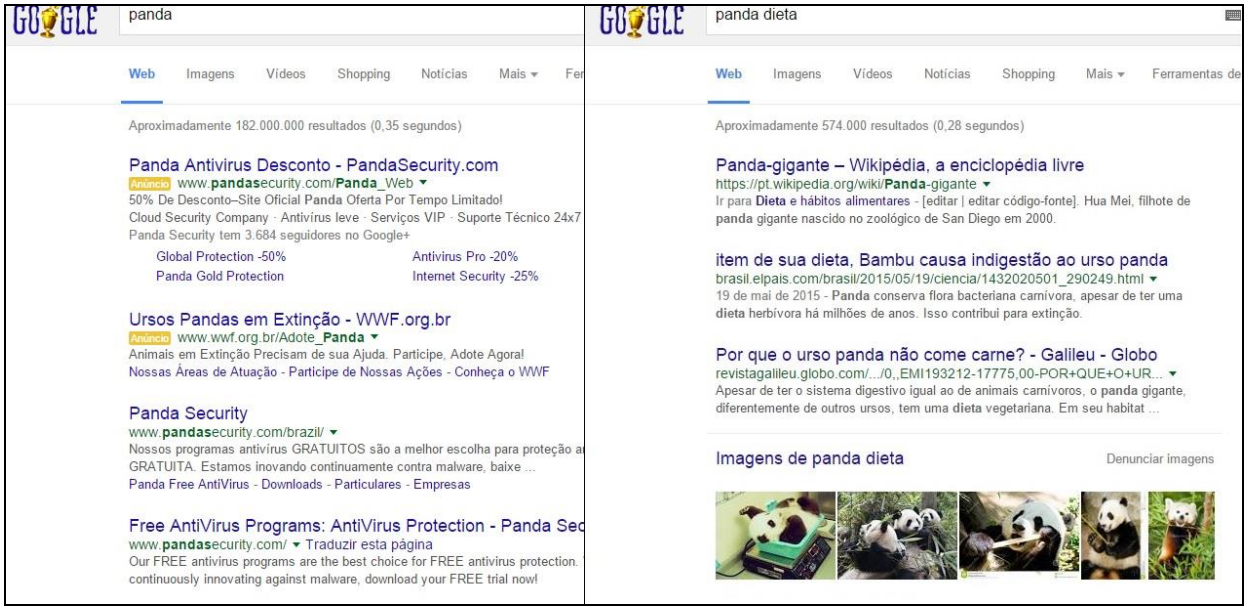
Este reforço da fase de entendimento significa que a Google começou a prestar mais atenção ao contexto em que uma busca é realizada e em quais conceitos aparecem nos documentos. Isto só é possível através da semântica (REDONDO, 2015, tradução nossa).

As palavras deixaram de ser apenas um conjunto de letras para se tornarem conceitos. Isto significa que o que uma vez era uma unidade única agora

se tornou um número de conotações que trazem um significado concreto (REDONDO, 2015, tradução nossa).

Segundo Redondo (2015) a busca semântica funciona da seguinte maneira, ao realizarmos uma busca pela palavra ‘panda’, sem especificar nada mais, poderíamos pensar que estamos nos referindo a uma espécie de urso nativa da China, um dos algoritmos da Google ou até mesmo um software antivírus, já ao pesquisar as palavras ‘panda dieta’, está claro que o Google sabe perfeitamente que tipo de resultados mostrar, isto porque nesta pesquisa, o termo está melhor definido e estamos dando um contexto, conforme pode ser visto na Figura 3.

Figura 3 - Exemplo de busca semântica.



Fonte: Adaptado de Redondo (2015).

3 WEB SEMÂNTICA

3.1 CONCEITO DE SEMÂNTICA

Para melhor entendimento do tipo de busca utilizado no projeto é preciso entender o significado de seu termo.

Semântica é “[...] componente do sentido das palavras e da interpretação das sentenças e dos enunciados [...].” (HOUAISS; VILLAR; FRANCO, 2004, p. 2540).

3.2 APLICAÇÃO DA SEMÂNTICA NA WEB

Com o crescimento da *World Wide Web*, devido ao aumento da informação, tem-se percebido que os métodos atualmente disponíveis para encontrar e utilizar informações na Web são muitas vezes insuficientes. Estes que são basicamente limitados à busca por palavras-chave ou sub-palavras (*sub-strings*) não oferecem suporte a estruturas mais profundas que podem estar ocultas nos dados ou que as pessoas costumam utilizar para raciocinar e orientar sua pesquisa, isso pode dificultar e ocasionar perda de informação na pesquisa feita na web (ENGEHOFER, 2002, tradução nossa).

Mesmo tendo sido projetada para facilitar a recuperação das informações, a Web acabou se tornando um repositório desorganizado de informações e documentos, no qual deixa muito a desejar quando precisamos recuperar o que realmente estamos buscando (SOUZA; ALVARENGA, 2004).

Existiria um potencial muito maior para a exploração da Web se as ferramentas disponíveis melhor combinassem com o raciocínio humano. Nesse sentido, a comunidade científica começou um esforço para investigar a base para o próximo estagio da Web, chamado de Web Semântica (ENGEHOFER, 2002, tradução nossa).

A Web Semântica não é uma Web separada, mas uma extensão da atual, em que é dado à informação um significado bem definido, permitindo que

computadores e pessoas trabalhem em cooperação (BERNERS-LEE; HENDLER; LASSILA, 2001, tradução nossa).

O objetivo básico da web semântica é criar uma camada na Web existente que possibilite o processo avançado do conteúdo da Web, permitindo o compartilhamento e processamento dos dados por humanos e softwares (KUMAR, 2012).

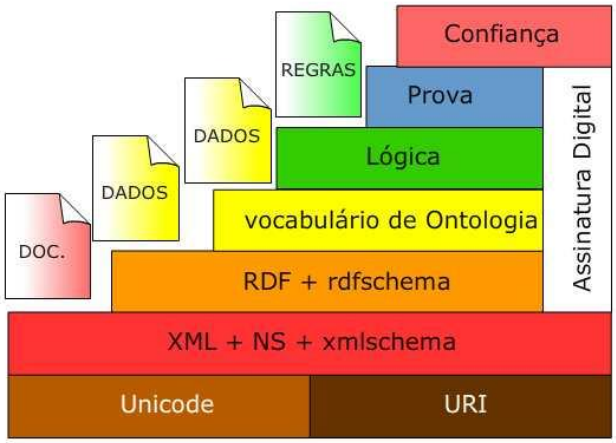
Para a Web Semântica funcionar, os computadores devem ter acesso a coleções estruturadas de informações e conjuntos de regras de inferência que eles podem usar para conduzir raciocínio automático (BERNERS-LEE; HENDLER; LASSILA, 2001, tradução nossa).

Segundo Pinheiro (2009), o desenvolvimento da Web Semântica é composto por diversas tecnologias, inclusive algumas que ainda estão sendo desenvolvidas, dentre as quais, as mais básicas padronizam a sintaxe utilizada para representar os dados.

Segundo Berners-lee; Hendler e Lassila (2001), para desenvolver aplicações que envolvam a Web Semântica, e necessária uma arquitetura dividida nas seguintes camadas:

a) unicode / URI;
b) camada XML / Namespaces / XML Schema;
c) RDF;
d) ontologia;
e) lógica;
f) prova;
g) confiança e Assinatura Digital.

Figura 4 - Arquitetura da web semântica.



Fonte: Adaptado de Berners-Lee; Hendler e Lassila (2001).

3.3 ARQUITETURA DA WEB SEMÂNTICA

Algumas das tecnologias necessárias para o desenvolvimento da Web Semântica já existem.

3.3.1 Unicode

Fundamentalmente, os computadores lidam com números. Eles armazenam letras e outros caracteres através da atribuição de um número para cada um. Antes de Unicode ser inventado, havia centenas de diferentes sistemas de codificação para a atribuição destes números, e qualquer computador particular especialmente os servidores precisavam suportar diversas codificações destes caracteres (THE UNICODE CONSORTIUM, 2008, tradução nossa).

O Unicode fornece um número único para cada caracter, não importa a plataforma, não importa o programa, não importa a língua. Ele é suportado em muitos sistemas operacionais, todos os navegadores modernos, e muitos outros produtos. O surgimento do Padrão Unicode, e da disponibilidade de ferramentas de apoio dele, estão entre as recentes tendências globais de tecnologia de software mais significativos (THE UNICODE CONSORTIUM, 2008, tradução nossa).

3.3.2 URI

Uniform Resource Identifiers (URI) são maneiras de identificar recursos, especialmente, mas não exclusivamente na Web.

As URIs possuem dois usos principais, é uma UR pode ser utilizada em ambos sentidos:

a) ***uniform Resource Names*** **(URNs)** são URIs são utilizadas para dar nome a algo, mesmo que seja um objeto abstrato que não esteja disponível na Web;
b) ***uniform Resource Locators*** **(URLs)** são URIs utilizadas para especificar o local de algo. URIs devem iniciar com um identificador de protocolo, e as URLs geralmente utilizam protocolos que uma representação técnica bem estabelecida no qual ferramentas podem utilizar para acessar lugares específicos, por exemplo http.

3.3.3 Extensible Markup Language (XML)

O XML disponibiliza uma sintaxe para estruturar documentos, mas não possuem nenhuma regra semântica nos mesmos (PINHEIRO, 2009).

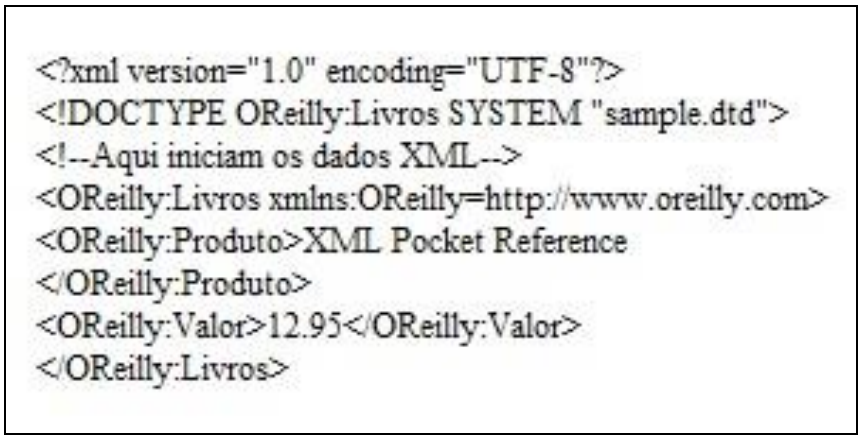
De acordo com Berners-Lee; Hendler e Lassila (2001, tradução nossa), o XML permite que todos criem suas próprias tags, que anotam páginas Web ou seções de textos em uma página.

O XML é um padrão de processamentos de documentos que é uma recomendação oficial da *World Wide Web Consortium*, e se trata de uma meta-linguagem que permite a você criar e editar suas próprias marcações de documentos. Muitos esperam que o XML e suas tecnologias irmãs se tornem a linguagem de marcação de escolha para conteúdos gerados dinamicamente, incluindo páginas não estáticas da Web (ECKSTEIN; CASABIANCA, 2001, tradução nossa).

O XML é na verdade uma forma simplificada do Standard Generalized Markup Language (SGML), uma documentação internacional que existe desde 1980.

Entretanto, o SGML é extremamente complexo, especialmente para a Web (ECKSTEIN; CASABIANCA, 2001, tradução nossa).

Figura 5 - Exemplo de XML.



Fonte: Adaptado de Eckstein e Casabianca (2001).

Um documento XML bem formatado é ou um XML, e automaticamente, bem-formatado, ou é apenas texto. Mas conforme utilizado de maneira comum, um XML bem formatado significa um documento que segue as recomendações do W3C para o XML, com todas as suas regras, conforme as seguintes (FAWCETT; QUIN; AYERS, 2012, tradução nossa):

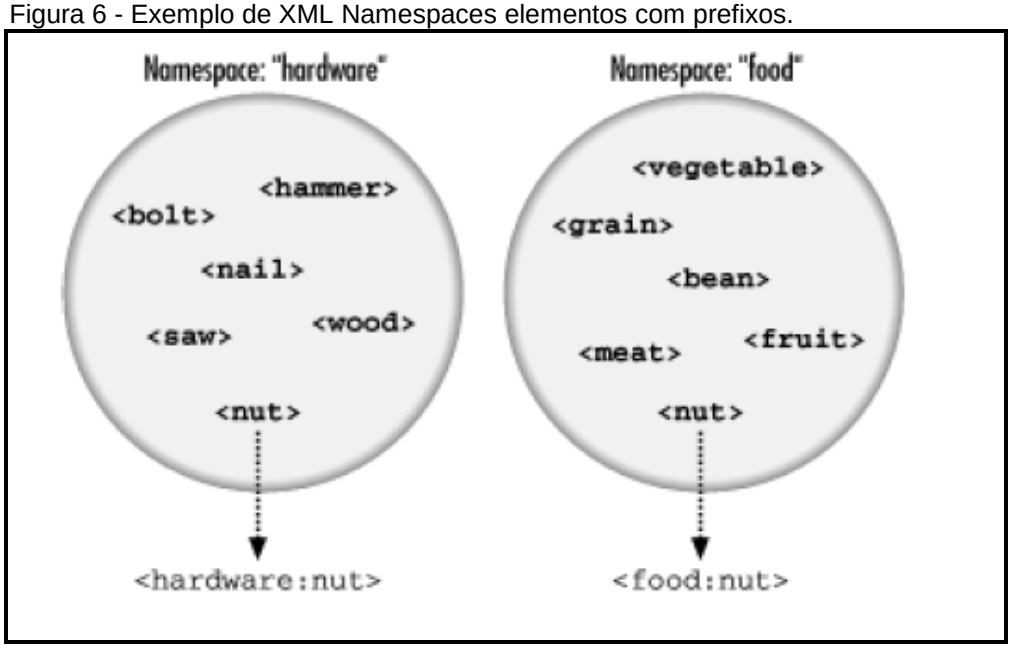
a) como o conteúdo é separado das marcações;
b) o que é utilizado para identificar a marcação;
c) o que as partes constituintes são;
d) em qual ordem e onde estas partes podem aparecer.

3.3.3.1 XML Namespaces

De modo simples, namespaces são maneiras de agrupar elementos e seus atributos em um cabeçalho comum, com o objetivo de diferenciá-los de elementos com nomes semelhantes (FAWCETT; QUIN; AYERS, 2012, tradução nossa).

Segundo Ray (2001), um namespace é um grupo de nomes de elementos e atributos. Pode-se declarar que aquele elemento existe dentro de uma namespace específica. Adicionando um prefixo a um elemento ou nome de atributo.

Figura 6 - Exemplo de XML Namespaces elementos com prefixos.



Fonte: Ray (2001).

3.3.3.2 Document Type Definitions (DTDs)

Você pode escrever um XML bem formatado seguindo regras simples. Estas regras descrevem como determinar uma das características chave do XML: Sua estrutura. Mas a boa formatação só chega até aqui. Muitas outras tecnologias estão disponíveis para validar documentos XML, cada uma com suas vantagens e desvantagens. Os DTDs oferecem uma maneira de especificar mais regras que ajudam na interpretação de documentos e sua estrutura. Utilizando *namespaces* é possível criar grupos de elementos e atributos distintos de acordo com o propósito desejado (FAWCETT; QUIN; AYERS, 2012, tradução nossa).

Um DTD especifica como os elementos dentro de um documento XML deveriam se relacionar e também fornece regras gramaticais para o documento e cada um de seus elementos. Um documento aderente às especificações XML e suas regras apontadas pelo seu DTD é considerado válido (ECKSTEIN; CASABIANCA, 2001, tradução nossa).

3.3.3.3 XML Schemas

Assim como as DTDs, os XML Schemas são utilizados para definir vocabulários XML. Eles descrevem a estrutura e conteúdo em mais detalhes que as DTDs, permitindo uma validação mais precisa (FAWCETT; QUIN; AYERS, 2012, tradução nossa).

Alguma das vantagens do XML Schemas, conforme Fawcett, Quin e Ayers (2012), são:

a) XML Schemas são criados utilizando XML básico, enquanto DTDs utilizam uma sintaxe diferente;
b) XML Schemas suportam as recomendações de Namespaces;
c) XML Schemas possibilitam a validação do conteúdo de elementos texto e tipos de dados definidos pelo usuário;
d) XML Schemas facilitam a criação de modelos de conteúdos complexos e reutilizáveis;
e) XML Schemas possibilitam a modelagem de conceitos de programação como a herança de objetos e substituição de tipos.

Conforme Ray (2001), *XML Schemas* utilizam fragmentos de XML. Chamados *templates* para demonstrar como um documento deve ser. O benefício de usar *schemas* é que eles são por si só, uma forma de XML, possibilitando que sejam editados com as mesmas ferramentas que se utilizaria para documentos XML. *Schemas* também introduzem uma checagem de tipos de dados mais poderosa, tornando possível encontrar erros em conteúdos, assim como na utilização de *tags*.

Hoje, os XML Schemas são uma tecnologia madura utilizadas em uma variedade de aplicações XML. Além do seu uso em validações, aspectos do XML Schema são utilizados em várias outras tecnologias, como o SOAP, por exemplo (FAWCETT; QUIN; AYERS, 2012, tradução nossa).

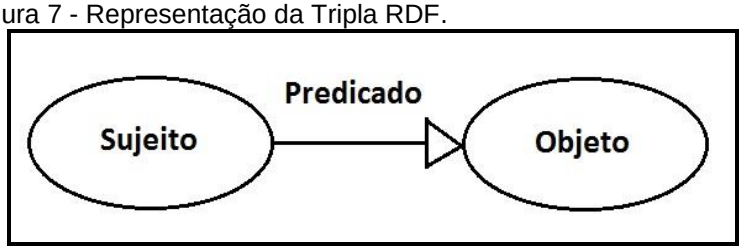
3.3.4 Resource Description Framework (RDF)

O RDF trata-se de um modelo de dados utilizado para representar objetos e seus relacionamentos, e pode ser representado através da sintaxe XML (PINHEIRO, 2009)

O RDF é o responsável por expressar o significado, e o representa em grupos de triplas, onde cada tripla consiste do sujeito, predicado e objeto (BERNERS-LEE; HENDLER; LASSILA, 2001, tradução nossa).

