Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

ContextWS: Infraestrutura para auxílio à criação de aplicações sensíveis ao contexto.

Marcelo de Oliveira Costa Machado

JUIZ DE FORA
JUNHO, 2015

ContextWS: Infraestrutura para auxílio à criação de aplicações sensíveis ao contexto.

Marcelo de Oliveira Costa Machado

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Eduardo Barrére

JUIZ DE FORA

JUNHO, 2015

ContextWS: Infraestrutura para auxílio à criação de aplicações sensíveis ao contexto.

Marcelo de Oliveira Costa Machado

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Eduardo Barrére
Doutor em Engenharia de Sistemas e Computação, UFRJ.

Daves Marcio Silva Martins
Mestrado em Sistemas Computacionais, UFRJ.

Edmar Welington Oliveira
Mestrado em Informática, UNIRIO.

JUIZ DE FORA

29 DE JUNHO, 2015

Resumo

Dispositivos computacionais, com capacidade de captura de mídias, cada vez mais fazem parte do cotidiano dos usuários. Neste cenário, sistemas computacionais e aplicações têm o desafio de fornecer melhores conteúdos e experiências aos usuários em vários aspectos, como por exemplo em usabilidade ou eficiência. Uma boa alternativa que vem sendo proposta na literatura é o uso de informações de contexto, que são basicamente parâmetros de uma entidade que definem o que é importante ou não para os usuários. No entanto, a construção de sistemas baseados em informações contextuais não é algo trivial, tendo em vista que essas informações vêm de bases heterogêneas. Neste trabalho é apresentada uma infraestrutura genérica de armazenamento e consulta de informações de contexto, baseada em mídias, que poderão ser consumidas para a criação de aplicações em diversas plataformas.

Palavras-chave: Sistemas Sensíveis ao Contexto, Informações de Contexto, Mídias, Sistemas Multimídia.

Abstract

More and more computer devices with capability of capture media are part of our daily life, thus computer systems and applications are challenged to provide the best experiences for users on several aspects like in usability or efficiency. A good alternative that has been proposed on the literature is the use of context informations, which are basically parameters of an entity that define what is important or not for users. However, building systems based on context information is not a trivial task, given that these informations are gathered from heterogeneous databases. This paper presents a generic infrastructure for storage and query context informations of media that can be consumed by applications of multiples platforms.

Keywords: Context Aware Systems, Context Information, Media, Multimedia Systems.

Agradecimentos

Agradeço primeiramente a Deus por todas bençãos despejadas em minha vida até o momento.

Agradeço minha família pela educação, apoio e paciência em toda minha jornada. Principalmente, minha mãe que com toda sua humildade me deu uma educação digna.

Agradeço aos amigos que sempre estiveram ao meu lado nos momentos de alegria ou tristeza, dando palavras motivadoras ou "puxões de orelha" quando necessário. Não citarei nomes, afinal, a lista seria imensa. Mas sei que cada um que ler este trabalho saberá que contribuiu de alguma forma e que com certeza foi lembrado por mim.

Por fim, agradeço aos meus professores e arte-educadores que ajudaram a formar minha educação, caráter e personalidade. Neste momento agradeço principalmente ao professor e orientador Eduardo Barrére pela paciência e ensinamentos não somente para o presente trabalho, mas também por toda minha vida acadêmica.

"Run, rabbit run

Dig that hole, forget the sun

And when at last the work is done

Don't sit down it's time to dig another one."

Pink Floyd

Sumário

Lista de Figuras **8**

Lista de Abreviações **9**

1 Introdução **10**
1.1 Contextualização . . . 10
1.2 Justificativa . . . 11
1.3 Objetivos . . . 12
1.3.1 Objetivos Gerais . . . 12
1.3.2 Objetivos Específicos . . . 12
1.4 Metodologia . . . 12
1.5 Estrutura do trabalho . . . 13

2 Pressupostos e Revisão da Literatura **14**
2.1 Fundamentação Teórica . . . 14
2.1.1 Mídia . . . 14
2.1.2 Informações de Contexto . . . 15
2.1.3 Sistemas Sensíveis a Contexto . . . 16
2.1.4 Web Services . . . 16
2.2 Trabalhos Relacionados . . . 18
2.2.1 MultiS: Um Servidor de Contexto voltado à Computação Pervasiva 18
2.2.2 Development of a new Cliente-Server Architecture for Context-Aware Mobile Computing . . . 19
2.2.3 Uma Proposta Baseada em Web Semântica para Sensibilidade ao Contexto na Computação Ubíqua . . . 20
2.2.4 The Design and Use of a Generic Context Server . . . 21
2.2.5 Conclusão . . . 22

3 ContextWS: Servidor de Informações de Contexto **24**
3.1 Arquitetura de armazenamento . . . 25
3.2 Arquitetura de captura . . . 26
3.3 Modelagem das informações de contexto . . . 27
3.4 Banco de Dados . . . 28

4 Desenvolvimento e Implementação **30**
4.1 Serviços . . . 31
4.1.1 Adicionar recursos . . . 32
4.1.2 Capturar recursos . . . 34

5 Prova de conceito **35**
5.1 Testes com Advanced REST Client . . . 35
5.1.1 Capturar recursos . . . 35
5.1.2 Adicionar recursos . . . 37
5.2 Testes com aplicativo Android . . . 39

6 Conclusão **41**

Referências Bibliográficas **43**

Lista de Figuras

3.1 Arquitetura de armazenamento de recursos utilizando o ContextWS 26
3.2 Arquitetura de captura de recursos utilizando o ContextWS 26
3.3 Modelo Entidade-Relacionamento do Banco de Dados 29

4.1 Servidor em funcionamento. 32

5.1 Acesso ao servidor para captura de recurso a partir de um usuário. 35
5.2 Resposta do servidor para captura de recurso a partir de um usuário. . . . 36
5.3 Simulação de erro ao passar token vazio. 37
5.4 Acesso ao servidor para armazenamento de recurso. 37
5.5 Resposta do servidor para armazenamento de recurso. 38
5.6 Dados persistidos no banco de dados. 38
5.7 Interface da aplicação Android de teste. 39
5.8 Resposta da requisição feita ai servidor através de uma aplicação Android. 40

Lista de Abreviações

XML	*eXtensible Markup Language*
MER	Modelo de Entidade e Relacionamento
SGBD	Sistema de Gerenciamento de Banco de Dados
LApIC	Laboratório de Aplicações e Inovação em Computação
UFJF	Universidade Federal de Juiz de Fora
HTTP	*HyperText Transfer Protocol*
HTTPS	*Hyper Text Transfer Protocol Secure*
REST	*REpresentational State Transfer*
SOAP	*Simple Object Access Protocol*
URIs	*Uniform Resource Identifiers*
MVC	*Model-View-Controller*
API	*Application Programming Interface*
ISAM	Infraestrutura de Suporte às Aplicações Móveis
CD-XML	*Context Definition eXtensible Markup Language*
EXEHDA	*Execution Environment for Highly Distributed Applications*
URL	*Uniform Resource Locator*
IDE	*Integrated Development Environment*
OWL	*Web Ontology Language*
JSON	*JavaScript Object Notation*
CSV	*Comma-Separated Values*
DCC	Departamento de Ciência da Computação

1 Introdução

1.1 Contextualização

A cada dia que passa pode-se perceber o crescente uso de dispositivos móveis. Os famosos celulares que a pouco eram uma inovação, hoje são chamados de *smartphones* devido ao grande aumento de suas funcionalidades. Esses aparelhos, agora, não servem apenas para fazer uma ligação telefônica ou receber mensagens, mas também para acessar a internet, enviar correios eletrônicos, jogar, utilizar aplicativos, acessar contas bancárias, acessar redes sociais e capturar mídias — comportando-se como computadores.

No momento em que uma mídia é capturada, várias informações complementares são armazenadas. No caso de um vídeo, por exemplo, tem-se sua duração, formato, título, quantidade de quadros por segundo, resolução, entre outras. Porém, existem outras informações sobre essas mídias que são relevantes — como o momento e local onde foram capturadas. O conjunto de informações que descrevem uma entidade ou uma situação da entidade, são chamadas de informações de contexto, neste caso a entidade descrita é uma mídia, mas poderia ser uma pessoa, lugar ou objeto — desde que seja relevante para a interação entre um usuário e uma aplicação (Dey and Abowd, 1999).

A partir das informações de contexto, uma mídia deixa de ser apenas um conteúdo a ser exibido e passa a colaborar com a criação de aplicações de diversas naturezas. As aplicações que utilizam essas informações de contexto se encaixam em uma área da computação que vem sendo bastante abordada: a Computação Sensível ao Contexto do Inglês *Context-Aware*. Esse termo está relacionado à ideia dos computadores serem capazes de perceber e reagir ao ambiente em que estão inseridos, com o intuito de facilitar as atividades dos seres humanos (Araújo, 2003). Exemplos potenciais de aplicações sensíveis ao contexto são: Alterar o estado de um *smartphone*, dependendo do local que o usuário se encontra (e.g. ocupado em uma reunião); reconhecer, automaticamente, monumentos e pessoas que se encontram em uma imagem; aplicações que dão alertas relacionados à saúde do usuário, entre outras.

