UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
BACHARELADO EM ENGENHARIA DE SOFTWARE

RODRIGO GIRÃO DE FREITAS

ICA: UMA API PARA CONSTRUÇÃO DE APLICAÇÕES PARA ADAPTAÇÃO DE IMAGEM BASEADO EM CONTEXTO UTILIZANDO COMPUTAÇÃO EM NUVEM

QUIXADÁ
2014

RODRIGO GIRÃO DE FREITAS

ICA: UMA API PARA CONSTRUÇÃO DE APLICAÇÕES PARA ADAPTAÇÃO DE IMAGEM BASEADO EM CONTEXTO UTILIZANDO COMPUTAÇÃO EM NUVEM

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Bacharelado em Engenharia de Software da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Bacharel.

Área de concentração: computação

Orientador: Prof. Dr. Lincoln Souza Rocha

QUIXADÁ
2014

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca do Campus de Quixadá

F936i Freitas, Rodrigo Girão de
ICA: uma API para construção de aplicações para adaptação de imagem baseado em contexto utilizando computação em nuvem / Rodrigo Girão de Freitas. – 2014.
33 f. : il. color., enc. ; 30 cm.

Monografia (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Bacharelado em Engenharia de Software, Quixadá, 2014.
Orientação: Prof. Dr. Lincoln Souza Rocha
Área de concentração: Computação

1. Computação em nuvem 2. Computação em nuvem 3. Interfaces de usuário (Sistemas de computação) I. Título.

CDD 005.1

RODRIGO GIRÃO DE FREITAS

ICA: UMA API PARA CONSTRUÇÃO DE APLICAÇÕES PARA ADAPTAÇÃO DE IMAGEM BASEADO EM CONTEXTO UTILIZANDO COMPUTAÇÃO EM NUVEM

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Bacharelado em Engenharia de Software da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Bacharel.

Área de concentração: computação

Aprovado em: 13/ junho / 2014.

BANCA EXAMINADORA

_____

Prof. Dr. Lincoln Souza Rocha (Orientador)
Universidade Federal do Ceará-UFC

_____

Prof. Dr. Flávio Rubens de Carvalho Sousa
Universidade Federal do Ceará-UFC

_____

Prof. Msc. Leonardo Oliveira Moreira
Universidade Federal do Ceará-UFC

À Deus
Aos meus pais
À Letícia.

AGRADECIMENTOS

Agradeço aos meus pais que sempre insvestiram na minha educação e acreditaram que eu poderia ter um futuro melhor do que o deles.

Agradeço a minha família por ter me aponhado e ajudado nas decisões difíceis.

Agradeço a minha futura esposa Letícia por ter me ajudado e que sempre teve toda paciência do mundo comigo.

Agradeço aos professores Lincoln e Flávio por suas orientações e por terem confiado e não desistido de mim.

Agradeço aos professores presentes na banca por suas críticas e sugestões que foram bastante contrutivas para o trabalho.

Agradeço aos amigos e colegas de faculdades por todo este tempo convivendo e partilhando experiências juntos.

"Quando você tenta o seu melhor, mas não tem sucesso.
Quando você consegue o que quer, mas não o que precisa.
Quando você se sente cansado, mas não consegue dormir."
(Coldplay)

RESUMO

O uso de dispositivos móveis trouxe uma nova abordagem ao desenvolvimento de aplicações. Com isso os desenvolvedores passaram a fazer aplicações que identificasse o contexto do usuário sem a necessidade do mesmo fornecer dados de maneira direta. Como forma de atender a necessidade dos desenvolvedores em obter informações de contexto do usuário, foi realizado o desenvolvimento de uma API que facilita a criação de aplicações de adaptação de imagem levando em conta o contexto do usuário. Primeiramente foi feito um levantamento de quais características seriam relevantes para o usuário. Foram escolhidas algumas informações do dispositivo móvel como tamanho e resolução de tela, memória ram disponível, porcentagem da bateria e tipo conexão de dados. Após identificar as características relevantes, foi criado um protótipo para viabilizar a construção da API. Em seguida foi realizado o desenvolvimento da API, que é composta por uma biblioteca cliente e uma biblioteca servidor. Para exemplificar o uso da API foi construída uma aplicação exemplo mostrando como cada classe e método pode ser usado na criação de uma aplicação de adaptação de imagens. O desenvolvimento da aplicação exemplo servirá como uma documentação para os desenvolvedores que desejarem utilizar a API.

Palavras chave: Desenvolvimento de Software. Dispositivos móveis. Biblioteca

ABSTRACT

The use of mobile devices has brought a new approach to application development. With that developers started to make applications to identify the user context without the need to provide the same data directly. In order to meet the needs of developers in receiving information from the user's context was conducted to develop an API that facilitates the creation of adaptive imaging applications taking into account the user's context. First a survey was done of which features are relevant to the user. Some information from the mobile device was chosen such as screen size and resolution, available ram, battery percentage and data connection type. After identifying the relevant features, a prototype was created to enable the construction API. Then development was carried out API, which is composed of a server and a client library. To illustrate the use of the API was built a sample application showing how each class and method can be used in creating an application for adjustment of images. The development of the application example will serve as documentation for developers who want to use the API.

Keywords: Software development. Mobile devices. Library

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Os dispositivos móveis estão cada vez mais presentes na vida das pessoas. Uma pesquisa feita pela empresa Cisco Systems, Inc. (2013) revela que houve um crescimento de 70% do tráfico de dados móveis em 2012, atingindo 885 petabytes por mês contra 520 petabytes por mês no ano de 2011. A pesquisa também mostra que em 2012, 429 petabytes de tráfego de dados móveis foram baixados para a rede fixa a cada mês. Sem a opção de carregamento, o tráfego de dados móveis teria crescido 96%, em vez de 70% em 2012.

Outra pesquisa feita pela Symantec Corporation (2013) mostra que 1/3 das empresas na América Latina está adotando soluções de tecnologia móvel. As demais empresas não utilizam esta tecnologia por desconhecimento dos benefícios para seu negócio. Na mesma pesquisa é apontado que 42% das empresas latino americanas afirmaram ter perdido dados empresariais de seus clientes ou empregados por questões vinculadas ao uso de dispositivos móveis.