Cada tripla pode ser representada de acordo com a Figura 7.

Figura 7 - Representação da Tripla RDF.



Fonte: Adaptado de W3C (2015).

Para exemplificar melhor, Berners-Lee; Hendler e Lassila (2001, tradução nossa) explica que, no RDF, um documento faz asserções de que coisas específicas (pessoas, páginas Web ou outras coisas) possuem propriedades (tais como “é uma irmã de,” ou “é autor de”) com certos valores (uma pessoa, uma página Web). Esta estrutura acabou sendo a maneira natural de descrever a grande maioria dos dados processados por máquinas.

3.4 REPRESENTAÇÃO DO CONHECIMENTO

A representação do conhecimento é claramente uma boa ideia, e algumas demonstrações muito boas existem, mas ainda não mudou o mundo (BERNERS-LEE; HENDLER; LASSILA, 2001, tradução nossa).

Ela contém as sementes de aplicações importantes, mas para que seu total potencial seja percebido, ela deve estar ligada em um sistema global. Sistemas tradicionais de representação de conhecimento geralmente têm sido centralizados,

requerendo que todos compartilhem a mesma definição de conceitos comuns como “pais” e “veículo” (BERNERS-LEE; HENDLER; LASSILA, 2001, tradução nossa).

3.5 ONTOLOGIA

Segundo Gruber (1993), uma ontologia é uma especificação explícita da conceptualização. O termo é emprestado da filosofia, onde uma ontologia é uma conta sistemática da existência. Para sistemas baseados em conhecimento, o que “existe” é exatamente aquilo que pode ser representado.

Berners-lee; Hendler e Lassila (2001) explica que, na filosofia, uma ontologia é uma teoria sobre a natureza da existência, da qual tipos de coisas existem; Ontologia como disciplina estuda tais teorias. Pesquisadores de Inteligência Artificial e da Web tem escolhido este termo para o seu próprio jargão, e para eles, ontologia é um documento ou um arquivo que formalmente define as relações entre termos. O tipo mais típico de ontologia para a Web tem taxonomia e um grupo de regras de inferência.

4 TECNOLOGIAS DE DESENVOLVIMENTO MÓVEL

O mercado de smartphones em todo o mundo cresceu 28,2% ano a ano no quarto trimestre de 2014 (2014 Q4), com embarques de 377,5 milhões de unidades, de acordo com dados do *International Data Corporation* (IDC) *Worldwide Quarterly Mobile Phone Tracker*. Android ainda domina o mercado com uma quota de 76,6% no 4Q14. A Apple conseguiu enviar 74,5 milhões de unidades neste trimestre com um crescimento ano-a-ano de 46,0%, fechando a lacuna de um empate perto com a Samsung. Para todo o ano, o mercado de smartphones em todo o mundo enviados um total de 1,3 bilhões de unidades. Este é um crescimento de 27,7% em relação aos 1,0 bilhões de unidades de embarques em 2013 (IDC, 2014, tradução nossa).

Tabela 1 – Mercado de smartphones.

Período	Samsung	Apple	Lenovo*	Huawei	LG Electronics	Outros
Q1 2015	24.6%	18.3%	5.6%	5.2%	4.6%	41.7%
Q1 2014	30.7%	15.2%	6.6%	4.7%	4.3%	38.6%
Q1 2013	31.5%	16.9%	4.7%	4.2%	4.7%	38.1%
Q1 2012	28.9%	22.9%	5.0%	3.4%	3.2%	36.6%

Fonte: Adaptado de IDC (2015).

Devido a esse grande aumento da usabilidade dos *smartphones* e *tablets*, na tecnologia móvel, surge à necessidade de softwares que sejam compatíveis a essa tecnologia, e dependendo do sistema operacional utilizado existe sua própria plataforma de desenvolvimento.

Segundo os dados acima analisado, o sistema operacional mais utilizado ainda é o Android, uma plataforma líder no mercado.

4.1 PLATAFORMA ANDROID

A Google, vendo um grande crescimento do uso da Internet e de pesquisa em dispositivos móveis, adquiriu a Android, Inc., em 2005, para concentrar o seu desenvolvimento em uma plataforma de dispositivos móveis (STEELE; TO, 2011, tradução nossa).

A plataforma Android surgiu da aliança da Google com a Open Handset Alliance (OHA), que consiste na aliança entre empresas que englobam as tecnologias móveis desde softwares a hardwares.

A OHA é um grupo de 84 empresas de tecnologia e tecnologia móvel que se uniram para acelerar a inovação nesta tecnologia e oferecer aos consumidores uma rica, mais barata e melhor experiência móvel (OPEN HANDSET ALLIANCE, 2015, tradução nossa).

O sistema operacional Android já percorreu um longo caminho desde o anúncio da OHA no final de 2007. A idéia de um sistema operacional (SO) de código aberto pra sistemas integrados não era nova, mas a Google agressivamente por trás disso ajudou a empurrar o Android ao topo em apenas alguns anos (STEELE; TO, 2011, tradução nossa).

O Android foi construído para permitir que desenvolvedores criem aplicativos móveis convincentes que tirem o máximo proveito de tudo que um aparelho tem para oferecer. Por exemplo, um aplicativo pode chamar a qualquer das funcionalidades básicas do telefone, como fazer chamadas, enviar mensagens de texto, ou utilizar a câmara, permitindo aos desenvolvedores criar experiências mais ricas e coesas para os usuários. O Android é construído sobre o aberto Kernel do Linux. Além disso, utiliza uma máquina virtual personalizada que foi projetada para otimizar a memória e recursos de hardware em um ambiente móvel. A plataforma vai continuar a evoluir à medida que a comunidade de desenvolvedores trabalha em conjunto para criar aplicativos móveis inovadores (OPEN HANDSET ALLIANCE, 2015, tradução nossa).

A diversidade de dispositivos e a rápida adaptação ajudaram o Android a aumentar sua base de usuários, mas isto aconteceu com desafios potenciais aos desenvolvedores. As aplicações precisam suportar múltiplos tamanhos de tela, resoluções, teclado, hardware de sensores, versões do sistema operacional taxa de dados wireless e configurações do sistema. Cada um pode levar a um imprevisível e diferente comportamento, mas testar as aplicações em todos os ambientes é uma tarefa impossível (STEELE; TO, 2011, tradução nossa).

O sistema operacional Android oferece um conjunto completo de softwares para dispositivos móveis: um sistema operacional, middleware e aplicativos móveis chave (OPEN HANDSET ALLIANCE, 2015, tradução nossa).

Leal (2015) explica que o sistema operacional Android é feito sobre o Kernel do Linux, que tem a reponsabilidades de administrar a memória, energia drives e processos. Já o middleware tem a responsabilidade de facilitar a comunicação entre aplicativos do aparelho. E as aplicações chaves são as aplicações comuns como contatos, discador etc.

O Android tem muitas versões e para cada uma é dado um nome de doce seguido a um número sequencial chamado de *API level,* no qual é importante entender para saber os recursos disponibilizados em cada versão, pois se executarmos um aplicativo que utiliza recursos de uma versão mais recente em uma versão inferior ocorrerá erro (LEAL, 2015).

Tabela 2 - Versões do Android.

Nome da versão	Versão	API Level
Cupcake	1.5	3
Donut	1.6	4
	2.0	5
Eclair	2.0.1	6
	2.1	7
Froyo	2.2	8
Gingerbread	2.3	9
	2.3.3	10
	3.0	11
HoneyComb	3.1	12
	3.2	13
Ice cream Sandwich	4.0	14
	4.0.3	15
	4.1	16
JellyBen	4.2	17
	4.3	18
KitKat	4.4	19
	4.4w	20
Lollipop	5.0	21

Fonte: Adaptado de Leal (2015).

4.1.1 Arquitetura

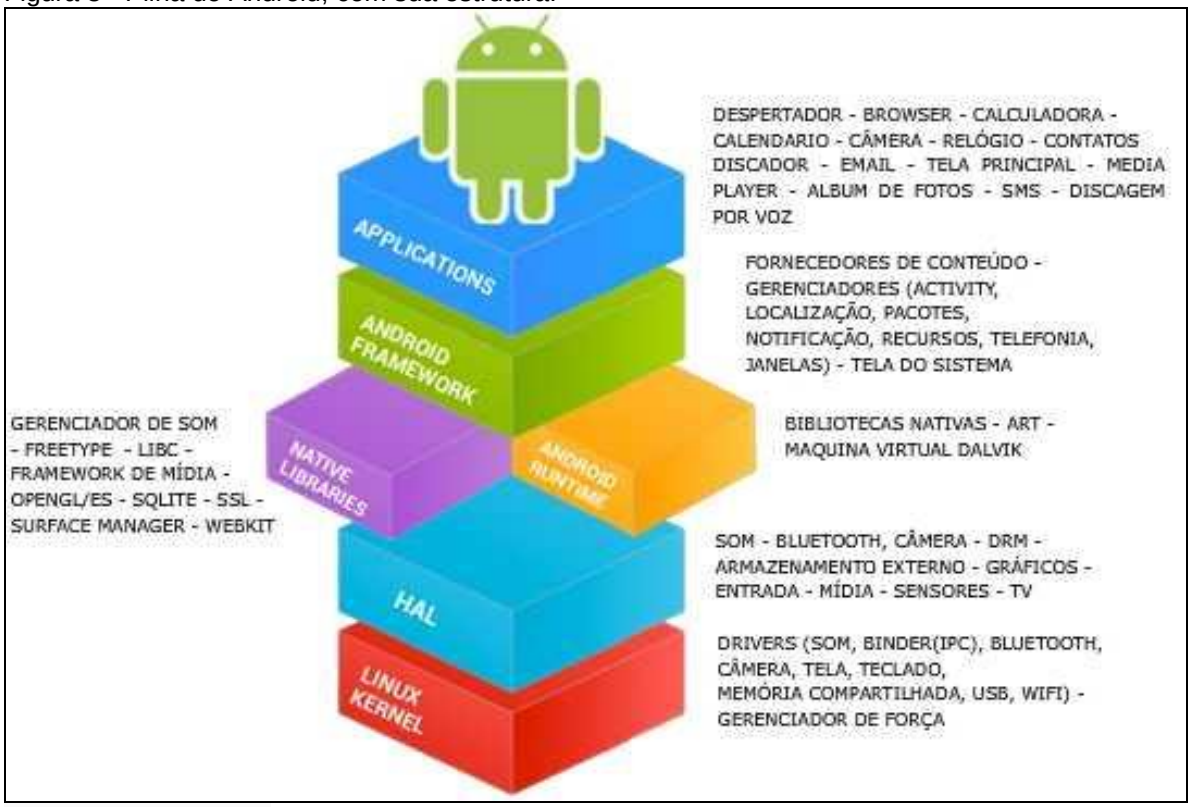
É importante entender a arquitetura do sistema do Android antes de portá-lo em seu Hardware. Pelo fato dos seus drivers e o *Hardware abstraction layer* (HAL) interagirem com o Android, saber como o Android funciona, pode ajudar a navegar entre as muitas camadas de código na árvore de fontes do Android Open Source Project (AOSP). (ANDROID, 2015, tradução nossa).

O Android é construído em cima do Linux que é um ótimo sistema operacional. Existem muitas boas razões para escolher o Linux como base da pilha Android. Alguns dos principais são a sua portabilidade, segurança e características (GARGENTA, 2011, tradução nossa).

Segundo Android (2015, tradução nossa), Android é uma pilha de software de fonte aberta para uma ampla gama de dispositivos móveis e um projeto de código aberto correspondente liderado pelo Google.

A figura 8 mostra a pilha de software do Android e suas diversas camadas.

Figura 8 - Pilha do Android, com sua estrutura.



Fonte: Android (2015).

Como podemos observar na figura 8, o Android é composto por cinco camadas que estão organizadas como pilha: Camada de Aplicações, Android Framework, Bibliotecas Nativas, Android *Runtime*, HAL, e o *Kernel* do Linux.

4.1.1.1 Camada de Aplicações

Nessa camada se encontra todas as aplicações feitas em Java, que fazem o funcionamento básico do sistema, feitas pela comunidade do Android, nela se encontra cliente de e-mail, mapas, navegadores, sistema de SMS, gerenciador de contatos entre outras aplicações (PEREIRA; SILVA; 2009).

Segundo Burnette (2010, tradução nossa), aplicações são programas que podem tomar conta da tela inteira do dispositivo e interagir com o usuário. Por outro lado *widgets* (que algumas vezes são chamados de *gadgets*) operam apenas em uma pequena área da tela principal.

Todas as aplicações, ambas nativas e de terceiros, são construídas na camada de aplicações, através das mesmas bibliotecas. A camada de aplicações é executada dentro do Android *Runtime*, utilizando classes e serviços disponíveis pelo Application Framework (MEIER, 2012, tradução nossa).

4.1.1.2 Android Framework

O *Application Framework* fornece blocos de construção que são utilizados para criar aplicações no Android. Este framework vem pré-instalado com o Android, mas é possível estendê-lo com seus próprios componentes.

Segundo Burnette (2010, tradução nossa) as partes mais importantes deste framework são:

a) *activity manager*: Responsável por controlar o ciclo de vida das aplicações;
b) *content providers.* Estes objetos encapsulam dados que necessitem ser compartilhados entre as aplicação, como os contatos por exemplo;
c) *resource manager*: *Resources* (Recursos) são qualquer coisa que existe em seu programa que não seja código;

d) *location manager*: Um telefone android sempre sabe onde está;
e) *notification manager*: Eventos como a chegada de mensagens, alertas de proximidades e etc, podem ser apresentados de maneira discreta ao usuário.

4.1.1.3 Bibliotecas Nativas

Esta camada contém as bibliotecas nativas do Android. Estas bibliotecas compartilhadas são escritas em C ou C++, compiladas para uma específica arquitetura de hardware utilizada pelo telefone, e pré-instalada pelo fabricante.

Segundo Burnette (2010, tradução nossa) algumas das mais importantes bibliotecas nativas:

a) surface Manager: O Android utiliza um gerenciador de composição de janelas similar ao Vista ou Compiz. Ao invés de desenhar diretamente no buffer da tela, seus comandos de desenhos se tornam bitmaps fora da tela, que são combinados com outros bitmaps para formar a imagem que o usuário vê. Isto permite que o sistema crie muitos tipos de efeitos e transações interessantes;
b) gráficos 2D e 3D: Elementos de duas e três dimensões podem ser combinados em uma única interface de usuário com Android;
c) codecs de Mídia: O Android pode reproduzir vídeo, gravar e reproduzir áudio em uma variedade de formatos;
d) banco de dados SQL: O Android inclui um motor leve de banco de dados chamado SQL Lite, o mesmo utilizado no Apple iPhone;
e) motor de browser: Para a exibição rápida de conteúdos HTML, o Android usa a biblioteca WebKit, a mesma utilizada no browser Google Chrome.

4.1.1.4 Android Runtime

O Android *Runtime* possui a máquina virtual (VM) Dalvik e as bibliotecas do núcleo do Java. A máquina virtual Dalvik é uma implementação da Google do

Java, otimizada para dispositivos móveis. Todo o código escrito para Android será escrito em Java e executará dentro desta máquina virtual (BURNETTE, 2010, tradução nossa).

O *runtime* é o que faz um telefone Android ser um telefone Android e não apenas uma implementação móvel do Linux. Incluindo as bibliotecas do núcleo e máquina virtual Dalvik, o *Android Runtime* é o motor que dá força a todas as aplicações e, junto com as bibliotecas, forma a base do *Application Framework* (MEIER, 2012, tradução nossa).

4.1.1.5 HAL

O Hardware Abstraction Layer (HAL) é uma interface padrão que permite ao sistema chamar a camada de drivers do dispositivo, sendo agnóstico em relação as implementações de baixo nível de hardware e drivers.

4.1.1.6 Kernel Linux

Criado por Linus Torvalds em 1991, o Linux pode ser encontrado em praticamente tudo, desde relógios de pulso até supercomputadores. O Linux fornece uma camada de abstração de hardware para o Android, permitindo que o Android seja portado em várias plataformas no futuro (BURNETTE, 2010, tradução nossa).

Serviços como drivers de hardware, gerenciamentos de memória, segurança e rede, são manipulados pelo *Kernel* do Linux (MEIER, 2012, tradução nossa).

5 COMUNICÃO ENTRE SISTEMAS

Com a evolução e o surgimento de novas tecnologias, surgiu a necessidade de realizar a comunicação entre sistemas desenvolvidos utilizando tecnologias diferentes, seja entre sistemas locais ou na nuvem.

Conforme Martins (2005), o surgimento da Internet exigiu uma integração entre os mais diversos tipos de informações, obrigando a gestores de Tecnologia da Informação (TI) a adotar estratégias que pudessem suprir tal necessidade. Surgiram então tecnologias como Service Oriented Architecture (SOA) e Web Services (WS).

5.1 WEB SERVICE

O WS é uma evolução dos modelos de computação distribuída, amplamente utilizados na segunda metade da década de 90. Estes modelos e tecnologias foram bem sucedidos na integração de software em ambientes de redes locais. Com o avanço da internet, houve a necessidade de integrar sistemas além destas redes locais e em ambientes heterogêneos, surgiram então os WS, resultado de um consórcio formado por empresas como IBM, Microsoft, BEA e outras que integram o W3C (GOMES, 2014).