1.2 Justificativa

Seres Humanos, na maioria das vezes, são bem sucedidos em transferir uma ideia entre si e então reagir apropriadamente (Dey and Abowd, 1999). Isso se deve a vários fatores: riqueza da língua que compartilham, comum entendimento de como o mundo funciona e um implícito conhecimento das situações diárias. Infelizmente, essa ideia ainda não é bem adaptada para a relação humano-computador. De fato, mesmo com a constante evolução da computação, ainda é necessário que os usuários de uma aplicação se esforcem — ajustando suas preferências de uso, respondendo questionários e transmitindo informações — na tentativa de mostrar seus estados atuais e atividades que estão realizando.

A computação sensível ao contexto busca resolver este problema, criando aplicações que, através de informações de contexto, consigam perceber a intenção do usuário e, assim, reagir a uma solicitação — minimizando os esforços do mesmo na execução de suas tarefas.

Mídias são largamente utilizadas em aplicações, devido à riqueza de representação e apresentação da informação. São a partir dessas entidades, por exemplo, que a educação à distância é possível. Mídias capturadas via dispositivos móveis podem agregar uma série de informações de contexto, as contidas na própria mídia (locais, objetos, pessoas) e as contidas nos dispositivos (localidade, data, tipo de dispositivo, etc.). Trabalhar com essas informações de contexto pode ser bem útil em aplicações de propósitos gerais, como de segurança pública (desastres, problemas diversos) e também para cobertura de eventos (jogos, shows). Muitos trabalhos científicos têm sido publicados com o intuito de promover a computação sensível ao contexto. Entretanto, os trabalhos já realizados focam, na maioria das vezes, em aspectos específicos do contexto (Kappel et al., 2002), assim como pode ser visto na seção 2.2. O trabalho apresentado nesta monografia busca criar uma infraestrutura que irá possibilitar o armazenamento e o consumo de informações de contexto, baseadas em mídias, para auxiliar na criação de sistemas sensíveis ao contexto — independente de plataformas — fazendo com que seja possível a produção de mais serviços computacionais úteis, além de colaborar com aspectos da criação de sistemas ubíquos[1] escaláveis — um dos sete desafios para a ciência da computação nos próximos

[1] A Computação Ubíqua vem da ideia de tornar a interação homem computador invisível, ou seja,

vinte anos, de acordo com trabalho divulgado por cientistas da *The British Computer Society* no início de 2005 (Hoare and Milner, 2005).

1.3 Objetivos

1.3.1 Objetivos Gerais

O principal objetivo deste trabalho é a criação de uma infraestrutura — baseada em serviços relacionados à informações de contexto de mídias — que irá apoiar a criação de ambientes sensíveis ao contexto. Os serviços oferecidos por essa infraestrutura deverão ser acessados a partir de URLs que irão armazenar e servir informações relacionadas a uma dada mídia.

1.3.2 Objetivos Específicos

Para construção da infraestrutura supracitada se faz necessário modelar como serão apresentadas as informações de contexto. Assim, este trabalho também tem como objetivo a criação de um modelo de representação dessas informações. Por fim, objetiva-se a criação de uma aplicação de teste que será utilizada como prova dos conceitos apresentados.

1.4 Metodologia

Para realização deste trabalho, primeiramente, foi feito um levantamento teórico acerca dos termos utilizados. Em seguida, foi feito o estudo de casos e levantamentos de requisitos para delimitação de módulos e funcionalidades da infraestrutura. Então, foi feita a modelagem e seleção de ferramentas para que ela fosse implementada.

Por fim, testes foram feitos utilizando aplicações que armazenam e consomem informações de contexto desta infraestrutura de forma a provar o funcionamento de seus serviços, mostrando que esses podem ser utilizados para construção de aplicações reais sensíveis ao contexto.

integrar a informática com as ações e comportamentos naturais das pessoas de forma que elas nem percebam que estão dando comandos a um computador, mas sim conversando com alguém (Kahl and Floriano, 2011).

1.5 Estrutura do trabalho

Além deste capítulo introdutório, o presente trabalho está dividido conforme a seguir:

No segundo capítulo, é realizada uma revisão da literatura promovendo um maior detalhamento dos conceitos que compõem este trabalho.

O terceiro capítulo, propõe uma infraestrutura que irá armazenar e servir informações contextuais para construção de aplicações sensíveis ao contexto baseada em mídias.

No quarto capítulo, são apresentados os detalhes dos serviços implementados.

No quinto capítulo, são mostrados resultados de testes realizados utilizando-se aplicações que consomem os serviços implementados.

Por fim, o sexto capítulo apresenta as considerações finais.

2 Pressupostos e Revisão da Literatura

2.1 Fundamentação Teórica

Com o crescimento do uso de dispositivos computacionais, cresce o interesse em fazer com que as aplicações executadas por esses dispositivos apresentem caráter adaptativo, sendo capazes de realizar análises a respeito do contexto atual de interesse do usuário e se adaptar em atendimento às necessidades particulares deste usuário. Dominar esses conceitos foi de suma importância para a realização dos objetivos deste trabalho.

2.1.1 Mídia

A palavra mídia, vem do latim "medius", que significa: "Meio; instrumento mediador; elemento de intermédio" (Houaiss and Villar, 2012). Um conceito um tanto quanto abstrato, fazendo com que a palavra mídia seja amplamente utilizada em diversos contextos. Na maioria das vezes em que se fala sobre o conceito de mídia este está ligado à área da Comunicação Social, podendo ser facilmente substituída por meio de comunicação. O conceito de mídia na ciência da computação também tem vários significados, dentre os quais o de Mídia de Representação é o mais relevante para este trabalho.

De acordo com Herrtwich and Stwinmetz (1990), mídia de representação é qualquer forma de representar a informação a partir de um sistema computacional. Por exemplo, a mídia texto é representada em forma de sequências visuais de caracteres e a mídia áudio é representada acusticamente a partir de ondas sonoras. Herrtwich and Stwinmetz (1990) cita ainda duas classificações das mídias de representação: mídias discretas e mídias contínuas. As mídias discretas são aquelas cuja informação codificada é atemporal, ou seja, não depende da dimensão tempo, tais como imagem estática e texto. Já as mídias contínuas dependem da dimensão tempo, como o vídeo e o áudio. Percebe-se então que os termos "discreta" e "contínua" não se referem à uma representação de dados interna mas sim da "visão" do usuário perante a mídia.

Neste trabalho quando citada a palavra mídia isoladamente, significa mídia de representação.

2.1.2 Informações de Contexto

Embora existam várias definições sobre informações de contexto, as apresentadas neste trabalho são as mais utilizadas na literatura. Segundo Fleischmann (2012), de forma simples, pode-se entender que contexto se trata de tudo que ocorre ao redor do usuário e que influencia na forma com que este interage com o ambiente físico e com as outras pessoas. Para Vieira et al. (2009), contexto é o que está por trás da habilidade de discriminar o que é ou não importante em um dado momento e, com isso, permite ajudar indivíduos a melhorar a qualidade de conversação e a compreender certas situações, ações ou eventos.

Schilit et al. (1994) considera que existam três aspectos importantes sobre o contexto: onde você está, com quem você está e quais recursos se encontram próximo de você. Além disso, categoriza o contexto em 4 itens:

- Contexto computacional: rede, conectividade, custo da comunicação, banda passante, recursos, etc.
- Contexto do usuário: perfil do usuário, posição, velocidade, entidades próximas, etc.
- Contexto físico: luminosidade, temperatura, nível de ruído, etc.
- Contexto temporal: hora, mês, ano, época do ano, etc.

Já Dey and Abowd (1999) tem uma definição mais abrangente: "Contexto é qualquer informação que pode ser utilizada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, lugar ou objeto considerado relevante para a interação entre um usuário e uma aplicação, incluindo o usuário e a aplicação por si próprios.". Outra definição semelhante é fornecida por Moore (2009), que define o contexto como informações relativas às propriedades que se combinam para descrever e caracterizar uma entidade e seu papel de uma forma legível ao computador.

No presente trabalho foram adotadas, principalmente, a definição de Dey and Abowd (1999), onde as entidades citadas são as mídias. E para modelagem das informações de contexto foram utilizadas partes da categorização de Schilit et al. (1994).

2.1.3 Sistemas Sensíveis a Contexto

Sistemas sensíveis ao contexto são definidos como aqueles que utilizam informações de contexto para prover serviços relevantes ao usuário, onde a relevância depende das tarefas desempenhadas pelo mesmo. (Dey and Abowd, 1999). Uma definição parecida é exposta por Zimmermann et al. (2007) que considera aplicações sensíveis ao contexto como uma entidade, na qual utiliza elementos de contexto para adaptar seu comportamento de forma a prover serviços ao usuário final.