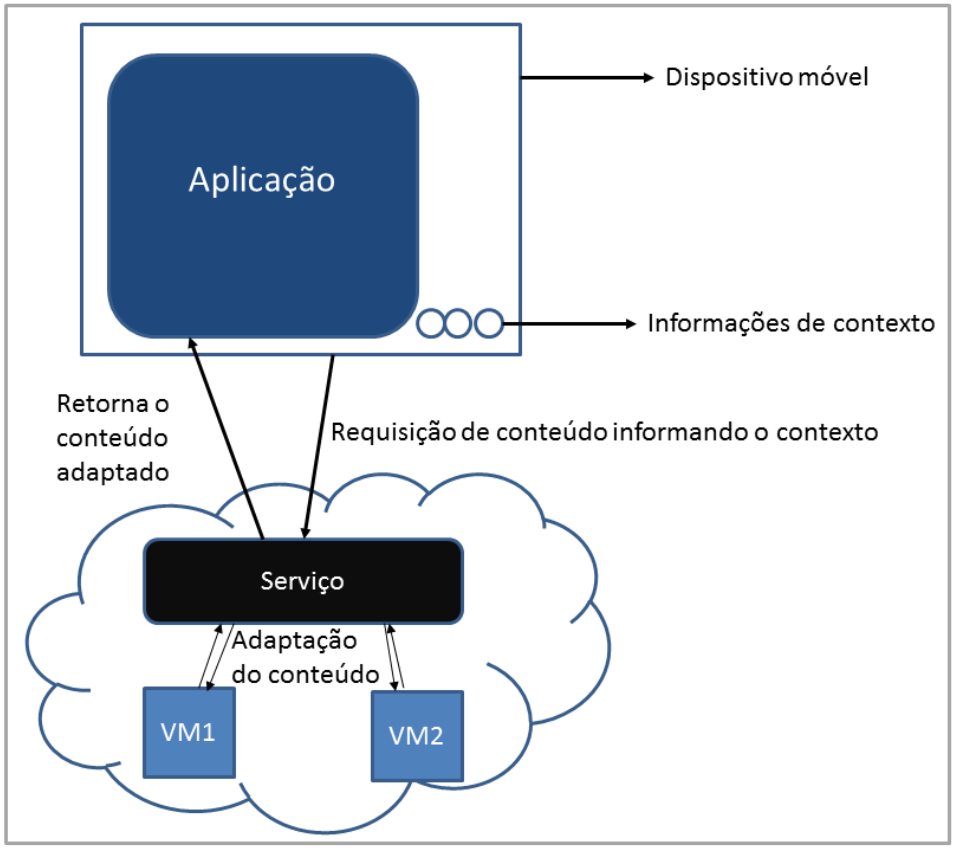
Com o aumento da capacidade dos dispositivos móveis (e.g., aparelhos celular e tablets), os usuários passam a poder executar aplicações potentes e com uma grande capacidade de uso de sensores (KOVACHEV; CAO; KLAMMA, 2011). A popularização das aplicações se deu a partir das lojas de download de aplicativos como a App Store (APPLE, 2013) e Google Play (GOOGLE, 2013). Existem aplicações de várias categorias como jogos, entretenimento, finanças, música, negócios, notícias, viagens, dentre outras. Entretanto, mesmo com os ganhos em termos de mobilidade e melhorias em termos de processamento e capacidade de armazenamento, os dispositivos móveis ainda apresentam limitações em termos de poder de processamento e armazenamento quando comparado com computadores de mesa (desktops). Além disso, a fonte de energia dos dispositivos móveis é restrita (i.e., baseada em bateria), a resolução e o tamanho da tela variam de um dispositivo para outro e a conectividade, por ser baseada em tecnologias de comunicação sem fio, está sujeita a vários tipos de interferência, tornando-se intermitente (FERNANDO; LOKE; RAHAYU, 2013). Portanto, essas características precisam ser levadas em consideração quando uma aplicação móvel é desenvolvida para consumir serviços disponíveis na internet.

Nesse cenário, a computação em nuvem surge como uma alternativa viável para lidar com algumas destas limitações, provendo acesso transparente a recursos computacionais na forma de serviço para que possam ser consumidos pelas aplicações móveis (DINH et al., 2011). A computação em nuvem permite armazenar e processar grandes volumes de dados de

forma simples e eficiente (SOUSA; MOREIRA; MACHADO, 2009). Além disso, a Computação em Nuvem disponibiliza recursos de Tecnologia da Informação na forma de serviço, sendo estes fornecidos sob demanda e com pagamento baseado no uso.

Os sensores disponíveis na maioria dos dispositivos móveis atuais podem ser usados para capturar diversos tipos de informações de contexto (i.e., informações sobre o dispositivo, o usuário e o ambiente físico). Essas informações podem ser utilizadas para adaptar serviços providos por uma plataforma de computação em nuvem (e.g., software especializado, processamento, armazenamento e segurança) em função do contexto corrente da aplicação móvel que o consome. Por exemplo, o trabalho de Christensen (2009) apresenta um conjunto de considerações arquiteturais para criação de aplicações móveis que fazem uso de informações de contexto no acesso a serviços web RESTful em plataformas de computação em nuvem.

Figura 1 – Exemplo de serviço de adaptação de conteúdo



Fonte: Elaborada pelo autor.

De forma precisa, este projeto consiste no desenvolvimento de uma API para auxiliar os desenvolvedores na construção de aplicações de adaptação de imagens. Para facilitar o entendimento, considere a aplicação móvel e o serviço de nuvem descritos na

Figura 1. Trata-se de um serviço de adaptação de conteúdo, que adapta o conteúdo (imagem) acessado pela aplicação móvel em função do contexto do dispositivo. O fluxo de execução desse tipo de serviço funciona da seguinte forma: (1) as informações de contexto do dispositivo são capturadas (e.g., capacidade de processamento, largura de banda, resolução e tamanho da tela de exibição); (2) quando a aplicação solicita ao serviço algum tipo de conteúdo, as informações de contexto são enviadas junto com a requisição; (3) o serviço hospedado na nuvem recebe a requisição e adapta o conteúdo de acordo com as informações de contexto enviadas; e (4) o serviço envia o conteúdo adaptado ao dispositivo requisitante. Esse é o tipo de aplicação que a API proposta tem por objetivo dar suporte ao desenvolvimento.

Este trabalho tem como público alvo os desenvolvedores de aplicações para dispositivos móveis e desenvolvedores de serviços adaptativos de nuvem, permitindo que estes possam desenvolver aplicações móveis que melhorem a experiência do usuário incluindo, menor consumo de bateria, menor atraso de resposta da aplicação e menor número de dados trafegados.

O objetivo geral do trabalho é construir uma API de suporte ao desenvolvimento de serviços de adaptação de imagem baseados em informações de contexto para dispositivos móveis. Os objetivos específicos são: identificar as características de contexto relevantes para aplicações em dispositivos móveis; desenvolver uma API provendo classes e métodos de suporte ao desenvolvimento de serviços de adaptação de imagens; desenvolver uma aplicação exemplo utilizando a API para mostrar sua facilidade de uso aos desenvolvedores que pretendem construir aplicações de adaptação de imagem.