Para Cerami (2002, tradução nossa) WS podem ser definidos como qualquer serviço disponível na internet, que utilize um sistema de mensagem de XML padronizado, e que não esteja amarrado a qualquer sistema operacional ou linguagem de programação.

5.1.1 Padrões de desenvolvimento de Web Services

De acordo com Gomes (2014), atualmente existem dois padrões de desenvolvimento de WS, como o padrão SOAP e o padrão REST ou RESTfull.

5.1.1.1 SOAP

Simple Object Access Protoco (SOAP) é um protocolo baseado em XML para a troca de informação entre computadores. Embora o SOAP possa ser usado em uma variedade de sistemas de mensagens e pode ser entregue através de uma variedade de protocolos de transporte, o foco inicial do SOAP é a chamada a procedimentos remotos transportados via HTTP (CERAMI, 2002, tradução nossa).

Segundo Box et al (2000, tradução nossa), o SOAP consiste de três partes:

a) a construção do encapsulador SOAP define um quadro global para expressar o que está na mensagem, quem deveria lidar com ela, e quando ela é opcional ou obrigatória;
b) as regras de codificação SOAP definem um mecanismo de serialização que pode ser usado para trocar instâncias de tipos de dados definidos pela aplicação;
c) a representação *Remote Procedure Call* (RPC) define uma convenção que pode ser usada para representar chamadas e respostas aos procedimentos remotos.

5.1.1.2 REST ou RESTful

O termo REST vem da dissertação de PhD de Roy Fielding, publicada em 2000, e significa *Representational State Tranfer* (transferência de estado representacional). Sozinho, o REST não é uma arquitetura, mas sim uma série de restrições que, quando aplicadas a um sistema, cria um estilo de arquitetura de software (SANDOVAL, 2009, tradução nossa).

O termo *RESTful* é como o termo "orientado a objetos". Uma linguagem, um *framework*, ou uma aplicação podem ser desenhados de uma maneira orientada a objetos, mas isso não faz de sua arquitetura uma arquitetura orientada a objetos (RICHARDSON; RUBY, 2007, tradução nossa).

O REST facilita o desenvolvimento, assim como a escalabilidade e flexibilidade das aplicações garantindo que os dados estejam em camadas, sem

estado e bem definido. Para mudar do XML para o JSON, por exemplo, bastaria mudar a extensão na URL para fazer a requisição, sem a necessidade de redesenhar a aplicação ou mudar a plataforma de desenvolvimento (MAKICE, 2009, tradução nossa).

Roy Fielding chegou ao *REST* avaliando todos os recursos e tecnologias de rede disponíveis para a criação de aplicações distribuídas. Ele olha aos documentos de estilos de arquitetura de aplicações distribuídas, iniciando com o que ele chama de 'espaço-nulo' que representa a disponibilidade de toda a tecnologia e todo estilo de desenvolvimento de aplicação sem regras e sem limites, e termina com as seguintes restrições que definem um sistema *RESTful* (SANDOVAL, 2009, tradução nossa).

a) deve ser um sistema cliente-servidor;
b) não deve manter estado - não deve haver necessidade para o serviço manter as sessões dos usuários; Em outras palavras, cada requisição deve ser independente das outras;
c) deve suportar um sistema de *cache;*
d) deve ser uniformemente acessível - cada recurso deve ter um endereço único e um ponto de acesso válido;
e) deve ser dividido em camadas;
f) deve fornecer código em demanda - embora esta seja uma restrição opcional, aplicações podem ser extensíveis em tempo de execução, permitindo o download de código sob demanda.

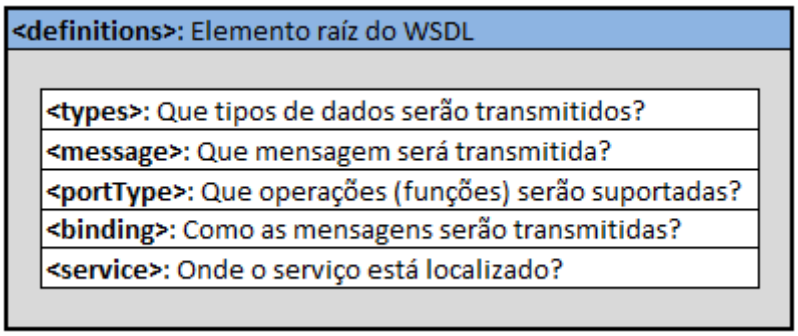
5.1.2 WSDL

Segundo Gomes (2014), o *Web Service Description Language* (WSDL) pode ser definido como um arquivo XML, que possui o objetivo de descrever as operações que compõem um WS, definindo como deve ser a entrada e saída dessas operações.

O WSDL é uma especificação que define como descrever Web Services em uma gramática comum do XML. O WSDL descreve quatro pedaços fundamentais de dados. (CERAMI, 2002, tradução nossa):

a) informações da interface descrevendo todas as funções disponíveis publicamente;
b) informação do tipo de dado para todas as mensagens de requisição e respostas;
c) informação de vínculo sobre o protocolo de transporte a ser usado;
d) informação do endereço para localizar o serviço especificado.

Figura 9 - Especificação WSDL resumida



Fonte: Adaptado de Cerami (2002).

5.1.3 WADL

Web Application Description Language (WADL) é uma descrição para um web service RESTful implementado, e é similar ao WSDL do SOAP.

Assim como o WSDL que mostra a estrutura, funcionalidade, parâmetros e aceita diferentes métodos HTTP dos web services SOAP, o WADL também fornece os mesmos recursos.

A diferença entre os dois é que o WADL é usado para Web services RESTful, e o WSDL é usado para web services SOAP (GULABANI, 2014, tradução nossa).

5.1.4 JSON

JavaScript Object Notation (JSON) é um formato muito popular de intercâmbio de dados. Criado por Douglas Crockford, JSON é um formato leve, baseado em texto, que pode ser facilmente lido por humanos para troca de dados

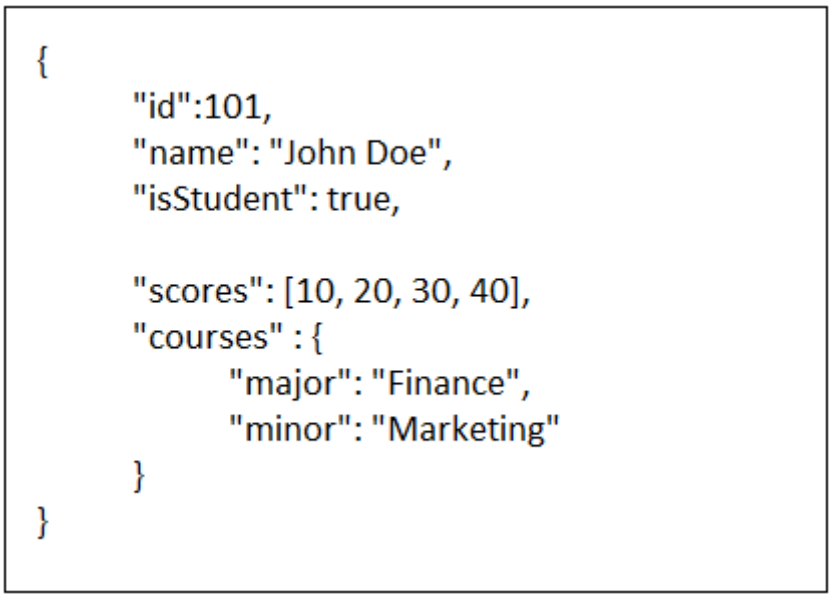
entre clientes e servidores. O JSON pode ser utilizado em aplicações Web para transferência de dados. Antes do JSON, o XML era considerado o ideal para o intercâmbio de dados (SRIPARASA, 2013, tradução nossa).

JSON, em resumo, é uma representação textual definida por um pequeno grupo de regras no qual os dados são estruturados. A especificação do JSON define que os dados podem ser estruturados em uma das seguintes composições (SMITH, 2015 , tradução nossa):

a) uma coleção de pares de nomes/valores;

b) uma lista ordenada de valores.

Figura 10 - Exemplo de dados em JSON.



Fonte: Adaptado de Sriparasa (2013).

6 OCR (RECONHECIMENTO ÓPTICO DE CARACTERES)

É muito comum hoje em dia precisarmos de textos que se encontram em imagens ou documentos e muitas vezes é preciso a reescrita destes textos. Para solucionar este problema, e obter estes textos novamente sem o trabalho de ter que digitar tudo manualmente, é muito comum a digitalização desses arquivos e a utilização de softwares OCR (ALVES, 2003).

Ainda de acordo com Alves (2003), esta tecnologia baseia-se na extração de textos de imagens digitais, ela torna textos encontrados em jornais, fax, livros, dentre outros em textos que podem ser modificados. É importante identificar que podem existir nestes documentos elementos textuais e gráficos, para que o OCR não perca tempo tentando processar documentos que tenham apenas elementos gráficos.

O princípio do reconhecimento automático de padrões é primeiramente ensinar a máquina quais classes de padrões podem ocorrer e como elas se parecem (EIKVIL, 1993, tradução nossa).

No OCR os padrões são as letras, números e alguns caracteres especiais. O ensinamento da máquina é realizado mostrando exemplos de caracteres de todas as classes diferentes. Baseada nos exemplos, a máquina constrói um protótipo ou uma descrição de cada classe de caractere. Durante o reconhecimento, os caracteres desconhecidos são comparados as descrições previamente obtidas e assim, relacionadas a classe mais parecida (EIKVIL, 1993, tradução nossa).

Com o surgimento das redes neurais, muitos estudos foram realizados e mais técnicas foram criadas para a transposição perfeita de imagem para textos. Para que se consigam resultados satisfatórios os programas devem ser treinados em diversos tipos de diferentes fontes, aumentando as probabilidades de um reconhecimento correto (MELO, 2012).

Na etapa de treinamento, a entrada dos dados será composta por imagens de caracteres. Durante o uso do reconhecimento óptico de caracteres com a rede neural, podem surgir caracteres não reconhecidos pela rede, ou que não foram utilizados em seu treinamento, onde a rede poderá não reconhecer o

caractere de imediato, mas poderá utilizar-se de artifícios para realizar a correta tradução dos caracteres, como o uso de um dicionário (MELO, 2012).

Além de tudo, um pré-processamento da imagem poderá ser indispensável para o aumento da probabilidade de reconhecimento dos caracteres (MELO, 2012).

6.1 PASSOS PARA O RECONHECIMENTO DOS CARACTERES

Segundo Souza (2011), para o processamento da conversão de imagem para texto, técnica OCR, se resume em quatro Etapas, abaixo definidas:

a) **aquisição dos dados:** Ocorre quando o documento impresso é digitalizado, seja por câmeras, *scanner* ou outros dispositivos;
b) **processamento da imagem:** Trata-se da redução de ruídos, análise da rotação da imagem, junção de caracteres entre outros processamentos, essa etapa é a mais importante, pois através dela que se obtêm melhores resultados na conversão;
c) **análise de características:** Encontra características para facilitar o processo de reconhecimento, tal como tamanho da fonte e análise de diagramação;
d) **geração do arquivo de saída:** gera o arquivo final com o texto extraído da imagem.

6.2 PREPARAÇÃO DA IMAGEM

A seguir serão explicadas as etapas essenciais que ocorrem no processo para deixar a imagem bem preparada para o reconhecimento de caracteres.

6.2.1 Processo de digitalização

Através do processo de digitalização, uma imagem digital do documento original é capturado. Nessa etapa, é usado um processo muito importante chamado Thresholding. Este processo é importante porque os resultados do reconhecimento

dependem totalmente da qualidade da imagem em dois níveis. Um limite fixo é usado, onde tonalidades de cinza abaixo deste valor são dito preto e níveis acima são referidos como sendo branco (EIKVIL,1993, tradução nossa).

Contudo, esta técnica pode não ser suficiente em grande parte dos documentos encontrados, por possuírem uma gama bastante grande em contraste. Em casos específicos como estes, serão necessárias técnicas mais sofisticadas a fim de se obter um bom resultado (EIKVIL,1993, tradução nossa).

6.2.2 Localização e segmentação

A segmentação é um processo que determina os componentes de uma imagem. É necessário localizar as regiões do documento onde os dados estão impressos e distingui-las de figuras e gráficos (EIKVIL,1993, tradução nossa).

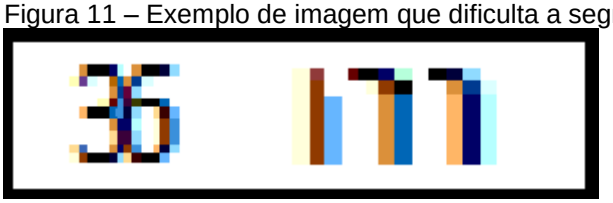
A segmentação é o isolamento de caracteres ou palavras. A maioria dos algoritmos de reconhecimento de caracteres segmentam as palavras em caracteres isolados para serem reconhecidos de maneira individual (EIKVIL,1993, tradução nossa).

Durante essa etapa podem existir problemas. Segundo EIKVIL (1993), os problemas podem ser divididos em quatro grupos:

1) extração de caracteres fragmentados que se tocam: essas distorções podem levar vários caracteres que estejam unidos a serem interpretados como um único caractere, ou parte de um caractere a ser interpretado como um símbolo inteiro. Este tipo de problema geralmente acontece em documentos de fotocópias escuras;
2) distinguir o ruído do texto: Pontos e acentos podem ser confundidos como ruído, e vice-versa;
3) confundir gráficos ou geometria com texto: Pode acontecer de um elemento não textual ser enviado para o reconhecimento;
4) confundir texto com gráficos ou geometria: neste caso, o texto pode ser confundido com gráfico e não será passado para a fase de reconhecimento. Isso ocorre com frequência se os caracteres estão conectados a gráficos.

Um exemplo de imagem no qual os textos ou caracteres estão impressos de maneira que possa influenciar na segmentação correta, pode ser visto na figura 11.

Figura 11 – Exemplo de imagem que dificulta a segmentação dos caracteres.



Fonte: Adaptado de EIKVIL (1993).

6.2.3 Pré-processamento

A imagem resultante do processo de análise pode conter certa quantidade de ruído. Dependendo da resolução e o sucesso da técnica aplicada de thresholding, os caracteres podem estar manchados ou quebrados. Alguns destes defeitos podem causar pobres taxas de reconhecimento (EIKVIL,1993, tradução nossa).

Segundo EIKVIL (1993, tradução nossa), esses problemas podem ser eliminados pela utilização de um pré-processamento para suavizar a imagem. A suavização implica tanto no enchimento e afinamento. No enchimento se elimina pequenas pausas, lacunas e buracos, enquanto no afinamento reduz a largura da linha dos caracteres.

Além de suavização, o pré-processamento inclui geralmente normalização. A normalização é aplicada para se obter caracteres de tamanho uniforme, inclinados e rodados de forma a ficar totalmente alinhados (EIKVIL,1993, tradução nossa).

7 TRABALHOS CORRELATOS

Hoje em dia devido ao grande uso de tecnologia móvel e ao grande número de acessos à internet, surgem novas aplicações utilizando esta tecnologia. Nós tópicos subsequentes serão abordados trabalhos que usam temas ou tecnologias semelhantes a este trabalho proposto.

7.1 PROTÓTIPO DE APLICATIVO ANDROID PARA EXTRAÇÃO DE TEXTO EM IMAGENS E CONVERSÃO EM VOZ, ORIENTADO AO APOIO A LEITURA PARA DEFICIENTES VISUAIS

Projeto de pesquisa feito por Claiton de Melo Marcilio para obtenção de grau de Bacharel, no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

O objetivo do projeto foi desenvolver um protótipo de aplicativo para Android que possam realizar leitura de documentos impressos para usuários com deficiência visual, para isso foi utilizado extração de texto de imagem e a conversão deste texto em voz. (MARCILIO, 2013).

7.2 EPLANTS: PROTÓTIPO DE APLICATIVO ACADÊMICO EM PLATAFORMA ANDROID PARA CATALOGAÇÃO DE PLANTAS

Projeto de pesquisa realizado por Vanderson Zanoni Campanholi para Conclusão de Curso e obtenção do grau de Bacharel no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Com o objetivo de desenvolver um protótipo de aplicativo móvel na plataforma Android para identificação e classificação de plantas destinadas a acadêmicos da área ambiental (CAMPANHOLI, 2014).

7.3 METAIMAGEM: ANOTAÇÃO DE IMAGENS COM USO DE WEB SEMÂNTICA PARA SUPORTE A CLASSIFICAÇÃO EM APLICAÇÕES DE RECUPERAÇÃO

Projeto de pesquisa para a Conclusão de Curso para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense, realizado por Marcelo Vinicius Santos.

O projeto propôs um modelo de anotação de imagens através da representação de seu conteúdo com uso de Web Semântica para dar suporte à sua classificação em aplicações de recuperação.

7.4 A WEB SEMÂNTICA NO CONTEXTO EDUCATIVO UM SISTEMA PARA A RECUPERAÇÃO DE OBJECTOS DE APRENDIZAGEM BASEADO NAS TECNOLOGIAS PARA A WEB SEMÂNTICA, PARA O E-LEARNING E PARA OS AGENTES

Dissertação submetida à Universidade do Porto, de Portugal para à obtenção do grau de Doutor em Engenharia Eletrotécnica e de Computadores, realizada por Vitor Manuel Barrigão Gonçalves.

A finalidade deste projeto era explorar a aplicação das tecnologias para a Web Semântica no contexto educativo. Recorrendo a um estudo de campo, este projeto teve como propósito propor uma arquitetura para a recuperação de conteúdos educativos e, com base nela, implementar um protótipo experimental e determinar a influência de um conjunto de variáveis associadas (especificações para os metadados e para as ontologias e mecanismos para a inferência) num determinado contexto educativo – os sistemas de e-Learning e os repositórios de objetos de aprendizagem, para orientar a localização e recuperação de informação (GONÇALVES, 2007)

8 PROTÓTIPO DE APLICATIVO ANDROID PARA EXTRAÇÃO DE TEXTO EM IMAGENS DE RÓTULOS DE CERVEJAS PARA BUSCA SEMÂNTICA DAS INFORMAÇÕES

A pesquisa proposta tem como resultado um protótipo de aplicativo para dispositivos móveis que contém informações relevantes sobre as diferentes cervejas comercializadas no mercado.

