WebMedia2012

XVIII SIMPÓSIO BRASILEIRO DE SISTEMAS MULTIMÍDIA E WEB

Foto: Museu do Ipiranga - São Paulo-SP por YuriAlexandre

ANAIS

XVIII Simpósio Brasileiro de Sistemas Multimídia e Web

EDIÇÃO//
Sociedade Brasileira de Computação
ORGANIZAÇÃO//
Graça Bressan (USP) e Regina Melo Silveira (USP)

15 a 18 de outubro de 2012
Frei Caneca Shopping & Convention Center
São Paulo / SP
http://sws2012.ime.usp.br/webmedia

SIMPÓSIOS'12
SBBD / WebMedia / SBSC

Webmedia 2012
XVIII Simpósio Brasileiro de Sistemas Multimídia e Web

ANAIS DOS WORKSHOPS E MINI CURSOS DO WEBMEDIA 2012

XII Workshop de Teses e Dissertações (WTD)
XI Workshop de Ferramentas e Aplicações (WFA)
IX Workshop de Trabalhos de Iniciação Científica (WTIC)
Mini Cursos do Webmedia 2012

Coordenação Geral

Graça Bressan
Regina Melo Silveira

São Paulo, 15-18 Outubro de 2012

Anais dos Workshops do Webmedia 2012

Prefácio

O Simpósio Brasileiro de Sistemas Multimídia (WebMedia) é um evento anual, promovido pela Sociedade Brasileira de Computação (SBC), em cooperação com a ACM/SIGWEB e SIGMM, estando em sua décima oitava edição. Desde a sua primeira edição em 1995, o WebMedia transformou-se no mais importante fórum de debates para pesquisadores e profissionais das áreas de multimídia e hipermídia no Brasil. O WebMedia é um fórum de discussão onde pesquisadores, acadêmicos, profissionais e sociedade em geral podem trocar conhecimentos e experiências no uso de tecnologias capazes de projetar, criar, armazenar e trocar dados multimídia e hipermídia pela Web. Este ambiente rico de discussões engloba diversas áreas tais como Sistemas de Informação Multimídia, TV Digital, Ensino a Distância, Rede de Telecomunicações e Teleconferências.

A décima oitava edição do evento (WebMedia 2012) ocorre no período de 15 a 18 de outubro de 2012, na cidade de São Paulo, SP. Conforme incentivo da SBC, o WebMedia é normalmente organizado em conjunto com outros eventos. Em 2012, o XVIII Simpósio Brasileiro de Sistema Multimídia e Web (WebMedia 2012) acontece em conjunto com o XXVII Simpósio Brasileiro de Banco de Dados (SBBD 2012) e IX Simpósio Brasileiro de Sistemas Colaborativos (SBSC 2012). Dada a interseção dos diversos tópicos de interesse das edições destes três simpósios, a organização conjunta oferece uma ótima oportunidade para estimular ainda mais o intercâmbio de idéias e trabalhos entre pesquisadores, profissionais e estudantes das áreas de pesquisa abrangidas pelos eventos.

A organização do WebMedia 2012 é de responsabilidade da Universidade de São Paulo (USP) através da Escola Politécnica (EPUSP). O simpósio ocorre com patrocínio do Comitê Gestor da Internet no Brasil (CGI.br), Núcleo de Informação e Coordenação do Ponto BR (NIC.br), Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), CAPES, CNPq e Programa PRCEU da Pró-Reitoria de Cultura e Extensão da USP.

Os anais dos artigos completos e artigos cursos do WebMedia 2012 estão disponíveis na Digital Library da ACM. O presente volume dos Anais do WebMedia 2012 é composto dos artigos apresentados nos workshops organizados no escopo do WebMedia, a saber: IX Workshop de Trabalhos de Iniciação Científica (WTIC), XI Workshop de Ferramentas e Aplicações (WFA), e XII Workshop de Teses e Dissertações (WTD), bem como dos Minicursos. Deve ser salientado o enorme esforço do Comitê de Programa do WebMedia 2012 e dos Comitês de Programa dos vários workshops para realizar a seleção dos trabalhos a serem apresentados e publicados dentre tantas submissões de excelente qualidade. Agradecemos a todos os autores que submeteram seus artigos e minicursos e congratulamos aqueles que tiveram seus trabalhos aceitos. Por último, gostaríamos de agradecer aos membros dos comitês de organização e de programa, bem como aos revisores, pelo trabalho voluntário de fundamental importância para a realização do WebMedia 2012. Desejamos que o evento resulte em frutos para o trabalho de todos.

São Paulo, Outubro, 2012
Graça Bressan
Regina Melo Silveira
Coordenação Geral - Webmedia 2012

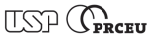
Maria da Graça Pimentel
Ethan V. Munson
André Santanchè
Coordenação do Comitê de Programa - Webmedia 2012

Patrocínio

PROMOÇÃO

ORGANIZAÇÃO

PATROCÍNIO



WebMedia em cooperação com

Organização

Organização Geral

Graça Bressan	Universidade de São Paulo
Regina Melo Silveira	Universidade de São Paulo

Coordenação do Comitê de Programa

Maria da Graça Pimentel	Universidade de São Paulo

Vice-coordenação de Trilhas

André Santanchè	Unicamp	Web e Redes Sociais
Ethan V. Munson	University of Wisconsin	Multimidia
Maria da Graça Pimentel	USP	TV Digital, Computação Ubíqua e Móvel

Coordenação do Comitê Especial

Celso Alberto Saibel Santos	UFES
Fernando Antônio Mota Trinta	UFC

Comitê Especial

Eduardo Barrére	UFJF
Fábio Gomes	IFPI
Guido Lemos de Souza Filho	UFPB
José Valdeni de Lima	UFRGS
Jussara Almeida	UFMG
Luiz Fernando Gomes Soares	PUC-Rio
Maria da Graça Pimentel	USP
Mário Meireles Teixeira	UFMA
Roberto Willrich	UFSC
Valter Roesler	UFRGS

Sumário

I XII Workshop de Teses e Dissertações (WTD)

II XI Workshop de Ferramentas e Aplicações (WFA)

III IX Workshop de Trabalhos de Iniciação Científica (WTIC)

Parte I

XII Workshop de Teses e Dissertações (WTD)

Prefácio

O Workshop de Teses e Dissertações (WTD) tem sido promovido tradicionalmente como um dos eventos integrantes do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia), sendo um fórum dedicado à apresentaç ão e à discussão de trabalhos de mestrado e doutorado em andamento nas áreas de Multimídia, Hipermídia e Web. O principal objetivo do workshop é prover um fórum onde alunos de mestrado e doutorado possam discutir sua pesquisa com membros experientes da comunidade na área. Outro objetivo é promover integração e cooperação entre pesquisadores da academia e da indústria.

Neste ano, o evento recebeu um total de 19 submissões de trabalhos de mestrado e doutorado em andamento, os quais foram avaliados de forma criteriosa pelos revisores. Após o processo de avaliação e submissão de revisões, foram selecionados 8 artigos, resultando em uma taxa de aceitação de 42,1%. Os artigos selecionados abordam temas relacionados a sistemas de recomendação, engenharia de documentos, TV Digital, vídeo 3D, computação ubíqua e engenharia Web.

Agradecemos aos membros do comitê de programa pela preciosa colaboração nas revisões dos artigos e aos autores que submeteram seus trabalhos. Agradecemos à Professora Graça Bressan, da Escola Politécnica da USP, pelo convite para coordenar o WTD e por todo intenso trabalho que vem desenvolvendo na organização do evento.

São Paulo, Outubro, 2012

José Maria Parente de Oliveira - ITA
Rudinei Goularte - ICMC-USP

Organização

Coordenação do WTD

José Maria Parente de Oliveira	ITA
Rudinei Goularte	ICMC-USP

Comitê de Programa do WTD

André Santanchè	Unicamp
Carlos Ferraz	UFPE
Cássio Prazeres	UFBA
Celso Alberto Saibel Santos	UFES
Clóvis Fernandes	ITA
Débora Paiva	UFMS
Graça Bressan	USP
Leonardo Andrade	UFSCar
Luciana Martimiano	UEM
Luciano Pansanato	CEFET-PR
Marcelo Manzato	USP
Mario M. Teixeira	UFMA
Renata Fortes	USP
Renato Bulcão Neto	UFG

Apoiando a construção de gráficos por usuários inexperientes através de recomendações

Taissa Abdalla Filgueiras de Sousa
Departamento de Informática
Pontifícia Universidade Católica, PUC-Rio
22451-900 Rio de Janeiro, Brasil
+55 (21) 35271510
tsousa@inf.puc-rio.br

Simone Diniz Junqueira Barbosa
Departamento de Informática
Pontifícia Universidade Católica, PUC-Rio
22451-900 Rio de Janeiro, Brasil
+55 (21) 35271500 Ramal: 4353
simone@inf.puc-rio.br

ABSTRACT
Visualization research emphasizes the need for systems that assist in decision-making and visual analysis. This paper outlines a recommender system that supports the interactive construction of charts using statistical data. Based on a sequence of simple user interactions, the interface automatically generates an elementary view. This process yields knowledge in an implicit manner, which enables the system to analyze the selected dataset, suggest visualization options and provide guidelines for selecting the most appropriate representation for a given problem.

Categories and Subject Descriptors
H.5.0 [**Information Systems**]: Information Interfaces And Presentation - General

General Terms
Design

Keywords
information visualization, recommender systems, human-computer-interaction

1. INTRODUÇÃO

Essa pesquisa visa contribuir para a área de interação-humano-computador (IHC) através de um sistema de visualização de informação que agrega recomendações de tarefas para apoiar a construção de gráficos para usuários inexperientes.

Gráficos são um meio de comunicação eficiente que fazem uso da percepção. No entanto, é possível confundir a percepção do leitor através da distorção das representações. Nesse sentido, Tufte [12] descreve as técnicas mais comuns. Assim, para uma interpretação correta, pesquisas demonstram que a interpretação de gráficos requer conhecimentos específicos que não são facilmente aprendidos [11][13][14]. Considerando ainda as elevadas taxas de analfabetismo funcional, problemas no ensino médio e o consumo de informações sem questionamento podemos verificar que existem problemas na área de visualização de informações, especialmente por usuários inexperientes. Motivados por esse problema, nossa proposta é criar uma ferramenta que auxilie na construção das visualizações com a utilização de dados estatísticos.

Este artigo está dividido da seguinte forma: A seção 2 cita ferramentas e *toolkits* de visualização disponíveis e descreve suas limitações. Além disso faz uma análise de técnicas utilizadas em projetos de interfaces de visualização de dados. A seção 3 descreve o problema que motivou a pesquisa. A seção 4 apresenta o objetivo da proposta. A seção 5 listas as contribuições esperadas para a área. A seção 6 descreve as metodologias. A seção 7 faz uma breve descrição de pesquisas já realizadas a fim de atingir o objetivo proposto e em que estágio o trabalho se encontra. A seção 8 compara a proposta com projetos recentes. Por último, a seção 9 conclui o artigo.

2. CONTEÚDO TEÓRICO

Diversos sistemas têm sido desenvolvidos com o objetivo de melhorar a experiência do usuário na visualização de dados e motivar o interesse por dados estatísticos. Entre eles, podemos citar: Manyeyes,[1] GapMinder,[2] Worldmapper,[3] Statplanet,[4] Google Public Data,[5] diversos atlas multimídia, SIDRA,[6] Séries estatísticas,[7] etc. Elias e Bezeriano [1] citam algumas *toolkits* (Flare,[8] Silverligth,[9] JavaScript InfoVis tk,[10] ivtk[11]) que permitem tanto a criação de gráficos quanto de combinações deles. Afirmam que elas não visam usuários novatos e normalmente necessitam de programação adicional para realizar as operações para a criação de visualizações. Neste caso, a curva de aprendizado é normalmente alta e visa usuários mais experientes. Elias e Bezeriano [1] afirmam ainda que as ferramentas para usuários novatos tais como ManyEyes, Sense.us e Polstar, normalmente restringem usuários a uma única visualização por vez.

Heer e Shneiderman [4] apresentam três alternativas utilizadas em projetos de interfaces de visualização de dados. A primeira é através de *chart typology*, uma paleta de templates de visualizações disponíveis na qual analistas podem apresentar seus dados. Segundo ele, esse método é familiar aos usuários de planilhas e possui benefícios como a simplicidade e familiaridade, mas limita os tipos de visualização e torna custoso o experimento de diferentes visualizações com o mesmo dado. A segunda alternativa apresentada é através do uso de *data-flow graphs*, no

[1] http://www-958.ibm.com/software/data/cognos/manyeyes/
[2] http://www.gapminder.org/
[3] http://www.worldmapper.org/
[4] http://www.sacmeq.org/statplanet/StatPlanet.html
[5] http://www.google.com/publicdata/directory
[6] http://www.sidra.ibge.gov.br/
[7] http://seriesestatisticas.ibge.gov.br/
[8] http://flare.prefuse.org/
[9] http://www.silverligth.net/
[10] http://thjit.org/
[11] http://ivtk.sourceforge.net

qual o processo de visualização é desconstruído em conjuntos refinados de operadores para importação, transformação, layout, etc. Através das combinações flexíveis, eles permitem maior variação de designs, porém requerem um maior esforço de entrada que o método anterior e ainda são limitados pelo conjunto de operadores. A terceira alternativa é baseada em *formal grammars* para a visualização de construções, constituídas por linguagens de alto nível que descrevem como o dado deve ser mapeado nas características visuais. Exemplos: *ggplot for the R statistical analysis platform*,[12] Protovis for HTML5 [5] e *Google Chart Tools*.[13] No entanto, todas elas requerem um mínimo de habilidade em programação. Ainda segundo Heer e Shneiderman [4], essas três alternativas não são mutuamente exclusivas. Pode-se aplicar um sistema de *data-flow* ou *formal grammars* para definir novos componentes para incluir dentro de uma *chart typology*, aproveitando melhor a expressividade da primeira e da facilidade de utilização da última.

Pesquisas sobre visualização ainda demonstraram a necessidade de criar significados formais sobre as formas de visualizações de dados [6][7][8]. Tory e Torsten [6] apresentam uma taxonomia que organiza as técnicas de visualização com base em modelos de dados e categoriza esses modelos de acordo com a natureza do domínio do dado (contínuo ou discreto) e em suas restrições, além de destacar o papel do usuário nos modelos conceituais. Amar et al. [23] apresentam um conjunto de dez tipos de tarefas de análise primitiva que representam os tipos de questões específicas que uma pessoa pode perguntar ao trabalhar com um conjunto de dados. Duke et al. [7][8] destaca a necessidade de uma ontologia para visualização que suporta trabalhos colaborativos e educação.

3. IDENTIFICAÇÃO DO PROBLEMA

Verificamos que existem problemas na interpretação e construção de gráficos [11][13][14][3] e necessidade de ferramentas que apóiem essas tarefas por usuários novatos [1][9]. Mackinlay et al. [9] confirmam a necessidade quando afirmam que todos os analistas têm conhecimento sobre o domínio de seus problemas, mas poucos têm habilidade para construir apresentações gráficas eficientes da informação. Acrescenta ainda que as pessoas precisam de sistemas de análises visuais que automaticamente apresentam dados usando as melhores praticas do design gráfico.

4. OBJETIVO

Pelo motivo apresentado, buscamos uma solução que auxilie usuários inexperientes na construção de visualizações. Nossa pesquisa visa o desenvolvimento de um sistema de visualização que auxilie usuários na construção de gráficos eficientes com dados estatísticos, evitando construções que levam a interpretações equivocadas. A eficiência da representação é definida por Bertin [16] pela seguinte proposição: "se, para tentar obter uma resposta correta e completa para uma questão dada, sendo todas outras coisas iguais, uma construção requer um tempo de observação mais curto que outra construção, nós podemos dizer que é mais eficiente para a questão" [16, p.139]. Bertin completa ainda que "as construções mais eficientes são aquelas nas quais quaisquer questões, não importando tipo ou nível, podem ser respondidas num simples instante da percepção, ou seja, em uma só imagem" [16, p. 146].

5. CONTRIBUIÇÕES ESPERADAS

Como descrito na seção 2, existem diversas ferramentas de visualização. No entanto, as ferramentas que possibilitam maior interação são voltadas para usuários mais experientes e muitas vezes até necessitam de conhecimento de programação por parte deles. As ferramentas voltadas para usuários inexperientes possuem muitas restrições, como limitação de interação com dados e limitação nas construções de gráficos.

Portanto, nossa contribuição direta será: i) na resolução de problemas de visualização e construção de gráficos por usuários inexperientes com uma ferramenta que permite diversos tipos de construções e operações avançadas através das recomendações; e indiretamente: ii) na avaliação da ontologia de visualização que define relações entre dados, modelos de visualização e perguntas dos usuários; iii) no sentido de estimular usuários inexperientes a uma análise das representações gráficas.

6. METODOLOGIA ADOTADA

O sistema de visualização proposto integra o conceito de *content-based recommender system*. O processo básico executado por esse sistema consiste na combinação de atributos do perfil do usuário com atributos do conteúdo do objeto para recomendar ao usuário novos itens de interesse [25]. Assim, a partir da seleção de um dado pelo usuário, o sistema apresentará perguntas que podem ser respondidas com a visualização apresentada e recomendará outras visualizações eficientes. As visualizações possíveis e as perguntas têm relação direta com o dado selecionado. O conhecimento dessa relação foi definido através de uma ontologia de visualização [2], explicada mais adiante.

As perguntas estão sendo elaboradas com base nas dez tarefas de [23]. Elas visam criar um diálogo com o usuário em linguagem natural para que ele identifique com mais facilidade as visualizações relacionadas ao dado desejado. A cada interação do usuário, as perguntas exibidas podem ser alteradas ou receber um destaque diferente na interface devido à sua relevância para o dado selecionado Dessa forma, o sistema pemitirá que o usuário encontre com mais facilidade visualizações eficientes para responder sua pergunta.

Como mecanismo de *relevance feedback*, o usuário retorna um *feedback* implícito para o sistema a cada interação e demonstra a finalização da realização de uma tarefa ao exportar uma visualização. Nesta hora pretendemos incluir ainda um tipo de *feedback* explícito para verificar se a opção selecionada tem relação com as perguntas mais relevantes sugeridas.

Para verificar se objetivo foi alcançado, realizaremos testes de IHC junto a usuários utilizando os métodos utilizados em pesquisa anterior [3]: Método de Avaliação de Comunicabilidade (MAC) [18, p.344] e *think aloud* [19] com co-participação [20]. Os testes de IHC realizados avaliaram as ferramentas Statplanet, SIDRA e Séries estatísticas e estão descritos na seção 7. A avaliação da ferramenta será através da utilização de um cenário que possibilite o uso de ViSC 2 (Visualization with Smart Charts 2) e das demais ferramentas avaliadas anteriormente [3] para a obter uma comparação entre as ferramentas e o novo sistema. O usuário, conforme avaliações anteriores, poderá construir um gráfico em mais de um sistema e optar pelo uso de um deles. Isso,

[12] http://had.co.nz/ggplot2/
[13] ttps://developers.google.com/chart/interactive/docs/index

porque além de avaliar a comunicabilidade de ViSC 2, identificar melhorias e avaliar se evoluímos no sentido possibilitar a construção de visualizações eficientes, temos o objetivo também de fazer uma análise comparativa entre as ferramentas.

7. ESTÁGIO ATUAL DO TRABALHO

Esta proposta foi fundamentada em duas pesquisas anteriores e no desenvolvimento de uma primeira versão da ferramenta ViSC como proposta preliminar.

7.1 Avaliação das ferramentas de visualização

Para avaliar como ferramentas de visualização interativas podem influenciar a interpretação dos dados [3], utilizamos dois métodos de avaliação com usuário e escolhemos as ferramentas Statplanet, SIDRA e Séries estatísticas. O MAC foi fundamental para encontrar signos que atrapalhavam na construção dos gráficos verificados através das rupturas de comunicação entre a metamensagem do designer e a recebida pelo usuário. O método *Think Aloud* com co-participação, possibilitou entender sobre os processos abdutivos na escolha das visualizações pelos usuários, seus modelos mentais e problemas de compreensão na leitura de gráficos.