O trabalho está dividido da seguinte maneira: na Seção 2 são apresentados os principais conceitos utilizados no trabalho, como computação móvel, contexto, computação em nuvem e computação em nuvem para dispositivos móveis; na Seção 3 são apresentados os passos que foram seguidos para realização do trabalho; na Seção 4 foram identificadas as características necessárias para o desenvolvimento da API e também foi criado um protótipo que serviu para identificar dificuldades no desenvolvimento da API; na Seção 5 foi feito o desenvolvimento da API utilizando as caraterísticas definidas na Seção 4; na Seção 6 é mostrado como deve ser feita a criação de aplicações utilizando a API através de uma aplicação exemplo; na Seção 7 são apresentados os trabalhos futuros e as considerações finais do trabalho.

2 REVISÃO BIBLIOGRÁFICA

Nesta seção são apresentados os principais conceitos utilizados no projeto. A Seção 2.1 é dedicada aos conceitos da computação móvel, mostrando sua diferença com os demais paradigmas e algumas das tecnologias relacionadas ao tema. Na Seção 2.2 são mostrados os principais conceitos relacionados a contexto e sensibilidade ao contexto e como estes se aplicam aos dispositivos móveis. A Seção 2.3 está focada nos conceitos sobre computação em nuvem, apresentando detalhes sobre o modelo Software como um Serviço (SaaS) e a abordagem de Arquitetura Orientada a Serviços (SOA). Por fim, a Seção 2.4 é dedicada à computação em nuvem para dispositivos móveis, descrevendo a forma como os conceitos de computação em nuvem são utilizados para tentar resolver os problemas advindos das limitações dos dispositivos móveis.

2.1 Computação Móvel

A popularização dos *smartphones* nos últimos anos, juntamente com a computação *ubíqua*, tem chamado bastante a atenção dos desenvolvedores. Estimativas recentes indicam que em 2015 mais de 70% dos dispositivos usados pelas pessoas serão *smartphones* capazes de rodar aplicações móveis. (Joorabchi, Mesbah, Kruchten,2013).

Os dispositivos móveis são “computadores” que possuem uma maior portabilidade, limitação de recursos e diferem entre si. Segundo B'Far (2005), “Sistemas de computação móvel são sistemas de computação que, fisicamente, podem ser facilmente movidos e cujas capacidades de computação podem ser usados enquanto estão sendo movidos.”. Isso traz novas abordagens para tipos de aplicações que podem ser desenvolvidas.

Com a ascensão dos dispositivos móveis, surgiu um novo paradigma de desenvolvimento de software. O novo paradigma permite novas maneiras de fazer negócio e novos tipos de negócio que afetam seus envolvidos como usuários, profissionais e desenvolvedores (ESPOSITO, 2012).

Quando olhamos para as funcionalidades das aplicações móveis, temos uma impressão que elas são mais simples do que aplicações desktop ou web. Porém, elas se tornam mais complexas quando são levadas em conta algumas características dos dispositivos móveis como entrada de dados, sensores, armazenamento de dados e ciclo de vida (ESPOSITO, 2012).

Os usuários estão usando as aplicações em situações mais rotineiras, o que faz com que estas sejam usadas mais vezes ao dia em um menor tempo contínuo. Esse fato deixa os usuários mais exigentes em relação ao tempo de espera e a usabilidade da aplicação, fazendo com que os desenvolvedores sejam obrigados a levar em consideração esses requisitos. Esposito (2012) fornece uma boa definição quanto ao que é preciso saber para desenvolver aplicações móveis:

> [...] o termo móvel refere-se a uma variedade de plataformas, cada uma com seu próprio conjunto de capacidades e características, e cada uma delas requer significativamente diferentes habilidades, diferentes sistemas operacionais, linguagens de programação diferentes, diferentes interfaces de programação de aplicativos (APIs), e até mesmo computadores diferentes. (ESPOSITO, 2012, p. xiii)

Seguindo essa definição do termo móvel, atualmente, é possível encontrar alguns sistemas operacionais móveis disponíveis no mercado, como Android, iOS e Windows Phone. Para cada sistema operacional móvel é preciso utilizar uma linguagem de desenvolvimento. Os sistemas citados acima usam respectivamente as linguagens, Java, Objetive C e C#. Ainda existem os web sites móveis ou web apps que são escritos em HTML5, CSS3, Javascript e rodam no navegador do dispositivo, independente de plataforma.

As aplicações móveis podem ser divididas em três categorias: nativas, web e híbridas. As aplicações nativas são desenvolvidas de acordo com o sistema operacional e necessitam ser adaptadas para cada sistema diferente. As aplicações móveis web rodam no navegador do dispositivo móvel. As híbridas são aplicações web encapsuladas para o sistema operacional.

Neste trabalho, foi escolhido o desenvolvimento de aplicações nativas utilizando o sistema operacional Android que utiliza a linguagem Java para suas aplicações. A escolha foi motivada por este ser um software livre e que não necessita de hardwares ou softwares proprietários para o desenvolvimento.

2.2 Contexto

Com o crescente avanço tecnológico, a computação passou a fazer parte do dia a dia das pessoas. Essa onipresença computacional tem como objetivo facilitar a execução das atividades cotidianas dos usuários, causando-lhe o mínimo de distração possível. Nesse cenário, os sistemas que conseguem capturar as informações do usuário de maneira transparente e adaptar-se de forma adequada, refletindo a necessidade e intenção do usuário, estão ganhando espaço sobre os que obrigam o usuário a registrar explicitamente suas informações e a desencadear ações (VIEIRA; TEDESCO; SALGADO, 2009). Para atender a

esta demanda, os desenvolvedores estão buscando desenvolver aplicações mais atrativas, adaptáveis e proativas (VIEIRA; TEDESCO; SALGADO, 2011). Dessa forma, o contexto surge para fazer com que as informações relevantes para o usuário sejam capturadas e disponibilizadas de acordo com sua necessidade.

Contexto pode ser entendido como a maneira de gerar interações do usuário com o ambiente a sua volta. Dey (2001, p. 5) define contexto como, "qualquer informação que pode ser usada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, lugar, ou objeto que é considerado relevante para a interação entre um usuário e uma aplicação, incluindo o usuário e as próprias aplicações.". Nesse caso, qualquer informação que possui relevância para a tarefa que está sendo realizada, e possa ser capturada para melhorar esta tarefa, pode ser caracterizada como contexto.

Segundo Dey (2001, p. 5) "um sistema é sensível ao contexto se este usa o contexto para prover informações relevantes ou serviços para o usuário, onde a relevância depende da tarefa do usuário". Um sistema sensível ao contexto deve proporcionar ao usuário um mínimo de trabalho possível na obtenção de informações necessárias, que serão utilizadas na realização de uma tarefa.