O protótipo oferece um meio ao consumidor de obter as informações relevantes dos produtos desejados através de uma imagem de seu rótulo, onde a única entrada de dados manual que o consumidor deve fazer é informar o tipo de cerveja que está tomando e tirar uma foto do rótulo do produto. Após a foto ter sido carregada pelo protótipo, seja através da câmera ou da galeria de imagens, será possível gravar estas fotos com informações que serão trazidas de uma busca realizada no Google.

Para a realização desse protótipo foram utilizadas tecnologias como: Android, Java EE, Tesseract, Web Services RESTful, Tomcat 7 e banco de dados PostgreSQL.

8.1 METODOLOGIA

O método utilizado durante o desenvolvimento do projeto foi primeiramente fazer um levantamento bibliográfico, a fim de entender o conceito e definição, para obter boas referências das tecnologias para futuramente ser feito a prática do protótipo.

A fundamentação teórica foi organizada de maneira a direcionar a implementação, aumentando a produtividade no desenvolvimento.

Primeiro foi indicado a importância dos buscadores na obtenção de conteúdos da web, que tanto tem a oferecer para o usuário na formação de conhecimento e o quanto os mesmos estão evoluindo e tentando de maneira inovadora, entender realmente o que o usuário está pesquisando, e então foi citado o Google e sua evolução. Logo após, foi realizado uma pesquisa sobre o conceito de semântica e analisado a estrutura da web semântica, que é uma das estruturas mais

novas para se organizar o conteúdo da web de maneira que possa ser compreendida pelos buscadores.

Foi também realizada uma abordagem sobre o tema de desenvolvimento mobile, e de seu crescimento devido ao aumento no uso de smartphones. Também foi realizado uma pesquisa sobre a plataforma Android, que foi a plataforma escolhida para o desenvolvimento do protótipo. Além disso, foi explicado a comunicação entre sistemas diferentes que utilizam de tecnologias como WS para se comunicar e como é feita essa comunicação. Por fim foi explicado a tecnologia do OCR e seus processos para o reconhecimento de texto em imagem.

Com base nessa fundamentação, para a realização do trabalho foi necessário desenvolver duas aplicações: uma aplicação cliente e uma aplicação web responsável por atender as solicitações da aplicação cliente através de serviços de um Web Service RESTful.

No sistema web está hospedado o servidor que administra o fluxo de dados, contendo a aplicação da tecnologia OCR e efetuando a busca das cervejas não cadastradas no banco diretamente de informações extraídas do Google.

A aplicação cliente, desenvolvida em Android, é responsável por acessar os serviços RESTful do servidor, trazendo as informações cadastradas no banco de dados e disponibilizando-as em sua interface.

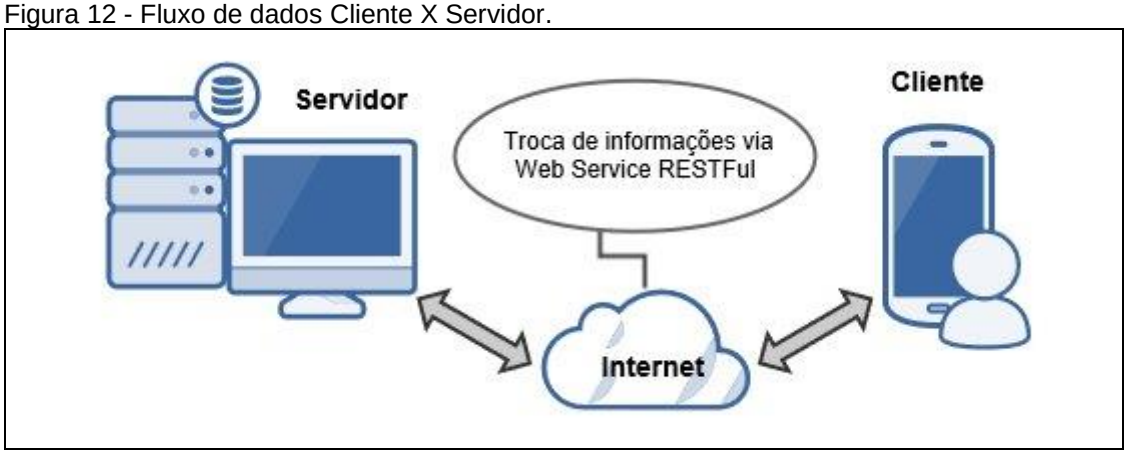
A aplicação web foi desenvolvida em Java através da IDE NetBeans 8.0 utilizando servidor Tomcat 7.0.63, uma biblioteca para o desenvolvimento de webservices RESTful chamada Jersey, a aplicação Tesseract, utilizada para o reconhecimento óptico de caracteres, banco de dados PostgreSQL 9.3 para realizar a persistência dos dados e Maven, responsável por obter as dependências necessárias para o projeto através de seu repositório.

A aplicação cliente foi criada com a plataforma Android Studio versão 1.3.2, incluindo a utilização da biblioteca Picasso para carregar as imagens do servidor, e a biblioteca Asynchronous Http Client para fazer as requisições ao WS.

8.2 FLUXO DAS INFORMAÇÕES

Por existir aplicações de plataformas diferentes como uma aplicação cliente e um servidor, a comunicação entre estas foi realizada através de um Web Service RESTful, que consumirá as requisições, produzindo respostas no formato JSON, que serão interpretadas pela aplicação cliente. Um exemplo deste fluxo pode ser visualizado na figura 12.

Figura 12 - Fluxo de dados Cliente X Servidor.



Fonte: Do autor.

O servidor é responsável por toda a regra de negócio do protótipo, assim como o reconhecimento dos caracteres das imagens enviadas, acesso e manipulação dos dados oriundos do banco de dados, armazenamento dos dados e a busca de informações não cadastradas através do motor de busca do Google.

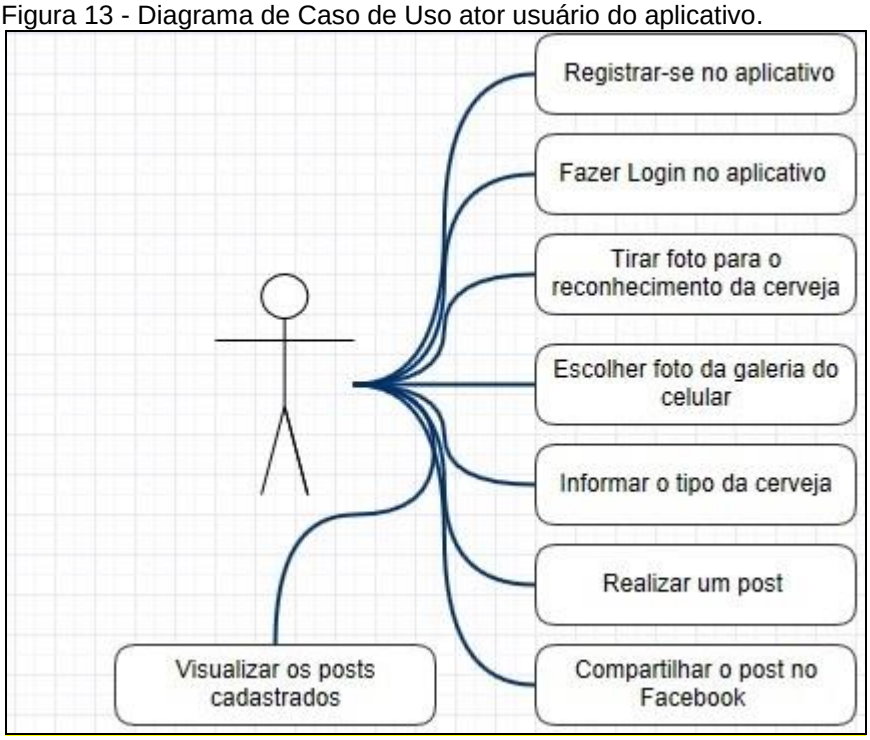
Para usar esta arquitetura de WS RESTful, foi utilizado a API Jersey versão 2.19, que trata-se de um framework responsável por simplificar o desenvolvimento de serviços RESTful.

8.3 MODELAGEM

Para ajudar na identificação de itens importantes e evitar erros futuros, foram criados modelos que servem para indicar os requisitos e a organização do protótipo a ser desenvolvido.

O primeiro modelo disponível na figura 13 mostra o ator (usuário) do aplicativo, e as funções que o mesmo poderá efetuar no protótipo, que é a aplicação cliente.

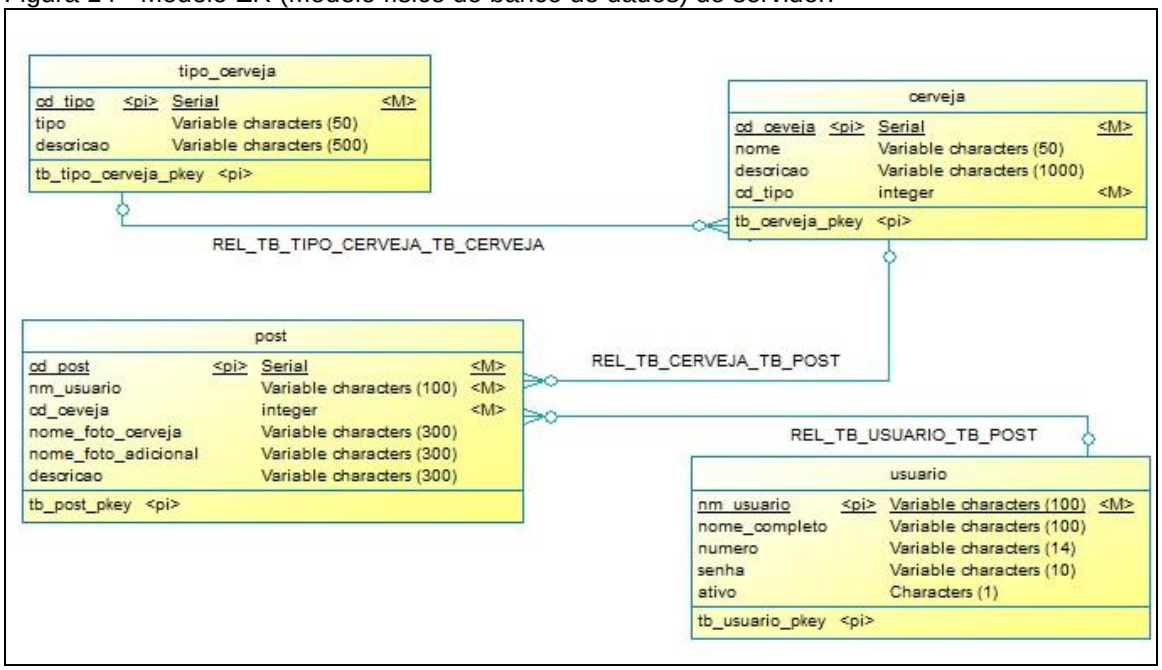
Figura 13 - Diagrama de Caso de Uso ator usuário do aplicativo.



Fonte: Do Autor.

Na figura 14 está o modelo físico do banco de dados, utilizado para o cadastro das informações e consultas do aplicativo. Para este, foi utilizado o Diagrama ER, que demonstra as entidades, seus atributos e como elas se relacionam, tal diagrama foi feito com o auxilio do software de modelagem de dados Power Design.

Figura 14 - Modelo ER (Modelo físico do banco de dados) do servidor.

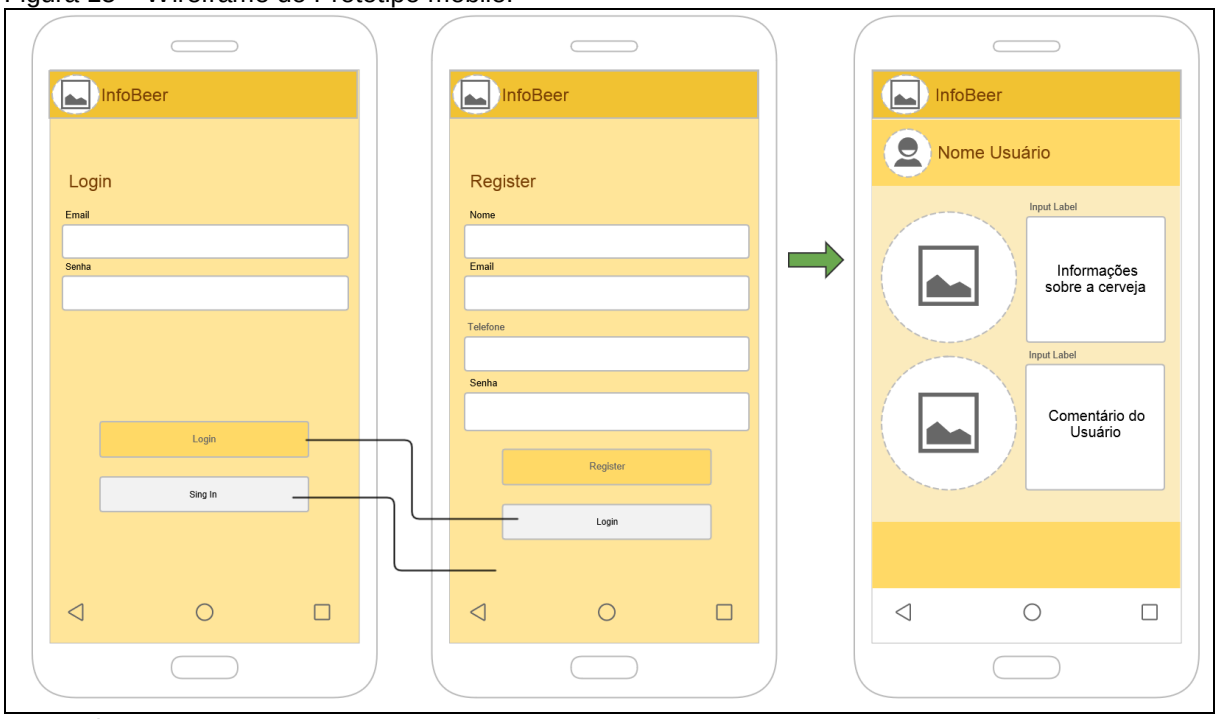


Fonte: Do Autor.

Podemos verificar na figura 14, que o banco conterá cadastrado as cervejas e os tipos de cervejas, assim como o cadastro de usuário os posts realizados. Com as informações armazenadas no banco de dados, o usuário terá acesso as informações sobre as cervejas, assim como consultar os momentos em que degustou uma cerveja diferente e até mesmo com quem esteve naquele momento, através do caminho da imagem que será armazenado no banco de dados, na tabela post.

Outro modelo feito foi o Wireframe, uma prototipação para auxiliar na identificação da posição dos campos e o funcionamento da aplicação. Nesse modelo é criado um esqueleto de como as informações serão visualizadas, os campos utilizados, e também a interface do Software, como mostra a figura 15.

Figura 15 – Wireframe do Protótipo mobile.



Fonte: do Autor.

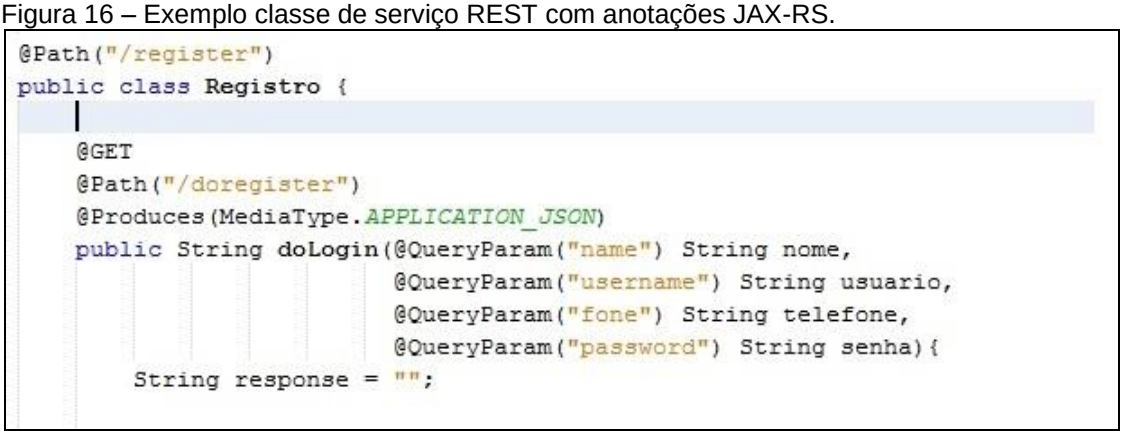
8.4 FRAMEWORK RESTFUL – JERSEY

Desenvolver Web Services RESTful que permitam expor seus dados em uma variedade de representações de tipos de mídia e abstraí-los dos detalhes de baixo nível da comunicação cliente-servidor, não é uma tarefa fácil sem boas ferramentas (ORACLE CORPORATION, 2015, tradução nossa).

Com o objetivo de simplificar o desenvolvimento de Web Services RESTful e de seus clientes em Java, uma API padrão e portável do JAX-RS foi criada. O framework Jersey para Web Services RESTful é open source e fornece suporte às APIs JAX-RS, servindo como implementação de referência (JSR 311 e JSR 339) do JAX-RS (ORACLE CORPORATION, 2015, tradução nossa).

Segundo JSR-311 (2009) no JAX-RS as requisições são tratadas por métodos de uma classe recurso. A classe recurso trata-se de uma classe java como outra qualquer, que possua as anotações da JAX-RS, indicando os mapeamentos e as operações existentes. Toda classe recurso deve possuir um construtor público, que poderá conter parâmetros de certas anotações, que realizarão a injeção de dependência do serviço REST, conforme podem ser visualizadas na figura 16.