Para Santaella [22], o conceito de abdução é o raciocínio que leva à adoção de uma hipótese para ser comprovada e indução é o teste da hipótese. Segundo a mesma autora, em 1901, Peirce definiu abdução como "aceitação ou criação de uma premissa menor como uma solução hipotética para um silogismo cuja premissa maior é conhecida e cuja conclusão descobrimos ser um fato" [22, p. 92]. Este conceito foi logo depois ampliado para a visão de que ela "consiste no exame de uma massa de fatos permitindo que estes fatos sugiram uma teoria" [22, p. 92].

Observamos que temos maior complexidade dos dados e flexibilidade de construções, maior foi a dificuldade encontrada pelos usuários. As ferramentas mais simples, no entanto, permitiram pouca interação e limitaram as construções. Além disso, as ferramentas oferecem pouco ou nenhum apoio ao usuário inexperiente na construção de visualizações eficientes para responder a sua pergunta e ainda apresentam problemas em construções geradas que podem levar a interpretações equivocadas.Observamos ainda que a linguagem e a gramática das ferramentas de visualização tem grande influência sobre as visualizações geradas pelos usuários. Encontramos diversos problemas de IHC em ferramentas de visualização que podem atrapalhar a interpretação dos dados e nas construções gráficas pelo usuário inexperiente.

7.2 Criação da Ontologia de Visualização

Esta pesquisa visava melhorar a comunicação entre usuários e sistemas de visualização. Vimos que a formulação da pergunta era um dos critérios para uma visualização eficiente [17]. Sobre o número de questões em cada visualização, Bertin [16] afirma que existem tantas questões quanto o número de dimensões. Cada dimensão produz um tipo de questão. Assim, cada visualização responde a determinadas perguntas. A pergunta que o usuário quer responder tem relação com a tarefa e com a eficiência da representação tem relação.

Visko [24] e a ontologia de visualização da UK Nacional e-Science Center [7][8] serviram de base para a criação da nova ontologia. Nosso esforço foi no sentido de criar uma ontologia que definisse as relações entre dados e modelos de visualização a fim de responder perguntas de usuários através de visualizações eficientes. Definimos então 5 classes de nível superior: dado, atributo de exibição, visualização, transformação e tarefa (definida por perguntas).

7.3 ViSC

Com a primeira versão da ferramenta ViSC buscamos resolver problemas apresentados na primeira pesquisa e avaliar a ontologia desenvolvida na segunda. ViSC já agregou um sistema de recomendação. As recomendações no ViSC são apresentadas através de perguntas pertinentes às visualizações e ao dado selecionado de acordo com a ontologia [2]. As perguntas são exibidas segundo a relevância para a visualização apresentada. Elas foram cadastradas e classificadas com pesos de acordo com uma condição de seleção. Enquanto as recomendações das visualizações dependem das características semânticas do dado, as perguntas verificam, além das visualizações disponíveis, o número de elementos selecionados em cada atributo de exibição e as visualizações selecionadas pelo usuário, atribuindo mais peso para perguntas que respondem a estas visualizações. Na interface, o grau de relevância das perguntas é comunicado pela localização na tela e pela gradação da cor. Quanto mais no topo da página e mais escuro é o verde do fundo da pergunta, mais relevante.

A Figura 1 apresenta a tela de construção de visualizações de ViSC. A área 1 corresponde ao menu de cada uma das dimensões do dado. A área 2 apresenta o menu de visualizações da ferramenta. Os itens desse menu são habilitados ou desabilitados de acordo com o dado selecionado. A área 3 exibe os botões de ordenação, a área 4 é onde se localizam os gráficos e a área 5 onde as perguntas estão localizadas.

Essa ferramenta foi avaliada através do Método de Inspeção Semiótica (MIS) [18, p.330] e identificamos necessidades a serem corrigidas antes de fazer avaliações com usuários.

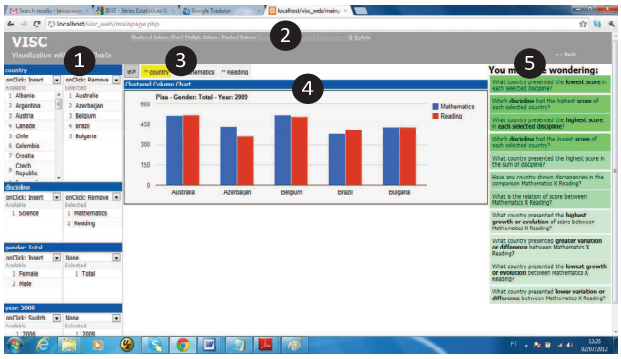


Figura 1: Tela de construção de visualizações da ferramenta ViSC

Para a proposta final ViSC 2, estamos ainda mapeando construções improdutivas a fim de incluir recomendações que indiquem problemas na construção gráfica com objetivo de evitar construções ineficientes. Necessitamos também de implementações complementares para permitir que o usuário tenha o nível de interação esperado tais como zoom, pan, alteração de configurações visuais e inclusão de botões de exportação dos resultados. Além disso, desejamos melhorar como as perguntas estão sendo exibidas na interface, de forma que se torne mais claro para o usuário a relação com as visualizações.

8. COMPARAÇÃO COM TRABALHOS RELACIONADOS

Na busca por automatização de apresentação de informação e sistemas de recomendação em sistemas de visualizações, encontramos *Show me* [9] e *Explorations Views* (EV) [1]. *Show me* é um conjunto integrado de comandos de interface que incorpora apresentações automáticas no Tableau, um sistema comercial projetado para ser usado tanto por usuários novatos quanto experientes e para construir visualizações eficientes baseado na semiologia de gráficos de Bertin [16]. *Show me* é um diálogo de escolhas com *tooltips* que descrevem condições para uma escolha estar ativa. EV também sugere gráficos e *templates* para a criação de *dashboards* por usuários novatos. Essa proposta partiu de pesquisas sobre as necessidade de usuários novatos em análises visuais. Assim, determinaram que a sequência de passos para a criação de *dashboards* deveria ser simples, com um contínuo *feedback* visual e além disso prover sugestões de gráficos e *templates* baseado nas tarefas do usuário.

O sistema proposto também parte da definição de valores *defaults* para algumas dimensões e gera automaticamente uma visualização elementar. A inclusão de um sistema de recomendação visa sugerir não apenas outras visualizações mas também estabelecer uma conversa com o usuário e compreender a tarefa a ser executada através da pergunta que o usuário deseja responder. Com isso, nossa proposta pretende estimular o interesse do usuário inexperiente pela busca de outras formas de visualização com custo baixo de experimentação. Assim esse sistema de recomendação pretende guiar o usuários para visualizações possíveis baseado não apenas nas características do dado como também na tarefa do usuário. Utiliza os métodos como *feedback* visual e definição de valores *defaults* para automatizar parte do processo de construção gráfica.

9. CONCLUSÕES

Este trabalho consiste no desenvolvimento da ferramenta Web para auxiliar na construção de gráficos. A fim de alcançar esse objetivo, projetaremos uma interface que visa oferecer apoio através de um sistema de recomendações com base em perguntas adequadas às visualizações e sinalização de problemas de construção. As avaliações com usuários serão realizadas após a implementação das melhorias para verificar se o objetivo foi alcançado.

10. REFERÊNCIAS

[1] Elias, Micheline; Bezerianos, Anastasia. 2011. Exploration Views: Understanding Dashboard Creation and Customization for Visualization Novices. Interact 2011, Part IV, pp. 274-291.

[2] Sousa, Taissa. 2011. Caracterização semântica de mecanismos de visualização. Monografia de fim de curso da disciplina Top. Hipertexto/Multimidia II

[3] Sousa, Taissa. 2011. Como sistemas de significação e comunicação influenciam na interpretação de dados estatísticos por usuários interessados em responder determinadas perguntas. Monografia de fim de curso da disciplina Engenharia Semiótica.

[4] Heer, Jeffrey; Shneiderman, Ben. 2012. Interactive Dynamics for Visual Analysis. A taxonomy of tools that support the fluent and flexible use of visualizations. ACMqueue.

[5] Bostock, Michael; Heer, Jeffrey. 2009. Protovis: A Graphical Toolkit for Visualization. InfoVis 2009.

[6] Tory, Melanie; Möller, Torsten. Rethinking Visualization: A High-Level Taxonomy

[7] Duke, D.J.; Brodlie K.W.; Duce, David.A. Building an Ontology of Visualization

[8] National e-Science Center. Visualization Ontologies. Último acesso em 13/11/2011. Disponível em http://www.nesc.ac.uk/talks/393/vis_ontology_report.pdf

[9] Mackinlay, Jock D.; Hanrahan, Pat; Stolte, Chris. Show Me: Automatic Presentation for Visual Analysis. IEEE Transactions on Visualizations and Computer Graphics, vol. 13, número. 6, november/ december 2007.

[10] Pinker, S. (1990). A theory of graph comprehension. In R. Freedle (Ed.), Artificial intelligence and the future testing (pp. 73-126). Hillsdale, NJ: Erlbaum.

[11] Goldenberg, E. Paul. (1988) Mathematics, metaphors, and human factors: Mathematical, technical, and pedagogical challenges in the educational use of graphical representation of functions. The Journal of Mathematical Behavior, Vol 7(2), 135-173.

[12] Tufte, Edward R. The Visual Display of Quantitative Information. 2001

[13] Clement, J. "Misconceptions in Graphing". Proceeding 9nd Annual Meeting of the International Group for the Psychology of Mathematics Education, 1985. p. 369-375.

[14] Gomes Ferreira, V. G. Exploring Mathematical Functions Through Dynamic Microworlds. 1997, 353 f. Tese (Doutorado em Educação). Institute Education, Universidade de Londres. Londres, 1997.

[15] Carzola, Irene. A relação entre a habilidade viso-pictórica e o domínio de conceitos estatísticos na leitura de gráficos. Campinas, 2002.

[16] Bertin, Jacques. Semiology of Graphics: Diagrams, Networks, Maps. 1918. esri first edition 2011, usa

[17] Steele Julie; Lliinsky, Noah. Beautiful Visualization. Looking at Data through the Eye of Experts. O'Reilly, 2010.

[18] Barbosa, S.D.J. & Silva, B.S. (2010) Interação Humano-Computador. Campus/Elsevier.

[19] Ericsson, K. Anders; Simon, Herbert A. Protocol Analysis. MIT, 1993

[20] Wilson, C. (1998). Usability techniques: Pros and cons of co-participation in usability studies. Usability interface, 4(4).

[21] Duke, D.J.; Brodlie K.W.; Duce, David.A; Herman, Ivan. 2005. Visualizations Viewpoints - Do you see what I mean? IEEE Computer Society 2005.

[22] Santaella, Lucia. O método anticartesiano de C. S. Peirce. Editora UNESP, 2004.

[23] Amar, Robert, Eagan, James, Stasko, John. 2005. Low-Level Components of Analytic Activity in Information Visualization. IEEE Symposium on Information Visualization 2005. USA

[24] Del Rio, N; da Silva, Paulo Pinheiro. Visualization Queries. Último acesso em 12/11/2011. Disponível em http://trust.utep.edu/visko/docs/docs/visquery.pdf

[25] Ricci, Francesco; Rocarch, Lior; Shapira, Bracha; Kantor, Paul B. Recommender System handbook. Springer 2011.

Uma Ferramenta de Autoria para Aplicações NCL e NCLua: uma Abordagem Orientada a Templates e com Suporte a Serviços Web

Thales Pordeus Ferreira
Trabalho de Mestrado
Início: março de 2011 – Término: dezembro de 2012
Programa de Pós Graduação em Informática (PPGI)
Universidade Federal da Paraíba (UFPB)
thales@lavid.ufpb.br

Raoni Kulesza
(Coorientador)
Centro de Informática (CIN)
Universidade Federal de Pernambuco (UFPE)
rk@cin.ufpe.br

Guido L. de S. Filho
(Orientador)
Laboratório de Aplicações de Vídeo Digital (LAVID)
Universidade Federal da Paraíba (UFPB)
guido@lavid.ufpb.br

RESUMO

As aplicações multimídia interativas tipicamente oferecem uma interface com o usuário (UI – User Interface) sofisticada e integrada com diferentes objetos de mídia. Complementar à UI multimídia, é comum a necessidade de apresentar informações visuais associadas a algum processamento de lógica de aplicação, uma característica comum na área das aplicações de informação, entretenimento ou jogos de computador. Neste contexto, a especificação da aplicação é feita em conjunto pelas equipes de software e UI, normalmente utilizando respectivamente ambientes de programação e ferramentas de edição visual. Atualmente, é possível identificar uma lacuna na definição de ferramentas de autoria que considerem a integração entre UI e lógica de aplicação complexa, por exemplo, os serviços web. O objetivo deste trabalho é definir uma ferramenta de autoria orientada a templates que integre melhor objetos de mídia e lógica de aplicação complexa, de forma a diminuir o tempo de desenvolvimento das aplicações. É empregada uma abordagem de desenvolvimento dirigida a modelos para gerar automaticamente as aplicações, reusando estruturas do template e funcionalidades presentes em serviços web.

PALAVRAS-CHAVE

Ferramenta de Autoria, Desenvolvimento Orientado a Templates, NCL, Lua, Serviços Web, TV Digital.

1. CONTEXTO TEÓRICO

As mudanças radicais da TV para tornar-se numa plataforma digital e interativa, como por exemplo, a TV Digital Interativa (TVDI) e as TVs Conectadas [1], possibilitam ao telespectador vivenciar experiências de informação e interação aproximadas daquelas já difundidas em aplicações presentes na Web. No cenário da TVDI, as formas já presentes de interação com os programas de TV (como votação, envio de mensagens, entre outras) podem ser aprimoradas com o uso das aplicações sobrepostas aos programas de TV [2]. Dessa forma, a interatividade torna mais transparente a interação do telespectador com o programa de TV.

Uma aplicação multimídia é definida como um software que possui interface com o usuário (UI – *User Interface*) com pelo menos um objeto de mídia (áudio, vídeo, imagens, animações, gráficos 2D ou 3D). Adicionalmente à UI multimídia, é comum em algumas das aplicações na TVDI o envolvimento com algum processamento de lógica de negocio, característica comum na área das aplicações informativas [4]. Estas aplicações possuem novos requisitos de ter objetos de mídia que estão intimamente vinculados à lógica de aplicação.

Um cenário atual que envolve este tipo de integração são os serviços web. Os serviços web são um tipo de código procedural disponibilizando funcionalidades para qualquer tipo de aplicação. As aplicações utilizando serviços da web, a exemplo das aplicações web *Mashups*, estão se tornando cada vez mais comuns e o reuso de serviços em aplicações multimídia também se faz necessário.

Para o desenvolvimento das aplicações é empregado linguagens textuais de autoria especializadas no domínio de criação multimídia. Por exemplo, aplicações para o padrão brasileiro de TV Digital tem a sua apresentação visual especificada pela linguagem de marcação NCL (*Nexted Context Language*) e a parte de código imperativo (comumente chamado de *script*) descrita pela linguagem Lua (NCLua) [3]. Uma deficiência da linguagem textual é demandar um esforço de codificação para transcrever em forma de texto a especificação da aplicação. A especificação feita por linguagem textual como NCL está relacionada basicamente a duas tarefas: (i) definição da sintaxe do conceito da linguagem a ser utilizado e (ii) de que forma eles serão usados (isto é, seus atributos preenchidos com valores). O trabalho [10] contorna em parte esse problema, pois tais ferramentas, apesar de acelerarem a escrita da sintaxe, não resolvem o segundo problema, que é a edição rápida dos valores dos atributos relacionados à especificação da aplicação. Ferramentas de autoria e uma sintaxe mais enxuta ajudariam a poupar tempo na especificação do documento [11].

Uma linguagem de autoria, por ser voltada para especificação da apresentação visual da aplicação, costuma ser abstraída utilizando ferramentas de autoria. Ferramentas de autoria oferecem técnicas simples e intuitivas para a construção visual de uma aplicação [5]. Dessa forma, uma das grandes vantagens é ajudar na redução da quantidade de código que os programadores precisam produzir, possibilitando uma melhoria considerável no tempo de construção da aplicação.

Outra forma para aumentar a produtividade do software é o uso dos métodos de autoria orientado a templates. Um template reaproveita partes de estruturas de aplicações que já foram

desenvolvidas previamente em novos casos de aplicações. Atualmente, as linguagens TAL (Template Authoring Language) [6] e XTemplate [7] são as principais linguagens para desenvolvimento baseado em templates de aplicações NCL. Aqui também se faz pertinente esconder os detalhes de implementação sintáticos e semânticos dessas linguagens textuais com o uso de ferramentas visuais.

Uma das metodologias que procura diminuir a complexidade do desenvolvimento de aplicações de domínio específico é o Desenvolvimento Generativo de Software (GSD – *Generative Software Development*). A principal característica do GSD é ser uma abordagem que busca modelar e desenvolver famílias de produtos de software ao invés de sistemas individuais [8]. Este trabalho aplica alguns dos conceitos existentes no trabalho [9] para o desenvolvimento de aplicações multimídia baseada em templates.

2. IDENTIFICAÇÃO DO PROBLEMA

O desenvolvimento de aplicações multimídia envolve basicamente a coordenação entre dois tipos de projetos: (i) projeto da interface com o usuário e (ii) projeto do software, que é a lógica da aplicação. Pelo fato deles serem comumente desenvolvidas independentemente (por profissionais e equipes distintas) ambas precisam ser integradas. Sendo assim, um requisito importante é a coordenação dos diferentes grupos de desenvolvedores e a integração dos seus artefatos advindos dos seus diferentes resultados produzidos.

Quando interfaces multimídia com o usuário tornam-se mais sofisticadas, as dependências entre objetos de mídias, interface e lógica de aplicação aumentam. Neste sentido, existe uma dependência da UI para a lógica e vice versa. Tais inter-relações são um gargalo critico dentro do desenvolvimento da aplicação e requer uma coordenação entre os diferentes grupos de desenvolvedores [15].

As ferramentas de autoria para NCL existentes não suportam a integração entre o projeto de software e o projeto de interface. As ferramentas tratam as tarefas de especificação da interface e do relacionamento entre as mídias utilizando uma única visão, o que dificulta o trabalho independente do designer e do programador. Por exemplo, no Composer [12] a visão estrutural permite criar e editar a aparência da UI como também definir lógica da apresentação.

Como visto anteriormente, o Composer [12] e o EDITEC [13] são algumas das ferramentas de autoria existentes para auxiliar na criação rápida de aplicações NCL. Entretanto, elas requerem do usuário conhecimento especializado da linguagem NCL para desenvolver uma aplicação. Em ambos, a notação visual faz-se uso de muitos conceitos presentes na NCL, os quais são bastante específicos ao domínio de aplicações multimídia interativas.

Outro problema mais pontual das ferramentas de autoria existentes é a inexistência do suporte a linguagem NCLua, a qual possibilita a integração do código procedural dentro da aplicação NCL. Por exemplo, hoje não é possível que o programador possa de forma visual e rápida visualizar ou realizar uma chamada dos métodos existentes nas mídias procedurais e que um valor de retorno seja utilizado para ser apresentado por outra mídia. Um cenário atual que envolve bastante este tipo de integração é a utilização dos serviços web dentro da aplicação. A automatização do uso do código procedural dos serviços web facilitaria bastante a vida do programador.

3. OBJETIVO

Buscando solucionar os problemas relacionados anteriormente, este trabalho endereça a integração entre o projeto da interface com o usuário e o projeto do software utilizando uma ferramenta de autoria. Procuramos aplicar os conceitos e ideias existentes na área de GSD em uma ferramenta para a autoria de *templates* (famílias de aplicações multimídia) suportando a geração automática de aplicações NCL. Duas visões distintas e integradas suportam a

O objetivo geral desta dissertação é propor e desenvolver uma ferramenta de autoria orientada a *templates* (família de aplicações). O intuito é que a ferramenta coordene os artefatos produzidos pelo designer e pelo programador durante o processo de desenvolvimento da aplicação. Complementar a autoria visual da aplicação a especificação do *template* também é feita visualmente.