Neste trabalho, as informações de contexto serão capturadas a partir de sensores presentes nos dispositivos móveis. Estas informações serão utilizadas para informar qual o melhor tamanho da imagem que deve ser enviada para o dispositivo.

2.3 Computação em Nuvem

Computação em nuvem é uma tendência recente da tecnologia. 'Nuvem' é uma metáfora para a internet e seus serviços, podendo ser composta por milhares de computadores interconectados. O objetivo é proporcionar serviços de tecnologia da informação (TI) de acordo com a demanda de cada usuário. Dados podem ser acessados a partir de um navegador, independentemente de sua localização geográfica. Isso significa que usuários podem armazenar e acessar seus documentos pessoais na internet, por meio de aplicativos como Dropbox ou Sugarsync, por exemplo; podem utilizar um editor de textos on-line como o Google Docs; e empresas ou universidades podem utilizar grandes infraestruturas de computadores como o Amazon Web Services para processar e analisar dados, reduzindo custos e permitindo mais foco em inovação (SOUSA; MOREIRA; MACHADO, 2010).

Na nuvem os recursos são monitorados, possibilitando a tarifação de acordo com o uso, da mesma maneira que os serviços de telefonia e energia. Com isso, os usuários podem

alugar capacidade de computação e armazenamento, em vez de investir na aquisição e instalação de equipamentos. (SOUSA; MOREIRA; MACHADO, 2009).

Um dos modelos de serviço da computação em nuvem é o Software como um Serviço (SaaS). Esse modelo fornece uma maneira diferente de disponibilizar aplicações para os usuários, que agora podem acessar aplicações apenas pelo navegador, sem se preocupar com atualizações e limitações de hardware (SOUSA, 2010).

Uma Arquitetura Orientada a Serviços (SOA) permite que aplicações sejam criadas a partir de combinações de serviços, sejam estes novos ou já existentes. SOA está sendo muito utilizado como software na indústria (SOUSA, 2010).

Segundo SOUSA (2010, p. 12) em "uma SOA, um serviço é uma função de um sistema computacional que é disponibilizado para outro sistema". Como um serviço pode operar de forma independente ou em conjunto com outros serviços é necessário estabelecer uma interface de comunicação bem definida. Os WebServices são quem fazem a comunicação entre os clientes e os serviços. A comunicação é feita por meio de dados em um formato padrão, como XML e JSON. As aplicações SOA são em sua maioria WebServices do tipo, SOAP, REST e WDSL (SOUSA, 2010).

Neste trabalho será criado um WebService do tipo REST que irá enviar e receber dados do tipo JSON. Esta escolha foi feita, pois a combinação do REST, JSON e o protocolo HTTP permite fazer serviços simples que estão sendo muito usados na Web 2.0 (FILHO; FERREIA, 2011). Este WebService ficará disponível em uma nuvem da Amazon, que permite uma realocação de recursos caso necessário.

2.4 Computação em Nuvem para Dispositivos Móveis

Computação em nuvem para dispositivos móveis é geralmente abordada como a execução de uma aplicação em um servidor remoto, enquanto o dispositivo apenas acessa essa aplicação através da rede de dados (FERNANDO; LOKE; RAHAYU, 2013).

Em Kovachev, Cao e Klamma (2011) computação em nuvem para dispositivos móveis é definida como:

> [...] um modelo para o aumento da transparência e elasticidade dos recursos dos dispositivos móveis através do acesso sem fio ubíquo para armazenamento em nuvem e recursos de computação, com ajuste dinâmico sensível ao contexto de descarga em relação à alteração das condições operacionais, preservando o sensoriamento disponível e as capacidades de interatividade dos dispositivos móveis. (KOVACHEV; CAO; KLAMMA, 2011, p. 1)

De acordo com esta definição, a computação em nuvem para dispositivos móveis mescla os recursos fornecidos pelos dispositivos móveis com os recursos fornecidos pela computação em nuvem, permitindo que o usuário tenha um acesso online a uma grande capacidade de armazenamento e processamento.

A maneira de liberar todo o potencial das aplicações móveis em nuvem é transferindo para a nuvem o processamento e armazenamento sem atingir a interatividade com o usuário, a latência ou limitando as possibilidades das aplicações. Para isso, devem ser considerados parâmetros internos e externos na troca de dados entre a nuvem e o dispositivo (KOVACHEV; CAO; KLAMMA, 2011).

As razões para a distribuição do processamento dos dispositivos móveis para a nuvem são: a capacidade computacional limitada, energia da bateria limitada, conectividade limitada, a oportunidade de reunir mais dados de sensoriamento, acesso a diferentes conjuntos de dados e conteúdos e fazer o uso do poder de processamento ocioso (FERNANDO; LOKE; RAHAYU, 2013).

No trabalho de DINH et al. (2011) são listados algumas abordagens para resolução das limitações dos dispositivos móveis. A primeira abordagem citada soluciona o problema da desconexão. Neste modelo a solicitação do usuário vai primeiramente para um gateway de serviço, que irá escolher um proxy para atender a requisição e retorna a resposta para o usuário. Caso aconteça uma desconexão, são estabelecidos outros proxies que possuem suporte a reconfiguração dinâmica, com um mínimo de interrupção. Em outra abordagem é construído um módulo de middleware embutido no dispositivo móvel, que monitora os recursos e o contexto do dispositivo móvel, ajustando dinamicamente os requisitos dos usuários em tempo de execução.

Neste trabalho a computação em nuvem para dispositivos móveis será usada para otimizar os recursos limitados dos dispositivos ao receber conteúdos. O conteúdo escolhido foi imagem, por ser bastante utilizado pelos usuários. As imagens recebidas não devem consumir mais do que o necessário dos recursos, deixando o trabalho de adaptação para a nuvem.

3 PROCEDIMENTOS METODOLÓGICOS

Nesta seção são mostrados os passos para a execução do trabalho. Inicialmente será realizada a identificação das características que serão utilizadas na criação da API de adaptação. Após a identificação das características será feito o desenvolvimento de um protótipo que irá viabilizar o desenvolvimento da API. Após isso, será desenvolvida a API utilizando as características identificadas inicialmente. Por último, será desenvolvida uma aplicação exemplo utilizando a API proposta como o intuito de avaliar sua viabilidade e conformidade ao contrato de design estabelecido.

3.1 Identificar Características para a Construção da API

Inicialmente, foram identificadas as características referentes às informações de contexto relevantes para a adaptação de imagens para dispositivos móveis, tais como características dinâmicas (CPU, memória, energia) e estáticas (tamanho e resolução da tela) destes dispositivos. Essas características identificadas servirão de base para o desenvolvimento da API proposta que utilizará estas para retornar a imagem adequada ao dispositivo.