Segundo Fleischmann (2012), a adaptação e sensibilidade ao contexto são temas que andam juntos em diferentes momentos. A sensibilidade ao contexto não necessariamente implica adaptação, mas um sistema adaptativo, em geral, precisa estar ciente da ocorrência de determinados eventos no sistema e reagir de forma adequada a estes eventos. Independentemente da aplicação, pode-se citar uma série de áreas que têm feito uso de adaptação e/ou sensibilidade ao contexto, sendo mais característico de aplicações autodenominadas móveis, pervasivas[2] e/ou ubíquas. Fleischmann (2012) complementa que nesses tipos de sistemas a sensibilidade ao contexto é usada para manter o sistema consistente em termos de funcionamento, e também mantê-lo capaz de proativamente perceber as melhores condições de ação e agir de acordo com esta percepção enquanto o usuário se movimenta, por exemplo.

Não há muitas variações nos conceitos deste termo, deixando qualquer ambiguidade relacionada às informações de contexto. Portanto, para este trabalho foram levados em consideração os conceitos supracitados, sendo então, sistemas sensíveis ao contexto aqueles que basicamente gerenciam dados relacionados a uma aplicação em um domínio e usam as dimensões do contexto para apoiar um agente de execução de alguma tarefa.

2.1.4 Web Services

Os *Web Services* são tecnologias emergentes, muito utilizadas na integração de aplicações. Uns apontam essa tecnologia como o caminho a seguir no desenvolvimento de aplicações

[2]No conceito de computação pervasiva o computador está embarcado ao ambiente de maneira invisível para o usuário tendo a capacidade de capturar informações de seu contexto e então utilizá-la para garantir melhor adequação as características do ambiente. O ambiente também pode e deve ser capaz detectar outros dispositivos que adentrem a ele.(ARAÚJO, 2003)

distribuídas, enquanto que outros vêm nela apenas mais uma evolução de um conceito antigo.

De acordo com Lopes and Ramalho (2004) os *Web Services* são aplicações modulares, auto-descritivas, acessíveis através de um URL, independente das plataformas de desenvolvimento e que permitem a interação entre aplicações sem intervenção humana . Lopes and Ramalho (2004) concluem que os *Web Services* apresentam-se como a solução para muitos problemas associados aos sistemas distribuídos, nomeadamente nas questões relacionadas com a integração de sistemas heterogêneos.

Para o W3C (2004) um *Web Service* define-se como um sistema de software projetado para suportar a interoperabilidade entre máquinas sobre rede. Utilizando uma maneira mais prática, pode-se dizer que um *Web Service* é uma tecnologia que visa oferecer serviços, disponíveis publicados na *web* (Menéndez, 2002).

Cada aplicação pode ter a sua própria linguagem, que é traduzida para uma linguagem universal em um formato como o XML, JSON, CSV, etc.

As bases para a construção de um *Web Service* são os padrões como o XML e SOAP. O transporte dos dados é realizado normalmente via protocolo HTTP ou HTTPS para conexões seguras. Os dados são transferidos no formato XML, encapsulados pelo protocolo SOAP, também é bastante comum usar o protocolo REST(Representational Transfer Protocol), para transferir o estados do dado para a aplicação (W3C, 2004).

O entendimento sobre os *Web Services* é fundamental para este trabalho, pois a infraestrutura a ser implementada se baseia nos aspectos dessa tecnologia

2.2 Trabalhos Relacionados

2.2.1 MultiS: Um Servidor de Contexto voltado à Computação Pervasiva

O trabalho de Fehlberg (2007) tem como objetivo apresentar uma solução para a aquisição e tratamento de informações de contexto, oriundas de multi-sensores, para colaborar com a computação pervasiva. O problema em questão é, portanto, a criação de um sistema genérico de tratamento de informações contextuais capaz de compor novas informações a partir de inúmeros sensores. A solução deve oferecer facilidade de utilização, capacidade de reação às modificações no ambiente e, ainda deve estar preparada para suportar inúmeras aplicações executando concorrentemente. Diversos requisitos são necessários para resolver o problema supracitado. O primeiro deles é a criação de um sistema de reconhecimento de contexto capaz de atender as necessidades das aplicações móveis, pervasivas e conscientes de contexto, conforme a semântica estabelecida em uma arquitetura chamada de ISAM (Infraestrutura de Suporte às Aplicações Móveis). O principal requisito desse sistema é a capacidade de gerar novos dados de contexto a partir de diferentes sensores e de disponibilizar essa informação às aplicações interessadas. É importante que o sistema esteja preparado para comunicar às aplicações as modificações que ocorrem no ambiente, atendendo requisições do tipo inscrição/notificação. Além disso, a solução proposta dever ser facilmente extensível, possibilitando a inserção de novos mecanismos para criação e processamento de informações de contexto.

Outro requisito diz respeito à comunicação entre aplicações e o sistema de reconhecimento de contexto. Cada aplicação tem sua própria maneira de compor informações de sensores e seus próprios requisitos. A descrição sobre como as informações dos sensores precisam ser compostas, para formar os dados de contexto de alto nível, precisa, portanto, ser repassada pela aplicação ao sistema de reconhecimento de contexto. Para que essa comunicação seja simples para ser escrita por programadores sem que eles tenham conhecimentos profundos da infraestrutura é desenvolvida uma linguagem para transformação e composição de informações de contexto.

A arquitetura ISAM, citada, foi concebida para habilitar as aplicações a obter as

informações do ambiente onde executam e então permite que elas se adéquem às alterações que o correm durante o curso da execução (Augustin et al., 2001). É ainda proposto uma linguagem de programação e um ambiente de desenvolvimento, especialmente desenvolvidos para atender às necessidades das aplicações da computação pervasiva. A integração de informações de diferentes sensores é feita com a linguagem CD-XML (Context Definition Extensible Markup Language), uma linguagem simples que pode inclusive ser programada através de ferramentas de desenvolvimento populares, que foi criada para que as aplicações conscientes do contexto possam descrever como devem ser relacionados os dados dos sensores para produzir informações de contexto de alto nível, permitindo que as aplicações reajam a modificações no ambiente e adaptem seu comportamento.

2.2.2 Development of a new Cliente-Server Architecture for Context-Aware Mobile Computing

O principal objetivo da dissertação de Gui (2009) é desenvolver uma aplicação de busca na qual provê dados e serviços relevantes aos usuários para dada situação e cuja concepção é capaz de adaptar-se ao dia a dia e alterações dinâmica de usuários.

Gui (2009) diz que para apoiar utilitários sensíveis a contexto, que possam ser utilizados em qualquer lugar, uma infraestrutura apropriada é essencial. A aquisição do contexto logo no início do processo realmente afeta a precisão da consciência do contexto em níveis mais elevados, erros ou viés nesse momento certamente irá se propagar e multiplicar em fases posteriores. Resumindo, a infraestrutura para captura de contexto é crucial para o sucesso de aplicações nesse âmbito. Uma solução para a aquisição de contexto foi proposta por Salber et al. (1999) que é o *Context Toolkit* que tem como principal objetivo gerar "widget" orientado a objetos na qual interage com sensores de baixo nível. O widget encapsula sensores de baixo nível e gera acesso à aplicações sensíveis ao contexto.

O projeto para desenvolvimento do software foi divido em várias partes:

- A interpretação do contexto
- Registro de serviços

- Gerenciamento de serviços
- Interpretação de uma consulta/requisição
- Política de registro
- Motor de busca com reconhecimento de contexto.

O trabalho em questão mostra alguns passos de como o usuário interage com o sistema de busca e, também, alguns passos de como o sistema responde as requisições, assim como detalhes de implementação.

2.2.3 Uma Proposta Baseada em Web Semântica para Sensibilidade ao Contexto na Computação Ubíqua

Ontologias têm sido largamente utilizadas em áreas como gerenciamento de conteúdo e conhecimento, comércio eletrônico e Web semântica. Na representação de ambientes ubíquos elas comumente descrevem entidades envolvidas e suas respectivas propriedades, tipos de aplicações, serviços, dispositivos, usuários, entre outros. Além disso, definem descrições padrões para localização, atividades, informações sobre temperatura, etc. Ou seja, elas são utilizas para caracterizar as informações de contexto de uma entidade.

A intensão do trabalho de Venecian et al. (2009) é definir um mecanismo de sensibilidade ao contexto que atue sobre células distintas com a perspectiva de ser integrado ao **middleware** EXEHDA(*Execution Environment for Highly Distributed Applications*) e que tenha por base o uso de ontologias para a representação dos estados contextuais.. Tem como objetivo também, difundir o conhecimento pertinente à área de suporte à execução de aplicações na computação ubíqua, sobre tudo no que diz respeito aos aspectos de sensibilidade ao contexto.