Um dos principais requisitos deste trabalho é abstrair os conceitos envolvendo tanto a linguagem NCL como a definição do template. Aspectos como a definição da interface e o relacionamento entre as mídias NCL e NCLua, ou seja, sincronismo e interatividade são feitas utilizado um mecanismo visual. A ferramenta diminui o esforço de criação da aplicação ganhando tempo em relação às linguagens textuais.

4. CONTRIBUIÇÕES

A principal contribuição esperada nesse trabalho é uma ferramenta de autoria para suportar o projeto da UI e projeto de software de maneira integrada e independente da sintaxe NCL. Como contribuições secundárias, mostramos como aplicar técnicas de GSD ao desenvolvimento de aplicações multimídia baseadas em templates.

5. METODOLOGIA

Como descrito anteriormente, este trabalho tenta resolver o problema de integração entre as atividades envolvidas no projeto da UI e no projeto de software (UI e Software).

A metodologia de pesquisa consiste em três fases: (i) análise do problema; (ii) proposta e projeto da solução e por fim a (iii) validação da proposta.

A primeira fase analisa os trabalhos existentes e investiga seus problemas. Para isso é feito uma revisão de literatura para determinar os requisitos da nossa abordagem que são advindos de outros trabalhos. Mais ainda, ela serviu para descobrir algumas limitações que precisam ser resolvidas. Tais limitações foram discutidas na seção 2 e constituem nos principais problemas endereçados. Primeiro foi feito um estudo sobre as principais abordagens para tratar o desenvolvimento de aplicações multimídia com foco na integração entre UI e Software. No segundo estudo, analisaram-se os métodos e ferramentas de autoria orientadas a templates.

Na segunda fase, os requisitos e problemas da etapa anterior são usados para projetar uma solução. O principal requisito é entender quais as tarefas e necessidades de integração o projeto de UI e software requerem. Baseado neles foi definido uma ferramenta de autoria com duas visões integradas e propomos um método de autoria baseado em template.

Finalmente, na terceira fase, é feito um estudo para investigar em que medida a ferramenta de autoria pode melhorar no tempo de desenvolvimento de aplicações multimídia em comparação aos métodos existentes de autoria orientada a templates. O objetivo é avaliar se a questão do nível de abstração utilizando um editor visual deve aumentar a produtividade, diminuindo o esforço de codificação durante o desenvolvimento de aplicações multimídia.

6. ESTÁGIO ATUAL DO TRABALHO

Como descrito na seção anterior, o trabalho proposto possui as etapas para analisar o problema e em seguida propor e validar a solução. Destas, encontra-se feito a análise do problema (seção 2) e uma proposta inicial da solução (seção 7).

No tocante ao projeto, para o método de autoria orientado por template já foi desenvolvida uma prova de conceito através de um protótipo que realiza a geração automática de aplicações baseada em ferramentas tipo *wizard* usando 2 (dois) estudos de casos. Um deles instancia aplicações a partir de um template para aplicações do tipo Enquete e outro template para aplicações de S*lideshow*. Nos dois templates são tratados cenários que utilizam Serviços *web*.

Adicionalmente, foi realizado um estudo piloto através de metodologias de engenharia de software experimental com o objetivo de avaliar de forma quantitativa a ferramenta em relação a trabalhos relacionados que representam ferramentas de autoria existentes.

7. SOLUÇÃO PROPOSTA

A criação de uma aplicação multimídia NCL envolve uma cooperação entre o projeto da interface com o usuário e o projeto de software, que é a lógica de aplicação. O projeto da interface com o usuário corresponde à criação da UI por meio das tarefas de (i) escolher as mídias adequadas, (ii) posicionar corretamente as mídias (leiaute) e (iii) especificar o conteúdo e aspecto visual das mídias. O projeto da lógica de aplicação refere-se ao relacionamento entre os objetos de mídia podendo incluir a especificação (i) da sincronização temporal entre as mídias e (ii) da interatividade do usuário com a aplicação.

Estas atividades possuem relações de dependências entre seus artefatos desenvolvidos dentro do processo. Os fatores que influenciam na dependência estão: (i) criação, exclusão, alteração de objetos de mídia a partir da lógica de aplicação em tempo de execução; (ii) atualização dos objetos de mídia usando conteúdo dinâmico; e (iii) geração de eventos de objetos de mídia propagados para a lógica da aplicação e vice versa [15]. Sendo assim, é necessário que a ferramenta de autoria defina duas visões distintas, sendo uma para cada projeto, e que as inter-relações entre elas sejam suportadas.

O relacionamento entre os artefatos produzidos no projeto da UI e no projeto da lógica de aplicação exige uma comunicação entre o designer e o programador. As atividades que necessitam integrar os artefatos produzidos por estes dois desenvolvedores estão: (i) relacionamento dos elementos da interface com os objetos do domínio e (ii) uso dos elementos que foram definidos na UI pelo designer para realizar o projeto da lógica de aplicação, isto é, especificação do relacionamento entre os elementos.

Para cumprir estes requisitos, a ferramenta de autoria disponibiliza dois editores visuais voltados para modelar os aspectos relacionados ao projeto da interface com o usuário (UI), o projeto da lógica de aplicação e paralelamente a definição do template. Ambos suportam as tarefas de cada etapa e a partir do uso de modelos os dois projetos são integrados de maneira rápida.

A Figura 1 apresenta um *workflow* de integração entre os dois principais editores presentes na ferramenta. O fluxo de dados através da ferramenta mostra como o projeto da interface coordena-se com o projeto do software. No primeiro passo, utilizando o **editor de interface**, o designer define todos os elementos da UI que irão fazer parte da aplicação. Ele utiliza como base uma paleta com componentes visuais que podem ser arrastados para uma área de edição especifica. Assim que os elementos da UI são definidos, o projetista de software utiliza estes elementos como ponto de partida para iniciar o projeto da lógica da aplicação. Sendo assim, ele não necessita que a versão final da interface esteja totalmente pronta.

No design da interface ocorre o primeiro ponto de integração entre UI e a lógica de aplicação. Além das respectivas tarefas de edição da UI, o designer deve descrever qual será o conteúdo de cada elemento da UI. Esta tarefa acarreta na programação do mapeamento entre a UI e os objetos do domínio. O editor auxilia o designer disponibilizando uma interface para mapear o elemento da UI com o domínio do domínio.

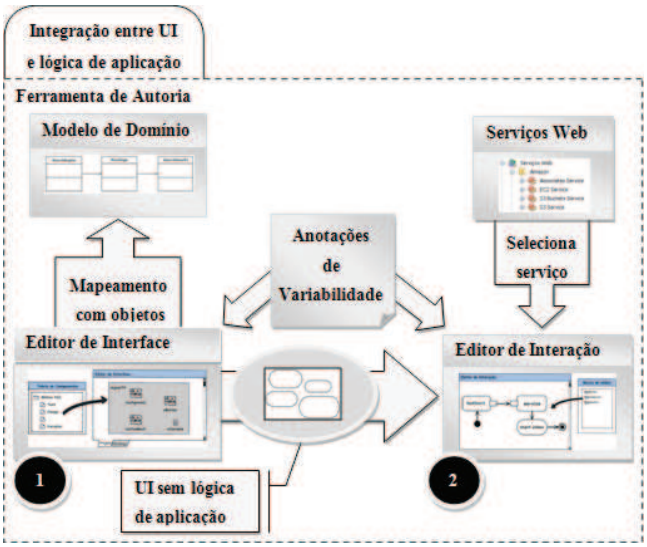


Figura 1 – Visão geral dos editores presentes na ferramenta.

No segundo passo, o **editor de interação** especifica a lógica de aplicação a partir dos elementos da UI presentes no editor de interface. O editor de interação modela visualmente o relacionamento entre as mídias NCL e Lua da aplicação tratando os aspectos de sincronização temporal e interatividade. Ele utiliza como base um grafo de blocos para modelar o comportamento da aplicação em decorrência da interação feita pelo usuário ou por eventos ocorridos nas mídias. A notação visual abstrai a sintaxe NCL/NCLua e é baseada no diagrama de ligação de sinais proposto em [14].

O editor inclui blocos para representar tanto os elementos da UI como o código imperativo de objetos NCLua (ou seja a lógica da aplicação). Eles são conectados para descrever o fluxo de ações a serem realizadas e dessa forma definindo a lógica da aplicação. Funcionalidades de serviços web também são reusadas em

termos de blocos e são selecionados a partir da paleta de serviços web.

Paralelamente a criação da aplicação, o desenvolvedor utiliza um conjunto de anotações de variabilidade para definir o template da aplicação. As anotações são usadas para descrever como os elementos do modelo realizam os aspectos comuns e variáveis do template. Um elemento pode ser anotado por dois tipos de anotações: (i) condição de presença (CP) – presença ou não de determinado elemento, por exemplo, elemento presente apenas quando selecionado para existir na aplicação instanciada; (ii) multiplicidade – repetição de elementos do template.

A aplicação final é gerada automaticamente com base na configuração escolhida pelo usuário a partir das características do template. As características da aplicação são os pontos comuns e variáveis do template em questão. A aplicação é obtida passando por duas etapas de transformação, sendo uma transformação entre modelos, para instanciar o template a partir da configuração, e por fim a geração textual, para transformar a instancia do template na aplicação NCL.

8. TRABALHOS RELACIONADOS

Quanto às ferramentas de autoria podemos destacar: EDITEC [12] e Composer [13]. O EDITEC é um editor gráfico de templates de composição hipermídia baseado na linguagem XTemplate 3.0. Seu objetivo é auxiliar o processo de criação de templates através de uma abordagem gráfica de fácil uso sem a necessidade de conhecimento da linguagem XTemplate ou NCL. O Composer não considera templates durante a autoria, mas disponibiliza recursos de edição visual e textual para auxiliar o desenvolvedor em tarefas repetitivas.

A grande vantagem destes editores é auxiliar na criação rápida de aplicações NCL. Entretanto, eles requerem do usuário conhecimento especializado da linguagem para desenvolver uma aplicação. Em ambos, a notação visual faz-se uso de muitos conceitos presentes na NCL, os quais são bastante específicos ao domínio de aplicações multimídia interativas.

Como explicado na seção 2, as ferramentas de autoria existentes não suportam a integração entre o projeto de software e o projeto de interface (UI). Outro problema é a inexistência do suporte a integração entre o código NCL e o código procedural que trata a lógica da aplicação. O uso dos serviços *web* também faz parte deste cenário de integração.

Em relação às abordagens de desenvolvimento é possível destacar o trabalho [15]. Ele enfoca o desenvolvimento de aplicações multimídia interativas usando a MML (*Multimedia Modeling Language*) como linguagem para modelagem. O principal objetivo é permitir que o processo envolva diferentes especialistas pertencentes aos três tipos de projeto existente: projeto de software; interface com o usuário e produção de mídias. Apesar de ser uma linguagem voltada para integrar UI e software, a MML não considera templates, nem serviços web no seu metamodelo. Outra deficiência é que nenhum mecanismo é criado para abstrair os conceitos da UML durante a definição do projeto de software através do diagrama de atividades. Dessa forma, a linguagem é voltada apenas para especialista em UML.

9. AGRADECIMENTOS

O autores agradecem à CAPES e ao CTIC/RNP pela suporte financeiro a esse projeto de pesquisa.

10. REFERÊNCIAS

[1] Alfonsi B., “I Want My IPTV: Internet Protocol Television Predicted a Winner,” *IEEE Distributed Systems Online,* 2005.

[2] Bachmayer S., Lugmayr A. e Kotsis G., “Convergence of collaborative web approaches and interactive TV program formats,” *International Journal of Web Information Systems*, 2010.

[3] Soares L. F. G., Moreno M. F. e Sant'Anna F., “Relating declarative hypermedia objects and imperative objects through the NCL glue language,” em *Proceedings of the 9th ACM symposium on Document engineering*, 2009.

[4] Pleub A., “Modeling the user interface of multimedia applications,” em Proceedings of the 8th international conference on Model Driven Engineering Languages and Systems, 2005.

[5] Kaskalis T., Tzidamis T. D. e Margaritis K. G., “Multimedia Authoring Tools: The Quest for an Educational Package,” *Educational Technology & Society*, 2007.

[6] Soares Neto C., “Autoria de Documentos Hipermídia Orientada a Templates,” Tese de Doutorado, 2010.

[7] dos Santos J. e Muchaluat-Saade D. C., “XTemplate 3.0: spatio-temporal semantics and structure reuse for hypermedia compositions,” *Multimedia Tools and Applications*, 2011.

[8] Czarnecki K., “Overview of generative software development,” em I*nternational conference on Unconventional Programming Paradigms*, 2005.

[9] Czarnecki K. e Antkiewicz M., “Mapping Features to Models: A Template Approach Based on Superimposed Variants,” em *International Conference on Generative Programming and Component Engineering*, 2005.

[10] Azevedo R. G. A., Neto C. S. S., Teixeira M. M., Santos R. C. M. e Gomes T. A., “Textual authoring of interactive digital TV applications,” *EuroITV '11,* 2011.

[11] Soares Neto C., de Souza C. S. e Soares L. F. G., “Linguagens computacionais como interfaces: um estudo com nested context language.,” em *IHC*, 2008.

[12] Guimarães L. R., Guimarães R. M. R. e Soares L. F. G., “Composer: Authoring Tool for iTV Programs,” *Euro iTV,* 2008.

[13] Damasceno J., Santos J. e Muchaluat-Saade D. C., “EDITEC: Editor Gráfico de Templates de Composição para Facilitar a Autoria de Programas para TV Digital Interativa,” *Simpósio Brasileiro de Sistemas Multimídia e Web,* 2010.

[14] Faison T., Event-Based Programming: Taking Events to the Limit, 1ª ed., Apress, 2011.

[15] PleuB A., “MML: A Language for Modeling Interactive Multimedia Applications,” em *International Symposium on Multimedia*, 2005.

Recomendação de Conteúdo em Ambientes de Convergência Digital Incorporada ao *Middleware* Ginga

Priscilla Kelly Machado Vieira
Laboratório de Aplicações de Vídeo Digital (LAVID)
Universidade Federal da Paraíba
Departamento de Informática
João Pessoa – Paraíba – Brazil
+55 (83) 3216-7093
priscilla@lavid.ufpb.br

Natasha Queiroz Lino
Laboratório de Aplicações de Vídeo Digital (LAVID)
Universidade Federal da Paraíba
Departamento de Informática
João Pessoa – Paraíba - Brazil
+55 (83) 3216-7093
natasha@ci.ufpb.br

ABSTRACT
The emerging scenario of interactive Digital TV (iDTV) is promoting the increase of interactivity in the communication process and also in audiovisual production, thus rising the number of channels and resources available to the user. This reality makes the task of finding the desired content becoming a costly and possibly ineffective action. The incorporation of recommender systems in the iDTV environment is emerging as a possible solution to this problem. This paper aims to propose an approach to content recommendation in iDTV, incorporated to the middleware Ginga, based on data mining clustering techniques and knowledge representation, considering the iDTV as a digital convergence environment (DTV and Web).

Categories and Subject Descriptors
H.3.5 [**Information Storage and Retrieval**]: On-line Information Services – data sharing, web-based services.

General Terms
Algorithms, Design.

Keywords
Interactive Digital TV, Ginga, Recommendation Systems.

1. INTRODUÇÃO
Com o advento da TV Digital interativa (TVDi), nota-se o aumento de interatividade no processo de comunicação além do incremento das produções audiovisuais [1], o que torna um ambiente favorável ao desenvolvimento de aplicações dedicadas a esta tecnologia. Neste sentido, a TV Digital (TVD) promoveu um cenário de aumento da quantidade de canais, serviços e conteúdo disponíveis ao usuário. Esta nova realidade faz com que a tarefa de encontrar o conteúdo desejado se torne uma ação onerosa e, em alguns casos, ineficiente. É neste contexto que Sistemas de Recomendação [2] emergem como solução possível para auxiliar esta escolha.

No contexto tratado, o trabalho propõe um módulo de recomendação adequado à arquitetura da TVDi brasileira, considerando-a como um ambiente de convergência digital (TV e Web) [3], inserido no *middleware* Ginga [4], permitindo o direcionamento de conteúdo de acordo com o usuário. O serviço de recomendação utilizará uma abordagem baseada em técnicas de Mineração de Dados [5] integradas a conceitos semânticos da Web Semântica [6].

2. CONTEXTO TEÓRICO
A evolução tecnológica associada com a inserção da Televisão Digital no Brasil propicia transformações profundas em todas as instâncias relativas ao meio, caracterizado por um processo de convergência a uma nova plataforma de comunicação baseada em tecnologias digitais de codificação, compressão, multiplexação, transporte, transmissão e recepção de informações. Adicionalmente, a TVDi congrega uma gama de possibilidades que incrementam o processo de comunicação: (i) Inclui a interatividade e (ii) Possibilita conectividade com a Web (TVDi Conectada). Estas características redemocratizam o acesso à informação [1], principalmente se considerarmos que a TV ainda é o principal meio de transmissão de informações e entretenimento para a população brasileira, estando presente em mais de 95% das residências; enquanto que os computadores possuem uma cobertura de pouco mais de 38% das residências brasileiras [7].

Arquiteturalmente, o sinal da TVD é recebido e decodificado por um dispositivo chamado de Set-Top-Box (STB). Por possuir tais funcionalidades, esse dispositivo assemelha-se a um computador com poder limitado de processamento, podendo, inclusive, ser conectado na Web. Torna-se permissivo, assim, o envio e recebimento de dados via STB. Considerando o Sistema Brasileiro de TV Digital (SBTVD) [8], uma das camadas que compõem a arquitetura do receptor é o *middleware*, responsável por abstrair as particularidades do sistema para aplicações e usuários. No cenário brasileiro, o software responsável por estas atividades é o Ginga [4].

Analisando o surgimento da TVD, observa-se o aumento da quantidade de canais e recursos disponíveis ao usuário [9]. Esta realidade propicia o desenvolvimento de sistemas de recomendação de conteúdo, que podem aperfeiçoar a busca por conteúdo, tornando-a menos onerosa e mais adequada aos interesses dos usuários.

2.1.1 Sistema de Recomendação de Conteúdo

Diversos aspectos devem ser considerados para a construção de um sistema de recomendação, a exemplo de como as informações são coletadas, e os métodos de recomendação empregados [10].

A coleta de dados implícita é transparente ao usuário, onde seu comportamento é monitorado por meio do histórico do sistema [11]. A principal vantagem desta abordagem é ser discreta ao objetivo do usuário, não sendo necessária sua interação direta com o sistema. Em contrapartida, a precisão do sistema é atenuada, pois depende intrinsecamente dos dados coletados ao longo do uso do sistema. Por outro lado, na coleta explícita o sistema necessita que o usuário informe claramente suas preferências [11]. A principal vantagem desta abordagem é a precisão para expressar preferências ou interesses. Em contrapartida, a necessidade de interação direta torna o processo muitas vezes dispendioso, sendo dificultado com a forma de entrada de dados via controle remoto. Esta abordagem pode não ser a mais adequada para o domínio da TV, por ser caracterizado, predominantemente, por interações passivas dos usuários [11].

Entre as técnicas utilizadas para recomendação de conteúdo inclui-se a Mineração de Dados [5], passo mais importante no processo de Descoberta de Conhecimento em Banco de Dados (em inglês, *Knowledge Discovery in Data Bases* - KDD) [12].

KDD pode ser definido como o processo não trivial de identificação de padrões válidos, potencialmente úteis em um conjunto de dados [12]. Esses padrões extraídos devem ser confiáveis, compreensíveis e úteis, podendo ser empregados nos âmbitos científico ou governamental.

A etapa de Mineração de Dados, inclusa no processo de KDD, consiste na aplicação de análise de dados e algoritmos que produz uma relação particular de padrões de dados [12]. Um das tarefas de mineração de dados é a Clusterização [13], que consiste em dado uma base de dados X, agrupar (clusterizar) os objetos de X de modo que os mais similares sejam alocados no mesmo grupo (cluster) e os menos similares em clusters distintos.

Observando a gama de dados a serem minerados e as informações que podem ser extraídas após a mineração, pode-se raciocinar automaticamente sobre estes dados estruturando-os em Ontologias [14].

Ontologias podem ser definidas como um modelo abstrato capaz de organizar de forma explícita e formal os conceitos e restrições relacionados a um domínio de interesse. Desta forma, por serem formais, Ontologias possibilitam o processamento automático por máquinas e facilitam o compartilhamento de informações por um grupo de pessoas ou máquinas [15].