Figura 16 – Exemplo classe de serviço REST com anotações JAX-RS.



```
@Path("/register")
public class Registro {

    @GET
    @Path("/doregister")
    @Produces(MediaType.APPLICATION_JSON)
    public String doLogin(@QueryParam("name") String nome,
                          @QueryParam("username") String usuario,
                          @QueryParam("fone") String telefone,
                          @QueryParam("password") String senha){
        String response = "";
```

Fonte: Do autor.

O framework Jersey, no entanto, é mais do que uma implementação de referência do JAX-RS. O Jersey fornece sua própria API que adiciona recursos e utilidades para simplificar o desenvolvimento de serviços RESTful e de seus clientes (ORACLE CORPORATION, 2015, tradução nossa).

8.5 FERRAMENTA OCR TESSERACT

Para o reconhecimento dos textos nos rótulos das cervejas, foi escolhida a ferramenta Tesseract, por ser provavelmente a ferramenta OCR de código aberto mais precisa disponível, tornando-se uma das três primeiras ferramentas no teste de precisão UNLV de 1995, sendo melhorada desde então pela Google. De acordo com o site oficial da ferramenta, é possível compilar o Tesseract para funcionar em outras plataformas, incluindo plataformas móveis, como o Android, por exemplo (CODE GOOGLE, 2015, tradução nossa).

A fim de testar esta ferramenta, primeiramente foi desenvolvida uma aplicação de testes separada do resto do protótipo, utilizando a linguagem Java e a IDE NetBeans 8.0.2. Esta aplicação de testes teve como objetivo testar a eficácia da ferramenta e sua usabilidade, assim como adquirir o conhecimento de suas funcionalidades e deixar preparado e organizado o código necessário para sua utilização posterior no Web Service RESTful.

Em testes, enviando imagens dos rótulos para a ferramenta, foi possível observar uma baixa taxa de reconhecimento dos caracteres contidos nas imagens. Foram então realizados testes utilizando uma técnica de pré-processamento de imagens, chamada Threshold, que converte as cores das imagens para preto ou branco, sem utilizar escalas de cinza. Com esta técnica, se tornou notável a melhora no reconhecimento dos caracteres nas fotos de rótulos das cervejas enviadas, conforme ilustra a figura 17.

Figura 17 - Técnica de Threshold.

Fonte: Do Autor.

Outro problema enfrentado foi que as imagens enviadas para a ferramenta Tesseract, por se tratarem de rótulos de cerveja, não continham apenas textos, mas sim figuras gráficas e outros textos além do que se desejava buscar, nos quais a ferramenta tentou reconhecer, poluindo o resultado desejado, que seria apenas o nome da cerveja.

Para a resolução deste problema, foi configurado um dicionário de palavras, disponibilizado pela própria ferramenta Tesseract, que faz com que em casos em que a ferramenta não reconheça todos os caracteres de uma palavra, ela buscará a palavra similar mais próxima encontrada neste dicionário, que não é nada além de um arquivo de texto com o vocabulário desejado, permitindo assim, reconhecer palavras específicas de um determinado tema.

Para o protótipo, as funcionalidades da ferramenta de OCR Tesseract foram abstraídas através do uso de uma biblioteca Java chamada Tess4J, em sua

versão 2.0, sendo esta incluída para utilização pelo Web Service RESTful através do Maven.

O protótipo foi projetado prevendo falhas no sistema OCR, no qual caso o nome da cerveja que estará contido no rótulo não for reconhecido e não existir no dicionário de palavras da ferramenta será então solicitado ao usuário informar manualmente o nome contido no rótulo, assim como o estilo da cerveja.

8.6 CUSTOM SEARCH GOOGLE

O Google Custom Search permite criar um motor de busca para o seu site, blog, ou uma coleção de sites. Você pode configurar seu mecanismo de busca para procurar tanto em páginas da Web como em imagens (GOOGLE DEVELOPERS, 2015, tradução nossa).

Ainda segundo Google Developers (2015, tradução nossa), há dois casos de uso principais para a pesquisa personalizada - você pode criar um motor de busca que procura somente o conteúdo de um site (pesquisa do site), ou você pode criar um que incida sobre um determinado tópico a partir de vários sites. É possível utilizar o Custom Search também para priorizar ou ignorar sites.

Existem dois tipos de licenças, um gratuito e um pago, chamado Google Site Search. A diferença é que o Google Site Search permite criar motores de busca que não incluam anúncios e que remova a marca Google (se você preferir). Além disso, os clientes do site de pesquisa Google podem recuperar resultados em XML, para que se tenha mais controle sobre como os resultados são apresentados aos usuários (GOOGLE DEVELOPERS, 2015, tradução nossa).

Há duas maneiras de começar o motor de busca personalizado, seja pelo painel de controle de desenvolvedor do Google ou por XML (GOOGLE DEVELOPERS, 2015, tradução nossa).

Segundo Google Developers (2015), após criar um Mecanismo de pesquisa personalizado, é possível usar uma página padrão oferecido pelo Google para mostrar seus resultados para o usuário, ou então incorporar a funcionalidade de pesquisa diretamente no seu site. Existem ferramentas disponibilizadas para o desenvolvedor personalizar os resultados da pesquisa:

a) a API de controle personalizado de Pesquisa permite incorporar elementos Google Custom Search em páginas da web e outras aplicações web usando JavaScript;
b) a API JSON / Atom permite desenvolver sites e programas para recuperar e exibir resultados de pesquisa do Google Custom Search programaticamente. Com esta API, pode-se usar as solicitações RESTful para obter resultados de pesquisa em qualquer formato JSON ou Atom;
c) a API XML, através de um aplicativo cliente, é possível recuperar uma lista de motores de busca personalizado (CSE) em uma conta, criar e excluir os motores de busca personalizados e recuperar o código para uma caixa de pesquisa. Este recurso está disponível apenas para clientes do Google Site Search.

Para o protótipo, primeiramente foi criada uma chave de API, através do Google Developers Console. Uma chave de API é um identificador único que se cria através do Google Developers Console. Clientes premium, geralmente possuem um ID ao invés de uma chave. Cada plataforma exige um diferente tipo de chave:

a) chave do servidor: para uso com APIs de Web Services;
b) chave de browser: para uso em APIs Web;
c) chave do android: para uso com APIs Android;
d) chave do iOS: para uso com APIs do iOS (GOOGLE DEVELOPERS, 2015, tradução nossa).

Nesse caso foi criado uma chave de servidor, já que as requisições de pesquisa do Google são realizadas pelo servidor do protótipo. O servidor RESTful do protótipo, utiliza um motor de busca personalizado, criado através do Google Custom Search Engine. Este motor de busca personalizado está configurado para realizar buscas no idioma português, dando prioridade aos resultados referentes ao site www.brejas.com.br.

Para que o motor de busca personalizado seja utilizado, o servidor realiza uma requisição a uma URL do servidor RESTful da Google, passando como parâmetro o ID do mecanismo de pesquisa, que foi obtido no momento da criação do

motor de busca personalizado, e o contexto da pesquisa, como pode ser verificado na figura 18.

Figura 18 – Pesquisa utilizando o Custom Search da Google.

```
public class PesquisaGoogle {
    final String URL_GOOGLE = "https://www.googleapis.com/customsearch/v1";
    String key = "AIzaSyB_8MFHsU6kq2gw9XD-lgC0p5F-KK2zY3c";
    String cx = "006006104017362856352:q7syobkayk0";

    public String pesquisar(String nomeCerveja, String tipoCerveja) {
        String pesquisa = nomeCerveja + " " + tipoCerveja;
        pesquisa = pesquisa.replaceAll(" ", "%20");
        String descricao = null;
        try {
            URL url = new URL(URL_GOOGLE + "?key="+key+ "&cx=" + cx + "&q="+ pesquisa + "&alt=json");
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestMethod("GET");
            conn.setRequestProperty("Accept", "application/json");
            BufferedReader br = new BufferedReader(new InputStreamReader(
                    (conn.getInputStream()), "UTF-8"));

            String output;
            while ((output = br.readLine()) != null) {

                if(output.contains("\"snippet\": \"")){
```

Fonte: Do autor.

Após o Web Service Google realizar a pesquisa e retornar as informações em formato JSON, o servidor do protótipo trata estas informações, e caso esta cerveja não conste no banco de dados do protótipo, o servidor realizará o cadastro automático das informações obtidas. Em uma próxima vez que esta mesma informação for requerida, o servidor utilizará as informações cadastradas, melhorando o tempo de resposta.

8.7 PICASSO

Picasso é uma biblioteca para Android que realiza o download de imagens de determinada URL, realizando automaticamente o cache das imagens para quando estas forem visualizadas novamente, facilitando a implementação de requisições assíncronas para o download das imagens.

As principais características desta biblioteca, conforme indicado no site oficial são:

a) transformação complexa de imagens com um uso mínimo de memória;

b) caching de disco e memória automáticos (SQUARE 2015, tradução nossa).

Esta biblioteca foi utilizada para carregar as imagens já postadas pelo usuário anteriormente, no protótipo cliente, através de seus nomes, armazenados no banco de dados do Servidor.

Ao enviar uma imagem para o servidor RESTful, a mesma é armazenada em um diretório específico do servidor, e para que estas imagens estejam visíveis pela aplicação Web, foi utilizado um recurso do servidor Tomcat, através do arquivo de configuração server.xml, que disponibiliza o acesso aos arquivos contidos em um diretório do servidor para a aplicação Web, através de uma URL, conforme pode ser visualizado na figura 19.

Figura 19 – Configuração no server.xml do servidor Tomcat.

```
<Host appBase="webapps" autoDeploy="true" name="localhost" unpackWARs="true">
  <!-- As imagens do diretório d:\uploaded são disponibilizadas através da URL /uploaded,
  logo após o endereço do servidor, por exemplo http://localhost:8081/uploaded/,
  pela configuração realizada na linha abaixo -->
  <Context  docBase="D:\uploaded"   path="/uploaded" />

  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
  pattern="%h %l %u %t &quot;%r&quot; %s %b" prefix="localhost_access_log." suffix=".txt"/>

</Host>
```

Fonte: Do autor.

Um exemplo de código utilizando a biblioteca Picasso para carregar uma imagem de uma URL dentro de um ImageView, pode ser visualizado na figura 20.

Figura 20 - Código carregar imagem com a biblioteca Picasso.

```
Picasso.with(context).load("http://i.imgur.com/DvpvklR.png").into(imageView);
```

Fonte: Square (2015).

8.8 FACEBOOK SDK

O Facebook SDK para o Android é a maneira mais fácil de integrar uma aplicação Android com o Facebook, permitindo:

a) login no Facebook: Permite autenticação com as credenciais do Facebook;

b) diálogos de compartilhamento e envio: Permite compartilhar conteúdos do aplicativo com o Facebook;
c) eventos do Aplicativo: Registra eventos na sua aplicação;
d) Graph API: Permite ler e escrever a API Graph (FACEBOOK DEVELOPERS, 2015, tradução nossa).

Para o protótipo foi utilizado a versão 4.7.0 do Facebook SDK, com o objetivo de possibilitar o compartilhamento de um link pelo Facebook. Este link direciona a uma galeria personalizada que cada usuário possui, demonstrando todas as cervejas já enviadas pelo usuário de maneira interativa, através de uma página Web.

Para que seja realizado o compartilhamento das informações no Facebook, é necessário seguir os seguintes passos (FACEBOOK DEVELOPERS, 2015, tradução nossa):

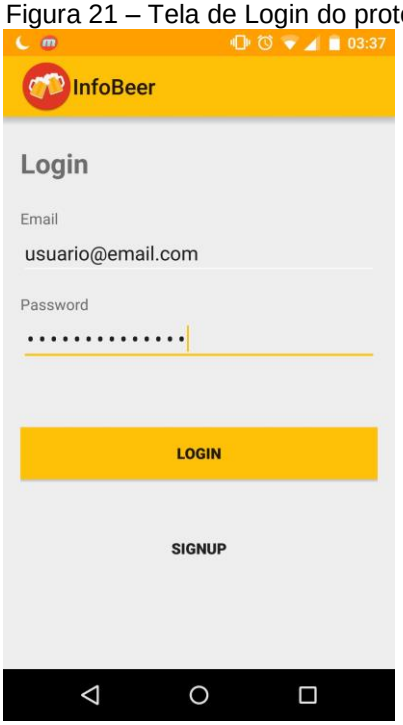
a) adicionar o Facebook SDK para Android no seu ambiente de desenvolvimento móvel;
b) conseguir um Facebook App ID configurado de maneira apropriada e relacionar a sua aplicação Android;
c) gerar uma chave Android e adicioná-la ao seu perfil de desenvolvedor;
d) adicionar uma Facebook Activity e incluí-la ao AndroidManifest.xml.

8.9 DESENVOLVIMENTO PROTÓTIPO MOBILE

O protótipo mobile permite realizar requisições e consumir as respostas em JSON geradas pelo servidor REST. Primeiramente foram criadas duas classes genéricas, sendo uma para efetuar requisições assíncronas simples, chamada RequisicaoActivity.java, responsável por realizar o envio de informações textuais através do método GET e POST do protocolo HTTP, utilizando a biblioteca AsyncHttpClient. Desta forma, classes especializadas como as classes LoginActivity.java e RegisterActivity.java, responsáveis respectivamente pelo Login e Registro do usuário, poderão utilizar, através da herança, os métodos genéricos de requisição e resposta desta classe abstrata, sobrescrevendo-os quando necessário.

Na figura 21 pode ser verificada a tela de Login do protótipo móvel, que realizará uma requisição assíncrona para a validação das credenciais do usuário.

Figura 21 – Tela de Login do protótipo.



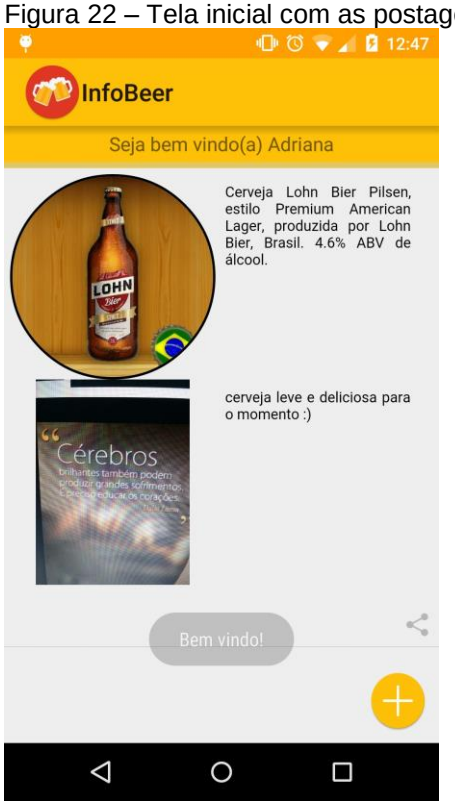
Fonte: Do Autor.

Uma segunda classe chamada RequisicaoUploadImagem.java foi criada para o envio assíncrono de requisições com conteúdo binário, como imagens, para o servidor REST. Para isto, foram utilizadas as bibliotecas HttpMime e HttpClient por motivo de conhecimento prévio do acadêmico. No entanto, um dos problemas em utilizar a biblioteca HttpClient, é que a mesma não realiza automaticamente requisições assíncronas. Desta forma, foi necessário que esta classe genérica utilize recursos da classe AsycTask, por possuir um método chamado doInBackground, onde tarefas assíncronas são realizadas, estendendo suas funcionalidades por meio da herança e sobrescrevendo este método, onde então pôde-se utilizar a biblioteca HttpClient, obtendo como resultado, uma requisição assíncrona.

Através das requisições assíncronas, o protótipo mobile permite ao usuário realizar seu cadastro ou o seu login, informando suas credenciais, que serão enviadas para o web service RESTful, através da classe genérica criada para requisições assíncronas chamada RequisicaoActivity.java, e então validadas. Após

ter suas credenciais validadas, o protótipo iniciará a *Activity* HomeUserActivity.java através do método abrirListaPosts(), que irá realizar uma requisição assíncrona ao web service RESTful solicitando a lista de publicações anteriores do usuário, através do método invokeWS, definido na classe pai abstrata RequisicaoActivity.java, e disponibilizando estas publicações em uma lista visível ao usuário, conforme pode ser visto na Figura 22.

Figura 22 – Tela inicial com as postagens realizadas.



Fonte: Do Autor.

Uma nova publicação pode ser adicionada através do evento de clique do botão btnAddPost, que iniciará a *Activity* CameraActivity.java.

Para cada publicação serão necessárias duas fotos, uma foto da cerveja no qual se deseja obter informações, e uma foto adicional, onde ambas podem ser tiradas através da câmera, ou selecionadas da galeria.