Os problemas em questão são promover a interoperabilidade entre as ontologias que caracterizam o contexto dos diferentes domínios administrativos (células de execução) e, a construção e processamento de um contexto que represente a agregação de diversas células de execução, selecionadas no ambiente ubíquo por um critério, por exemplo, vizinhança. A proposta é uma integração semântica de ontologias na qual cada célula tem

sua própria ontologia, mas o vocabulário das mesmas está relacionada a uma ontologia em comum. A razão para escolha dessa estratégia é sua complexidade computacional reduzida a qual faculta alta escalabilidade, permitindo a manipulação de um número maior de células de execução.

As ontologias foram utilizadas como tecnologias, pois proporcionam a interoperabilidade semântica entre os contextos, além de facilitar o reúso.

2.2.4 The Design and Use of a Generic Context Server

Embora a computação sensível a contexto seja um componente chave para interfaces de usuário, ainda são necessárias infraestruturas genéricas para o desenvolvimento de aplicações sensíveis ao contexto. No trabalho de Salber and Abowd (1998), é proposta uma infraestrutura genérica baseada em servidores de contexto que armazenam e compartilham dados contextuais que são acessíveis por aplicações de rede executadas em dispositivos móveis, computadores de mesa e dispositivos embarcados. É descrito também algumas aplicações que foram construídas que aproveitam do compartilhamento de contextos históricos.

É dito que a falta de infraestrutura genérica requer que os desenvolvedores contem com soluções personalizadas para a manipulação de contexto e impede o desenvolvimento de novas aplicações tendo em vista que não existe uma maneira simples de se trabalhar com esses dados. Sabendo da existência desse problema e as necessidades gerais de se ter um servidor de informações contextuais são citadas três metas:

- Permitir acesso à dados de contexto em ambientes heterogêneos, a partir de redes remotas ou locais.
- Acomodar uma variedade de aplicações, sensores e operações nos dados de contexto
- Preservar o histórico de dados de contexto sensoreados.

No modelo proposto os dados de contexto são detectados pelos dispositivos e reúnem-se em um repositório de um computador. As informações de sensores distintos são fundidas gerando uma descrição resumida. Um computador é ligado a uma pessoa (indivíduos ou grupos) ou locais (e.g. salas, prédios, veículos). Cada computador executa

um servidor de contexto que realiza o gerenciamento de contexto e permite o acesso aos dados. Além disso, os servidores executam serviços, chamados de sintetizadores de contexto, que atuam sobre esses dados, sendo locais ou remotas, para gerar informações de contexto em um nível mais alto de abstração.

Para permitir o acesso por clientes heterogêneos, a API proposta é uma API de rede com base em XML e HTTP.

Por fim o trabalho cita que servidores de contexto são particularmente adequados para o compartilhamento de dados de contexto e fornecimento de acesso a esses dados. A partir destas características se torna possível explorar novas aplicações promissoras.

2.2.5 Conclusão

Os trabalhos apresentados mostram semelhanças entre si e com o tema deste trabalho.

Ressaltando os pontos em comum, foi possível notar que todos têm intuito de colaborar com a computação ubíqua. No primeiro trabalho apresentado embora o termo em ênfase seja a computação pervasiva é fácil perceber que o termo está facilmente ligado a computação ubíqua, não somente por ser um termo que faz parte do seu conceito, mas sim por estar sendo utilizado em vários momentos como se fossem sinônimos. Todos os trabalhos têm a pretensão de criar uma plataforma onde as informações de contexto possam ser reutilizadas — o que é um fator relevante tendo em vista que caso contrário, poder-se-ia armazenar inúmeras informações representadas de formas diferentes, mas com o mesmo significado — gerando um problema de sobrecarga de informações assim como existe hoje em dia na web sintática. Essa abordagem mostra então que não basta ter um servidor de armazenamento de informações; deve-se, também, criar soluções de armazenamento inteligente. Uma abordagem largamente utilizada com intuito de melhorar o problema da web sintática é a utilização de ontologias, dando grande relevância ao trabalho de Venecian et al. (2009).

Outro ponto comum entre os trabalhos é a utilização da linguagem de marcação XML para criação de informações de contexto, mesmo o trabalho de Venecian et al. (2009) que utiliza ontologias na qual a linguagem é a OWL(*Web Ontology Language*), pode-se manter a comparação, pois não deixa de ser uma linguagem baseada em XML. Esse fato

acontece por essa linguagem já ser bem difundida no sentido de descrever objetos, além da existência de vários *parsers* em várias linguagens de programação que manipulam estes arquivos.

Um fator em comum entre os trabalhos citados, porém que difere deste, é o foco dado. Estes trabalhos têm como primazia as informações contextuais dos usuários, concentrando-se em como criar estratégias para sua representação. Já este trabalho tem como principal foco trabalhar com informações contextuais de mídias, especializando os tipos de aplicações que podem ser criadas para facilitar e melhorar a vida cotidiana e melhorar a interação entre humanos e computadores.

Embora os outros trabalhos citem uma infraestrutura de armazenamento de informações contextuais, eles não demonstram sua criação, citando apenas como APIs irão utilizar esse ambiente para criação de aplicativos ou geração de conteúdo. Já o trabalho apresentado na presente monografia tem como diferencial justamente o fato de focar nessa infraestrutura — que tem como objetivo fornecer serviços que possibilitem gerenciar informações de contextos, de mídias, para criação de diversas aplicações sensíveis ao contexto.

3 ContextWS: Servidor de Informações de Contexto

Até o presente momento, foi citado que o objetivo deste trabalho é a criação de uma infraestrutura que irá fornecer e armazenar informações de contexto relacionadas a mídias. Essa infraestrutura é basicamente um servidor, baseado em serviços que podem ser consumidos a partir de requisições HTTP. Este capítulo tem o intuito de apresentar a arquitetura geral deste servidor — que juntamente com outros módulos já desenvolvidos, pelo grupo de pesquisa do Laboratório de Aplicações e Inovação em Computação(LApIC) do Departamento de Ciência da Computação(DCC) da Universidade Federal de Juiz de Fora(UFJF), e outros que ainda serão desenvolvidos, formarão um conjunto poderoso de ferramentas para a construção de aplicações sensíveis ao contexto.

A ideia geral do servidor de contexto é que ele esteja preparado para receber informações de contexto de várias bases de dados — relacionadas a mídias — para isso, em usos reais desta infraestrutura, se faz necessário um tratamento rigoroso das requisições. O presente trabalho não tem como intuito criar módulos de captura de contexto, porém, foi considerado o uso do **ContextF** proposto por Barbosa and Barrére (2013).

O ContextF é um *framework* que busca facilitar o desenvolvimento de aplicações sensíveis ao contexto, utilizando uma API simplificada que permite que aplicações passem recursos a serem anotados, com informações, sem se preocupar com todo processo de extraí-las do contexto, sincronizar com um servidor remoto ou armazenar em caso de indisponibilidade de rede (Barbosa and Barrére, 2013). Ele é baseado em uma arquitetura orientada a eventos e o elemento que viabiliza essa arquitetura é chamado de *Bus*. O *Bus* registra os componentes para reagirem aos eventos disparados por ele. O ponto de partida da arquitetura é uma chamada da API do *framework* por uma aplicação, passando um recurso que se deseja contextualizar (Barbosa and Barrére, 2013). Cada etapa do processo de contextualização tem um comportamento bem definido implementado por componentes baseados em técnicas e linguagens orientadas a objetos, os componentes que valem ser

citados são:

- Componente de Extração: É responsável pela extração das informações de contexto de um recurso. Essas informações podem vir de várias fontes, como sensores e outras aplicações instaladas no dispositivo. O ContextF por padrão conta com 4 fontes de dados — Espacial, Temporal, Dispositivo e Social. Essas fontes foram consideradas para a presente monografia.

- Componente de Sincronização: O ContextF é responsável apenas por capturar as informações para contextualização, portanto, para inferir outras informações é necessário a sincronização com um servidor com maior poder de processamento. Este servidor é a infraestrutura implementada por este trabalho.

- Componente de Armazenamento: É responsável por garantir que as informações de contexto, de um recurso, não sejam perdidas no momento em que elas não puderam ser sincronizadas com o servidor remoto. Porém, quando o *framework* percebe que a rede está disponível ele tenta sincronizar todas as informações de contexto obtidas anteriormente pelo componente de extração.