Comumente os sistemas de recomendação são classificados de acordo com o modo que realiza o serviço. Classicamente, utilizam: (i) Filtragem Baseada em Conteúdo [16], parte do princípio de que, se um usuário visualizou determinado programa, é provável que no futuro goste de outros similares. (ii) Filtragem Colaborativa [16], parte do princípio de que grupos de pessoas similares possuem comportamentos semelhantes e (iii) Filtragem Hibrida [16], combinação de abordagens.

Mais recentemente, surgiram os sistemas de recomendação semântica, os quais são caracterizados por utilizarem uma abordagem semântica, que utiliza conceitos e técnicas de Representação de Conhecimento (RC) [17], como por exemplo, Ontologias, durante o processo de recomendação. RC é a subárea da Inteligência Artificial que avalia como o conhecimento pode ser representado simbolicamente e manipulado de forma automática por máquinas [17].

Em sistemas de recomendação, em geral, são utilizadas bases para armazenamento e gerenciamento das informações dos usuários [18, 19]. Adicionalmente, estas bases podem ser definidas por meio de ontologias, de forma a manter os dados padronizados, permitindo o compartilhamento de informações, provendo semântica e inferência automática de relacionamentos entre os dados [18].

3. IDENTIFICAÇÃO DO PROBLEMA

Esta proposta está inserida no projeto *Knowledge TV* (KTV) [3] que propõe uma camada semântica, baseada nos conceitos da Web Semântica, para prover serviços na plataforma da TVDi. Desta forma, dentro do contexto analisado, propomos a recomendação de conteúdo incorporada ao *middleware* Ginga. Este serviço deverá ser baseado em Clusterização de dados utilizando a técnica *K-means* [20], algoritmo de agrupamento caracterizado por ser popular, com desempenho linear e largamente utilizado em ambientes multimídia [18].

Todos os dados resultantes do processo de Clusterização deverão ser estruturados em uma ontologia, permitindo o compartilhamento de informações entre sistemas e a inferência sobre o conhecimento armazenado.

Para a definição do trabalho proposto foram considerados diversos requisitos, tais como: (i) Ser multi usuário, considerando a TV como um ambiente de vários usuários, (ii) Ser multi plataforma, podendo recomendar conteúdo tanto em outros sistemas de TV quanto em outros ambientes, como a web (iii) Ser genérico, possibilitando o serviço de recomendação independente do ambiente provedor dos dados.

Considerando os requisitos anteriores foi proposta a arquitetura baseada em módulos da Figura 1.

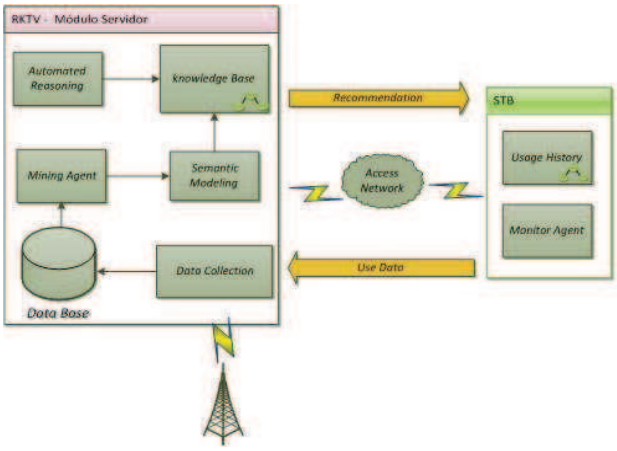


Figura 1. Arquitetura proposta.

A abordagem é baseada em uma arquitetura cliente-servidor. No lado servidor serão especificados todos os módulos que formarão o componente de recomendação proposto e inserido na arquitetura do KTV o *Recommender Knowledge TV* (RKTV). Desta forma, a arquitetura é estruturada em 7 (sete) módulos (Figura 1), distribuídos em 2 (dois) componentes: (i) o cliente, representado por um STB e (ii) o servidor, local onde será realizado o

processamento para a disponibilização do serviço de recomendação. Cada módulo possui funcionalidades especificas, como destacado a seguir:

- *Monitor Agent*: Monitorar o comportamento dos usuários do STB em relação ao conteúdo exibido na TVDi Conectada;
- *Usage History:* Responsável por estruturar e gerenciar em uma ontologia as informações da programação visualizada no STB (atividade realizada no Monitor Agent);
- *Data Collection:* Capturar os dados da programação das emissoras e dos usuários;
- *Mining Agent:* Responsável por todo o processo de mineração de dados sobre os dados coletados do STB e da emissora;
- *Semantic Modeling*: Tornar homogênea a descrição do conteúdo multimídia minerado, estruturado por meio de ontologias;
- *Knowledge Base*: Armazenar todo o conhecimento gerado no processo proposto em forma de ontologias;
- *Automated Reasoning*: Responsável por processar as informações presentes nas bases de conhecimento, gerando recomendações de conteúdo compatíveis aos interesses do usuário.

Os módulos do componente STB, apresentado na Figura 1, serão integrados ao Núcleo Comum do *middleware* Ginga onde são definidas funções básicas de sistemas de TVD, tais como: exibição e controle de mídias, controle de recursos do sistema, canal de retorno, dispositivos de armazenamento, acesso a informações de serviço e sintonização de canais [4]. A conexão entre o componente STB e o componente RKTV permitirá o serviço de recomendação semântica ao ambiente da TVD.

O módulo *Monitor Agent* atuará de forma implícita [11], monitorando o comportamento dos usuários do STB. Esta coleta de dados será por meio de uma comunicação constante com os módulos *Tuner* [4], responsável pela sintonização dos canais, e *SI* [4], responsável por extrair os metadados transmitidos pelas emissoras, do *middleware* Ginga.

O módulo *Data Collection* do componente RKTV captará a programação semanal das emissoras. Estes dados serão submetidos ao processo de Mineração de Dados. Após a atividade de Clusterização, grupos de conteúdo inter-relacionados serão gerados.

Os dados do histórico do uso do STB serão minerados, da mesma forma que os dados coletados da emissora. Duas formas de recomendação poderão ser geradas: (i) Os grupos de dados do histórico serão avaliados e qualquer conteúdo que estiver inserido em grupos semelhantes, gerados na mineração dos dados das emissoras, poderão ser recomendados, o que possibilitaria recomendações inesperadas e novas ao usuário. (ii) O conteúdo visualizado no momento em que a recomendação for solicitada será analisado e serão recomendados conteúdos que estejam classificados no mesmo grupo do conteúdo atual, o que possibilitaria recomendações mais próximas do esperado pelo o usuário. O raciocínio descrito será realizado sobre as ontologias, resultantes do processo de Mineração de Dados, no módulo *Knowledge Base* do componente RKTV. Basicamente, propomos um sistema de recomendação Hibrido, baseado em Filtragem Colaborativa e Semântica.

4. OBJETIVOS

Como objetivo geral propõe-se um sistema de recomendação adequado a arquitetura da TVDi brasileira, permitindo o direcionamento de conteúdo de acordo com o usuário. O trabalho inclui uma abordagem para recomendação em ambientes de convergência (TV e Web). Essa abordagem, baseada em conceitos de Representação do Conhecimento, como Ontologias, também propõe a integração de tecnologias da Web Semântica [21], permitindo, assim, o raciocínio automático sobre os dados e o compartilhamento de informação do sistema proposto com outros.

Para a obtenção do objetivo geral, foram definidos os seguintes objetivos específicos:

- Permitir recomendação de conteúdo multimídia em TVDi Conectada utilizando conceitos e técnicas da Web Semântica [6];
- Possibilitar que o *middleware* Ginga forneça o serviço de recomendação à TVDi Conectada;
- Obter um sistema de recomendação multi usuário, realizando recomendação de conteúdo multimídia a partir do uso do STB;
- Propor uma arquitetura genérica de recomendação de conteúdo multimídia, possibilitando o serviço independente do ambiente provedor dos dados.

5. CONTRIBUIÇÕE ESPERADAS

Ao final deste trabalho espera-se avançar no estado da arte em termos dos métodos para recomendação de conteúdo em TVDi, incorporando ao contexto da TV conceitos e técnicas baseadas nos princípios da Web Semântica. De modo suplementar, espera-se estender o *middleware* Ginga, com a incorporação de um módulo que disponha serviço de recomendação de conteúdo.

6. METODOLOGIA

A pesquisa deve consistir das seguintes etapas:

I. Realizar revisão bibliográfica relativa a métodos de recomendação no ambiente da TVD bem como à Descoberta de Conhecimento em Banco de Dados por meio de ontologias;

II. Propor um mecanismo de recomendação semântica, baseado em clusterização dos dados e ontologias, no ambiente da TVDi;

III. Desenvolver um módulo que possa ser inserido na arquitetura do SBTVD e que possibilite a recomendação de conteúdo;

IV. Avaliar a eficácia e relevância do módulo desenvolvido no item III.

7. ESTÁGIO DO TRABALHO

Atualmente, foi finalizado o levantamento do estado da arte em sistemas de recomendação de conteúdo. Foram avaliadas as técnicas e as características do processo de recomendação de trabalhos relacionados. A próxima etapa sob desenvolvimento trata da definição, implementação e avaliação do algoritmo de clusterização sobre os dados coletados do usuário e das emissoras de TV. Adicionalmente será definido o modelo de ontologia que estruturará o conhecimento proveniente da mineração, possibilitando, assim, o raciocínio automático sobre os dados.

Como trabalho futuro, destaca-se a necessidade de se implantar algoritmos de diversas tarefas da mineração de dados, como por

exemplo, classificação, regras de associação, entre outros, que permitirão um comparativo com a proposta, sendo possível, inclusive, a combinação de técnicas, fornecendo solução para problemas de múltiplas áreas. Também poderá ser estudada a integração desta solução em outros *middlewares* de sistemas de TVD, e em outros ambientes, tais como a web. Outro ponto que poderá ser explorado é o raciocínio avançado sobre a ontologia proposta neste trabalho, especificação de novos conceitos e relacionamentos que proporcionem melhor precisão no processo de recomendação.

8. TRABALHOS RELACIONADOS

Outros trabalhos propuseram sistemas de recomendação na TV. Ávila [9] apresenta um sistema integrado ao *middleware* Ginga baseado em mineração por regras de associação sem nenhuma estruturação semântica do conteúdo. Diferentemente, este trabalho propõe o uso de ontologias para estruturar o conhecimento após o processo de clusterização dos dados.

Aroyo [19] propõe a incorporação da semântica na TVD, considerando-a no ambiente da Web, a fim de possibilitar o desenvolvimento de aplicações distribuídas e a personalização do serviço. Aroyo baseia sua proposta de personalização nos conceitos da Web Semântica e na Mineração de Dados. Assim como Aroyo, nossa proposta também é baseada nos conceitos da Web semântica, no entanto, propomos recomendação no ambiente do SBTVD incorporando no *midleware* Ginga um módulo responsável por prover o serviço proposto.

9. REFERÊNCIAS

[1] Médola, A. S. L.. Televisão Digital Brasileira e os Novos Processos de Produção de Conteúdos: Os Desafios para o Comunicador. Revista da Associação Nacional dos Programas de Pós-Graduação em Comunicação | E-compós, Brasília, v.12, n.3, set./dez. 2009.

[2] Resnick, P. and Varian, H. R. 1997. Recommender systems. Communications of the ACM. ACM 40, 3 (Mar. 1997), 56-58.

[3] Lino, N., Araújo, J., Anabuki, D., JR, J. P., Batista, M., Nóbrega, R., Amaro, M., Siebra, C. and Lemos, G.. Knowledge TV. EuroITV, Lisboa, Portugal, 2011.

[4] Soares, L. F.; Lemos, G. Interactive Television in Brazil: System Software and the Digital Divide. In European Interactive TV Conference - EuroITV2007. Amsterdam, 2007.

[5] Han, J. and Kamber, M. Data Mining Concepts and Techniques, 2ª edição, Editora Elsevier, Reino Unido.

[6] Berners-Lee,T,; Lasila, O. Hendler, J. The Semantic Web. Scientific American, 284 (5):34-43, 2011.

[7] IBGE - Instituto Brasileiro de Geografia e Estatística. Síntese de Indicadores Sociais. Brasil, 2012.

[8] SBTVD – Sistema Brasileiro de TV Digital. Ministério das Comunicações. Disponível em: <http://sbtvd.cpqd.com.br/>. Acessado em 20 de Junho de 2012.

[9] Ávila, P. M. RecommenderTV: Suporte ao Desenvolvimento de Aplicações de Recomendação para o Sistema Brasileiro de TV Digital. Dissertaçao de Mestrado. Universidade Federal de São Carlos, Brasil, 2010.

[10] Ricc, F.; Rokach, L.; Shapira, B.; Kantor, B.P.. Recommender Systems Handbook. Springer Sciense Busines Media, LLC, 2011

[11] Chorianopoulos, K.. Personalized and mobile digital TV applications. Proc. 2008 MultimediaTools and Applications, Volume 36, Issue 1-2. Kluwer Academic Publishers, pp. 1-10, 2008.

[12] Fayyad, U. M., PIATESKY-SHAPIRO, G., SMYTH, P. From Data Mining to Knowledge Discovery: an Overview". Advances in Knowledge Discovery and Data Mining, AAAI Press, 1996.

[13] Ochi, L. S., DIAS, C. R., Stênio, S. F. Clusterização em Mineração de Dados. *Escola Regional de Informática Rio de Janeiro*. Espírito Santo. Mini Curso. Rio das Ostras, 2004.

[14] Guarino, N. Formal Ontology, Conceptual Analysis and Knowledge Representation, International Journal of Human-Computer Studies, 43(5-6):625–640, 1995.

[15] Gruber, Thomas R., A Translation Approach to Portable Ontology. Journal Knowledge Acquisition – Special issue: Current issues in knowledge modeling, 1993, v: 5. 199-220p.

[16] Adomavicius, G., Tuzhilin, A. Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering, 2005, v: 17. 734-749p.

[17] Russell, S. and Norvig, P. Artificial Intelligence - A Modern Approach. 2 ª edição, Prentice Hall, 2002.

[18] Kim, J., Kwon, E., Cho, Y., Kang, S.. Recommendation System of IPTV TV Program Using Ontlogy and K-means Clustering. Ubiquitous Computing and Multimedia Applications, Daejeon, Korea, 2011.

[19] Aroyo, L.; Conconi, A.; Dietze, S.; Kaptein, A.; et *al*. NoTube – Making TV a medium for personalized interaction. EuroITV2009, Leuven, Belgium, 03-05 Jun 2009.

[20] Berkhin. P.. Survey of Clustering Data Mining Techniques. Technical report, Accrue Software, San Jose, CA, 2002.

[21] Breitman, K. . Web Semântica: O Futuro da Internet. 1. ed. Rio de Janeiro: LTC - Livros Tecnicos e Cientificos Editora S.A., 2005. v. 1. 190 p.

Um Framework para a Publicação de Dados Abertos Governamentais a partir de Bases de Dados Relacionais

Clayton Martins Pereira
Mestrando em Engenharia Eletrônica e Computação
Instituto Tecnológico de Aeronáutica, Brasil
clayton@ita.br

José Maria Parente de Oliveira
Professor da Divisão de Ciência da Computação
Instituto Tecnológico de Aeronáutica, Brasil
parente@ita.br

ABSTRACT
This work aims to define a framework, based on Semantic Data Layer of DIGO architecture for open government data, for the purpose of automating the generation and publication of open government data (semantic data) obtained from structured data maintained in Relational Databases.

RESUMO
Este trabalho visa definir um *framework*, baseado na camada de dados semânticos (*Semantic Data Layer*) da arquitetura de dados abertos governamentais DIGO, com a finalidade de automatizar os processos de geração e publicação de dados abertos governamentais (dados semânticos) obtidos a partir de dados estruturados mantidos em Bases de Dados Relacionais.

Categories and Subject Descriptors
D.3.3 [**Programming Languages**]: Language Constructs and Features – frameworks, modules, packages.

General Terms
Management, Design, Experimentation, Languages.

PALAVRAS-CHAVE
Dados Abertos Governamentais, *Resource Description Framework*, Ontologias, Bases de Dados Relacionais.

INFORMAÇÕES
Categoria: Mestrado
Universidade: Instituto Tecnológico de Aeronáutica
Programa: Pós-Graduação em Engenharia Eletrônica e Computação – Área Informática
Início: 2011.2
Previsão de Defesa: 2012.2

1. CONTEXTO TEÓRICO
A *Web* Semântica representa um novo campo de pesquisa e desenvolvimento na área da Tecnologia da Informação, que tem por objetivo trazer estrutura para o conteúdo significativo de páginas da *Web*, dado que o conteúdo disponibilizado na *Web* atualmente, de uma forma geral, é destinado apenas à leitura humana, não possibilitando que este seja reutilizado ou manipulado por computadores e seus *softwares*. Com a *Web* Semântica, pretende-se que as informações a serem disponibilizadas na *Web* sejam estruturadas, com significados bem definidos, de forma a habilitar que computadores, tais como *desktops* e dispositivos portáteis, sejam capazes de processar e entender automaticamente estes dados, os quais hoje são apenas mostrados em tela ou impressos [1].

Dados estruturados são aqueles organizados segundo um padrão rígido e predefinido, respeitando diversos critérios como campos (ou atributos) de dados, extensão do campo, domínio (valores possíveis) do dado, tipo de dado, etc. Este é o caso, por exemplo, dos dados mantidos em tabelas de Bases de Dados Relacionais (BDR ou *RDB – Relational DataBases*), usados pelos sistemas de informação na maioria das instituições [2].

O *RDF* (*Resource Description Framework*) é um modelo que permite esta representação estruturada de dados e informações, onde significados são codificados em conjuntos de triplas objeto-atributo-valor, chamadas de declarações, que podem ser comparadas a uma oração onde temos: sujeito, predicado e objeto. O próprio usuário define sua terminologia através de uma linguagem chamada *RDF Schema* (que não tem qualquer relação com o *XML Schema*). Nela é possível definir um vocabulário, as propriedades de espécies de objetos e os valores que podem assumir, bem como descrever relacionamento entre estes [3].

As Ontologias, como artefato de engenharia, consistem em estruturas formais de conceitos e relações entre conceitos, além de um conjunto de axiomas que restringe a interpretação dessa estrutura e proporciona a derivação de conhecimento do conhecimento factual representado na estrutura. Através de Linguagens Ontológicas os usuários podem gravar conceituações explícitas e formais de modelos de domínio. Os principais requisitos de uma linguagem ontológica são: a sintaxe bem definida, o suporte a raciocínio eficiente, uma semântica formal, poder expressivo suficiente e conveniência de expressão. As linguagens ontológicas mais utilizadas atualmente são a *OWL* (*Ontology Web Language*) e a *RDFS* (*Resource Description Framework Schema*) [4].

O termo "dados abertos" é definido pelo *World Wide Web Consortium - W3C* como a publicação de dados em seu formato bruto, com semântica embutida, de forma que possam ser interpretados por máquina dentro do contexto semântico para os quais foram definidos, permitindo assim seu reuso por outras aplicações. Isso é possível através da disponibilização na *Web* de *datasets* na forma de triplas *RDF*, acessíveis por meio de

Identificadores Únicos de Recursos (*URI*, similar a uma *URL HTTP*), e consultáveis por uma *Query Language* (*SparQL*, considerada a linguagem *SQL* da *Web* semântica) [5].

Dados Abertos Governamentais referem-se à disponibilização de dados em formato aberto por órgãos e entidades governamentais, de maneira que possam ser prontamente publicados e acessados por todos os interessados, além de permitir seu reuso por outras aplicações [2]. Atualmente, as instituições governamentais publicam parte dos seus dados em Portais *Web*, utilizando as linguagens e tecnologias da *Web* atual, o que não oferece facilidades para reutilização desses dados na geração de novas informações. Sendo assim, o acesso à informação relevante, precisa e passível de reutilização por outros aplicativos, torna-se cada vez mais complexo.