3.2 Construção do Protótipo

Foi construído um protótipo utilizando as tecnologias Android e Java Web. O protótipo é uma mini aplicação, onde o dispositivo móvel requisita uma imagem enviando informações de contexto, como resolução e tamanho da tela. O servidor recebe as informações de contexto e adapta a imagem com valores fictícios. Esse protótipo serve para verificar a viabilidade da construção da API. Através do protótipo é possível identificar dificuldades e analisar como será feito o desenvolvimento da aplicação.

3.3 Construção da API

Foi desenvolvida uma API para construção de serviços adaptativos para dispositivos móveis. A API gerencia a coleta de informações de contexto de forma transparente, enviando-as ao WebService junto com as requisições da aplicação. Por outro lado, a API provê um conjunto classes e métodos que servirão como um contrato de design para a construção dos serviços adaptativos. Nesse caso, o serviço foi construído para receber as informações de contexto enviadas pelo dispositivo móvel e adaptar o processamento da requisição de forma a prover respostas mais adequadas ao contexto observado.

3.4 Construção da Aplicação Exemplo

Foi desenvolvida uma aplicação exemplo que utiliza a API proposta, tanto na parte do consumo como no desenvolvimento do serviço adaptativo propriamente dito. O objetivo do desenvolvimento dessa aplicação exemplo é oferecer uma prova de conceito do uso da API proposta, evidenciando a sua viabilidade. Nesse caso, foi desenvolvida uma aplicação móvel na plataforma Android que fez o uso das classes e métodos da API para fazer comunicação com o servidor e receber as imagens adaptadas. A aplicação Web foi feita para estabelecer o contrato de comunicação com a aplicação móvel e utilizar o método de adaptação da API.

4 REQUISITOS PARA A CONSTRUÇÃO DA API

Nesta seção serão identificados os pré requisitos para guiar o desenvolvimento da API. Os pré requisitos identificados servirão como base para o desenvolvimento da API. Inicialmente, foram identificadas as características necessárias para obter as informações de contexto relevantes do dispositivo móvel. A partir das características obtidas, foram selecionadas algumas características para desenvolver um protótipo descartável a fim de viabilizar o desenvolvimento da API.

4.1 Identificação das Características

Em Held, Buchholz e Schill (2002) são abordadas 3 categorias de informações de contexto consideradas relevantes para proporcionar uma melhor experiência para o usuário de dispositivos móveis. As categorias abordadas por ele são: capacidade dos dispositivos móveis, características da conexão de rede e informações específicas do usuário. Para a capacidade dos dispositivos ele leva em consideração as características como poder de processamento da CPU, memória, tamanho e resolução da tela, cores, som, etc. A conexão de rede é caracterizada principalmente pela largura de banda e atraso da rede. As informações específicas do usuário são as preferências escolhidas por ele, como, por exemplo, tamanho da fonte.

Com base nas características mostradas acima, para esse trabalho foram escolhidas 5 características consideradas importantes para o contexto do dispositivo móvel: tamanho e resolução de tela, memória ram disponível, porcentagem da bateria e tipo conexão de dados. Essa características serão utilizadas como base para o desenvolvimento da API, pois a partir delas será determinado qual o tamanho mais adequado para a imagem requisitada pelo dispositivo.

4.2 Criação do Protótipo

O protótipo desenvolvido consiste em uma aplicação simples que redimensiona uma imagem de acordo com as informações de tamanho e resolução de tela e memória disponível do dispositivo. A seguir são listadas as tecnologias utilizadas no desenvolvimento do protótipo.

4.2.1 Desenvolvimento

O desenvolvimento do protótipo foi dividido em duas etapas: desenvolvimento da aplicação cliente e o desenvolvimento do servidor. O protótipo desenvolvido serve para avaliar as dificuldades e necessidades que a API deve fornecer para o desenvolvedor. Com base no desenvolvimento do protótipo foi construído a API. Nas duas seções seguintes será detalhado como foram realizadas as duas etapas do desenvolvimento do protótipo.

4.2.1.1 Cliente

Foi desenvolvida uma aplicação cliente em Android, que fará requisições de imagens a um servidor desenvolvido em Java. O Android é um software de código aberto constituído de sistema operacional, middleware e um conjunto de bibliotecas que permitem o desenvolvimento de aplicações para os dispositivos móveis. As bibliotecas possuem APIs que permitem utilizar os recursos dos dispositivos como acesso ao hardware, gravação de vídeo, serviços baseados em localização, mapas, Bluetooth e NFC. (MEIER, 2012)

Na aplicação, ao solicitar uma imagem através de uma requisição POST para o servidor, é enviado um JSON, contendo as informações do tamanho e resolução da tela e memória ram disponível. As informações do dispositivo foram capturadas e convertidas em JSON. Para fazer essa conversão foi utilizado o framework Gson, que faz a conversão de Objetos JAVA para o formato JSON e vice-versa (SINGH, LEITCH e WILSON, 2014). A resposta dessa requisição é a imagem adaptada de acordo com o dispositivo.

4.2.1.2 Servidor

A aplicação servidor consiste em receber uma requisição de imagem com as informações do dispositivo e enviar a imagem adaptada de acordo com essas informações. A aplicação foi desenvolvida em Java web, utilizando o framework Jersey e a biblioteca Gson.

Jersey é um framework de código aberto que provê uma API para desenvolver web services RESTful em Java de uma maneira simples. Ele fornece suporte para Java API for RESTful Services (JAX-RS) (ORACLE CORPORATION, 2012), mas provê um conjunto maior de funcionalidades para simplificar o desenvolvimento do serviço RESTful (ORACLE CORPORATION, 2014).