3.1 Arquitetura de armazenamento

O ContextF se comunica com uma aplicação para capturar informações de contexto de uma mídia, que por exemplo pode estar sendo sincronizada em uma rede social. Ao capturar as informações de contexto deve ser criado um XML de acordo com o modelo proposto por este trabalho(o modelo aqui proposto estendeu o proposto por (Barbosa and Barrére, 2013), devido à necessidade de padronização dos dados e de forma a atender outros tipos de mídias além de imagens). A partir da criação do modelo o ContextF irá tentar se comunicar com o ContextWS acessando um de seus serviços — no caso desta seção, um serviço de armazenamento de informações contextuais — a partir de uma requisição REST. O ContextWS será responsável por verificar a integridade das informações passadas, como por exemplo, se os dados passados conferem com o tipo de mídia abordado pelo serviço e caso tudo esteja de acordo com o esperado o armazenamento das informações será feito. Por fim, um XML será retornado indicando o sucesso no armazenamento, caso contrário

será retornado um XML de erro. O ContexF é o responsável por levar as respostas do servidor para a aplicação. A figura 3.1 mostra a arquitetura de armazenamento de recursos:

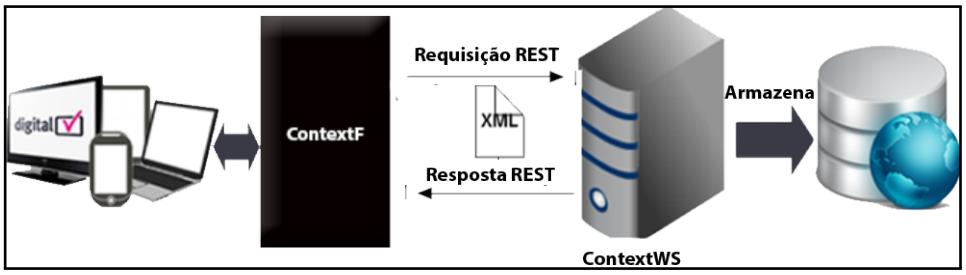

Figura 3.1: Arquitetura de armazenamento de recursos utilizando o ContextWS

No presente trabalho o ContextF é apresentado como uma caixa-preta, pois não se tem o interesse implícito em sua implementação, afinal o foco dado foi em cima da criação dos serviços da infraestrutura que está sendo apresentada.

3.2 Arquitetura de captura

Para capturar um recurso é necessário que uma aplicação informe ao ContextF o que se deseja, por exemplo: recuperar informações de mídias capturadas em uma região, recuperar informações de mídias criadas por um usuário específico e etc. O ContextF deverá então criar um modelo específico, de requisição, relacionado com as informações passadas pela aplicação. O ContextWS será responsável por processar as informações passadas pela requisição vinda do ContextF e então caso tudo esteja de acordo com o esperado, o servidor irá buscar os recursos no banco de dados. A figura 3.2 mostra a arquitetura de captura de recursos.

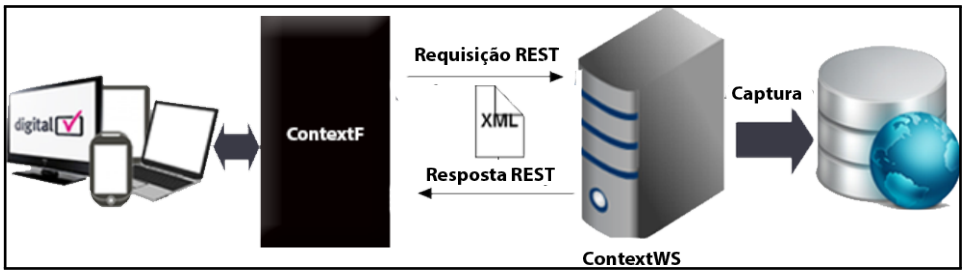

Figura 3.2: Arquitetura de captura de recursos utilizando o ContextWS

3.3 Modelagem das informações de contexto

O primeiro passo para se utilizar as informações de contexto é definir como elas serão representadas. Uma característica de sistemas sensíveis ao contexto é que os elementos contextuais podem (e devem) ser procedentes de fontes diversas e heterogêneas, muitas vezes externas ao sistema. Assim, a representação das informações contextuais provavelmente não irá considerar um modelo único, padrão e globalizado de representação (Power, 2003). Para o presente trabalho a representação das informações contextuais foram feitas utilizando a linguagem de marcação XML por ser um formato amplamente aceito, facilmente interpretado por diferentes *parsers* e integrável a diferentes linguagens de programação.

Para construção das dimensões do modelo de representação das informações contextuais uma série de perguntas foram consideradas baseadas em Truong et al. (2001):

- Quem são os usuários?
- O que é capturado e acessado?
- Quando a captura e acesso ocorrem?
- Onde a captura e o acesso ocorrem?

Com base nessas perguntas foi construído o modelo a seguir:

```
<resource>
    <user>
        <type></type>
        <token></token>
    </user>
    <context>
        <createdAt></createdAt>
        <syncedAt></syncedAt>
        <latitude></latitude>
        <longitude></longitude>
        <deviceModel></deviceModel>
        <deviceResolution></deviceResolution>
    </context>
    <media>
        <url></url>
        <type></type>
        <title></title>
        <comments></comments>
        <tags></tags>
        <video>
            <duration></duration>
            <fps></fps>
            <resolution></resolution>
        </video>
    </media>
</resource>
```

Um *Resource* é a representação de uma entidade de contexto a ser trabalhada por aplicações onde, *User* representa as informações dos usuários, *Context* informações de contexto que se dão a partir do dispositivo utilizado para captura e *Media* representa as informações de uma mídia. O XML mostrado anteriormente representa informações de uma mídia de vídeo.

Esse modelo é uma extensão do modelo proposto por Barbosa and Barrére (2013), e para trabalhos futuros deverá ser implementado no ContextF para uso prático em conjunto com o ContextWS.

3.4 Banco de Dados

As informações de contexto precisam ser armazenadas para que possam ser consumidas a partir de requisições. Dessa forma foi criado um banco de dados relacional utilizando o SGDB MySQL por ser largamente utilizado, além de ser uma ferramenta *Open Source*. O Modelo de Entidade-Relacionamento do banco de dados é apresentado na figura a seguir.

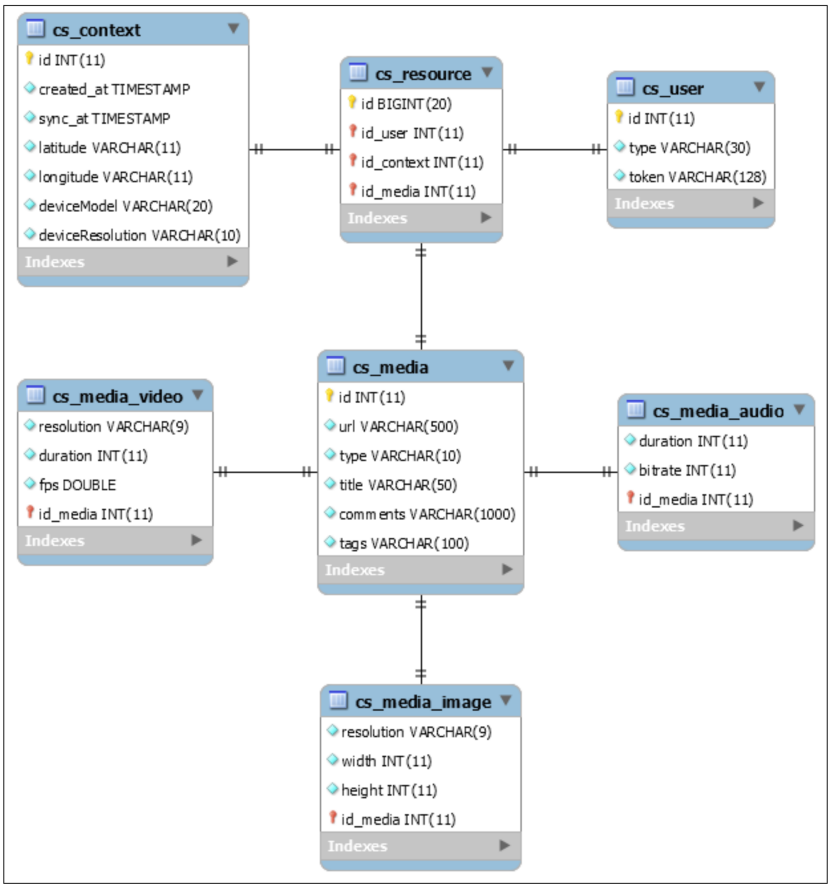

Figura 3.3: Modelo Entidade-Relacionamento do Banco de Dados

A modelagem deste banco se deu a partir do tipo de informação manipulada pelo XML citado na seção anterior. Essa situação é atípica, mas justifica-se pelo fato do XML ter sido baseado em pesquisas anteriores (Barbosa and Barrére, 2013), sendo apenas modificado para fins relativos a este trabalho.

4 Desenvolvimento e Implementação

O servidor de contexto criado pelo presente trabalho foi baseado em serviços implementados em um *Web Service*. Os Web Services são soluções amplamente utilizadas na integração de sistemas e comunicação entre aplicações. É a partir desta tecnologia que aplicações e sistemas que já existiam conseguem interagir com plataformas criadas atualmente. Para que a comunicação entre diferentes aplicações seja possível os Web Services adotam uma linguagem universal para atender requisições e fornecer serviços, a linguagem universal mais utilizada por esses serviços é o XML que também será utilizada neste trabalho.

Embora existam diferentes tipos de Web Services, optou-se neste trabalho por se utilizar o REST — um estilo de arquitetura para sistemas de hipermídia distribuídos. REST é baseado no *design* do protocolo HTTP, que possui diversos mecanismos para representar recursos. A caracterísa principal desta arquitetura são os URLs, os quais se referem a um endereço de rede onde se encontra algum recurso informático, por exemplo, um XML de informações de contexto.