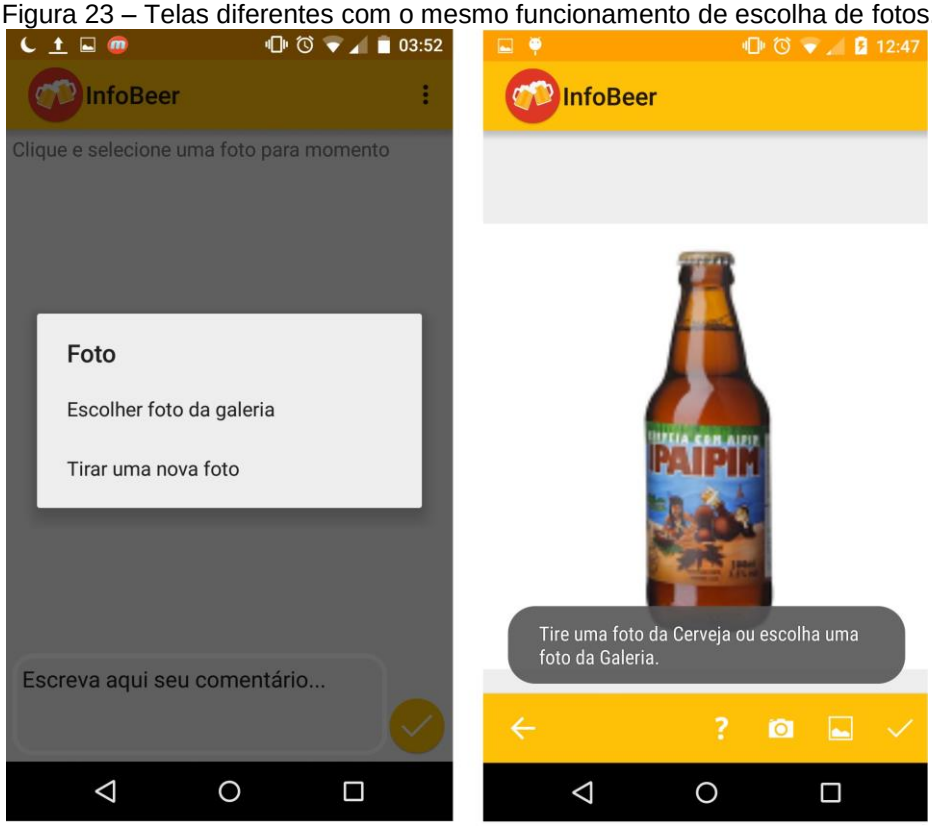
A *Activity* CameraActivity.java é responsável pela entrada da imagem da cerveja, seja através de uma nova foto ou de uma foto da galeria. Caso seja optado por tirar uma nova foto, será chamado o método novaMidia() da classe UtilityCamera.java, passando um parâmetro pré-determinado chamado

“MIDIA_FOTO”, pertencente a esta mesma classe. Este método é responsável por salvar a foto recém-tirada em uma pasta própria do aplicativo, chamada info-beer. Caso seja optado por uma imagem da galeria, será passado a uma nova *Intent* o parâmetro Intent.ACTION_PICK, interno do Android, responsável pela ação de escolher uma imagem da galeria.

Ao selecionar uma imagem, seja através de uma foto nova ou de uma imagem retirada da galeria, o caminho da imagem é enviada como parâmetro para a próxima *Activity,* chamada TipoCervejaActivity.java.

Na classe TipoCervejaActivity.java, existe o método invokeWSTipoCerveja(), responsável por fazer um requisição assíncrona ao web service RESTful e trazer os tipos de cervejas cadastrados no banco, para então exibi-las em uma lista, onde um tipo de cerveja deverá ser selecionado pelo usuário. Após ser selecionado o tipo da cerveja, é realizado o upload da foto da cerveja ao web service RESTful, e enviado como parâmetro o código do tipo de cerveja selecionada. Ao finalizar o upload da imagem e obter as informações da cerveja vindas do web service, essas informações são passadas para a classe PostActivity.java, onde se encontra a etapa final da publicação. Nesta classe existem métodos similares aos da classe CameraActivity.java, onde há a opção de tirar uma nova foto adicional ou selecionar uma foto adicional da galeria. Além disso, têm-se a possiblidade de informar um comentário que será relacionado à publicação, como pode ser verificado na Figura 23.

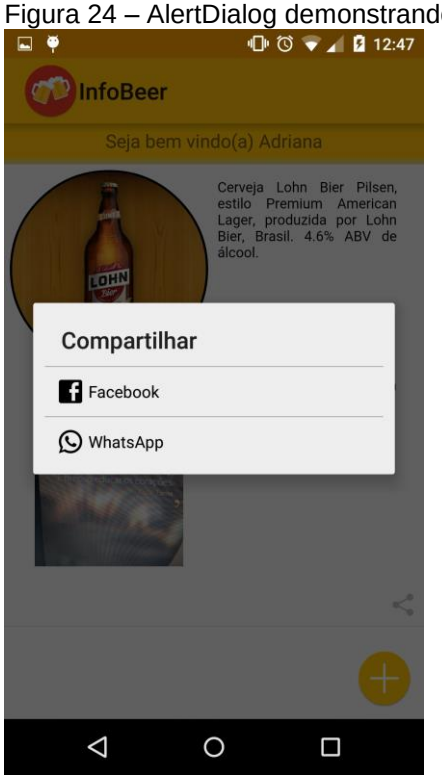
Figura 23 – Telas diferentes com o mesmo funcionamento de escolha de fotos.



Fonte: Do Autor.

O protótipo também possibilita ao usuário compartilhar esta publicação, na classe HomeUserActicity.java, ao clicar sobre o ícone de compartilhamento na lista de publicações do usuário, onde será exibido um *AlertDialog*, para que seja selecionado onde a publicação será compartilhada, no WhatsApp ou Facebook, conforme pode ser verificado na Figura 24.

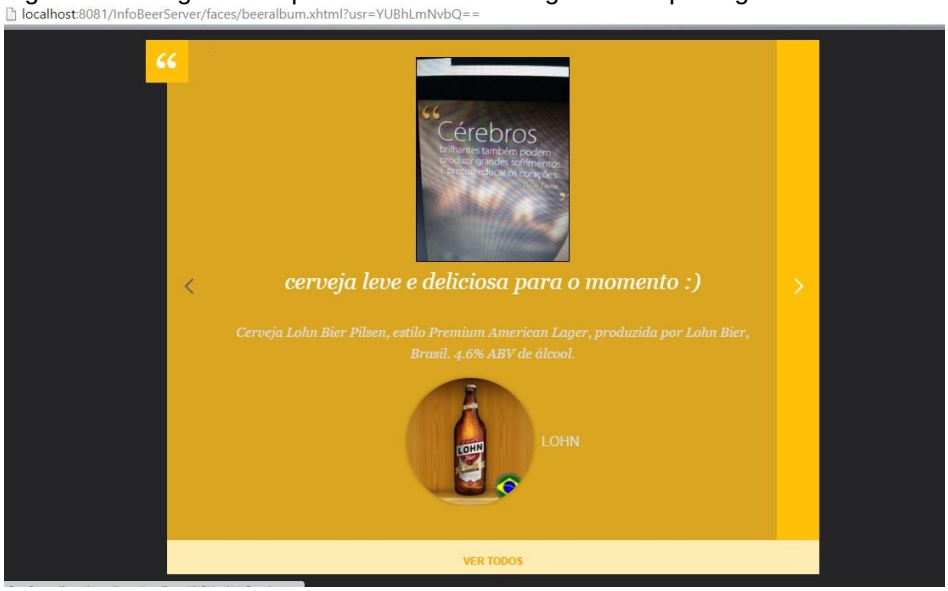
Figura 24 – AlertDialog demonstrando as opções de compartilhamento.



Fonte: Do Autor.

Caso selecionado o compartilhamento pelo Facebook, será exigido as credenciais do Facebook apenas se o usuário ainda não estiver logado no aplicativo nativo do Facebook em seu *smartphone*. O conteúdo da publicação compartilhada pelo Facebook, trata-se de um link de uma página web contida no mesmo servidor que o web service RESTful, com uma galeria personalizada, exibindo os dados das suas últimas 20 postagens do usuário, como pode ser verificado na figura 25.

Figura 25 – Página web personalizada com a galeria de postagens do usuário.



Fonte: Do Autor

Caso for escolhido compartilhar a informação pelo WhatsApp, existe nessa classe uma chamada a uma ação de compartilhar mídia e texto, nativo do próprio Android, realizada pelo envio do parâmetro Intent.ACTION_SEND, enviado para uma *Intent* através do método setAction(). É escolhido o pacote da aplicação do WhatsApp através do método setPackage(), passando como parâmetro a *string* com.whatsapp, pois sem definir o pacote da aplicação, o Android forneceria varias outras aplicações não relevantes como opção de compartilhamento.

8.10 RESULTADOS OBTIDOS

Por meio da pesquisa realizada e da aplicação das ferramentas estudadas, como por exemplo, a ferramenta Tesseract e o Google Custom Search Engine, foi desenvolvido um protótipo com o objetivo de auxiliar o usuário a obter informações sobre cervejas de diferentes estilos e fabricantes de maneira prática.

Com a realização de testes no protótipo, foi possível verificar que o mecanismo de OCR do Tesseract funciona de maneira adequada em rótulos de cervejas que não possuam muitos símbolos ou fontes estilizadas. Para o reconhecimento de rótulos complexos, ou seja, que envolvam muitas figuras, seria necessário um treinamento extensivo da ferramenta, assim como a aplicação de

técnicas de pré-processamento de imagens, como normalização e outras que vão além da técnica de thresholding, utilizada neste protótipo.

Em testes de reconhecimento realizado com fotos retiradas do rótulo de 10 cervejas, em um ambiente claro, foi possível reconhecer o nome de 06 cervejas, ou seja, uma taxa de acerto de 60%, que se mostrou uma taxa aceitável, dado o número de figuras e gráficos contidos em alguns dos rótulos. Entre as 06 cervejas reconhecidas, para 05 destas, as informações obtidas pela busca foram relevantes às imagens enviadas, enquanto que em uma delas, a pesquisa retornou uma descrição em inglês, porém com seu significado também relevante. Para cada uma das 04 cervejas não reconhecidas, foi possível incluir o nome de maneira manual, onde após este processo, foram retornadas as informações conforme o esperado.

Foi observado também, que a posição em que a foto é tirada, assim como a rotação da câmera ou da própria garrafa, além de fatores como iluminação do ambiente e foco da imagem, podem influenciar, causando taxas pequenas de reconhecimento, exigindo intervenção manual por parte do usuário.

Em testes também foi constatado a importância de um contexto na busca, pois em muitos casos, o nome da cerveja coincide com o nome de alguma outra coisa, seja um país ou animal. O contexto acrescentado na busca da cerveja para este protótipo refere-se ao tipo da cerveja pesquisada, no qual sem este contexto, ou seja, buscando apenas pelo nome da cerveja, foi possível observar em alguns casos, um retorno de informações não relacionadas.

O protótipo se demonstrou uma maneira simples e portável de se obter informações das cervejas desconhecidas, buscando rapidamente estas informações e disponibilizando-as em uma lista de publicações, assim como a possibilidade de compartilhamento das publicações por meio de redes sociais.

Levando em consideração os resultados analisados, o protótipo demonstrou atingir todos os objetivos propostos, possibilitando aprimoramentos futuros.

9 CONCLUSÃO

As pesquisas e os estudos realizados, assim como o desenvolvimento do protótipo proposto durante a execução deste trabalho de conclusão de curso, proporcionaram conhecimento em diversas áreas.

Desde o início do desenvolvimento do protótipo utilizando a plataforma móvel da Google, chamada Android, foi perceptível o quanto cada vez mais, os recursos, as bibliotecas, e até mesmo o próprio hardware, evoluem para melhor atender a necessidade dos consumidores, tornando possível a oferta de serviços específicos através do acesso às informações de seus usuários. Além disto, foi perceptível também a praticidade no próprio desenvolvimento para esta plataforma, já que conta com uma vasta variedade de bibliotecas que possuem o objetivo de abstrair a complexidade de determinados recursos.

Utilizando-se dos conhecimentos obtidos, foi possível atingir os objetivos propostos, dentro os quais o principal, que se trata da extração das informações impressas contidas no rótulo de cervejas e da realização de busca semântica através destas informações extraídas.

Os estudos e testes realizados foram imprescindíveis para o cumprimento dos objetivos, onde todos foram atingidos com sucesso, tanto o objetivo geral quanto os específicos. Estes estudos auxiliaram na pesquisa e desenvolvimento do trabalho, fornecendo conhecimentos teóricos e práticos sobre as tecnologias, que envolveram a leitura óptica de caracteres, a busca semântica por informações através do motor de busca Google, o desenvolvimento de aplicações para plataforma móvel Android, e o desenvolvimento de web services RESTful com Java, oferecendo a oportunidade de aprimoramento em futuros trabalhos que utilizem estas tecnologias. Os estudos auxiliaram também a superar as dificuldades encontradas durante a fase de implementação do protótipo, incluindo dificuldades como a criação de um protótipo móvel que seja de fácil usabilidade, a extração apenas do nome contido em um rótulo de cerveja, e o envio de informações binárias para um web service RESTful.

O protótipo desenvolvido neste trabalho possibilita o compartilhamento das informações da cerveja na rede social Facebook, a fim de difundir a informação,

buscando incentivar o consumo moderado e racional de novas cervejas, tendo como objetivo a degustação.

Durante testes com a tecnologia OCR utilizando Tesseract, o protótipo mostrou-se sensível na resposta, ou seja, vários fatores podem afetar a resposta dependendo do processo de aquisição da imagem, como por exemplo, problemas na iluminação, alinhamento do texto e rotação da imagem ou da câmera. Para o protótipo móvel foi desenvolvido uma solução alternativa para contornar este problema, que consiste em uma solução simples, que se trata de solicitar ao usuário que informe o nome da cerveja quando a mesma não for reconhecida automaticamente pelo OCR.

Para trabalhos futuros que utilizarem tecnologias semelhantes, pode-se sugerir:

a) aplicar outras técnicas de pré-processamento da imagem, tais como normalização, correção de ruídos e correção de luminosidade da imagem;
b) realizar a busca de informações de determinado produto utilizando outros motores de busca quando as informações encontradas não forem satisfatórias ou não preencherem uma quantidade mínima de caracteres;
c) incorporar um sistema de ranking, onde cada usuário do aplicativo poderá realizar uma avaliação do produto consumido;
d) criar uma Rede Social para apreciadores de cervejas com sua própria busca e anotações semânticas.
e) utilizar a tecnologia OCR e a busca semântica de informações para propósitos sociais, como o auxílio a deficientes visuais.
f) desenvolver um motor de busca próprio que faça uso da web semântica
g) implementar a leitura de QR code impresso nos rótulos dos produtos.

REFERÊNCIAS

ALVES, Neide Ferreira. **Estratégias para melhoria do desempenho de ferramentas comerciais de reconhecimento óptico de caracteres**. 2003. 108 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Federal de Pernambuco, Recife, 2003.

ANDROID. **Android system architecture**. [2015]a. Disponível em: <https://source.android.com/devices/#Android system architecture>. Acesso em: 06 jun 2015.

ANDROID. **The Android Source Code.** [2015]b. Disponível em: <https://source.android.com/index.html>. Acesso em: 19 jun 2015.

BERNERS-LEE, Tim; HENDLER, James; LASSILA, Ora. The Semantic Web. **Scientifc American,** Canadá, v. 284, n. 5, p.28-37 (1,2), 2001.

BORTOLIN, Ricardo Soares. **O modelo de negócio Google:** Estudo de caso sobre o Google Adwords. 2009. 65 f. Monografia (Especialização) - Curso de Ciência da Computação, Universidade Federal de Lavras, Minas Gerais, 2009.

BOX, Don et al. **Simple Object Access Protocol (SOAP) 1.1.** 2000. W3C Note 08 May 2000. Disponível em: <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>. Acesso em: 17 abr. 2015.

BRANSKI, Regina Meyer. Localização de Informações na Internet: características e formas de funcionamento dos mecanismos de busca. **Transinformação,** Campinas, v. 12, n. 1, p.11-19, jan. 2000. Quadrimestral. Disponível em: <http://www.scielo.br/pdf/tinf/v12n1/08.pdf>. Acesso em: 19 jun. 2015.

BURNS, Christa; SAUERS, Michael P.. **Google Search Secrets.** Chicago: American Library Association, 2014. 202 p.

CAMPANHOLI, Vanderson Zanoni. **Eplants**: Protótipo de aplicativo acadêmico em plataforma android para catalogação de plantas. 2014. 85f. Trabalho de Conclusão de Curso – Universidade do Extremo Sul Catarinense, Criciúma, 2014.

CENDÓN, Beatriz Valadares. Ferramentas de Busca na Web. **Ciência da Informação,** Brasília, v. 30, n. 1, p.39-49, jan. 2001. Disponível em: <http://www.scielo.br/pdf/ci/v30n1/a06v30n1>. Acesso em: 26 jun. 2015.

CERAMI, Ethan. **Web Services Essentials:** Distributed Applications with XML-RPC, SOAP, UDDI & WSDL. Publisher: O'Reilly, 2002.

CODE GOOGLE. **Tesseract Ocr.** Disponível em: <https://code.google.com/p/tesseract-ocr/>. Acesso em: 23 out. 2015.

CUNHA, Tiago vargas. **Competitividade e segmentação na indústria cervejeira**: Uma Análise da Competitividade das Microcervejarias Catarinense. 2011. Disponível em: <http://tcc.bu.ufsc.br/Economia299002.pdf>. Acesso em: 07 out. 2014.

ECKSTEIN, Robert; CASABIANCA, Michel. **XML:** Pocket Reference**.** 3. ed. Sebastopol: O'reilly, 2001.

EIKVIL, Line. **OCR - Optical Character Recognition.** 1993. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.25.3684>. Acesso em: 19 jun. 2015.

ENGENHOFER, Max J.. **Toward the semantic geospatial web**. Proceedings of the 10th ACM international symposium on Advances in geographic information systems, p.1-4, November 08-09, 2002, McLean, Virginia, USA. Disponível em: <https://www.ischool.utexas.edu/~i385t-sw/readings/Egenhofer-2002-Geospatial_Web.pdf>. Acesso em: 25 maio 2015.

FAWCETT, Joe; QUIN, Liam; AYERS, Danny. **Beginning XML.** 5. ed. Indianapolis: Wrox, 2012.

GARGENTA, Marko. **Learning Android.** Sebastopol: O'reilly, 2011.