Um modelo proposto para permitir a publicação de dados abertos governamentais e a correspondente forma de acesso a esses dados é apresentado em [2]. A Figura 1 apresenta a arquitetura geral de apoio ao modelo, denominada DIGO - Disponibilizando Informações de Governo. Ela é um modelo comum e tem por objetivo estipular um acordo semântico entre fontes heterogêneas de dados e permitir a fusão de dados de Portais *Web*. Ela está dividida em cinco camadas. A Figura 1 ilustra as suas camadas. A Camada 1 é denominada Geração de dados e Conhecimento. A Camada 2 é denominada Dados Sintáticos. A Camada 3 é denominada Dados Semânticos. A Camada 4 é denominada Fusão de dados. A Camada 5 é a camada de Informação. As linhas tracejadas em preto delimitam as camadas.

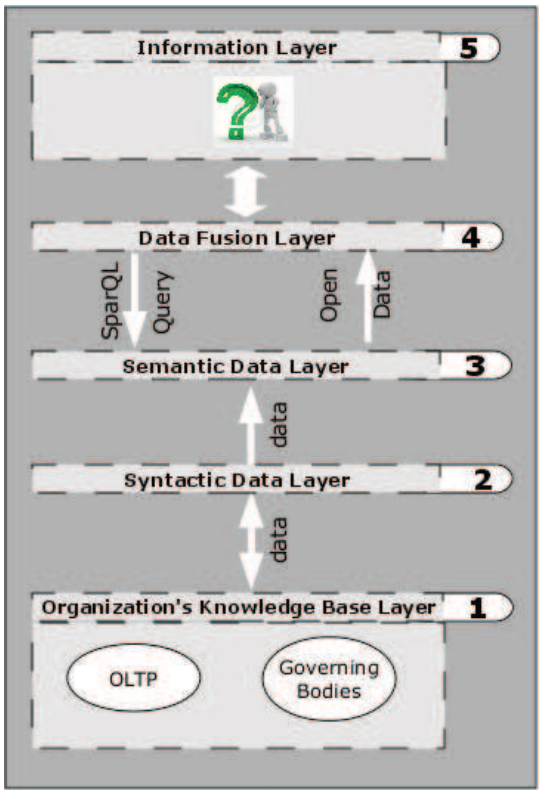


Figura 1. Camadas da arquitetura DIGO. Fonte: [2].

O escopo deste trabalho está em alguns elementos da Camada 3 da arquitetura (dados estruturados). A Camada 3 é a camada de disponibilização dos dados abertos e é responsável por disponibilizar os dados contendo a semântica embutida neles, mantendo o contexto semântico para o qual foram definidos em sua fonte originadora. Ela é composta por quatro subcamadas, apresentadas na Figura 2: Camada de Extração de dados, Camada de Transformação e Carga, Camada de Persistência dos Dados Semânticos e Camada de Manipulação de dados.

2. IDENTIFICAÇÃO DO PROBLEMA

Os processos de geração e publicação de dados abertos semânticos (em formato de triplas *RDF*) a partir de bases de dados relacionais (dados estruturados) contam atualmente com uma série de ferramentas de *software*, grande parte delas produtos de projetos de pesquisa acadêmica (vinculadas a trabalhos de mestrado ou de doutorado), as quais, no entanto, ainda exigem um elevado nível de conhecimento técnico e de interação manual do usuário para sua instalação, configuração e operação. Além disso, algumas dessas ferramentas possuem limitações de compatibilidade com alguns dos diversos sistemas gerenciadores de banco de dados (SGBDs) e sistemas operacionais atualmente encontrados no mercado, ou ainda implementam linguagens específicas para o processo de mapeamento entre ontologias ou vocabulários e as tabelas e colunas das bases de dados relacionais, o que dificulta o suporte e as eventuais necessidades de alterações ou correções manuais nesse mapeamento.

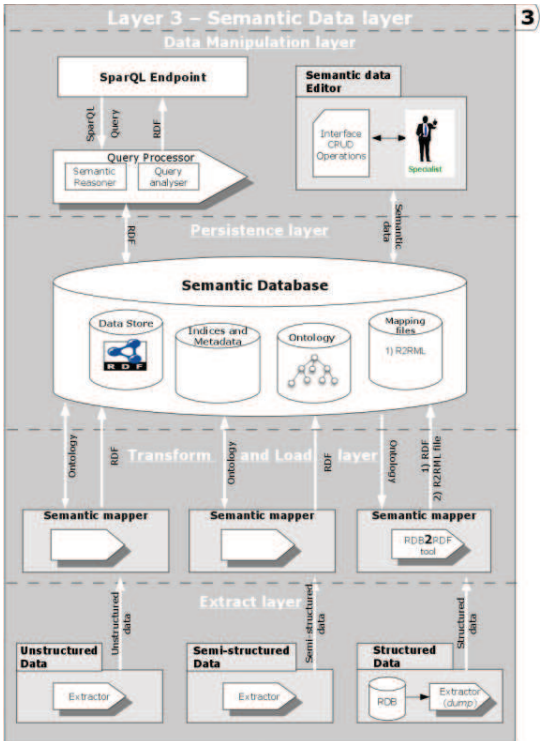


Figura 2. Visão detalhada da camada 3 da arquitetura DIGO e suas subcamadas. Fonte: [2].

Neste cenário, verifica-se a carência de *frameworks*/*toolkits* para automatizar as tarefas de extração, mapeamento e conversão de dados estruturados (em bases de dados relacionais) para dados abertos semânticos (geração de triplas *RDF*), de armazenamento (em *open datasets*) e de manipulação (publicação e consulta) das triplas geradas, de forma a oferecer uma solução única que integre métodos e ferramentas de *software* já desenvolvidas para cada

uma dessas tarefas, através de uma interface amigável para o usuário, que seja multiplataforma e compatível com os principais SGBDs disponíveis no mercado.

Com a entrada em vigor, em maio de 2012, da Lei de Acesso à Informação[1], que regula o acesso às informações sob a guarda de órgãos e entidades públicas de todos os poderes e entes federativos, todos estes deverão, ao divulgarem suas informações, "utilizar todos os meios e instrumentos legítimos de que dispuserem, sendo obrigatória a divulgação em sítios oficiais da rede mundial de computadores (*internet*)"[2], sendo que tais sítios deverão "possibilitar o acesso automatizado por sistemas externos em formatos abertos, estruturados e legíveis por máquina"[3] [6]. Considerando que grande parte dos dados a serem divulgados por tais órgãos e entidades são processados por Sistemas de Informações e persistidos em bases de dados relacionais, justifica-se a relevância do problema ora identificado.

3. OBJETIVO

O trabalho de mestrado ora apresentado visa definir um *framework*, baseado na camada de dados semânticos (*Semantic Data Layer*) da arquitetura de dados abertos governamentais DIGO [2], com a finalidade de automatizar os processos de geração, armazenamento e publicação de dados abertos governamentais (dados semânticos) obtidos a partir de dados estruturados mantidos em bases de dados relacionais (BDR).

Tal trabalho tem por objetivos específicos: 1) integrar, através de uma interface gráfica a ser desenvolvida em linguagem *Java*, as ferramentas de *software* selecionadas, após uma etapa prévia de pesquisa e avaliação, para executarem as funções de extração, mapeamento e conversão de dados estruturados (persistidos em BDR) para dados semânticos (geração de triplas *RDF*), de armazenamento das triplas *RDF* geradas (*RDF Store*), e de publicação (visualização e consulta) de dados semânticos, sendo que estas ferramentas deverão ser aderentes aos padrões do *W3C* (*RDF*, *OWL* e *R2RML*); 2) obter um método e desenvolver uma ferramenta de *software*, a ser implementada na referida interface gráfica, para a integração de uma ontologia *OWL* (em formato *turtle*) ao arquivo de mapeamento semântico da BDR, a fim de conferir maior expressividade e poder de inferência na geração e publicação dos dados semânticos (triplas *RDF*); 3) Validar o *framework* através de sua implementação em dois estudos de caso, um deles utilizando uma base de dados (SGBD *MySQL*) de teses defendidas em um dos programas de pós-graduação do Instituto Tecnológico de Aeronáutica (ITA), e outro utilizando uma base de dados (SGBD *MS-SQL Server*) do Sistema Único de Saúde (SUS) sobre tratamento e mortalidade de câncer de cérebro.

4. CONTRIBUIÇÕES ESPERADAS

Como contribuições esperadas deste trabalho estão: a obtenção de um modelo e de uma interface gráfica para integrar métodos e ferramentas de *software* voltados à geração e publicação de dados abertos governamentais, com base na camada de dados semânticos da arquitetura DIGO [2]; possibilitar ao usuário, através da referida interface gráfica, configurar as ferramentas do *framework* e fazer a integração de uma ontologia *OWL* ao mapeamento semântico da BDR, de forma a não exigir deste usuário elevados conhecimentos técnicos; facilitar e estimular a publicação de dados abertos governamentais, bem como a composição de aplicações para o consumo destes nas camadas de fusão e de informação da referida arquitetura; oferecer uma solução automatizada para que os órgãos e entidades da Administração Pública disponibilizem seus dados em formato aberto, de forma que os usuários possam obtê-los através de consultas customizadas de acordo com suas necessidades, bem como reaproveitá-los para a combinação com outras diferentes fontes de dados a fim de gerar novas informações (*mashup*), cumprindo assim o disposto na Lei de Acesso à Informação.

5. METODOLOGIA ADOTADA

Inicialmente foi conduzido um estudo detalhado da arquitetura DIGO [2] e dos conceitos relacionados, seguido de pesquisa na literatura e na *internet* em busca, respectivamente, dos trabalhos relacionados e das ferramentas existentes (determinação do estado da arte), a fim de possibilitar a identificação do problema de pesquisa e a análise de sua viabilidade técnica para, em seguida, determinar o objetivo e as contribuições esperadas do trabalho.

Assim, verificada a viabilidade técnica do trabalho, foram realizados alguns experimentos em laboratório a fim de avaliar as ferramentas encontradas, possibilitando a escolha daquelas que irão compor a versão inicial do *framework*. Para a realização dos experimentos foram testadas amostras das duas bases de dados a serem utilizadas nos referidos estudos de caso. As ferramentas foram escolhidas com base na aderência destas aos padrões do *W3C*, na compatibilidade com os SGBD utilizados (*MySQL* e *MS-SQL Server*), e na possibilidade de edição do arquivo de mapeamento semântico gerado, para a integração da ontologia.

Escolhidas as ferramentas que irão compor a versão inicial do *framework*, será então desenvolvida a interface gráfica, em linguagem *Java*, para integração destas ferramentas. Em paralelo, será elaborado o método e desenvolvida a ferramenta, a ser implementada na referida interface gráfica, para integração da ontologia *OWL* ao arquivo de mapeamento semântico da BDR.

Por fim, o *framework* será validado através de sua implementação nos dois estudos de caso citados anteriormente, onde, além de comprovar o funcionamento da interface gráfica desenvolvida, será possível comparar as visualizações e as consultas (*queries SparQL*) de dados semânticos obtidos a partir de duas situações: primeiramente através do mapeamento padrão de cada uma das BDR utilizadas (*default* da ferramenta) e, em seguida, através da integração da ontologia *OWL* do respectivo domínio a estes mapeamentos. Os resultados obtidos nestes estudos de caso possibilitarão a conclusão do trabalho.

6. ESTÁGIO ATUAL DO TRABALHO

A revisão da literatura e os experimentos de avaliação das ferramentas encontradas foram concluídos, possibilitando a seleção daquelas que irão compor a versão inicial do *framework*, as quais são relacionadas na tabela 1.

As ontologias de domínio para cada um dos estudos de caso foram obtidas, assim como o método para integração destas ao mapeamento semântico da BDR foi elaborado, possibilitando o projeto, especificação e início do desenvolvimento da respectiva

[1] Lei 12.527, de 18 de novembro de 2011. Regula o acesso a informações previsto no inciso XXXIII do art. 5o, no inciso II do § 3º do art. 37 e no § 2º do art. 216 da Constituição Federal.

[2] §2º do art. 8º da Lei 12.527/11.

[3] Inciso III do §3º do art. 8º da Lei 12.527/11.

ferramenta. Neste momento, também encontra-se em desenvolvimento a interface gráfica do *framework*.

Tabela 1. Ferramentas e padrões selecionados para comporem a versão inicial do *framework*.

Ferramenta/Padrão	Objetivo
D2RQ Engine	Mapeamento, extração e conversão de dados estruturados (geração de dados semânticos / triplas *RDF*).
D2R-Server	Visualização e consulta, via console *Web*, de dados semânticos gerados a partir de dados estruturados.
Jena API	Interface com aplicações locais *Java* e Armazenamento de triplas *RDF* (*RDF store*).
OWLtoD2RQ-Mapping tool[4]	Ferramenta para integração de ontologia *OWL* ao arquivo de mapeamento semântico do BDR.
Turtle	Formato do arquivo de mapeamento semântico do BDR e da ontologia *OWL* a ser integrada.

7. COMPARAÇÃO COM TRABALHOS RELACIONADOS

O *LOD2 Stack*[5] [7] é um *framework*, desenvolvido por um consórcio de empresas, centros de pesquisas e universidades da Europa e da Ásia, que agrupa uma série de ferramentas para a publicação de Dados Abertos Ligados[6] (ou *Linked Open Data - LOD*), abrangendo as tarefas de extração, consulta e exploração, criação, descoberta semi-automática de *links* entre as fontes de dados e, enriquecimento e reparação de bases de conhecimento. Dentre as características principais do *LOD2 Stack*, também encontradas no *framework* proposto neste trabalho, estão a adoção da ferramenta *D2RQ* para a extração de dados semânticos a partir de bases de dados relacionais, bem como para a visualização e consulta (através de console *SparQL*) destes, e sua interação com o usuário através de interface gráfica. Entretanto, a aplicação não permite a integração de ontologias ao mapeamento semântico da base de dados relacional, e está disponível para instalação somente na plataforma *Linux*, devendo ainda ser acessada através de um *browser Web*, pois apesar de algumas das ferramentas serem instaladas localmente no usuário, as demais ferramentas oferecidas são aplicações *on-line* na *internet*, diferentemente do *framework* proposto neste trabalho, que será multiplataforma, com todas suas ferramentas instaladas localmente no usuário.

O *Neon Toolkit*[7] [8] é um *framework* para a construção e o gerenciamento de ontologias, desenvolvido por um consórcio de instituições europeias, que conta com uma grande quantidade de ferramentas e *plugins* para estas finalidades. Apesar de seu propósito ser diferente ao deste trabalho, o *Neon* conta com um *plugin* (*ODE Mapster*) [9] para o mapeamento (através de interface gráfica) entre uma ontologia e uma base de dados relacional, porém compatível somente com os SGBDs *MySQL* e *ORACLE*, além de gerar o arquivo de mapeamento semântico em linguagem não aderente ao padrão *W3C* (*R2O*), mas que servirá de referência para o desenvolvimento da aplicação que visa integrar ontologia ao arquivo de mapeamento semântico da base de dados relacional, proposta neste trabalho.

O *Virtuoso*[8] [10] é uma plataforma, desenvolvida pela *OpenLink Software*, que tem por objetivo integrar dados, serviços e processos de negócio através de uma arquitetura que possibilita, em um único sistema, oferecer serviços de gerenciamento e integração de dados (com suporte a dados abertos semânticos), de integração de aplicações (*Web* services e *SOA*) e de integração e gerenciamento de processos. Possui uma versão comercial e uma *open-source* (com várias limitações). Possibilita a publicação de dados abertos gerados a partir do SGBD interno do produto (em ambas as versões) ou de um SGBD externo (somente na versão comercial).

8. CONCLUSÕES

Este trabalho vem preencher uma lacuna encontrada nas ferramentas disponíveis para a geração e publicação de dados semânticos a partir de dados estruturados persistidos em bases de dados relacionais, que é a falta de uma interface gráfica que automatize todo o processo e integre as diferentes ferramentas necessárias, bem como gerar as triplas *RDF* com base em uma ontologia, de forma a conferir maior expressividade e poder de inferência às visualizações e consultas desses dados semânticos.

O método para integração de ontologia *OWL* ao mapeamento semântico da BDR, nos testes efetuados manualmente, mostrou-se eficaz no sentido de tornar mais amigável para o usuário a visualização das triplas *RDF*, ao rotular sujeito, predicado e objeto de cada tripla com o respectivo termo da ontologia, o que facilita a formulação das *queries SparQL*.

A conclusão do desenvolvimento da interface gráfica e da ferramenta de integração de ontologia ao mapeamento semântico do BDR, e a implementação dos estudos de caso, permitirão aferir se o *framework* atingiu os objetivos e contribuições esperadas.

9. REFERÊNCIAS

[1] Berners-Lee, T.; Hendler, J.; Lassila, O. The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American vol. 284, Mai. 2001, pg. 28-37.

[2] Machado, A. L; Parente de Oliveira, J. M. DIGO: An Open Data Architecture for e-Government. In: 15th IEEE International Enterprise Distributed Object Computing Conference, 2011, Helsinki. 15th IEEE International Enterprise Distributed Object Computing Conference - Workshop Proceedings. IEEE Computer Society Press, 2011.

[3] Antoniou, G.; Harmelen, F. A Semantic Web Primer. 2ed. Cambridge: MIT Press, 2008.

[4] Guizzardi, G. Ontological Foundations for Structural Conceptual Models. Telematica Instituut Fundamental Research Series No. 015 (TI/FRS/015). Enschede: Universal Press, 2005.

[4] Nome provisório da ferramenta, em desenvolvimento pelo autor.

[5] http://demo.lod2.eu/lod2demo

[6] Dados abertos ligados refere-se à criação de *links* entre uma fonte de dados abertos e outras fontes disponíveis na *internet*, de forma a permitir que sejam combinadas e produzam novas informações e aplicações.

[7] http://neon-toolkit.org/

[8] http://virtuoso.openlinksw.com/

[5] W3C (e-Gov). eGovernment at W3C: improving access to government through better use of the Web, Online 2009. Disponível em: <http://www.w3.org/2007/eGov>.

[6] Cartilha Acesso à Informação Pública: Uma introdução à Lei nº 12.527, de 18 de novembro de 2011. Brasília: Controladoria Geral da União, 2011. Disponível em: <http://www.acessoainformacao.gov.br/acessoainformacao gov/publicacoes/CartilhaAcessoaInformacao.pdf.>

[7] Auer, S. et al. Managing the Life-Cycle of Linked Data with the LOD2 Stack. Disponível em: <http://lod2.eu/Blog Post/1214-paper-about-lod2-stack-accepted-for-iswc.html>.

[8] Haase, P. et al. Ontology engineering and plugin development with the neon toolkit. In: The 6th International Semantic Web Conference. ISWC 2007 Tutorial. Busan, Korea: Semantic Web Science Association, nov. 2007.

[9] Rodriguez, J. B.; Gómez-Pérez, A. Upgrading relational legacy data to the semantic web. Proceedings of the 15th international conference on World Wide Web. Anais...: WWW '06. New York, NY, USA: ACM, 2006. Disponível em: <http://doi.acm.org/10.1145/1135777. 1136019>.

[10] Erling, O.; Mikhailov, I. Virtuoso: RDF Support in a Native RDBMS. In: De Virgilio, R.; Giunchiglia, F.; Tanca, L. (Eds.). Semantic Web Information Management. Berlin, Heidelberg: Springer, Berlin Heidelberg, 2010. p. 501-519.

Métricas de Análise de Links e Qualidade de Conteúdo: um estudo de caso na Wikipédia

Raíza Hanada*
Universidade de São Paulo
Av. do Trabalhador
São-Carlense, 400
São Carlos, Brasil
rhanada@icmc.usp.br

Marco Cristo†
Universidade Federal do Amazonas
Av. Rodrigo Otavio, 6200
Manaus, Brasil
marco.cristo@icomp.ufam.edu.br

Maria da Graça Campos Pimentel‡
Universidade de São Paulo
Av. do Trabalhador
São-Carlense, 400
São Carlos, Brasil
mgp@icmc.usp.br

ABSTRACT

Muitos apontamentos (links) entre páginas Web podem ser vistos como indicativos da qualidade e/ou importância da página apontada. Apoiando-se nessa ideia, diversas métricas baseadas em links foram propostas na literatura para identificar conteúdo de qualidade na Web. Vários métodos de avaliação demonstram que estas métricas são bem sucedidas na tarefa de ordenação(ranking) das páginas de resposta a consultas submetidas a máquinas de busca. Apesar disso, não é possível determinar qual a contribuição específica de fatores como qualidade, importância ou popularidade para o resultado obtido. Essa dificuldade se deve, em parte, ao fato de que tais informações não são de fácil obtenção para páginas da Web em geral. Diferentemente de páginas Web comuns, artigos da Wikipédia são avaliados por seres humanos segundo a sua qualidade, utilizando critérios estabelecidos previamente. As notas de qualidade dos artigos da Wikipédia estão disponíveis aos leitores. Com posse dessa informação, o objetivo deste trabalho é verificar a relação existente entre as métricas de análise de links e as notas de qualidade atribuídas aos artigos da Wikipédia. Para atingir este objetivo, a pesquisa foi dividida em etapas as quais serão relatadas.