O projeto foi desenvolvido na linguagem de programação JAVA, através da utilização da ferramenta NetBeans IDE 8.0.2. Foi, também, baseado no padrão de arquitetura de software MVC, que separa a representação interna da informação da maneira que ela é apresentada ou aceita pelo usuário; o uso dessa arquitetura permite que alterações feitas no *layout* não afetem a manipulação de dados, além de gerar um melhor entendimento, manutenção e reúso da implementação.

Para persistências dos dados foi utilizado o *framework Hibernate*, pois, ele facilita o mapeamento dos atributos entre a base de dados relacional e o modelo de objetos, mediante o uso de arquivos XML ou *annotations* JAVA, além de habilitar a troca fácil de SGBD, caso seja necessário. É válido ressaltar que se trata de um software livre de código aberto.

Para construção do Web Service baseado na arquitetura REST foi utilizada a API JAX-RS que usa anotações(*annotations*) para simplificar o desenvolvimento, fazendo com objetos se tornem entidades de um XML e vice-versa de maneira fácil. O JAX-RS usa

o JAX-B como serializador padrão, bastando colocar a anotação *@XmlElement* na classe desejada para informar que ela é uma dimensão do XML.

```
1  @XmlRootElement(name = "resource")
2  @XmlAccessorType(XmlAccessType.FIELD)
3  @XmlType(name = "CsRecursoType", propOrder = {"csUser","csContext",
       "csMedia"})
4  public class CsResource  implements java.io.Serializable {
5      @XmlAttribute
6      private Long id;
7      @XmlElement(required = true, name = "user")
8      private CsUser csUser;
9      @XmlElement(required = true, name = "context")
10     private CsContext csContext;
11     @XmlElement(required = true, name = "media")
12     private CsMedia csMedia;
```

4.1 Serviços

Após a apresentação dos conceitos, modelagem e ferramentas resta indicar como é feito o armazenamento e captura das informações de contexto a partir do servidor. No servidor foram implementados serviços que, a partir do envio de arquivos XML a partir de aplicações — feitos através de uma URL — respondem com um XML contendo as informações requeridas, ou com um XML de erro.

A partir do momento que uma aplicação acessa um serviço, através de uma URL, ela se comunica com o servidor via requisições HTTP que implementa os métodos citados a seguir:

- GET: Solicita a representação de um determinado recurso. É definido como um método seguro e não deve ser usado para disparar uma ação (remover um usuário, por exemplo);
- POST: As informações enviadas no corpo (body) da requisição são utilizadas para criar um novo recurso. Também é responsável por fazer processamentos que não são diretamente relacionados a um recurso.
- DELETE: Remove um recurso.

- PUT: Atualiza um recurso na URI especificada. Caso o recurso não exista, ele pode criar um.

Para o funcionamento do servidor foi utilizado o Tomcat — um servidor web Java *open source* que permite o gerenciamento de aplicações web construídas em Java. Este serviço foi hospedado em um ambiente de teste, online, utilizando o sistema operacional Ubuntu.

Ao acessar o servidor é exibida a tela a baixo:

Figura 4.1: Servidor em funcionamento.

4.1.1 Adicionar recursos

Baseado no modelo de informações de contexto apresentado em no capítulo anterior(3.3), foram criados serviços para armazenamento de informações de uma mídia, sendo que esses foram divididos de acordo com o tipo da mídia.

Para adicionar uma mídia de imagem foi definido um serviço chamado **addResourceImage** que espera como requisição um XML com informações da mídia imagem. Seguem, abaixo, as definições deste método:

```
@POST
@Path("/addResourceImage")
@Produces(MediaType.TEXT_XML)
@Consumes(MediaType.APPLICATION_XML)
```

A primeira linha indica o método HTTP utilizado(neste caso foi utilizado POST — pois, se trata de uma inserção). A segunda linha representa o endereço que deverá ser referenciado para acesso ao serviço, no caso: **http://ENDEREÇO-DO-SERVIDOR:8080/contextserver/addResourceImage**. Em seguida, tem se o tipo da informação que será retornada(XML) e por fim o tipo de informação que deve ser passada(XML) via aplicação que fez a captura das informações de contexto. Segue a baixo um XML para essa requisição:

```
<resource>
    <user>
        <type>Facebook</type>
        <token>T6t3vG39sJ5faH</token>
    </user>
    <context>
        <createdAt>2015-05-07T17:04:58-03:00</createdAt>
        <syncedAt>2015-04-21T10:04:00-03:00</syncedAt>
        <latitude>-50.853</latitude>
        <longitude>-72.271758</longitude>
        <deviceModel>Motorola Moto G</deviceModel>
        <deviceResolution>1280x638</deviceResolution>
    </context>
    <media>
        <url>https://www.facebook.com/photo.php</url>
        <type>image/jpeg</type>
        <title>Saudade da Patagonia</title>
        <comments>Foto tirada na cidade de El Calafate Argentina</
           comments>
        <tags>Patagonia, El Calafate, Argentina, Horizonte, Tierra
           del Fuego</tags>
        <image>
            <width>960</width>
            <height>638</height>
            <resolution>96dpi</resolution>
        </image>
    </media>
</resource>
```

Ao receber a requisição o servidor verifica se as informações passadas estão de acordo com o modelo proposto, caso contrário o servidor retorna um XML de erro e não persiste as informações no banco de dados.

Os serviços **addResourceVideo e addResourceAudio** que aparecem na lista de serviços da página principal do servidor têm aspectos semelhantes ao serviço que foi apresentado anteriormente, mudando apenas nas informações específicas da mídia. No

caso de um vídeo ou um áudio informações como suas durações deveriam ser informadas.

4.1.2 Capturar recursos

Varias maneiras de capturar as informações de uma mídia poderiam ter sido implementadas: a partir de ID, TAGs ou localização. Para aplicações reais é necessário a delimitação de serviços necessários. Como prova do conceito o serviço criado foi baseado nas informações de contexto de mídias capturadas por um usuário.

O **getResourceByUser** foi baseado no método POST do HTTP, pois, embora a expectativa seja capturar um recurso, é necessário primeiro enviar as informações de um usuário via XML. Segue um exemplo de arquivo XML para capturar informações contextuais a partir de um usuário:

```
1 <csUser>
2     <token>T6t3vG39sJ5faH</token>
3 </csUser>
```

5 Prova de conceito

Na seção anterior foram apresentados alguns dos serviços implementados no servidor proposto por este trabalho. De forma a provar o conceito da arquitetura apresentada foram feitos testes utilizando o **Advanced REST Client** — uma extensão do Google Chrome que permite criações customizadas de requisições HTTP (Advanced Rest Client, 2015) — e um aplicativo na plataforma Android.

5.1 Testes com Advanced REST Client

5.1.1 Capturar recursos

Ao acessar a extensão primeiramente deve-se colocar a URL do serviço, depois deve-se adicionar o método HTTP requerido. Em *Payload* coloca-se o corpo do XML de requisição e por último em *Content-Type* adiociona-se o tipo de informações que estamos produzindo, no caso **application/xml** como indicado no quarto capítulo:

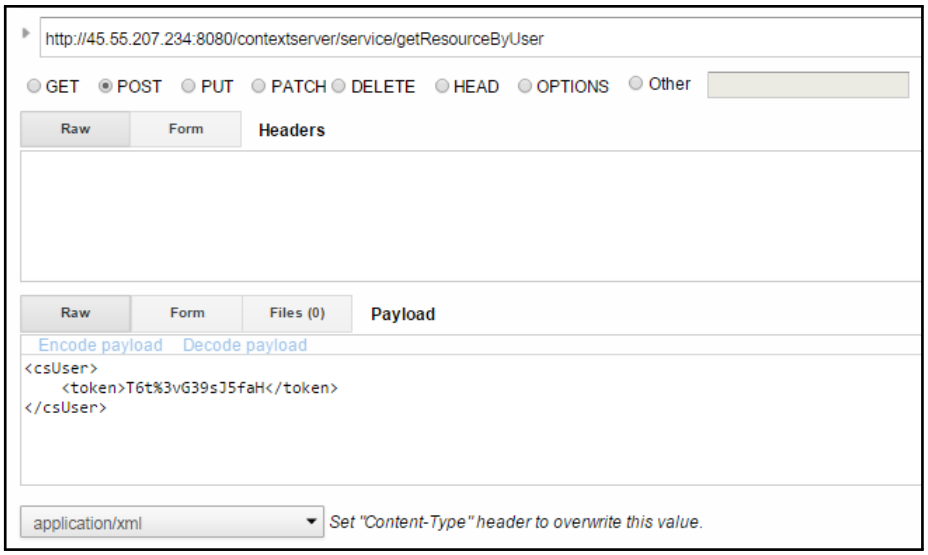

Figura 5.1: Acesso ao servidor para captura de recurso a partir de um usuário.