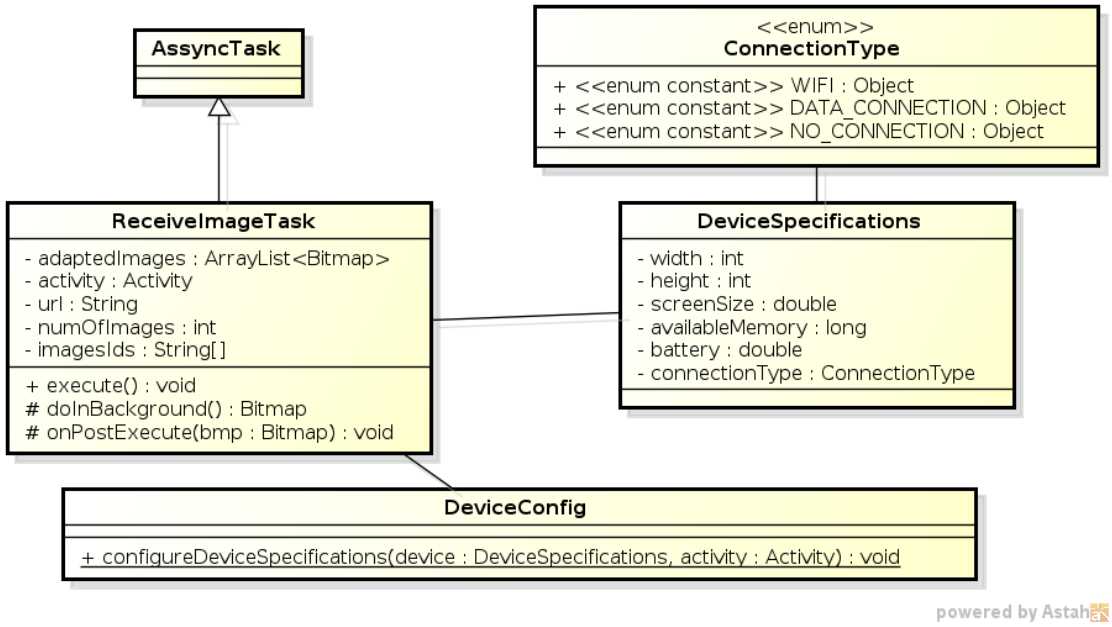
O web service desenvolvido recebe uma requisição de imagem e junto com ela um JSON com as informações do dispositivo. A imagem requisitada ficará com a resolução máxima do dispositivo. Caso o tamanho da tela do dispositivo seja menor que 3,5’’, a resolução da imagem é dividida pela metade. Por último, se o dispositivo possuir menos de

50Mb de memória RAM disponível, novamente a imagem é dividida pela metade e então enviada para a aplicação solicitante.

5 DESENVOLVIMENTO DA API

O desenvolvimento da API foi dividido em duas etapas: criação da biblioteca Android cliente e criação da biblioteca Java servidor. A biblioteca cliente é responsável por obter as informações de contexto do dispositivo, e enviar essas informações juntamente com a requisição da imagem. Os códigos da biblioteca estão disponíveis no link https://github.com/rodrigogirao/ICA. Na Figura 2 são listadas as classes, atributos e métodos existentes na biblioteca cliente da API.

Figura 2 – Diagrama de Classes da API Cliente



Fonte: Elaborada pelo autor.

A classe DeviceSpecifications contém as informações do dispositivo que serão enviadas ao servidor junto com a requisição da imagem. As informações armazenadas são resolução e tamanho da tela, memória ram disponível, porcentagem da bateria e tipo de conexão.

A classe ConnectionType é uma enumeração que contém as possibilidades de um dispositivo estar conectado à internet. O dispositivo pode estar conectado pelo WIFI, pela conexão de dados 2G, 3G ou 4G e também pode estar sem conexão com a internet.

A classe DeviceConfig contém um método estático que é responsável por buscar e calcular as informações do dispositivo.

A classe ReceiveImageTask herda da classe AssyncTask do Android. Nesta classe contém toda a lógica da biblioteca cliente da API. A classe possui dois construtores, um para receber apenas uma imagem e outro onde são recebidas várias imagens. Os parâmetros dos construtores são: um array de Bitmap, onde serão colocadas as imagens recebidas, a Activity

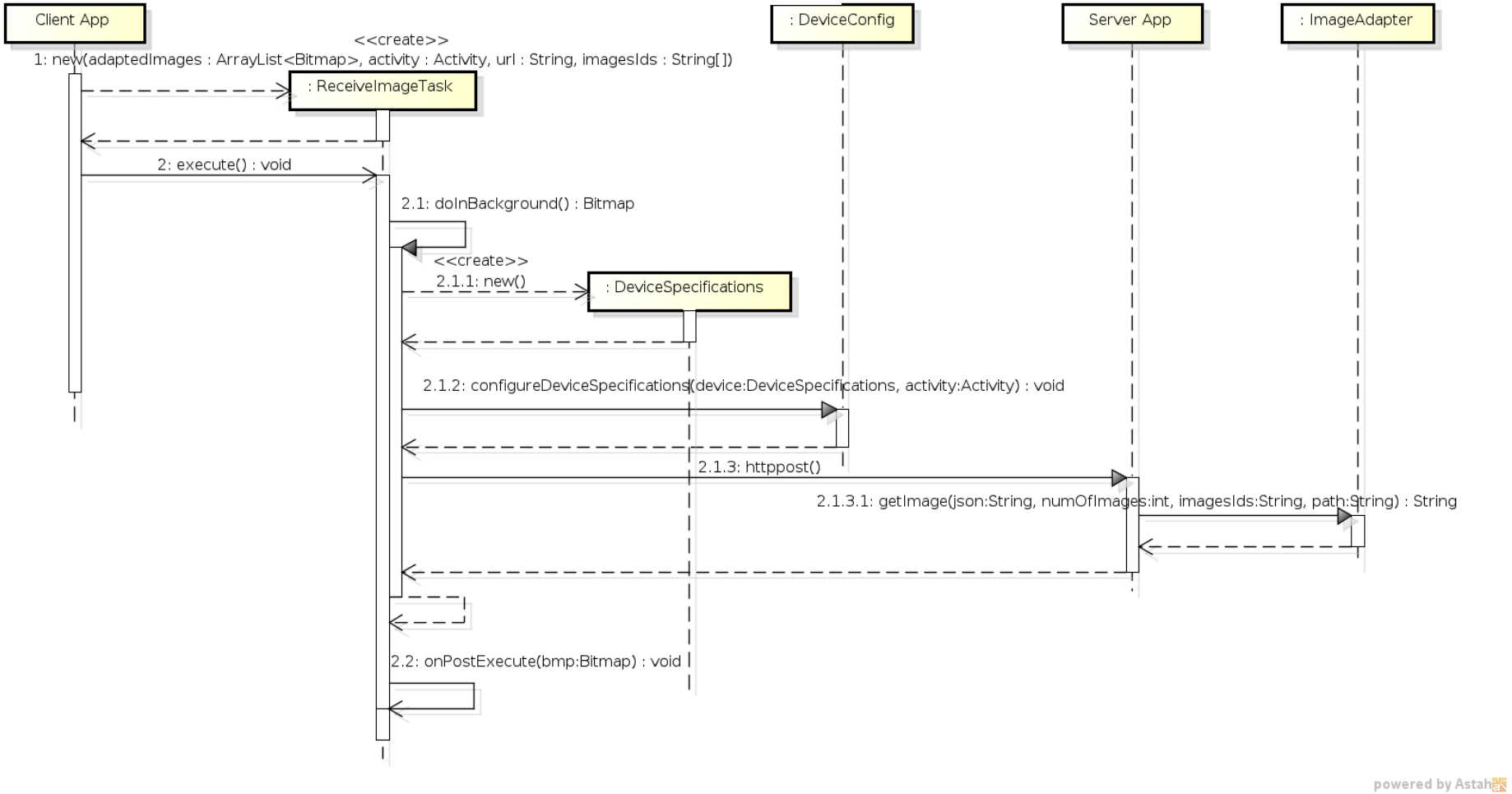
que iniciou a requisição da imagem, a url do servidor no formato String, o caminho das imagens em um vetor de String, e, no caso de várias imagens, o número de imagens requisitadas. Quando o método execute() é chamado ele aciona o método doInBackground() que cria um objeto do tipo DeviceSpecifications e chama o método da classe DeviceConfig passando esse objeto para receber as informações calculadas pelo DeviceConfig. Após isso é feita a requisição da imagem ao servidor utilizando a url e o caminho das imagens passados no construtor. Após receber a imagem, esta é adicionada ao array de Bitmap passado como parâmetro.