Categories and Subject Descriptors

H.4 [**Sistemas Web**]: Recuperação de Informação, Análise de Links, Wikipédia

General Terms

Métricas de Análise de Links[qualidade de conteúdo]

Keywords

Qualidade, Wikipédia, Métricas de Análise de Links

*Estudante do programa de mestrado do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo, com início em março de 2011 e defesa prevista para fevereiro de 2013

†Colaborador do Trabalho de Pesquisa

‡Orientadora do Trabalho de Pesquisa

1. INTRODUÇÃO

1.1 Contexto Teórico

A Web se expandiu ao longo das últimas décadas, se constituindo em uma enorme coleção de documentos, espalhados de forma descentralizada e desorganizada. Apesar disso, foram desenvolvidas ferramentas capazes de encontrar informações nesta coleção de forma bem sucedida. Tais ferramentas são as máquinas de busca. Elas permitem que um conjunto de páginas sejam buscadas de acordo com a sua relevância para um determinado conjunto de termos. O valor de relevância é atribuído por um algoritmo de *ranking*. Diversas evidências são combinadas pelos algoritmos de ranking para estipular valores de relevância. Dentre estas várias evidências estão as métricas que se baseiam em links. Neste caso, a Web pode ser vista como um grafo onde os nós são as páginas e as arestas são os hiperlinks. O estudo das propriedades e das relações nesse grafo chama-se Análise de Links.

Em análise de links, quando um autor de uma página cria um link para outra página, ele endossa o conteúdo da segunda, sugerindo que este é de qualidade e que a página apontada pode ser uma autoridade em um determinado assunto [9]. Outros fatores que contribuem para que uma página seja muito citada são a sua visibilidade e a sua popularidade [5], levando a ideia de que quanto mais uma página é apontada por outras maior é a sua relevância. Como consequência, páginas muito citadas obtêm posição de destaque em rankings de máquinas de busca.

Na literatura, são reportadas diversas métricas (e variações delas) para ordenar páginas aproveitando-se dos conceitos de análise de links [2]. O resultado desses métodos é avaliado por seres humanos que consideram o quão relevante são as páginas retornadas em relação aos termos dados. Uma vez que não se sabe determinar exatamente qual o grau de qualidade, de popularidade e de visibilidade de certa página,

não é possível verificar o impacto de cada um destes fatores para a relevância da mesma [1]. Porém, ao contrário de páginas da Web em geral, artigos da Wikipédia são explicitamente avaliados pela sua comunidade quanto à sua qualidade de conteúdo, o que fez com que muitos trabalhos sobre qualidade de páginas envolvendo a Wikipédia fossem publicados ([7], [13], [6]). Contudo, nenhum deles investigou, utilizando links internos e externos, em que grau métricas de análise de links são capazes de ordenar páginas da Wikipédia de acordo com a sua qualidade de conteúdo explicitamente indicada no processo de revisão humana.

1.2 Identificação do Problema

Existem na literatura vários trabalhos que consideram métricas baseadas em links para avaliar e ordenar páginas de acordo com a sua relevância [2]. Tais métricas foram elaboradas com base na hipótese de que links entre páginas podem indicar, entre outros fatores, maior ou menor qualidade de conteúdo [1]. Embora comum, até onde sabemos, esta hipótese não foi verificada adequadamente de forma a quantificar especificamente a importância do fator qualidade para os resultados obtidos pelas métricas de análise de links. Entre outros motivos, isso se deve ao fato de que páginas Web não possuem avaliação explícita de qualidade. Artigos da Wikipédia, entretanto, possuem avaliações de qualidade, o que nos permite determinar esta correlação. Contudo, artigos da Wikipédia podem possuir diferenças em relação a outras páginas Web, causando impacto no resultado de algumas métricas. A diferença existente entre o grafo de links da Web e da Wikipédia também deve ser estudada.

1.3 Objetivos

Este trabalho tem por objetivo investigar, com base na Wikipédia, qual é a relação existente entre a qualidade de conteúdo e os resultados obtidos pelas métricas de análise de links, utilizadas para avaliar e ordenar páginas de acordo com a sua relevância, e determinar como as conclusões observadas para Wikipédia podem ser estendidas para a Web em geral.

1.4 Contribuições Esperadas

Com este trabalho, esperamos fornecer as seguintes contribuições:

1. Verificar a correlação entre métricas de análise de links e qualidade de conteúdo através de um estudo em larga escala. Até onde sabemos, somente um trabalho analisou esta correlação [1]. Este, contudo, consistiu de um estudo limitado envolvendo poucas páginas, tópicos e avaliadores, o que motivou seus autores a afirmarem que um novo estudo, em maior escala, seria necessário para resultados mais conclusivos.

2. Possibilitar a criação de novas técnicas de inferência automática da qualidade de conteúdo. Esta é uma área de grande importância, dado seu impacto em várias outras aplicações como busca e recomendação. No caso particular da Wikipédia, a inferência automática de conteúdo também é desejável para a automatização de um processo essencialmente manual. Muitos trabalhos têm investigado esse problema, como por exemplo [16], [10] e [7].

3. Possibilitar a criação de novos algoritmos de ordenação em máquinas de busca através de uma melhor compreensão do impacto da qualidade em métricas de análise de links.

1.5 Organização

Este trabalho está organizado como descrito a seguir. A seção 2 aborda trabalhos relacionados a esta pesquisa e que deram conhecimento e suporte para a formulação dos procedimentos a serem seguidos e observados durante a resolução do problema; na Seção 3, são detalhadas as atividades que deverão ser realizadas para a execução do projeto; e, por fim, a Seção 4 apresenta quais atividades foram atingidas até o presente momento.

2. TRABALHOS RELACIONADOS

Durante o processo de revisão bibliográfica, foram encontrados trabalhos publicados na literatura que estão, de certa forma, relacionados ao trabalho proposto.

O trabalho considerado mais relevante para esta pesquisa é o realizado por Amento et. al (2000). Assim como desejamos fazer, os autores analisam a relação entre métricas de análise de links e qualidade de páginas. O principal problema encontrado neste trabalho é o pequeno número de especialistas, páginas e tópicos e as diferenças de opinião dos especialistas que dificultou a obtenção de resultados estatisticamente significativos. Ao usar a Wikipédia, poderemos confirmar seus resultados, pois possuiremos mais artigos revisados, cobrindo tópicos em um número maior de domínios e obtendo resultados estatísticamente significativos. Além disso, pretendemos considerar o impacto de outros fatores como popularidade e importância, que também podem ser obtidos de forma independente na Wikipédia.

Considerando que links na Wikipédia podem divergir de links na Web em geral, se assemelhando mais a citações em artigos cientificos, torna-se importante o trabalho de Smith et. al (2004), que aborda esta questão. Os autores concluem que citações e links mostraram-se semelhantes em apenas 20% da Web. Nesta mesma linha, Gleich et. al (2012), aplicaram o método PageRank sobre o grafo interno da Wikipédia, variando os valore do seu parâmetro de teletransporte (*coeficiente de dampenning*) e observaram que os rankings obtidos eram muito diferentes do esperado, levantando uma discussão sobre a diferença entre o grafo de links da Wikipédia e a Web. Finalmente, Kamps et. al (2009) realizaram um estudo e concluíram que a Wikipédia divergia da estrutura da Web em aspectos como a densidade de links, utilização de inlinks e outlinks para determinar importância da página, e utilização de evidências de links globais para realização de buscas. A discussão sobre as possíveis diferenças entre a Wikipédia e a Web são importantes, uma vez que desejamos saber o quanto é possível estender conclusões obtidas na Wikipédia para a Web em geral. Dessa forma, pretendemos realizar uma comparação da estrutura de links da Wikipédia com a nossa amostra da Web, similar à realizada por Kamps et. al (2009).

Outros trabalhos estudam o problema de estimativa automática de qualidade na Wikipédia, sugerindo o uso de informação de análise de links para esta tarefa (e.g. [6], [7] e [13]). Em todos os casos apresentados, os links usados

foram restritos à própria Wikipédia. Diferente destes trabalhos, pretendemos fazer uso de links internos e externos à Wikipédia, assim estaremos utilizando links independentes que não são usados com natureza enciclopédica e uma ampla vizinhança de páginas, necessárias para algumas métricas. Outra diferença é que nosso objetivo não é analisar métricas de links fora do contexto de previsão de qualidade na Wikipédia.

O trabalho apresentado por Berlt et. al (2010) faz adaptações de métricas agrupando páginas da Web por host e domínio e concluindo que todos os métodos apresentados tiveram desempenho melhor ou igual ao PageRank e ao Indegree tradicional com a vantagem adicional de serem menos sucestíveis a spam. Este trabalho está relacionado ao nosso estudo por apresentar adaptações dos métodos tradicionais que nós também pretendemos usar. Além disso, assim como faremos, Berlt et. al (2010) usam uma coleção derivada do domínio *br* como amostra da Web.

Finalmente, é importante verificar como fatores, tais como tópicos, influenciam no desempenho de métricas de análise de links no contexto da Wikipédia. Yamada et. al (2006) mostram que o comportamento dos links entre os artigos varia muito de acordo com a categoria ao qual eles pertencem. Assim, é interessante estudar como tais métricas se relacionam com qualidade quando artigos de diferentes áreas são considerados.

3. METODOLOGIA

O trabalho a ser realizado foi dividido nas etapas que estão descritas a seguir.

3.1 Estudo sobre a coleção e dados de interesse

O objetivo desta atividade é determinar quais dados são relevantes e devem ser capturados. Para a realização deste trabalho, é necessário uma amostra da Web com páginas que foram avaliadas quanto a sua qualidade (artigos da Wikipédia, neste trabalho). As bases obtidas foram as seguintes:

1. WBR10: base coletada no contexto do projeto INWEB[1], composta por documentos do domínio *br*. Essa coleção foi obtida em 2010 e contém 125 milhões de documentos HTML, com 6,8 bilhões de links entre eles, o que corresponde à maior amostra da Web brasileira disponível para estudo.

2. Ptwiki dump progress de 20100804: disponibilizada para o público pela Wikipédia no formato XML e diversos dumps SQL. A coleção corresponde ao mesmo período de coleta da base WBR10 e possui 2.363.341 páginas e 20.311.293 revisões.

Optamos por uma amostra da Wikipédia em português, visto que é provável que editores de páginas Web brasileiras citem mais artigos da Wikipédia em língua portuguesa do que em outros idiomas. Deve-se observar que os artigos da Wikipédia pertencem ao domínio *org* e, portanto, não fazem parte da coleção WBR10. Para a execução dos experimentos é necessária a união da coleção WBR10 com os artigos da Wikipédia.

Nesta etapa, desejamos verificar quais informações devem ser extraídas das coleções. Da amostra da Web WBR10, em princípio, estamos interessados no grafo para análise de links, podendo este ser enriquecido com informação de texto de âncora. Da amostra da Wikipédia, estamos interessados na qualidade do artigo, na popularidade (métricas de análise de links dão alta pontuação para páginas populares, devendo esse efeito ser considerado), e na categoria do artigo (útil para verificar o quanto a análise realizada vale para diferentes tópicos). Outras evidências podem ser consideradas de interesse para a pesquisa.

3.2 Construção das bases e extração dos dados

Esta etapa consiste na implementação dos extratores necessários para obter os dados de interesse, descritos anteriormente. Na Wikipédia, serão eliminados artigos que não possuem nota de qualidade e que não são citados nas páginas da WBR10 ou não citam páginas da WBR10. As informações necessárias serão extraídas por meio de consultas em SQL e depois combinadas às informações da coleção WBR10. Para obter o grafo de links final, serão utilizados ferramentas disponibilizadas com a própria coleção WBR10.

3.3 Comparação sistemática da Web e da amostra da Wikipédia

Como o objetivo deste trabalho é estabelecer a relação entre qualidade de conteúdo e métricas de análise de links na Web, é necessário determinar até que ponto a Wikipédia é representativa da Web. É importante observar que assumimos que a coleção WBR10 é representativa da Web. Baseados em trabalhos na literatura, sabemos que a Wikipédia possui diferenças em relação a amostras da Web, sendo importante determinar de forma sistemática como estas coleções (WBR10 e Wikipédia) se assemelham em termos da sua estrutura de ligações. Em particular, a comparação deve avaliar, pelo menos, características dos grafos de páginas, como a densidade de links e as distribuições de inlinks e outlinks.

3.4 Desenvolvimento e aplicação das métricas de ranking

Serão implementadas métricas encontradas na literatura baseadas em grafos (Indegree e Outdegree [4]) e baseadas em links (PageRank [11] e HITS [9]). Da mesma forma que foi feito por Berlt et. al (2010), as métricas serão usadas considerando páginas isoladas ou agrupadas em hosts e domínios. Será usada a mesma definição de hosts e domínios usada por Berlt et al. (2010). As métricas serão implementadas provavelmente utilizando as linguagens de programação C e/ou C++.

As métricas implementadas serão aplicadas sobre o grafo de páginas que integra as páginas da WBR10 e da Wikipédia. A aplicação de cada métrica implementada resultará em uma pontuação para cada artigo da Wikipédia que determinará a posição do artigo no ranking, permitindo a sua

[1] http://www.inWeb.org.br

comparação com a posição que teria no ranking baseado em sua qualidade.

3.5 Análise dos resultados e considerações finais

Para verificar a relação entre as métricas de análise de links e qualidade, usaremos uma estratégia de comparação de rankings. A ideia consiste em se obter rankings dos artigos da Wikipédia de acordo com cada métrica estudada e de acordo com a sua qualidade e aplicar métricas tradicionais de análise de ranking (Pearson [12] e Kendall Tau [8]). As métricas também serão comparadas entre si, de forma a permitir determinar qual a mais apropriada para o ambiente da Wikipédia. Além disso, as verificações deverão considerar os outros fatores que podem influenciar esta análise, como a popularidade e o tópico dos artigos, necessitando que outras comparações sejam realizadas.

Para o caso das métricas influenciadas pelo grafo interno da Wikipédia, as conclusões serão ponderadas de acordo com o estudo anterior sobre a similaridade entre a Wikipédia e a a Web. Em todas as análises, utilizaremos métodos estatísticos para garantir que conclusão serão tiradas apenas de resultados significativos em termos estatísticos.

4. CONTRIBUIÇÕES: ESTADO ATUAL DO TRABALHO

Inicialmente, foi feito um estudo preliminar no qual foram levantadas quais pesquisas estão sendo realizadas sobre a Wikipédia. A partir deste levantamento pudemos fundamentar esta pesquisa e encontrar trabalhos relacionados a ela.

O próximo passo foi obter as bases necessárias para a realização dessa pesquisa. Em parceria, com o grupo de pesquisa BDRI (Banco de Dados e Recuperacao de Informacao), da Universidade Federal do Amazonas, tivemos acesso à base WBR10 que contém informações sobre páginas com domínio *br*. Devido ao fato de a Wikipédia manter suas informações abertas, pudemos adquirir uma amostra da Wikipédia no mesmo período de coleta da WBR10 (Ptwiki dump progress de 20100804), sendo essa formada por arquivos XML e dumps SQL.

A amostra da Wikipédia possui arquivos em XML e dumps SQL. Foram realizadas operações para que pudéssemos possuir as informações da amostra em um banco de dados de fácil acesso. Na base WBR10, as informações estão armazenadas em imensos arquivos textuais. Está em andamento a implementação dos algoritmos pra manipulação dos dados existentes nas coleções escolhidas. Uma vez que todos os dados estierem em formatos fáceis de serem acessados, é possível fazer a combinação das duas bases para obter o novo grafo de links e ter acesso às informações importantes da Wikipédia.

5. REFERENCES

[1] B. Amento, L. Terveen, and W. Hill. Does authority mean quality? predicting expert quality ratings of web documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '00, 2000.

[2] R. Baeza-Yates, P. Boldi, and C. Castillo. Generalizing pagerank: damping functions for link-based ranking algorithms. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 308–315, New York, NY, USA, 2006. ACM.

[3] K. Berlt, E. S. de Moura, A. Carvalho, M. Cristo, N. Ziviani, and T. Couto. Modeling the web as a hypergraph to compute page reputation. *Inf. Syst.*, 35:530–543, July 2010.

[4] T. Bray. Measuring the web. *Proceedings of the 5th International World Wide Web Conference on Computer Networks and ISDN Systems*, page 993Ú1005, 1996.

[5] J. Cho and S. Roy. Impact of search engines on page popularity. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 20–29, New York, NY, USA, 2004. ACM.

[6] D. H. Dalip, M. A. Gonçalves, M. Cristo, and P. Calado. Automatic assessment of document quality in web collaborative digital libraries. *J. Data and Information Quality*, 2(3):14:1–14:30, Dec. 2011.

[7] P. Dondio and S. Barrett. Computational Trust in Web Content Quality: A Comparative Evalutation on the Wikipédia Project. *Informatica*, 31(2):151–160, 2007.

[8] M. Kendall. *Rank Correlation Methods*. Hafner Publishing Co, New York, 1955.

[9] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *JOURNAL OF THE ACM*, 46(5):604–632, 1999.

[10] E.-P. Lim, B.-Q. Vuong, H. W. Lauw, and A. Sun. Measuring qualities of articles contributed by online communities. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '06, pages 81–87, Washington, DC, USA, 2006. IEEE Computer Society.

[11] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web, 1999.

[12] K. Pearson. The grammar o science. J. M. Dent and Company, 1892.

[13] L. Rassbach, T. Pincock, and B. Mingus. Exploring the Feasibility of Automatically Rating Online Article Quality, 2008.

[14] A. G. Smith. Web link as analogues of citations. *School of Information Management*, 2004.

[15] K. S. T. Yamada and K. Kazama. Network analyses to understand the structure of wikipédia. *In Proc. of AISB'06*, 3:195–198, 2006.

[16] B. S. M. B. Twidale. Assessing information quality of a community-based encyclopedia. In *In Proceedings of the International Conference on Information Quality*, pages 442–454, 2005.

Os Limites das Folksonomias como Conceitualizações Compartilhadas na Especificação de Modelos Conceituais

Josiane M. P. Ferreira, Cesar Augusto Tacla
Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Sete de setembro 3165, CEP 80230-901
Curitiba – PR – Brasil
+55 44 3310-4685
josianempf@gmail.com, tacla@utfpr.edu.br

Sérgio Roberto P. da Silva
Universidade Estadual de Maringá
Av. Colombo, 5790, CEP 87020-900
Maringá – PR – Brasil
+55 44 3011-4076
sergio.r.dasilva@gmail.com

ABSTRACT
Looking to reduce the problem of knowledge acquisition bottleneck, this work takes on the hypothesis that the folksonomy induced from collaborative tagging data on the Web, based on parameters of authorship and motivation of categorization, can represent a shared conceptualization of a domain. Thus, it is expected that the use of such data can generate a reduction of divergences in the elicitation of terms that will be part of the conceptual model when compared with folksonomy induction algorithms that do not use these parameters.

Categories and Subject Descriptors
I.2.6 [**Artificial Intelligence**]: Learning – *Knowledge acquisition.* H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *Information filtering.*

General Terms
Algorithms, Design, Experimentation.

Keywords
Conceptual Models, Knowledge Acquisition, Collaborative *Tag*ging Systems, Folksonomies.