Após a requisição ter sido criada e solicitada ao servidor ele verifica se as informações estão de acordo com o esperado e caso positivo gera um XML indicando o sucesso na inserção, no caso, um XML com identificadores dos recursos adicionados. Pode-se ver

a resposta (no tipo escolhido pelo serviço) onde está escrito *Response* na imagem 5.2:

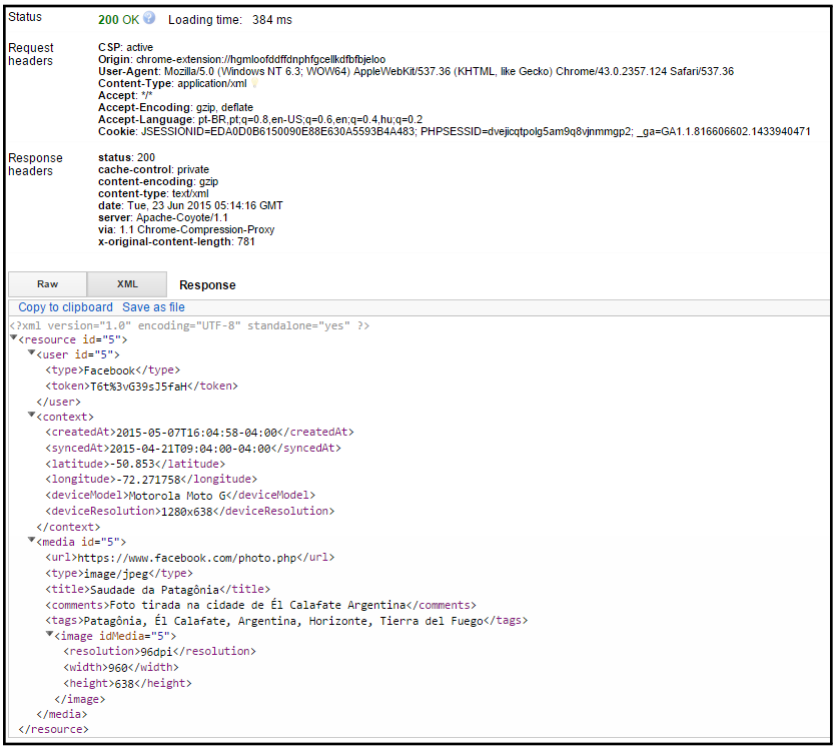

Figura 5.2: Resposta do servidor para captura de recurso a partir de um usuário.

Caso as informações passadas não tenham passado na verificação feita pelo ContextWS, ele irá gerar um XML de erro relacionado ao problema da requisição:

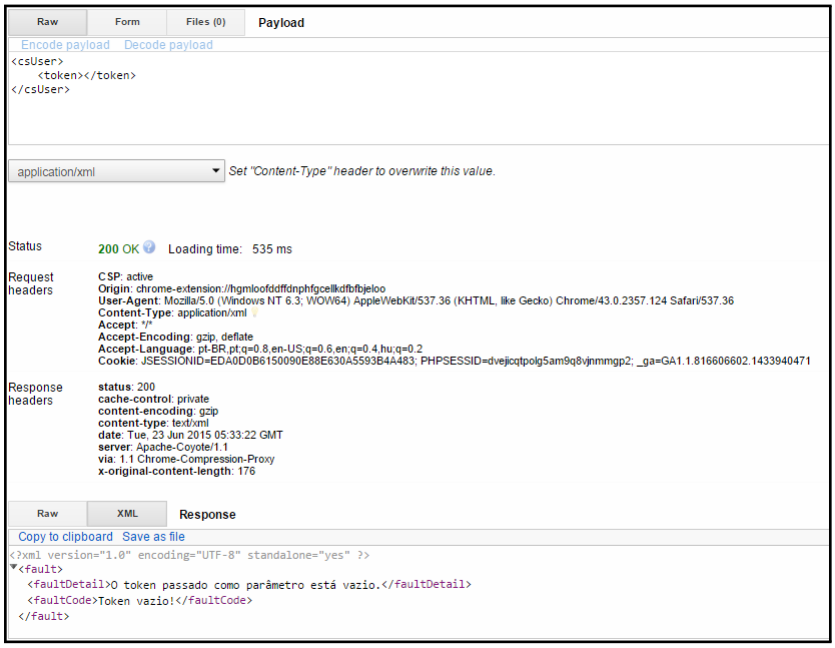

Figura 5.3: Simulação de erro ao passar token vazio.

5.1.2 Adicionar recursos

Para enviar uma informação a ser armazenada a aplicação cliente deve escolher o serviço correto e enviar o XML de dados:

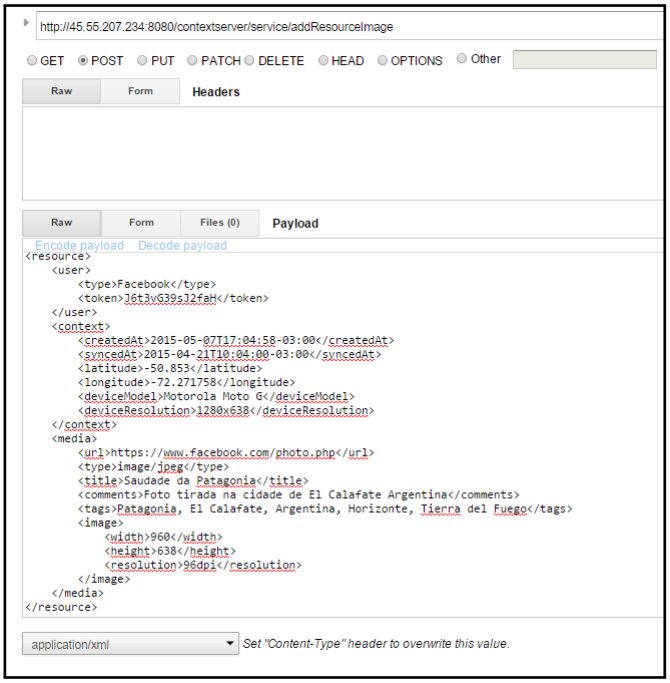

Figura 5.4: Acesso ao servidor para armazenamento de recurso.

O servidor recebe a requisição e verifica se ela é ou não válida, de acordo com os padrões do XML pré-estabelecido para aquele serviço, caso as informações sejam compatíveis com o esperado ela será atendida e a resposta poderá ser vista onde está escrito

Response na imagem 5.5 que representa o sucesso na inserção dos dados no banco de dados:

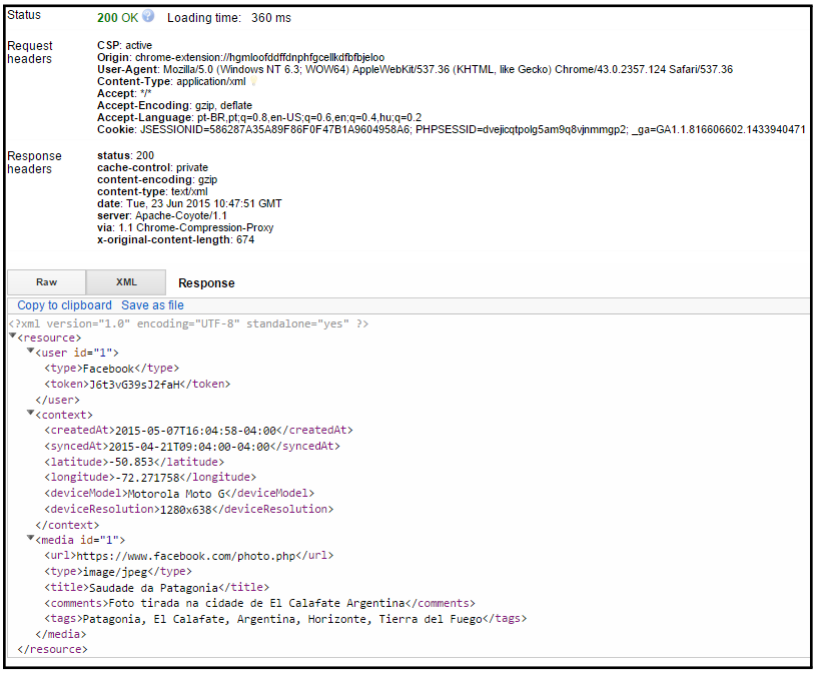

Figura 5.5: Resposta do servidor para armazenamento de recurso.

Os dados requisitados foram persistidos no banco de dados, para comprovação uma tabela que pode ser verificada é a **media**:

Figura 5.6: Dados persistidos no banco de dados.

5.2 Testes com aplicativo Android

Embora os testes utilizando o *Advanced REST Client* tenham mostrado claramente o funcionamentos dos serviços implementados foi criada também uma aplicação Android para simular o uso do servidor em uma plataforma real. Para desenvolvimento da aplicação foi utilizada a IDE *Android Studio 1.1.0*.