Já a biblioteca servidor é responsável por adaptar a imagem requisitada pelo cliente. Ela possui a classe ImageAdapter que recebe as informações de contexto e adapta as imagens requisitadas de acordo com estas. A classe DeviceSpecifications também está presente para receber as informações de contexto enviadas pelo cliente.

Na Figura 3 é mostrado um diagrama de sequência com o fluxo de uso da API sendo usada em uma aplicação cliente e uma aplicação servidor. Primeiramente a aplicação cliente cria uma instância da classe ReceiveImageTask da API passando parâmetros necessários. Após estar com uma instância do objeto da classe ReceiveImageTask a aplicação cliente chama o método execute(). Esse método chama o doInBackground() onde é feita a criação do objeto DeviceSpecifications. Quando o objeto é criado, o método configureDeviceSpecifications() da classe DeviceConfig faz a captura das informações de contexto e as armazena nesse objeto. Ainda no método doInBackground() é realizada uma requisição http do tipo POST para a aplicação servidor, onde são enviadas as informações de contexto armazenadas, e os identificadores das imagens requisitadas. A aplicação servidor chama o método getImage() da classe ImageAdapter da API, passando as informações recebidas na requisição POST e o caminho das imagens no servidor. O método getImage() adapta as imagens de acordo com as informações do dispositivo e as transforma em JSON. A aplicação servidor retorna esse JSON para o método doInBackground() que converte para Bitmaps e passa esses Bitmaps para o método onPostexecute(). O método onPostExecute() adiciona as imagens recebidas ao array passado como parâmetro na contrução da classe.

Tendo mostrado a estrutura interna e funcionalidades da API, na seção seguinte são mostradas as configurações de ambiente necessárias para o uso e também como os contratos de design devem ser estabelecidos.

Figura 3 – Fluxo de uso da API



Fonte: Elaborada pelo autor

6 APLICAÇÃO DO USO DA API

Para fazer o estudo de caso foi criada uma aplicação utilizando a API. A aplicação consiste em um álbum de fotos online, onde o usuário pode visualizar imagens sem estas estarem armazenadas no dispositivo. Os códigos da aplicação estão disponíveis no link https://github.com/rodrigogirao/ICA-Sample. A seguir será relatado como foi o desenvolvimento da aplicação utilizando a API.

6.1 Configuração do Ambiente

O ambiente necessário para poder utilizar a biblioteca cliente da API consiste apenas em uma aplicação Android, onde as dependências ica-client.jar e gson-2.2.4.jar foram colocadas em uma pasta libs dentro da aplicação. Além disso, o arquivo AndroidManifest.xml deve ser editado para incluir as permissões de internet e acessar o estado da rede: *<uses-permission android:name="android.permission.INTERNET"></uses-permission>*, *<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"></uses-permission>*.

Na aplicação servidor é necessário colocar as dependências gson-2.2.4.jar, asm-3.1.jar, jersey-bundle-1.18.jar e ica-server.jar dentro da pasta WebContent/WEB-INF/lib. Também é necessário modificar o arquivo web.xml dentro da pasta WebContent/WEB-INF para que o Jersey possa enxergar as suas classes de webservice. O código representado na Figura 4 deve ser colocado entre a tag <web-app> e a tag <param-value> deve ser modificada para conter o nome do pacote onde ficará as classes do webservice.

Figura 4 – Configuração do Jersey

```
<servlet>
  <servlet-name>ICAService</servlet-name>
  <servlet-class>
      com.sun.jersey.spi.container.servlet.ServletContainer
  </servlet-class>
  <init-param>
    <param-name>com.sun.jersey.config.property.packages</param-name>
    <param-value>nome.do.pacote.onde.ficara.o.webservice</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>ICAService</servlet-name>
  <url-pattern>/*</url-pattern>
</servlet-mapping>
```

Fonte: Elaborada pelo autor.

6.2 Uso da API na Aplicação Exemplo

Na aplicação servidor é preciso estabelecer um contrato com a API cliente. A API cliente envia dois parâmetros pela url, o número de images e o nome de cada imagem no servidor. O JSON com as informações de contexto do dispositivo é enviado dentro do conteúdo da requisição POST. A Figura 5 descreve como deve ser o contrato para o servidor poder receber e tratar as requisições.

Figura 5 – Contrato de comunicação da API

```
@POST
@Path("/{numOfImages}/{imagesIds}")
@Consumes("application/json")
@Produces("application/json")
public String sendImage(String json,
        @PathParam("numOfImages") int numOfImages,
        @PathParam("imagesIds") String imagesIds)
```

Fonte: Elaborada pelo autor.

O @POST é uma anotação para dizer qual o tipo de requisição será recebida. As anotações @Consumes e @Produces são para dizer que tipo de objeto o servidor receberá e enviará, no caso da API ambos são do tipo JSON. O @Path("/{numOfImages}/{imagesIds}") deve sempre ser o fim da url para se obter corretamente os parâmetros, além destes também serem declarados na assinatura do método: String json, @PathParam("numOfImages") int numOfImages, @PathParam("imagesIds") String imagesIds. Após receber esses parâmetros basta chamar o método getImage() da biblioteca do servidor passando os parâmetros recebidos e o local em que as imagens estão no servidor.

Com o estabelecimento do contrato na parte do servidor, agora vamos para a aplicação cliente. A primeira tela é apenas uma tela de carregamento enquanto as imagens são recebidas do servidor. Foi criado um vetor de String onde cada posição possui o nome da imagem a ser buscada no servidor. Também é criado um ArrayList de Bitmap que irá receber as imagens do servidor. Após isso, é criada uma instância da classe ReceiveImageTask que recebe o ArrayList de Bitmap, a activity atual representada pela palavra this, a url do servidor, o vetor de String com o nome das imagens e o número de imagens a serem requisitadas, como na Figura 6.