GONÇALVES, Vitor. **A web semântica no contexto educativo::** um sistema para a recuperação de objectos de aprendizagem baseado nas tecnologias para a web semântica, para o e-learning e para os agentes.. 2007. 150 f. Tese (Doutorado) - Curso de Faculdade de Engenharia, Universidade do Porto, Porto, 2007.

GOOGLE DEVELOPERS. **Usando as APIs do Google Maps**. [2015]a. Disponível em: <https://developers.google.com/maps/faq>. Acesso em: 28 out. 2015.

GOOGLE DEVELOPERS. **Overview**. [2015]b. Disponível em: <https://developers.google.com/custom-search/docs/overview>. Acesso em: 26 out. 2015.

GRUBER, Thomas R.. **A Translation Approach to Portable Ontology Specifications.** 1993. Appeared in Knowledge Acquisition , 5 (2):199-220, 1993. Disponível em: <http://tomgruber.org/writing/ontolingua-kaj-1993.pdf>. Acesso em: 26 jul. 2015.

GULABANI, Sunil. **Developing RESTful Web Services with Jersey 2.0:** Create RESTful web services smoothly using the robust Jersey 2.0 and JAX-RS APIs. Birmingham: Packt Publishing, 2014.

HOUAISS, Antônio; VILLAR, Mauro de Salles; FRANCO, Francisco Manoel de Mello. **Dicionário Houaiss da língua portuguesa**. Rio de Janeiro: Objetiva, 2004. 2540p.

IDC. **Smartphone Vendor Market Share, Q1 2015.** Disponível em: <http://www.idc.com/prodserv/smartphone-market-share.jsp>. Acesso em: 19 abr 2015.

JSR-311 e Jersey. **Java Magazine.** Rio de Janeiro, v. 144, n. 60, 2009. Mensal.

KUMAR, Sandeep. **Agent-Based Semantic Web Service Composition**. New York: Springer, 2012. 60p.

LEAL, Nelson Glauber de Vasconcelos. **Dominando o Android do Básico ao Avançado**. São Paulo: Novatec, 2015.

MAKICE, Kevin. **Twitter API:** Up and Running. Sebastopol: O'Reilly. 2009. 391p.

MARCILIO, Claiton de Melo. **Protótipo de Aplicativo Android para Extração de Texto em Imagens e Conversão em Voz, Orientado ao Apoio a Leitura para Deficientes Visuais**. 2013. 82f. Trabalho de Conclusão de Curso – Universidade do Extremo Sul Catarinense, Criciúma, 2013.

MARTINS, Victor Manuel Moreira. **Integração de Sistemas de Informação**: perspectivas, normas e abordagens. 2005. 218 f. Dissertação (Mestrado) – Universidade do Minho, Portugal: Guimarães, 2005.

MEIER, Reto. **Professional Android 4 Application Development**. Wrox, 2012.

MELO, Carlos Alexandre Barros de. **FILTRAGEM, COMPREENÇÃO E SINTESE DE IMAGENS DE DOCUMENTOS HISTORICOS.** 2012. Tese (Doutorado). Universidade Federal de Pernanbuco, Recife, 2012.

OPEN HANDSET ALLIANCE. **Overview**. Disponível em: <http://www.openhandsetalliance.com/oha_overview.html>. Acesso em: 19 abr 2015.

ORACLE CORPORATION. **Jersey:** RESTful Web Services in Java. Disponível em: <https://jersey.java.net/>. Acesso em: 14 out. 2015.

PEREIRA, Lúcio Camilo Oliva; SILVA, Michel Lourenço da. **Android para desenvolvedores.** Rio de Janeiro: Brasport, 2009.

PINHEIRO, José Maurício dos Santos. Web Semântica: Uma Rede de Conceitos. **Cadernos UniFOA**. Volta Redonda, ano IV, n. 9, abril. 2009. Disponível em: < http://web.unifoa.edu.br/cadernos/edicao/09/23.pdf>. Acesso em: 25 maio 2015.

RAY, Eric T. **Learning XML.** Sebastopol: O'reilly, 2001.

REDONDO, Sergio. **What is Semantic Search and Why Should I Care?** Disponível em: <http://www.searchenginejournal.com/seo-101-semantic-search-care/119760/>. Acesso em: 9 out. 2015.

RICHARDSON, Leonard; RUBY, Sam. **RESTful:** Web Services. Sebastopol: O'reilly, 2007.

SANDOVAL, Jose. **RESTful Java Web Services.** Birmingham: Packt Publishing, 2009.

SILVA, Viviane Soares Rodrigues; THOMÉ, Antonio Carlos Gay. **Um comitê de redes neurais para o reconhecimento de letras manuscritas**. In: VI ENCONTRO NACIONAL DE INTELIGÊNCIA ARTIFICIAL, 2007. Rio de Janeiro. Anais... Rio de Janeiro: UFRJ, p. 10.

SMITH, Ben. **Beginning with JSON**. Apress, 2015. 353p.

SOUZA, R. R.; ALVARENGA, L. **A Web Semântica e suas contribuições para a ciência da informação**. Ciência da Informação, Brasília, v. 33, n. 1, p. 132-141, 2004.

SOUZA, Vanderval Borges de. **Pré-Processamento de Imagens Para o Reconhecimento Ótico de Caracteres**. 2011. 78 f. Trabalho de Conclusão de Curso de Ciência da Computação, Unidade Acadêmica de Ciência e Tecnologia, Universidade do Extremo Sul Catarinense, Criciúma, 2011.

SQUARE. **Picasso:** A powerful image downloading and caching library for Android. Disponível em: <http://square.github.io/picasso/>. Acesso em: 26 out. 2015.

SRIPARASA, Sai Srinivas. **JavaScript and JSON Essentials**. Inglaterra, Birmingham: Packt Publishing, 2013.

STEELE, James; TO, Nelson. **The Android Developer's Cookbook**: Building Applications with the Android SDK. Boston: Pearson Education, 2011.

THE UNICODE CONSORTIUM (Org.). **What is Unicode?**. Disponível em: <http://www.unicode.org/standard/WhatIsUnicode.html>. Acesso em: 1 jun 2015.

VIRTUAL LIBRARY (Org.). **The WWW Virtual Library.** Disponível em: <http://vlib.org/>. Acesso em: 27 maio 2015.

ZAINA, Claudio Maximiliano de. **Visualização de Informação Aplicada a Resultados de Mecanismos de Busca.** 2005. 132 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Instituto de Computação, Universidade Estadual de Campinas, Campinas, 2005.

APÊNDICE

APÊNDICE A – Artigo Científico

Protótipo de aplicativo android para extração de texto em imagens de rótulos de cervejas para busca semântica das informações

Luana Gomes Silva[1], Gustavo Bisogni[2], Wiliam da Silva Bertam[2]

1Acadêmico do Curso de Ciência da Computação – Departamento de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brazil

2 Professor do Curso de Ciência da Computação – Departamento de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brazil

Luanagsilva20@gmail.com, gbisog@gmail.com, wiliam.bertam@gmail.com

Abstract. *The market for craft beers is in constant growing, causing the emergence of new and unknown products all the time. To assist the growth of this sector, this paper proposes the use of mobile technology by expanding the instant sharing of information. Using Android, it is possible through the use of Optical Character Recognition (OCR), the reading of beer labels in order to perform semantic search. This paper presents a prototype for the extraction of text within beer labels, performing the search for product information. For this, a study was carried out on the technologies involved. In performed tests it was possible to obtain relevant information for a product, aiding its popularity by sharing on social networks.*

Resumo. *O mercado das cervejas artesanais está em constante crescimento, fazendo com que surjam novos produtos ainda desconhecidos a todo instante. Para auxiliar o crescimento deste ramo, este trabalho propõe o uso da tecnologia móvel, por ampliar o compartilhamento instantâneo de informações. Utilizando o Android, é possível através do uso do Reconhecimento Óptico de Caracteres (OCR) a leitura de rótulos de cervejas a fim de realizar a busca semântica. Este trabalho apresenta um protótipo para a extração do texto do rótulo de cervejas, realizando a busca por informações do produto. Para isto, um estudo das tecnologias envolvidas foi realizado. Em testes, foi possível obter informações de um determinado produto auxiliando em sua popularidade através do compartilhamento em redes sociais.*

1. Introdução

A indústria cervejeira é apresentada nos seus processos produtivos, matérias-primas e tipos de cerveja. Nos ultimos anos esse ramo vem crescendo tanto a nivel nacional quanto internacional. No mercado surgem cervejas sofisticadas com destaque nas produzidas artesanalmente.

Santa Catarina se destaca nacionalmente por esse “movimento microcervejeiro”, sendo que algumas de suas microcervejarias têm reconhecimento nacional e mundial, por produzirem cervejas de qualidade superior.

Dentro desse contexto e dado o alto número de diferentes tipos de produtos com o qual o consumidor tem que lidar, surge a necessidade de obter informações sobre estes produtos, que podem ser retiradas muitas vezes de seus próprios rótulos.

Considerando o crescente uso de dispositivos como “smartphones”, surgiram diversos sistema operacionais para estes, dentre eles o Android, que é líder no mercado e desenvolvido pelo Google. Com algumas funcionalidades fornecidas pelo Android, como a câmera, é possível a automatização da pesquisa do rótulo desconhecido.

Para automatizar a entrada de dados nestes dispositivos, este projeto propõe utilizar softwares de reconhecimento óptico de caracteres em imagens, também chamados de OCR, que serão responsáveis por extrair as informações contidas em um rótulo, permitindo que seja realizada correção manual ou adição de informações.

Dado a importância do compartilhamento de informações referente a determinados segmentos ou produtos, este projeto permitirá compartilhar através de um aplicativo os dados extraídos por OCR, uma mensagem, a própria foto do produto, e informações adicionais trazidas através de busca semântica na Web, permitindo assim a divulgação e valorização da indústria nacional de cervejas especiais.

2. Mecanismo de buscas

Existem praticamente dois tipos de ferramenta de busca na WEB, os diretórios e os motores de busca.

Diretórios foi uma ferramenta iniciada ainda quando havia pouco conteúdo na web, caracterizada por uma organização hierárquica dos sites de sua base em categorias ou subcategorias de assuntos.

Os motores de busca utilizam de software robôs que também podem ser chamados de aranhas (spiders), agentes viajantes (wanderers), ou vermes (worms).

Os spiders vasculham as paginas e geralmente começam pelas mais populares. Através de links encontrados nas mesmas vão seguindo e assim visitando mais páginas guardando o que julgam necessário em sua base de dados. O resultado é mostrado em forma de lista com os links guardados da base, onde o usuário pode seguir e encontrar a informação necessária.

2.1. Motor de Busca Google

O Google nasceu em 1996, originalmente chamado BackRub, foi a criação dos estudantes da universidade de Stanford, Larry Page e Sergey Brin como uma melhor maneira de organizar e pesquisar a crescente Web.

Antes de pesquisar, o Google usa de robôs chamados de indexadores ou rastreadores, que anexam páginas para futuramente serem retornadas como resultado das pesquisas, caso estas pesquisas estiverem relacionadas de acordo com a palavra chave. Ao encontrar páginas relevantes, estas são organizadas por ordem de relevância e popularidade. No entanto, a Google mudou o seu famoso algoritmo para se focar na fase de entendimento das pesquisas. Isto significa que a empresa começou a prestar mais atenção ao contexto em que uma busca é

realizada e em quais conceitos aparecem nos documentos. Isto só é possível através da semântica

A busca semântica funciona da seguinte maneira, por exemplo, ao realizarmos uma busca pela palavra ‘panda’, sem mais especificações, é possível entender que o termo se refere a uma espécie de urso nativa da China, um dos algoritmos da Google ou até mesmo um software antivírus. Já ao pesquisar as palavras ‘panda dieta’, está claro que o Google sabe perfeitamente que tipo de resultados exibir. Isto acontece porque nesta pesquisa, o termo está melhor definido, onde um contexto é dado, conforme pode ser visto na Figura 1.

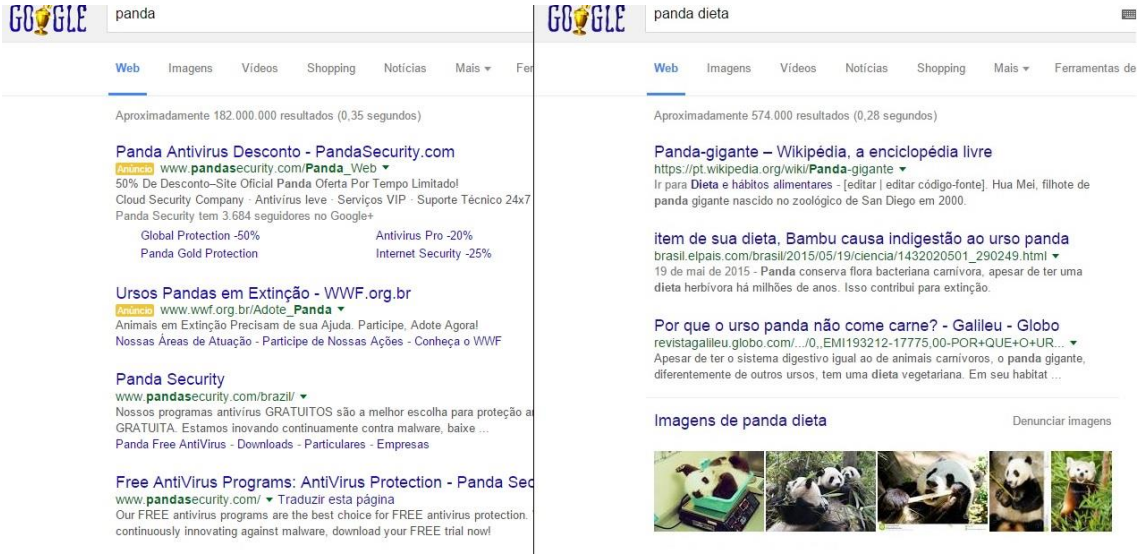


Figura 1 – Busca do google.

3. Web Semântica

Mesmo tendo sido projetada para facilitar a recuperação das informações, a Web acabou se tornando um repositório desorganizado de informações e documentos, no qual deixa muito a desejar quando precisamos recuperar o que realmente estamos buscando.

Nesse sentido, a comunidade científica começou um esforço para investigar a base para o próximo estagio da Web, chamado de Web Semântica.

A Web Semântica é uma extensão da atual, em que é dado à informação um significado bem definido, permitindo que computadores e pessoas trabalhem em cooperação. Para a Web Semântica funcionar, os computadores devem ter acesso a coleções estruturadas de informações e conjuntos de regras de inferência que eles podem usar para conduzir raciocínio automático.

O desenvolvimento da Web Semântica é composto por diversas tecnologias, inclusive algumas que ainda estão sendo desenvolvidas, dentre as quais, as mais básicas padronizam a sintaxe utilizada para representar os dados, a arquitetura e as tecnologias envolvidas podem ser verificadas na Figura 2.

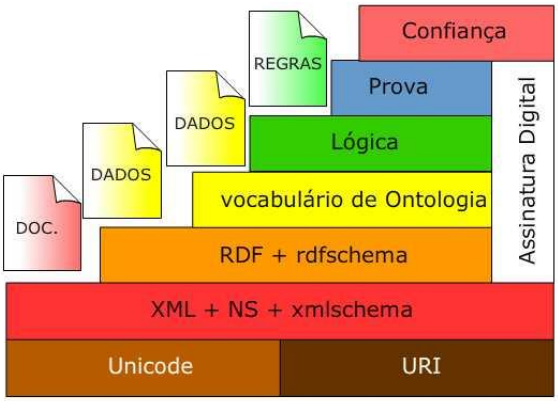


Figura 2 - Arquitetura web semântica

3.1. Unicode

Fundamentalmente, os computadores lidam com números. Eles armazenam letras e outros caracteres através da atribuição de um número para cada um. Antes de Unicode ser inventado, havia centenas de diferentes sistemas de codificação para a atribuição destes números, e qualquer computador particular especialmente os servidores precisavam suportar diversas codificações destes caracteres.

3.2. URI

Uniform Resource Identifiers (URI) são maneiras de identificar recursos, especialmente, mas não exclusivamente na Web.

As URIs possuem dois usos principais, e podem ser utilizado, para ambos os sentidos, uma é uniform Resource Names (URNs) que são URIs utilizadas para dar nome a algo, mesmo que seja um objeto abstrato que não esteja disponível na Web. A outra é Uniform Resource Locators (URLs) são URIs utilizadas para especificar o local de algo. URIs devem iniciar com um identificador de protocolo, e as URLs geralmente utilizam protocolos que uma representação técnica bem estabelecida no qual ferramentas podem utilizar para acessar lugares específicos, por exemplo http.

3.1. XML

O XML disponibiliza uma sintaxe para estruturar documentos, mas não possuem nenhuma regra semântica nos mesmos. Um documento XML bem formatado é ou um XML, e automaticamente, bem-formatado, ou é apenas texto. Mas conforme utilizado de maneira comum, um XML bem formatado significa um documento que segue as recomendações do W3C para o XML, com todas as suas regras.

3.2. RDF

O RDF trata-se de um modelo de dados utilizado para representar objetos e seus relacionamentos. Para exemplificar melhor, o RDF, é um documento que faz asserções de coisas específicas (pessoas, páginas Web ou outras coisas) e possuem propriedades (tais como “é uma irmã de,” ou “é autor de”) com certos valores (uma pessoa, uma página Web). Esta estrutura acabou sendo a maneira natural de descrever a grande maioria dos dados processados por máquinas.

4. Representação do Conhecimento

Sistemas tradicionais de representação de conhecimento geralmente têm sido centralizados, requerendo que todos compartilhem a mesma definição de conceitos comuns como “pais” e “veículo”.

5. Ontologia

Uma ontologia é uma teoria sobre a natureza da existência, da qual tipos de coisas existem; Ontologia como disciplina estuda tais teorias. Pesquisadores de Inteligência Artificial e da Web tem escolhido este termo para o seu próprio jargão, e para eles, ontologia é um documento ou um arquivo que formalmente define as relações entre termos.