Foi criada uma interface simples baseada em alguns botões que ao serem acionados disparam um serviço do *ContextWS*:

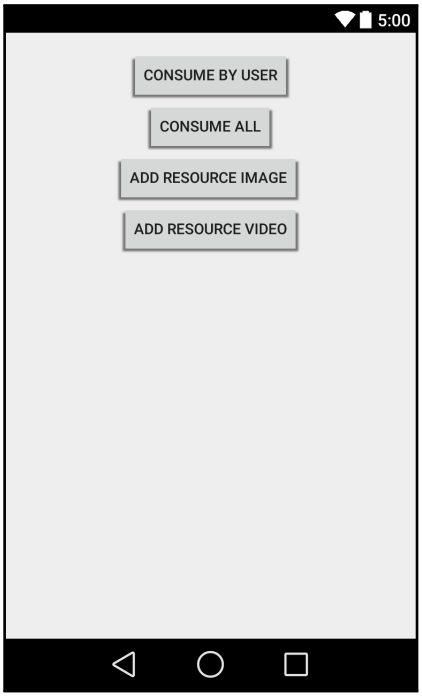

Figura 5.7: Interface da aplicação Android de teste.

Ao pressionar o botão **Consume By User**, por exemplo, o serviço **getResourceByUser** será acessado passando um XML com um *token*:

```
1  public void consumeResource(View view)
2  {
3      String xml ="<csUser><type></type><token>tokentest</token></
           csUser>";
4      String url = "http://45.55.207.234:8080/contextserver/service/
           getResourceByUser";
5      WebServiceClient webServiceClient = new WebServiceClient();
6      Intent intent = new Intent(this,ResultActivity.class);
7      intent.putExtra(responseXML, webServiceClient.post(url,xml));
8      startActivity(intent);
9
10 }
```

A resposta a essa requisição pode ser vista na imagem 5.8:

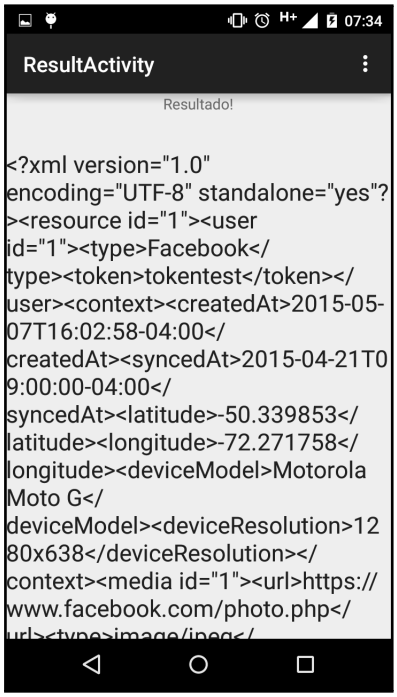

Figura 5.8: Resposta da requisição feita ai servidor através de uma aplicação Android.

Os outros botões têm acessos similares, ou seja, acessam um serviço e recebem uma resposta. Os tipos de respostas podem ser vistos nos testes utilizando a extensão Advanced REST Client do Google Chrome apresentada anteriormente.

6 Conclusão

Sistemas multimídias são utilizados em várias áreas de forma a melhorar a qualidade das informações transmitidas. Esses sistemas são baseados em mídias de representação tais como vídeos, imagens, textos e áudios. Com o crescente uso de dispositivos com capacidade de capturar esses elementos, diversas aplicações e sistemas estão sendo desenvolvidos na tentativa de fornecer serviços úteis à sociedade, melhorar a comunicação humano-computador e aspectos como usabilidade e eficiência. Essas mídias capturadas guardam informações que as caracterizam e, também, informações referentes — por exemplo, ao momento e local onde foram capturadas. Os artefatos referente a essa entidade são chamados de informações de contexto e são matérias-primas utilizadas para construção de ambientes sensíveis ao contexto. A criação de sistemas sensíveis ao contexto não é uma tarefa trivial, haja vista que essas informações vêm de bases heterogêneas de dados. Este trabalho apresentou uma arquitetura de um servidor para armazenamento e captura de informações de contexto, baseado em mídias. Esse servidor foi baseado em um Web Service REST, no qual os serviços são acessados via URLs através de métodos do protocolo HTTP. As informações de contexto foram modeladas utilizando a linguagem de marcação XML e o banco de dados baseado nessa arquitetura utilizou o SGBD MySQL. Toda implementação foi feita em JAVA utilizando a arquitetura de software MVC, foi utilizado o *Hibernate* para persistência de dados e a API JAX-RS para mapeamento de objetos em entidades XML.

A partir dos testes realizados, a infraestrutura se mostrou funcional, para a utilização na criação de aplicações sensíveis ao contexto, baseadas em mídias. Para utilização em aplicações reais, a infraestrutura deverá ser estendida para atender as demandas de uma aplicação — por exemplo: adicionar outras informações aos recursos, criar serviços que realmente sejam relevantes, ou seja, que processem as informações de contexto e façam alguma inferência, além de tratar de aspectos de segurança.

Como trabalhos futuros esta monografia propõe a adaptação do ContextF ao modelo aqui proposto, de forma a utilizá-lo em conjunto com o ContextWS. Além disso,

para que o servidor se torne ainda mais genérico, se faz necessário criar uma camada de transformação que ao receber requisições de armazenamento — de diversas bases de dados(sobre mídias) — deverá captar as informações relevantes e pertinentes ao modelo proposto por este trabalho, que por sinal pode e deve ser estendido.

Referências Bibliográficas

Advanced rest client, 2015. Access date: 29 junho. 2015.

Araújo, R. **Computação ubíqua: Princípios, tecnologias e desafios**. 2003.

Augustin, L.; Yamin, A.; Junior, E. S.; Barbosa, J.; Geyer, C. ; Cavalheiro, G. **Isam: um middleware para aplicações móveis distribuídas.** volume 8, p. 41 –58, 2001.

Barbosa, M. H.; Barrére, E. **Contextf: Framework de apoio a construção de aplicações móveis sensíveis ao contexto**. p. 1–3, 2013.

Dey, A. K.; Abowd, G. D. **Towards a Better Understanding of Context and Context - Awareness**. p. 304–307. Springer Berlin Heidelberg, 1999.

Fehlberg, F. W. **Multis: um servidor de contexto voltado à computação pervasiva**. 2007.

Fleischmann, A. M. P. **Sensibilidade à situação em sistemas educacionais na web**. 2012.

Gui, F. **Development of a new cliente - server architecture for context - aware mobile computing**. 2009.

Herrtwich, R. G.; Stwinmetz, R. **Towards integrated multimedia systems: Why and how**. volume 293, p. 328, 1990.

Hoare, T.; Milner, R. Grand challenges for computing research. **Grand Challenges in Computing Research Cnference 2008**, v.48, n.1, p. 6–7, 2005.

Houaiss, A.; Villar, M. **Minidicionário Houaiss da língua portuguesa.** 2012.

Kahl, M.; Floriano, D. **Computação ubíqua, tecnologia sem limites**. 2011.

Kappel, G.; Proll, B.; Retschitzegger, W. ; Schwinger, W. Towards a generic customisation model for ubiquitous web applications. **International workshop on Web Oriented Software Technology, Iwwost, Málaga, Spain**, 2002.

Lopes, C. J. F.; Ramalho, J. C. **Web services: Metodologias de desenvolvimento**. 2004.

Menéndez, A. I. M. **ma ferramenta de apoio ao desenvolvimento de web services.** p. 97, 2002.

Moore, P. **Intelligent context for personalized m-learning.** 2009.

Power, R. **Topic maps for context management**. In: Proceedings of the 1st International Symposium on Information and Communication Technologies, ISICT '03, p. 199–204. Trinity College Dublin, 2003.

Salber, D.; Abowd, G. **The design and use of a generic context server**. 1998.

Salber, D.; Abowd, G. **The context toolkit: aiding the development of context-enabled applications**. 1999.

Schilit, B.; Adams, N. ; Want, R. **ontext-aware computing applications,**. p. 85–90, 1994.

Truong, K.; Abowd, G. ; Brotherton, J. **Who, what, when, where, how: Design issues of capture and access applications**. In: Abowd, G.; Brumitt, B. ; Shafer, S., editors, Ubicomp 2001: Ubiquitous Computing, volume 2201 de **Lecture Notes in Computer Science**, p. 209–224. Springer Berlin Heidelberg, 2001.

Venecian, L.; Joao Lopes and, A. Y. ; Palazzo, L. **Uma proposta baseada em web semântica para sensibilidade ao contexto na computação ubíqua**. 2009.

Vieira, V.; Tedesco, P. ; Salgado, A. C. **Modelos e processos para o desenvolvimento de sistemas sensíveis ao contexto**. 2009.

Web services architecture - w3c working group note 11 february 2004. p. 7–8, 2004.

Zimmermann, A.; Lorenz, A. ; Oppermann, R. **An operational definition of context, modeling and using context**. p. 558–578, 2007.