1. CONTEXTO TEÓRICO
Guizzardi [6] adota o nome conceitualização para designar o conjunto de conceitos utilizados para articular abstrações do estado das coisas em um domínio. Modelo é uma abstração de uma porção da realidade articulada segundo uma conceitualização de um domínio. Ainda, para Guizzardi [6], tanto conceitualizações como modelos existem somente nas mentes das pessoas. O que há de concreto são **especificações de modelos conceituais** feitas em uma linguagem de modelagem que permitem expressar (representar) conceitualizações. Desta forma, a especificação do modelo conceitual – denominada de modelo conceitual, é um artefato concreto que permite aos atores envolvidos no processo de construção do modelo compreender o domínio, **atingir consenso** sobre o significado das entidades representadas e se comunicar.

Na passagem das conceitualizações e modelos abstratos para modelos concretos ocorre o problema descrito por Feigenbaum [4] denominado de **gargalo de aquisição de conhecimentos** que diz respeito à dificuldade que os engenheiros de conhecimentos têm em capturar e representar conhecimentos a partir de interações com especialistas. Entretanto, nos dias de hoje, além dos especialistas, existem outras fontes de informação que podem ser utilizadas no processo de especificação do modelo conceitual, como, por exemplo, a *Web*. O número de atores envolvidos (engenheiros de conhecimento, especialistas no domínio e usuários) no processo de especificação do modelo conceitual também pode ser maior [18].

Realizar aquisição de conhecimentos em larga escala é um processo demorado e custoso. Atingir consenso com um número elevado de atores torna-se difícil, pois aumentam as divergências, assim como o número de interações para resolvê-la. Há abordagens de aprendizado de ontologias que se utilizam de métodos e técnicas de processamento de linguagem natural, aprendizado de máquina e mineração de textos para extrair conceitos, relações e instâncias de fontes de informação processáveis (ex. esquemas de bancos de dados, textos, etc.) [11]. Algumas abordagens de aprendizado de ontologias atuais têm utilizado dados dos sistemas baseados em ***tagging* colaborativo** como fonte de informação para estes algoritmos.

Sistemas de *tagging* colaborativo são aplicações ditas sociais que permitem aos seus usuários atribuírem etiquetas (*tags*) a recursos da *Web*. Um recurso pode ser etiquetado por vários usuários com quantas e quais *tags* eles acharem convenientes. O fato interessante é que, apesar de não existir um vocabulário controlado, depois de certo tempo as *tags* utilizadas pelos usuários para etiquetar um recurso parecem se estabilizar [14]. Ao associarem as mesmas *tags* aos mesmos recursos, os usuários constroem um **vocabulário consensual** para um determinado conjunto de recursos que pode ser representativo em um domínio, como mencionado por vários autores [1, 8, 11, 13], e pode ser visto como uma forma simples de **conceitualização compartilhada** na forma de uma lista de termos (*tags*, neste caso). Por isso, várias abordagens [1, 2, 3, 6, 8, 11, 13, 14, 15, 18] utilizam estes dados como entrada para construir algum tipo de estrutura "consensual" (vocabulário compartilhado, taxonomia, ontologia) dos dados do *tagging*.

Alguns autores chamam os dados do *tagging* colaborativo de folksonomia. Neste trabalho, o termo folksonomia será utilizado para designar a estrutura coletiva consensual (vocabulário compartilhado, taxonomia, ontologia) que emerge do *tagging* colaborativo por meio de um algoritmo de indução de folksonomias [17].

2. IDENTIFICAÇÃO DO PROBLEMA
Várias das abordagens citadas que utilizam os dados do *tagging* para derivar uma estrutura consensual de um conjunto de recursos pressupõem que estes dados ajudam no desenvolvimento de modelos conceituais de consenso pelo simples fato de resultarem

de um processo humano e coletivo sem, no entanto, verificar com profundidade a natureza do conhecimento existente no *tagging*. A maioria delas procura avaliar a abordagem, ou o algoritmo utilizado que induz a estrutura consensual dos dados de *tagging*, sem, no entanto, avaliar a utilidade da estrutura derivada – se ela realmente representa um consenso; ou as características dos dados de entrada – se eles são mais interessantes do que outros conjuntos de termos para determinada tarefa. A questão principal aqui é se as folksonomias que emergem dos dados do *tagging* colaborativo são realmente consensuais, já que as abordagens que as utilizam não verificam isso na prática. Será que os dados do *tagging*, por possuírem a dimensão social, são mais úteis do que um conhecimento extraído automaticamente de textos por meio de algoritmos de extração de termos, por exemplo? Será que a forma na qual a folksonomia é induzida dos dados de *tagging* (os parâmetros utilizados) determina sua utilidade – grau de consenso – na construção de modelos conceituais?

3. OBJETIVO

O objetivo deste trabalho é buscar evidências, por meio de experimentos de modelagem conceitual, de que estruturas que emergem da dimensão social do *tagging*, de acordo com alguns parâmetros, atenuam o gargalo de aquisição que ocorre na especificação de modelos conceituais de domínios, por representarem uma conceitualização compartilhada em uma comunidade de usuários.

Especificamente, pretende-se construir um algoritmo que leve em conta informações de autoria das *tags* (o conhecimento de quem a fez) e de motivação de etiquetagem (para quê a fez) e avaliar se as folksonomias produzidas com base nestes parâmetros realmente auxiliam a atenuar o gargalo da aquisição de conhecimento na construção de modelos conceituais. Espera-se que a utilização destas folksonomias provoque a diminuição de divergências na elicitação de termos que farão parte de um modelo conceitual em comparação com algoritmos de indução de folksonomias que não utilizam estes parâmetros e, também, com algoritmos baseados em técnicas tradicionais de recuperação de informações (ex. *TF-IDF*).

4. CONTRIBUIÇÕES ESPERADAS

As contribuições desta proposta interessam aos pesquisadores que lidam com modelagem conceitual, em particular, com a atenuação do gargalo de aquisição de conhecimentos na modelagem conceitual, bem como no entendimento da utilização e dos limites de uso das folksonomias como fonte de informação na modelagem conceitual. Particularmente, propõe-se melhorar os algoritmos de indução de folksonomias pelo uso de autoria (autoridade cognitiva) e motivação das etique*tag*ens.

5. METODOLOGIA ADOTADA

Para determinar se as folksonomias induzidas podem realmente ser consideradas como conceitualizações compartilhadas em um domínio, pretende-se realizar uma série de experimentos para a construção colaborativa de modelos conceituais. Parte-se do princípio de que o grupo que utilizar como entrada a folksonomia induzida que realmente represente consenso sobre determinado domínio irá se deparar com um número menor de divergências durante um experimento de modelagem do que os grupos que utilizam outros dados de entrada.

Para avaliar o grau de divergência na modelagem conceitual, será utilizado o método *CoFolkconcept* [8]. O processo de modelagem no *CoFolkoncept* é colaborativo e se desenvolve da seguinte maneira: i) cada usuário constrói um modelo conceitual individualmente utilizando-se de um conjunto de *tags*/termos produzindo, desta forma, um modelo conceitual particular; ii) os diferentes modelos conceituais de cada usuário são comparados a fim de se detectar divergências nas *tags*/termos escolhidos por cada usuário quanto ao tipo (conceito, instância ou relação) e à posição taxonômica (quando forem conceito ou instância); iii) resolvem-se as divergências por meio de discussões estruturadas de acordo com a metodologia DILIGENT [18]; iv) gera-se uma nova versão do modelo conceitual que é consensual e repete-se o processo modificando-se individualmente o modelo consensual.

Para selecionar as *tags* sobre o domínio de interesse dos dados de *tagging* propõem-se um algoritmo que leve em conta informações de autoria e de motivação de etiquetagem, como mostra a Figura 1.

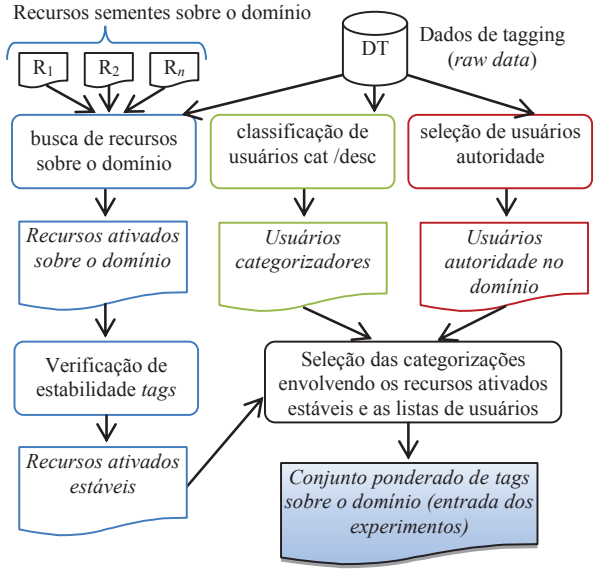


Figura 1: Algoritmo proposto para seleção do conjunto de *tags* que será utilizado como entrada dos experimentos.

Para selecionar os recursos pertinentes ao domínio de interesse em um sistema de *tagging* que permite a categorização de recursos sobre os mais diversos assuntos será utilizada a abordagem de García-Silva *et al.* [5]. Os autores desenvolveram um algoritmo que busca recursos sobre um domínio nos dados de *tagging* partindo de um conjunto de recursos sementes. As categorizações são modeladas na forma de um grafo não-direcionado no qual os vértices são os recursos e as arestas são *tags* em comum entre os dois recursos. Desta forma, dois recursos são adjacentes somente se eles contem ao menos uma *tag* em comum (independentemente de qual usuário fez a categorização). Partindo-se dos recursos sementes sobre o domínio a busca é feita em largura no grafo, com uma abordagem de *spreading activation* na qual a ativação de um recurso depende de quantas *tags* ele tem em comum com o recurso atual e do grau de ativação do recurso atual. Somente recursos com grau de ativação acima de um limiar serão visitados e considerados ativados. Outro ponto importante é verificar se o conjunto de *tags* utilizadas para categorizar estes recursos é estável, ou seja, é consensual. A abordagem descrita por Robu *et al.* [14] será utilizada para esta verificação.

Quanto à **motivação** do usuário ao realizar uma etiquetagem, pressupõe-se que ela pode ser reveladora do significado

pretendido para a *tag*, o que é importante no momento de se construir um modelo conceitual. Defende-se a ideia de que a motivação para criar uma *tag* tem influência no seu uso (ou não) durante a criação de um modelo conceitual. Körner *et al*. [9] abordam a motivação dos usuários durante a etiquetagem e tentam identificá-la automaticamente separando-as em dois grandes grupos: usuários categorizadores e usuários descritores de recursos. No conjunto de *tags* dos usuários categorizadores de recursos, há pouco uso de sinônimos (o que deve facilitar o consenso entre os atores envolvidos na especificação do modelo conceitual) e a estrutura induzida dos dados de *tagging* se aproxima de uma taxonomia. Por outro lado, no conjunto de *tags* dos usuários descritores de recursos, há uso mais proeminente de sinônimos e o vocabulário é, portanto, frequentemente maior, dificultando o consenso na especificação do modelo conceitual. Por isso, com base nas conclusões do autor, pretende-se, inicialmente classificar os usuários em categorizadores e descritores, e utilizar as *tags* dos usuários categorizadores, pois elas devem facilitar o consenso durante os experimentos de modelagem.

Quanto ao uso da autoria das *tags*, segundo Wilson [12], entidades consideradas autoridades em determinado assunto tendem a organizar melhor suas informações, possuírem conteúdos de qualidade, e manterem contato com pessoas que entendam ou tenham interesse no mesmo assunto. O autor define o conceito de **autoridade cognitiva** – uma autoridade fundamentada na competência e nas capacidades intelectuais de quem a recebe e cuja concessão é compreendida como o reconhecimento e o mérito por estas capacidades – uma autoridade que define "quem sabe o quê sobre o quê". Pressupõe-se, portanto, que pessoas que conhecem melhor determinado assunto tendem a organizar melhor as suas *tags*. Por isso, outro aspecto a ser considerado pelo algoritmo proposto é saber se quem realizou a etiquetagem possui autoridade no domínio de interesse.

Pretende-se avaliar se as folksonomias produzidas com base nestes parâmetros realmente auxiliam a atenuar o gargalo da aquisição de conhecimento na construção de modelos conceituais. Serão realizados experimentos com diferentes parâmetros de geração das folksonomias a fim de determinar em quais condições elas podem ser consideradas como conceitualizações compartilhadas.

Para fins de comparação, um algoritmo de indução de folksonomias, que não utiliza as informações de origem e de motivação, servirá de referência na avaliação (em princípio, será implementado o algoritmo de Hamasaki [7] – que utiliza a autoria das *tags*, mas não o conceito de autoridade). Um segundo algoritmo de controle fundamentado na técnica *TF-IDF* também fornecerá dados para comparação. Os três algoritmos (o proposto e os dois de comparação) utilizarão o mesmo conjunto de anotações como entrada e produzirão um conjunto de termos como saída, que será utilizado nos experimentos. O algoritmo baseado em *TF-IDF* gerará um conjunto de termos a partir das *URLs* encontradas nas mesmas anotações dos algoritmos de indução de folksonomias. No caso dos algoritmos de indução de folksonomias este conjunto de termos representa uma folksonomia em uma estrutura plana.

Os experimentos serão realizados com três grupos: o grupo de teste, que utilizará a folksonomia obtida pelo algoritmo proposto; o grupo de controle I, que utilizará a folksonomia obtida pelo algoritmo de Hamasaki; e o grupo de controle II, que utilizará o conjunto de termos obtido pelo algoritmo de *TF-IDF*. Espera-se que o grupo de teste se depare com um número menor de divergências, durante o processo de construção colaborativa do modelo conceitual, do que os grupos de controle.

6. ESTÁGIO ATUAL DO TRABALHO

Após a revisão de literatura e a especificação do modelo proposto, o algoritmo proposto e os algoritmos de comparação estão sendo implementados para gerar os conjuntos de termos/*tags* que devem ser utilizados nos experimentos iniciais. Para isso, alguns problemas de eficiência ao lidar com grandes bases de dados estão sendo resolvidos. A base de dados utilizada para testar os algoritmos possui aproximadamente 3,7 milhões de triplas usuário-*tag*-recurso, o que implica que o acesso a estes dados deve ser feito de forma muito eficiente para não se tornar um gargalo.

7. COMPARAÇÃO COM TRABALHOS RELACIONADOS

Como já citado, várias abordagens [1, 2, 3, 6, 8, 11, 13, 14, 15, 16, 18] também implementam algoritmos que induzem algum tipo de estrutura dos dados de *tagging*. Inclusive algumas delas derivam taxonomias ou ontologias leves [1, 3, 15, 16], muito próximas de modelos conceituais. O fato é que várias delas consideram a estrutura derivada como consenso [1, 8, 11, 13], mas nenhuma delas avalia este aspecto. O que se pretende avaliar neste trabalho é se esta estrutura derivada é realmente consensual e útil na especificação de um modelo conceitual a ponto de diminuir o número de divergências entre os atores envolvidos no processo. Algumas abordagens citadas até avaliam a estrutura resultante, mas em sua maioria, de forma empírica [8, 11, 13, 14], no contexto de busca [16] ou recomendação [6] para sistemas de *tagging*.

Cada abordagem, dependendo do tipo de algoritmo utilizado para derivar a estrutura (clusterização, baseadas em ontologias existentes, ou outros) e dos parâmetros adotados, deriva uma estrutura que pode ser bem diferente de outras abordagens, o que implica que a estrutura pode não ser consensual. Praticamente todas elas utilizam como ponto de partida a única relação explícita entre duas *tags* – a relação de coocorrência (duas *tags* coocorrem se elas fazem parte de uma mesma etiquetagem), sem, no entanto, levar em consideração qual o conhecimento/especialidade do usuário que fez a etiquetagem ou quais foram os motivos que o levaram a etiquetar aqueles recursos (características que o algoritmo proposto neste trabalho pretende levar em consideração). Algumas abordagens também utilizam informações sobre a autoria das *tags* [7, 9, 12, 15, 19] (em termos de qual usuário utilizou qual *tag* para etiquetar qual recurso) para extrair a relação de coocorrência entre as *tags*, mas sem avaliar o conhecimento do usuário sobre o recurso que está sendo categorizado.

REFERÊNCIAS

[1] Angeletou, S. et al. 2007. Bridging the Gap Between Folksonomies and the Semantic Web: An Experience Report. *Workshop Bridging the Gap between Semantic Web and Web 2.0, European Semantic Web Conference* (2007), 93.

[2] Begelman, G. et al. 2006. Automated *Tag* Clustering: Improving search and exploration in the *tag* space. *Collaborative Web Tagging Workshop at WWW'06* (Edinburgh, Scotland, 2006).

[3] Damme, C.V. et al. 2008. Deriving a Lightweight Corporate Ontology from a Folksonomy: a Methodology and its Possible Applications. *Scalable Computing: Practice and Experience - Scientific International Journal for Parallel and Distributed Computing*. 9, 4 (2008), 293–301.

[4] Feigenbaum, E.A. 1984. Knowledge Engineering. *Annals of the New York Academy of Sciences* (1984), 91–107.

[5] García-Silva, A. et al. 2012. *Building ontologies from folksonomies and linked data: Data structures and Algorithms*.

[6] Guizzardi, G. 2005. *Ontological Foundations for Structural Conceptual Models*. University of Twente, Enschede.

[7] Hamasaki, M. et al. 2007. Ontology Extraction using Social Network. *Proceeding of International Workshop on Semantic Web for Collaborative Knowledge Acquisition* (2007).

[8] Hauagge, J. M., Tacla, C. A., Freddo, A. R., Molinari, A. H., Paraiso, E.C. 2011. The Use of Well-founded Argumentation on the Conceptual Modeling of Collaborative Ontology Development. *International Conference on Computer Supported Cooperative Work in Design (CSCWD)* (2011).

[9] Jäschke, R. et al. 2008. Discovering shared conceptualizations in folksonomies. *Web Semantics: Science, Services and Agents on the World Wide Web*. 6, 1 (Feb. 2008), 38–53.

[10] Körner, C. et al. 2010. Of Categorizers and Describers: An Evaluation of Quantitative Measures for *Tag*ging Motivation. *Proceedings of the 21st ACM conference on Hypertext and hypermedia* (2010), 157–166.

[11] Maedche, A. and Staab, S. 2001. Ontology Learning for the Semantic Web. *IEEE Intelligent Systems*. 16, 2 (2001), 1–18.

[12] Mika, P. 2007. Ontologies are us: A unified model of social networks and semantics. *Web Semantics: Science, Services and Agents on the World Wide Web*. 5, 1 (2007), 1–15.

[13] Patrick, W. 1983. *Second-hand knowledge: An Inquiry into Cognitive Authority*. Westport: Greenwood Press.

[14] Robu, V. et al. 2009. Emergence of consensus and shared vocabularies in collaborative *tag*ging systems. *ACM Transactions on the Web*. 3, 4 (Sep. 2009), 1–34.

[15] Schmitz, P. 2006. Inducing ontology from Flickr *tags*. *In Collaborative Web Tagging Workshop, 15th WWW Conference, Edinburgh.* (2006).

[16] Specia, L. and Motta, E. 2007. Integrating Folksonomies with the Semantic Web. *4th European Semantic Web Conference* (Berlin Heidelberg, Germany, 2007), 624–639.

[17] Strohmaier, M. et al. 2012. Evaluation of Folksonomy Induction Algorithms. *Transactions on Intelligent Systems and Technology*. (2012).

[18] Tempich, C. et al. 2005. An argumentation Ontology for DIstributed, Loosely-controlled and evolvInG Engineering processes of oNTologies (DILIGENT). *The Semantic Web: Research and Applications – Lecture Notes in Computer Science* (2005), 241–256.

[19] Wu, X. et al. 2006. Exploring social annotations for the semantic web. *Proceedings of the 15th international conference on World Wide Web - WWW '06*. (2006), 417.