Figura 6 – Requisição de imagem

```
new ReceiveImageTask(images, this,
        "http://192.168.1.7:8080/ICA-Server-Sample/image/", imagesIds,
        imagesIds.length).execute();
```

Fonte: Elaborada pelo autor.

Após isso, é chamado o método execute(), que rodará a ReceiveImageTask. Na Figura 7 é mostrado como as imagens recebidas são colocadas em um GridView através do

seu Adapter que adiciona cada imagem em cada view do GridView. Essas imagens são apresentadas para o usuário.

Figura 7 – Mostrando imagens adaptadas para o usuário

```
gridView = (GridView) findViewById(R.id.gridview);
GridAdapter imageAdapter = new GridAdapter(ICASampleActivity.this,
        images);
gridView.setAdapter(imageAdapter);
```

Fonte: Elaborada pelo autor.

Após o desenvolvimento a aplicação servidor a mesma foi colocada em uma nuvem da Amazon para ser testada em uma ambiente de nuvem. A aplicação foi instalada em dois dispositivos: um *tablet* e um *smartphone*. O *tablet* possui uma tela de 7'' com resolução 1024 x 600 *pixels*, com 1Gb de memória ram e com uma conexão WIFI. O *smartphone* pussui uma tela de 4'' com resolução de 960x540 pixels, com 512 Mb de memória ram e com uma conexão 3G. Após alguns testes com estes dois dipositivos foi verificado que o tempo de recebimento das imagens estava muito parecido e logo a API estava funcionando.

7 CONSIDERAÇÕES FINAIS

Neste trabalho foram apresentados todas as etapas de uma proposta de desenvolvimento de uma API para criação de aplicações de adaptação de imagens. Inicialmente foram identificadas características relevantes para o usuários de aplicações móveis. Depois foi criado um protótipo para verificar a viabilidade da construção da API. No desenvolvimento da API foram feitas duas partes a parte cliente e a parte servidor. Utilizando a API desenvolvida foi feita uma aplicação exemplo, a fim de mostrar como seria o uso da API em uma aplicação real.

Após o desenvolvimento da aplicação exemplo foi constatado que a API tinha funcionado da maneira proposta porém com alguns ajustes na parte do servidor, como o contrato de comunicação que era pra ser entre as duas partes da API mas acabou sendo entre a parte cliente da API e a parte servidor da aplicação exemplo.

Com a API os desenvolvedores agora podem criar aplicações que façam uso de imagens de uma maneira simples e rápida. Com a configuração do ambiente pronta, eles podem utilizar as funções da API em qualquer parte de suas aplicações que necessite requisitar imagens a um servidor.

Para trabalhos futuros poderia ser criada uma comunicação direta entre a parte cliente e a parte servidor da API. Também poderiam ser usados os recursos de elasticidade da computação em nuvem para tratar de muitas e mais pesadas adaptações. Além disso, poderia ser trabalhada a parte da adaptação como criar algoritmos que otimizassem o processo, para poder incluir adaptações de outros conteúdos como, por exemplo, vídeo.

REFERÊNCIAS

APPLE, INC. Itunes Preview. **Itunes**, 2013. Disponível em: <https://itunes.apple.com/br/genre/ios/id36?mt=8>. Acesso em: 05 jul. 2013.

B'FAR, R. **Mobile Computing Principles**. [S.l.]: Cambridge University Press, 2005.

CHRISTENSEN, J. H. Using RESTful Web-Services and Cloud Computing to Create Next Generation Mobile Applications, 2009.

CISCO SYSTEMS, INC. **Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012–2017**. Cisco Systems, Inc. [S.l.]. 2013.

DEY, A. K. Understanding and Using Context. **Personal and Ubiquitous Computing**, v. 5, n. 1, p. 4-7, 2001.

DINH, H. T. et al. A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches. **Wireless Communications and Mobile Computing**, 2011.

ESPOSITO, D. **Architecting Mobile Solutions for the Enterprise**. [S.l.]: O'Reilly Media, Inc., 2012.

FERNANDO, N.; LOKE, S. W.; RAHAYU, W. Mobile cloud computing: A survey. **Future Generation Computer Systems**, v. Volume 29, p. 84-106, Janeiro 2013.

FILHO, O. F.; FERREIRA, M. Serviços semânticos: uma abordagem RESTful, 2011. 35-46.

GOOGLE. Loja. **Google Play**, 2013. Disponível em: <https://play.google.com/store>. Acesso em: 05 jul. 2013.

HELD, A.; BUCHHOLZ, S.; SCHILL, A. Modeling of Context Information for Pervasive Computing Applications. **Proceedings of SCI**, 2002.

KOVACHEV, D.; CAO, Y.; KLAMMA, R. Mobile Cloud Computing: A Comparison of Application Models. **arXiv**, 2011.

MEIER, R. **Professional Android 4 Application Development**. 3. ed. [S.l.]: Wrox, 2012.

ORACLE CORPORATION. JAX-RS Specification Wiki, 2012. Disponivel em: <https://java.net/projects/jax-rs-spec/pages/Home>. Acesso em: 15 Abril 2014.

ORACLE CORPORATION. Jersey - RESTful Web Services in Java, 2014. Disponivel em: <https://jersey.java.net/>. Acesso em: 15 Abril 2014.

SINGH, I.; LEITCH, J.; WILSON, J. Gson User Guide, 2014. Disponivel em: <https://sites.google.com/site/gson/gson-user-guide>. Acesso em: 12 Abril 2014.

SOUSA, F. R. C.; MOREIRA, L. O.; MACHADO, J. C. Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios. **ERCEMAPI**, 2009.

SOUSA, F. R. C.; MOREIRA, L. O.; MACHADO, J. C. Gerenciamento de Dados em Nuvem: Conceitos, Sistemas e Desafios. **Simpósio Brasileiro de Banco de Dados**, 2010. 101-130.

SOUSA, F. R. C. **Software como um Serviço.** Universidade Federal do Ceará. Fortaleza. 2009.

SYMANTEC CORPORATION. **State of Mobility Latin America Results**. [S.l.]. 2013.

VIEIRA, V.; TEDESCO, P.; SALGADO, A. Modelos e Processos para o desenvolvimento de Sistemas Sensíveis ao Contexto. **André Ponce de Leon F. de Carvalho, Tomasz Kowaltowski.(Org.). Jornadas de Atualização em Informática**, 2009. 381-431.

VIEIRA, V.; TEDESCO, P.; SALGADO, A. Designing context-sensitive systems: An integrated approach. **Expert Systems with Applications**, v. 38, n. 2, p. 1119-1138, 2011.