6. Tecnologias de desenvolvimento móvel

O mercado de smartphones em todo o mundo cresceu 28,2% ano a ano no quarto trimestre de 2014 (2014 Q4), com embarques de 377,5 milhões de unidades. Android ainda domina o mercado com uma quota de 76,6% no 4Q14.

6.1. Plataforma android

A plataforma Android surgiu da aliança da Google com a Open Handset Alliance (OHA), que consiste na aliança entre empresas que englobam as tecnologias móveis desde softwares a hardwares.

O Android é construído sobre o aberto Kernel do Linux, utilizando uma máquina virtual personalizada, projetada para otimizar a memória e recursos de hardware em um ambiente móvel. O Android tem muitas versões, e para cada uma é dado um nome de doce seguido a um número sequencial chamado de API level. Este número é importante para saber os recursos disponibilizados em cada versão, pois se executarmos um aplicativo que utiliza recursos de uma versão mais recente em uma versão inferior, ocorrerá erro.

6.1.1. Arquitetura Android

A arquitetura Android é composta por cinco camadas que estão organizadas como pilha, sendo a camada de aplicações onde se encontram todas as aplicações desenvolvidas em Java que fazem parte do funcionamento básico do sistema, feitas pela comunidade do Android, como por exemplo, cliente de e-mail, mapas, navegadores, sistemas de SMS, gerenciador de contatos entre outras. A camada Android Framework fornece blocos de construção utilizados para criar aplicações no Android, vindo pré-instalado, onde é possível estendê-lo com seus próprios componentes. A camada de bibliotecas nativas é a camada que contém todas as bibliotecas nativas do Android, escritas em C ou C++ e compiladas para uma arquitetura específica de hardware utilizada pelo telefone e pré-instalada pelo fabricante. A camada de execução, conhecida como Android Runtime, possui a máquina virtual Dalvik e as bibliotecas do núcleo do Java. Basicamente esta máquina virtual é uma implementação da Google do Java, otimizada para dispositivos móveis, tornando possível executar código Java no Android. A camada Hardware Abstraction Layer (HAL) se trata de uma interface padrão que permite ao sistema realizar as chamadas aos drivers do dispositivo.

7. Web Service

O WS é uma evolução dos modelos de computação distribuída e surgiu com a necessidade de integrar sistemas heterogêneos através da internet. O WS pode ser definido como qualquer

serviço na internet que faça uso de um sistema de mensagem padronizado e não esteja amarrado a qualquer sistema operacional ou linguagem de programação.

Para o desenvolvimento de Web Services existem dois padrões que podem ser adotados, como o padrão SOAP e o padrão REST ou RESTful.

O Simple Object Access Protocol (SOAP) é um protocolo baseado em XML para a troca de informação entre computadores. O SOAP realiza a chamada a procedimentos remotos via HTTP.

O termo REST surgiu da dissertação de PhD de Roy Fielding, e significa *Representational State Tranfer* (transferência de estado representacional).

O REST facilita o desenvolvimento, a escalabilidade e flexibilidade das aplicações garantindo que os dados estejam em camadas, sem estado e bem definido.

Através do REST é possível mudar de XML para JSON modificando apenas a extensão na URL para realizar a requisição, sem necessitar redesenhar a aplicação ou modificar a plataforma de desenvolvimento.

O JavaScript Object Notation (JSON) é um popular formato para intercâmbio de dados. Criado por Douglas Crockford, o JSON é um formato leve, baseado em texto, que pode ser facilmente lido por humanos para realizar a troca de dados entre clientes e servidores, sendo que antes do JSON, o XML era considerado o ideal para o intercâmbio de dados.

A especificação do JSON define que os dados podem ser estruturados dentro das seguintes composições por uma coleção de pares de nomes/valores, ou por uma lista ordenada de valores.

8. OCR (reconhecimento óptico de caracteres)

É comum hoje em dia necessitarmos de textos contidos em imagens ou documentos, sendo muitas vezes, necessária a reescrita destes textos. A fim de resolver o problema e automatizar o reconhecimento destes textos, é comum a utilização de softwares OCR. A tecnologia OCR se baseia na extração de textos contidos em imagens digitais, possibilitando, por exemplo, sua edição. Estas imagens podem conter, por exemplo, elementos gráficos, dificultando o reconhecimento correto do conteúdo. Sendo assim, é importante identificar o que são figuras e elementos gráficos para que o OCR não perca tempo em seu processo.

Para aumentar as taxas de acerto da ferramenta OCR, pode-se ensinar a máquina, através das redes neurais, os padrões que podem ocorrer e como eles se parecem. No OCR estes padrões são as letras, números e caracteres especiais. Além de tudo, um pré-processamento da imagem poderá ser indispensável para o aumento da probabilidade de reconhecimento dos caracteres.

8.1. Processo de digitalização

No processo de digitalização, uma imagem digital do documento original é capturada. Durante esta etapa, utiliza-se um processo chamado Thresholding. O thresholding é importante porque a taxa de acerto do reconhecimento depende da qualidade da imagem em dois níveis. Um limite fixo é usado, onde tonalidades de cinza abaixo deste valor são dito preto e níveis acima são referidos como sendo branco. Entretanto, apenas esta técnica pode não ser o suficiente, necessitando de técnicas mais avançadas de pré-processamento.

Dependendo da resolução e do sucesso da técnica aplicada de thresholding, os caracteres podem estar manchados ou quebrados. Este tipo de problema pode ser eliminado utilizando-se de um pré-processamento para suavizar a imagem. A suavização implica tanto no enchimento e afinamento dos caracteres. Além das técnicas de pré-processamento para tornar possível o reconhecimento, grande parte dos algoritmos de reconhecimento de caracteres segmentam as palavras em caracteres isolados, reconhecendo-os individualmente.

9. Protótipo de aplicativo android para extração de texto em imagens de rótulos de cervejas para busca semântica das informações

O método utilizado durante o desenvolvimento do projeto foi primeiramente fazer um levantamento bibliográfico, a fim de entender o conceito e definição, para obter boas referências das tecnologias e futuramente ser feito a prática do protótipo. A fundamentação teórica foi organizada de maneira a direcionar a implementação, aumentando a produtividade no desenvolvimento.

O protótipo oferece um meio ao consumidor de obter as informações relevantes dos produtos desejados através de uma imagem de seu rótulo, onde a única entrada de dados manual que o consumidor deve fazer é informar o tipo de cerveja e tirar uma foto do rótulo do produto. Após a foto ter sido carregada pelo protótipo, seja através da câmera ou da galeria de imagens, será possível gravar estas fotos com as informações obtidas através de uma busca realizada no Google.

Com base nessa fundamentação, para a realização do trabalho foi necessário desenvolver duas aplicações, sendo uma aplicação cliente e uma aplicação web, responsável por atender as solicitações da aplicação cliente através de serviços de um Web Service RESTful.

No sistema web está hospedado o servidor que administra o fluxo de dados, contendo a aplicação da tecnologia OCR e efetuando a busca das cervejas não cadastradas no banco diretamente de informações extraídas do Google.

A aplicação web foi desenvolvida em Java através da IDE NetBeans 8.0 utilizando servidor Tomcat 7.0.63, uma biblioteca para o desenvolvimento de webservices RESTful chamada Jersey, a aplicação Tesseract, utilizada para o reconhecimento óptico de caracteres, banco de dados PostgreSQL 9.3 para realizar a persistência dos dados e Maven, responsável por obter as dependências necessárias para o projeto através de seu repositório.

A aplicação cliente foi criada com a plataforma Android Studio versão 1.3.2, incluindo a utilização da biblioteca Picasso para carregar as imagens do servidor, e a biblioteca Asynchronous Http Client para fazer as requisições ao WS.

9.1. Desenvolvimento do Protótipo

No protótipo mobile, responsável por realizar requisições e consumir as respostas em JSON geradas pelo servidor REST, foram criadas duas classes genéricas, sendo uma para efetuar requisições assíncronas simples, responsável por realizar o envio de informações textuais através do método GET e POST do protocolo HTTP, utilizando a biblioteca AsyncHttpClient. Esta classe genérica é herdada pelas classes especializadas de login e registro de usuário, sobrescrevendo os métodos genéricos de requisição e resposta desta classe abstrata quando necessário.

Uma segunda classe foi criada para o envio assíncrono de requisições com conteúdo binário, como as imagens, por exemplo, para o servidor REST. Para isto, foram utilizadas as bibliotecas HttpMime e HttpClient. No entanto, um dos problemas em utilizar a biblioteca HttpClient, é que a mesma não realiza automaticamente requisições assíncronas. Desta forma, foi necessário que esta classe genérica utilize recursos da classe AsycTask, por possuir um método chamado doInBackground, onde tarefas assíncronas são realizadas, estendendo suas funcionalidades por meio da herança e sobrescrevendo este método, onde então pôde-se utilizar a biblioteca HttpClient, obtendo como resultado, uma requisição assíncrona.

Quando o usuário realiza o login, suas credencias são verificadas e então é realizada uma requisição simples com os dados informados ao WS. Após as credenciais serem validadas, o usuário é encaminhado para a tela dos posts que realizou anteriormente. Para trazer estas informações, é realizada uma requisição ao WS que obtém uma lista de posts do usuário. Como nesta lista contém o nome das fotos que usuário postou, é possível realizar o download das imagens diretamente do servidor, através da biblioteca Picasso, que já efetua também o cache destas imagens.

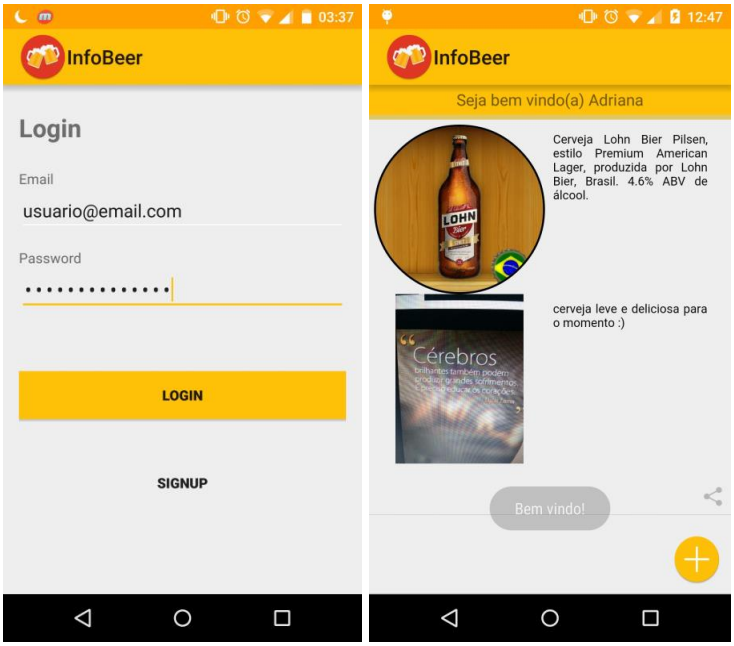


Figura 3- Tela logins e lista de posts do usuário

Para realizar um novo post, são necessárias três etapas, como pode ser visualizada na figura 4, uma por tela, sendo a primeira tela onde o usuário seleciona a foto da cerveja, que pode ser tirada com a câmera ou selecionada a partir da galeria do smartphone. Após selecionar a foto, esse caminho é passado para a próxima tela, na qual é solicitado ao usuário que selecione o tipo da cerveja. Com o tipo e arquivo da imagem, é realizada uma requisição ao WS, passando o conteúdo binário da imagem e o código do tipo da cerveja para o servidor. No servidor esta imagem é armazenada em uma pasta e passa por um pré-processamento, onde é aplicada a técnica de thresholding. Após a aplicação desta técnica, a imagem é direcionada para a ferramenta OCR, onde a resposta obtida do OCR é comparada a um dicionário de dados. Caso a cerveja seja reconhecida com sucesso, é realizado uma pesquisa com o nome e o tipo da cerveja no banco de dados, e caso a cerveja não tenha sido cadastrada previamente, é realizada uma requisição a uma URL do Custom Search da Google, que nada mais é do que uma pesquisa personalizada, que foi configurada para trazer apenas

informações de cervejas. Com as informações da cerveja obtidas do Custom Search, é possível então realizar o cadastro desta cerveja de maneira automática, já com as informações preenchidas para uso posterior. Caso a cerveja não seja reconhecida, será solicitado ao usuário através da aplicação cliente, que seja informado o nome da cerveja, onde então será realizada uma requisição para pesquisar os dados da cerveja. Ao obter todas as informações necessárias como nome e descrição da cerveja, é enviada uma resposta ao protótipo mobile com estas informações no formato JSON, onde o mesmo interpreta a resposta obtida e transfere estas informações para a ultima tela de post. Nesta última tela, o usuário pode capturar uma nova foto ou escolher uma foto adicional na galeria. Esta foto também é enviada para o servidor e então armazenada. Por fim, todas as informações referentes ao novo post são enviadas para armazenamento no banco de dados do servidor para consulta posterior ou compartilhamento através do Facebook ou WhatsApp.

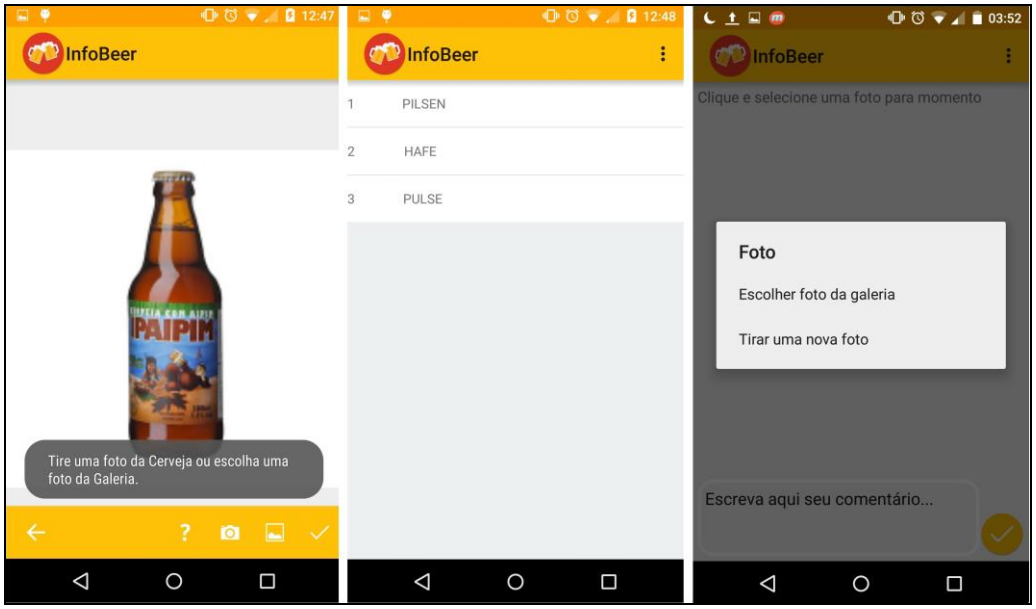


Figura 4- Telas etapa do cadastro de Post do usuário

10. Resultados Obtidos

Por meio da pesquisa realizada e da aplicação das ferramentas estudadas, como por exemplo, a ferramenta Tesseract e o Google Custom Search Engine, e através dos resultados obtidos o protótipo se demonstrou uma maneira simples e portável de se obter informações das cervejas desconhecidas, buscando rapidamente estas informações e disponibilizando-as em uma lista de publicações, assim como a possibilidade de compartilhamento das publicações por meio de redes sociais.

References

Alves, Neide Ferreira. Estratégias para melhoria do desempenho de ferramentas comerciais de reconhecimento óptico de caracteres. 2003. 108 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Federal de Pernambuco, Recife, 2003.

Berners-lee, Tim; Hendler, James; LASSILA, Ora. The Semantic Web. Scientifc American, Canadá, v. 284, n. 5, p.28-37 (1,2), 2001.

Branski, Regina Meyer. Localização de Informações na Internet: características e formas de funcionamento dos mecanismos de busca. Transinformação, Campinas, v. 12, n. 1, p.11-19,

jan. 2000. Quadrimestral. Disponível em: <http://www.scielo.br/pdf/tinf/v12n1/08.pdf>. Acesso em: 19 jun. 2015.

Burns, Christa; and Sauers, Michael P.. Google Search Secrets. Chicago: American Library Association, 2014. 202 p.

Cendón, Beatriz Valadares. Ferramentas de Busca na Web. Ciência da Informação, Brasília, v. 30, n. 1, p.39-49, jan. 2001. Disponível em: <http://www.scielo.br/pdf/ci/v30n1/a06v30n1>. Acesso em: 26 jun. 2015.

Cunha, Tiago Vargas. Competitividade e segmentação na indústria cervejeira: Uma Análise da Competitividade das Microcervejarias Catarinense. 2011. Disponível em: <http://tcc.bu.ufsc.br/Economia299002.pdf>. Acesso em: 07 out. 2014.

Eckstein, Robert; casabianca, Michel. XML: Pocket Reference. 3. ed. Sebastopol: O'reilly, 2001.

Eikvil, Line. OCR - Optical Character Recognition. 1993. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.25.3684>. Acesso em: 19 jun. 2015.

Fawcett, Joe; Quin, Liam; Ayers, Danny. Beginning XML. 5. ed. Indianapolis: Wrox, 2012.

Gruber, Thomas R.. A Translation Approach to Portable Ontology Specifications. 1993. Appeared in Knowledge Acquisition , 5 (2):199-220, 1993. Disponível em: <http://tomgruber.org/writing/ontolingua-kaj-1993.pdf>. Acesso em: 26 jul. 2015.

Redondo, Sergio. What is Semantic Search and Why Should I Care? Disponível em: <http://www.searchenginejournal.com/seo-101-semantic-search-care/119760/>. Acesso em: 9 out. 2015.