Estudo Comparativo de Codificação Tridimensional para o SBTVD

Adenilson J. A. Tomé[*]
Universidade Federal do ABC
Rua Santa Adélia, 166
Santo Andre, SP, 09210-580, Brasil
adenilson.tome@ufabc-.edu.br

Celso S. Kurashima[†]
Universidade Federal do ABC
Rua Santa Adélia, 166
Santo Andre, SP, 09210-580, Brasil
celso.kurashima@ufabc-.edu.br

Mario Minami[‡]
Universidade Federal do ABC
Rua Santa Adélia, 166
Santo Andre, SP, 09210-580, Brasil
mario.minami@ufabc-.edu.br

ABSTRACT

The Brazilian Digital TV System (SBTVD) started signal transmission in São Paulo on 2 December 2007. Currently, all the Brazilian main cities already have DTV channels available. The adoption of the H.264/MPEG-4 AVC standard provides advanced technologies for encoding digital video signals with high definition video. This standard, also allows transmission of three-dimensional video, but this is not considered by SBTVD yet. This research aims to study the encoding of three-dimensional video content with H.264/MPEG-4 AVC standard, and the insertion of these encoded signals in the SBTVD broadcasting stream. Was investigated techniques for capturing video from two and more cameras, encoding three-dimensional video streams, and transmission of 3D video in the Brazilian Digital TV system. The multiple views were coded with the side-by-side technique as well as with multiple-view coding. The videos were transmitted, decoded, and analyzed in a laboratory simulation environment. The objective quality of each video was measured within a wide range of transmission bit rate, and we concluded that good quality 3D video is possible to be allocated in the transmission stream.

Categories and Subject Descriptors

I.3.7 [**Computing Methodologies**]: Three-Dimensional Graphics and Realism; I.4.1 [**Computing Methodologies**]: Digitization and Image Capture—*Camera calibration*

General Terms

Experimentation, Measurement, Theory and Verification

[*]Programa de Pós Graduação em Engenharia da Informação (UFABC). Dissertação de Mestrado. Data de início: FEV.2010. Data da defesa: JUL.2012.

[†]Orientador.

[‡]Coorientador.

Keywords

Video encoding, 3DTV, SBTVD, Digital TV

1. INTRODUÇÃO

A implantação da TV Digital no Brasil iniciou oficialmente em 2007 e mudanças significativas na televisão brasileira surgem com a tecnologia digital. A transmissão de imagens em alta definição, áudio com qualidade superior ao de CD e a possibilidade de interatividade com os usuários são uma pequena parte das vantagens que o sistema pode propiciar.

A *International Telecommunication Union* (ITU) definiu um roteiro para a implantação da TV 3D. Esse roteiro é composto de três gerações, sendo que a primeira geração já está sendo utilizada por algumas emissoras, a segunda geração são estudos para um futuro próximo e a terceira prevê um tempo de desenvolvimento de 15 a 20 anos [7].

Foi realizado um estudo sobre captura de fluxos de vídeo a partir de *N* câmeras *firewire*, de modo sincronizado, visando permitir a codificação de vídeos tridimensionais no formato *side by side* (lado a lado) e *Multiview Video Coding* (MVC), Codificação de Vídeo Multi-vistas, apoiados pela norma H.26-4/MPEG-4 AVC. A camada de transporte foi considerada para a inserção dos sinais de vídeo tridimensional dentro do feixe de transmissão digital do SBTVD, o conteúdo 3D produzido foi encapsulado em fluxos de transportes e transmitido no SBTVD. Os conteúdos transmitidos tiveram sua qualidade medida através da métrica PSNR e analisada posteriormente.

1.1 Objetivo

O objetivo geral da dissertação é investigar e produzir contribuições sobre a viabilidade da utilização de conteúdo tridimensional no SBTVD. Atualmente as normas do SBTVD não preveem a transmissão de conteúdo tridimensional, devido a isso foram realizados estudos sobre a codificação de vídeos tridimensionais e inserção desses sinais de vídeo dentro do fluxo de transporte do SBTVD, efetuando a multiplexação de fluxos de bits válidos.

1.2 Contribuições Esperadas

A implantação do SBTVD no Brasil que deve abranger todo território nacional até 2016 extinguindo a transmissão do sistema analógico de TV país. A produção cada vez maior de aparelhos de TV com suporte à tecnologia 3D assim como

de filmadoras e câmeras digitais é motivo de reflexão quanto à utilização de técnicas eficientes para a disponibilização de conteúdo 3D de qualidade de modo gratuito ao usuário final, possibilitando abranger todas as classes sociais. As tecnologias a serem adotadas devem garantir além de robustez e qualidade que, sejam o estado na arte em 3D e que permitam possíveis expansões futuras. O estudo de técnicas de codificação de vídeo 3D, como a *side by side* e a MVC, voltadas para a transmissão no sistema aberto de TV digital brasileiro é estudada a fim de gerar resultados que visam propiciar fundamentos e conclusões sobre quais técnicas de codificação 3D que podem ser utilizadas para transmissão no SBTVD. As transmissões consideraram a camada de transporte e a largura de banda disponível por canal, podendo assim ter uma visão sobre as transmissões para dispositivos móveis e, TVs e receptores com resolução HD e *Full* HD.

1.3 Metodologia Adotada

Para dar suporte à esta dissertação e atingir os objetivos propostos a seguinte metodologia foi seguida: (i) Revisão bibliográfica abordando os assuntos relevantes para a elaboração dessa dissertação como técnicas de captura de imagens com múltiplas câmeras sincronizadas, composição de conteúdo 3D estereoscópico, sistemas de TV Digital, Multiplexação de dados e Serviços de Informação aplicados à TV Digital; (ii) Desenvolvimento de um aplicativo de captura de vídeo sincronizado a partir de 2, 4 e 8 câmeras *firewire*; (iii) Composição e codificação de conteúdo 3D estereoscópico e com múltiplas vistas com diversas taxas de bits; (iv) Encapsulamento dos fluxos elementares de vídeo 3D dentro de um fluxo de transporte válido para o SBTVD; (v) Transmissão dos fluxos de transporte no SBTVD; (vi) Análise de qualidade dos fluxos elementares de vídeo após a transmissão com base na resolução e taxa de bits.

As etapas descritas acima compõem um ambiente *offline*, necessário para análises individuais de cada etapa de um ambiente de TV 3D. Cada etapa será melhor detalhada a seguir. Todas as codificações foram realizadas com CPU Intel ® Core ™ i7-960 3.2Ghz, 6GB de memória RAM e GPU Nvidia GTS 250 de 1GB.

1.4 Estágio Atual do Trabalho

A dissertação encontra-se em fase final de análise final dos resultados e composição dos gráficos e tabelas.

2. CONTEXTO TEÓRICO

2.1 Captura de vídeo

Todo sistema multimídia envolve diversos fatores que devem ser considerados para o tratamento e manipulação dos dados e um sistema de captura de vídeos não é diferente. No processo de aquisição dos fluxos de vídeo os dados devem ser tratados de forma que possibilitem sua manipulação para construção de conteúdos tridimensionais [10]. Uma das principais distorções é a radial, que devido ao seu formato, provoca uma distorção da localização dos *pixels* próximos das bordas das imagens [3]. O processo de calibração de câmeras fornece um modelo da geometria da câmera e um modelo de distorção da lente. Estes dois modelos informativos definem os parâmetros intrínsecos da câmera. Através desses parâmetros é possível efetuar a correção da distorção causada pelas lentes.

A composição de conteúdo tridimensional requer que a captura dos fluxos de imagens e vídeos seja efetuada de modo sincronizado, para que as diversas imagens das câmeras captem o mesmo cenário ao mesmo tempo. Para a aquisição sincronizada de conteúdo faz-se necessário uma arquitetura de hardware robusta combinada com alguns recursos de softwares que possibilitem esse funcionamento.

2.2 Embasamento Teórico Tridimensional

Ao olhar ao nosso redor os dois olhos humanos nos dão dois pontos de vista simultâneos dos objetos em uma cena. Por um processo chamado de estereopsia, o cérebro funde essas e gera automaticamente informações de profundidade, e é isso que nos permite executar operações do nosso dia a dia com segurança, por exemplo, dirigir ou praticar esportes [4]. A visão 3D se destina a diversos métodos de visualização, onde os *displays* 3D se dividem em duas categorias: (i) estereoscópicos e (ii) autoestereoscópicos. A visualização estereoscópica requer dois pontos de vista enquanto os autoestereoscópicos necessitam de oito ou mais pontos de vista para fornecer a funcionalidade de ponto de vista livre, também conhecido como *Free-View Point* [11].

A estereoscopia é a técnica utilizada para a criação de um par de imagens estéreo planares. Plano-estereoscópico é o termo exato para descrever *displays* 3D que conseguem reproduzir um efeito de profundidade binocular, proporcionando ao espectador uma perspectiva com imagens ligeiramente diferentes em uma tela planar comum [8]. Na técnica conhecida como *side by side* cada quadro é composto por duas imagens, uma para cada olho. Cada quadro é reduzido horizontalmente, assim, cada imagem terá metade da resolução na horizontal, sem alterar a resolução vertical. Ao receber a imagem nesse formato, o *display* divide o quadro e redimensiona cada metade para o formato original e os exibe de modo sequencial conforme ilustrado na Figura 1.

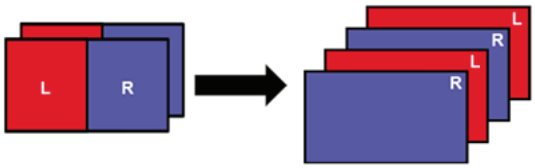


Figure 1: Sequência de quadros *side by side* transformados em quadros sequenciais.

O H.264/ AVC é um padrão de codificação de vídeo (considerado atualmente o estado da arte), desenvolvido em conjunto pela ITU-T *Video Coding Experts Group* (VCEG) e a ISO/IEC *Moving Pictures Experts Group* (MPEG). O H.264/ AVC foi desenvolvido para fornecer compressão eficiente e alta confiabilidade na transmissão de vídeo [9]. O desenvolvimento mais recente dos grupos VCEG e MPEG é o projeto *Joint Video Team* (JVT), as atividades desse projeto concentram-se na codificação com múltiplas vistas, também conhecido como *Joint Multiview Video Coding* (JMVC). O MVC é um codificador que possibilita representar uma cena tridimensional a partir da captura e codificação de duas ou mais câmeras utilizando ângulos ligeiramente diferentes da mesma cena [8].

2.3 SBTVD

O padrão base utilizado no SBTVD é o *Integrated Services Digital Broadcasting Terrestrial* (ISDB-T), desenvolvido pelo Japão. Assim, o sistema adotado no Brasil é o ISDB-TB, a sigla "B" foi adicionada para identificar que o sistema é o do Brasil [5]. Diversas atualizações foram acrescentadas no SBTVD, para codificação de vídeo foi adotado o padrão MPEG-4 parte 10, também conhecido como H.264, e para codificação de áudio o HE-AAC v2. O padrão MPEG-4 foi desenvolvido pela ITU e pelo *Video Coding Experts Group* (VCEG) em conjunto com a ISO/IEC MPEG. A ferramenta de compressão de vídeo do sistema de televisão digital terrestre brasileiro deve obrigatoriamente estar de acordo com a ITU-T *Recommendation* H.264 [2].

No sistema de transmissão foi utilizada a mesma técnica de modulação do ISDB-T, o *Orthogonal Frequency Division Multiplexing* (OFDM). O canal de 6 MHz é dividido em 14 segmentos de 428,57 kHz, dos quais 13 segmentos podem ser utilizados na transmissão de até três serviços distintos e simultâneos. A largura de banda de cada segmento do ISDB-TB varia de acordo com o tipo de modulação e das configurações dos parâmetros utilizados. Cada segmento pode ter taxa máxima entre 280 kbps e 1787 kbps. Na soma dos 13 segmentos a largura de banda máxima fica entre 3,6 Mbps e 23,2 Mbps [1]. O serviço de multiplexação do SBTVD é padronizado pelas normas ABNT NBR: 15602-1, 15602-2 e 15602-3. A principal referência para estas normas é a norma de camada de transporte, criada em conjunto pela ITU e pela ISO/IEC, respectivamente com os nomes de ITU-T H.222 e ISO/IEC 13818-1 [6].

3. RESULTADOS

Os resultados obtidos foram divididos em várias partes que juntos compõem o cenário desenvolvido para atingir os objetivos da dissertação em questão. A etapa de captura visou a captura dos fluxos de vídeos utilizando N câmeras *firewire*. A aplicação desenvolvida, designada *MultipleSync-Capture* é capaz de capturar fluxos de vídeos a partir de 2, 4 e 8 câmeras. O aplicativo é capaz de efetuar a sincronização das câmeras pelo barramento IEEE 1394a, efetuar a calibração das câmeras extraindo os parâmetros intrínsecos e de distorção e assim efetuar a correção de distorção das lentes. Por fim o aplicativo realiza a gravação de um vídeo no formato AVI sem compressão.

3.1 Codificação de vídeo 3D

As codificações de todos os fluxos de vídeo seguiram a norma MPEG-4 Parte 10, também conhecida como H.264 AVC. Para realizar as medições os fluxos foram codificados utilizando a mesma resolução, tendo como entrada duas vistas de 160x240 *pixels*.

Os fluxos de vídeo foram codificados com várias taxas de bits para o MVC e o *side by side*. As codificações dos fluxos elementares MVC foram efetuadas utilizado o software JMVC, software de referência para codificação MVC desenvolvido pelo JVT. As codificações MVC com duas vistas foram realizadas utilizando os parâmetros de entrada sugeridos no manual do JMVC, com exceção do comprimento do GOP, resolução e taxa de quadros dos vídeos, que devem ser exatamente informados de acordo com a codificação desejada. Para possibilitar fluxos com varias taxas de bits para o mesmo fluxo de vídeo o parâmetro de quantização do codificador JMVC foi alterado, assim cada cenário foi codificado variando o QP. Os valores de QP utilizados para os cenários foram: 2, 6, 9, 12, 15, 18, 21, 24, 30, 36 e 42. Cada valor gera uma taxa de bits e uma qualidade diferente no vídeo codificado. O processo de decodificação é mais simples que o processo de codificação bastando apenas informar ao decodificador o *bitstream* a ser decodificado e o número de vistas. As codificações MVC demandam elevado tempo, o que justifica a utilização de baixas resoluções nas codificações.

A composição dos fluxos elementares *side by side* foi realizada através dos mesmos fluxos elementares de vídeo utilizados na codificação MVC, ou seja, o codificador recebeu como entrada dois fluxos elementares de vídeo com resolução de 160x240 *pixels*, como no *side by side* os quadros são colocados lado a lado e somente são reescalados na decodificação a resolução final de saída do codificador é o um fluxo elementar com resolução de 320x240 *pixels*. Um fator importante a ser considerado para a realização da codificação *side by side* é que o codificador JMVC não gera fluxos elementares de vídeo com taxa de bits constantes, ou seja, não é possível determinar os valores exatos. Os valores das taxas de bits são gerados pela variação do parâmetro de quantização do codificador JMVC. Devido a isso, as taxas de bits utilizadas na codificação *side by side* tiveram como ponto de partida as taxas geradas pelo codificador JMVC para cada fluxo elementar codificado. Foram gerados fluxos de vídeo elementares utilizando as mesmas taxas de bits geradas pelo codificador JMVC, sendo assim, para cada fluxo elementar de vídeo MVC foi codificado um fluxo elementar de vídeo *side by side* correspondente, com os mesmos parâmetros utilizados na codificação MVC.

A transmissão de um conteúdo de vídeo 3D, ou seja, um fluxo elementar de vídeo 3D dentro do SBTVD deve ser feito seguindo todas as normas técnicas de tal forma que o ambiente de testes se aproxime o máximo possível de uma transmissão realizada por uma emissora de televisão. Os fluxos elementares devem ser encapsulados em fluxos de transporte e multiplexados seguindo a norma internacional ISO/IEC 13818-1 (ITU-T *Recommendation* H.222) e as normas ABNT NBR 15603-1, 15603-2 e 15603-3. Visando simular um ambiente real utilizado pelas emissoras de TV digital, todas as normas foram seguidas e as tabelas obrigatórias de *Serviço de Informação* (SI) e Programação de Informação e Sistema (PSI) foram inseridas em todos os fluxos de transporte gerados.

3.2 Medidas PSNR

Para os cálculos de qualidade foi utilizada a métrica *Peak Signal-to-Noise Ratio* (PSNR), efetuada com o apoio do software *MSU Video Quality Measurement Tool*[1], sempre analisando o componente de luminância. Para realizar os cálculos PSNR foram selecionados 8 sequências de vídeo com conteúdo capturados a partir de duas câmeras (estéreo). Cada sequência possui características diferentes que visam representar diversos tipos de cenários possíveis.Para cada sequência de vídeo foram realizadas 11 codificações MVC variando o parâmetro de quantização. Cada uma dessas 11 codificações geraram uma taxa de bits diferente. O valor medidas

[1] http://www.compression.ru/index_en.htm

PSNR foram efetuadas comparando o fluxo elementar codificado com o seu respectivo fluxo elementar original. O valor das medidas de qualidade foram calculadas de maneira independente para cada vista para os fluxos MVC e depois calculada a média aritmética. O valor PSNR do fluxo elementar de vídeo *side by side* foi calculado sobre os quadros contendo as duas vistas. As análises visam comparar somente os métodos MVC e *side by side*, o que pode ser avaliado através de testes objetivos. Os gráficos das sequências com as medidas PSNR das sequências *side by side* são demonstradas na Figura 2 e a MVC na Figura 3.

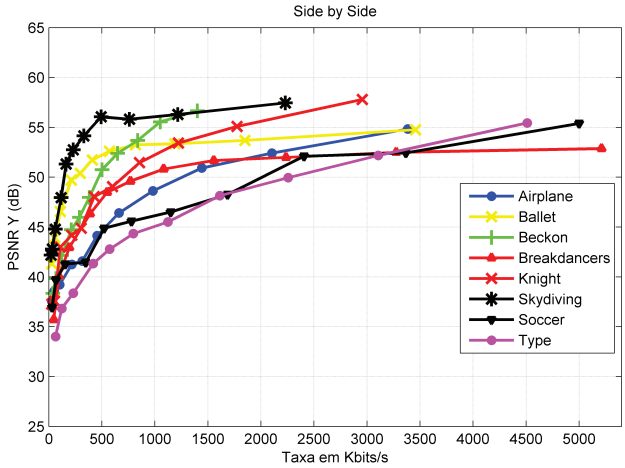


Figure 2: PSNR x taxa de bits das sequências *Side by Side*.

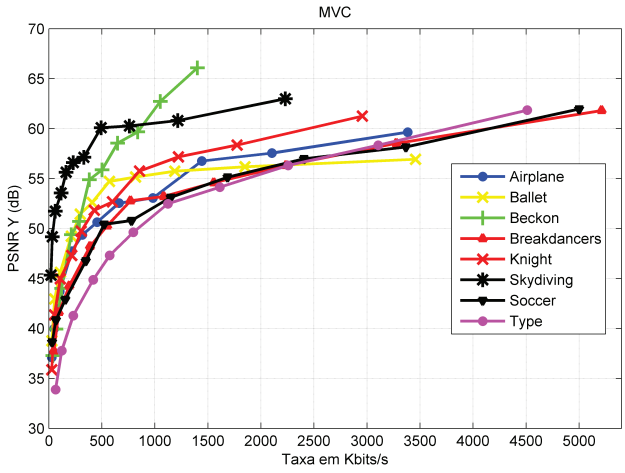


Figure 3: PSNR x taxa de bits das sequências MVC.

4. CONCLUSÕES

A codificação e análise das medidas PSNR dos fluxos de vídeo estereoscópicos *side by side* e MVC permitiram realizar uma comparação entre os dois codificadores. Essa comparação é importante uma vez que o conteúdo *side by side* é utilizado por algumas emissoras na Europa, nos Estados Unidos e por uma emissora aberta no estado de São Paulo/BR. Os resultados das medidas PSNR dos fluxos de vídeo mostram que a codificação *side by side* conseguiu ser melhor que a MVC somente em taxas de bits baixas. Em cinco dos oito fluxos 3D foi possível visualizar essa diferença nas taxas baixas, porém, cada fluxo foi codificado com onze taxas de bits diferentes e em quatro sequências a codificação *side by side* superou a MVC na taxa mais baixa. Dessas quatro sequências o melhor desempenho da codificação *side by side* foi apenas 3,77%, quando a taxa de bits foi de 28 Kbps.

Em 80 das 88 comparações efetuadas a codificação *side by side* obteve resultados piores que a MVC. O ganho de qualidade das sequências nem sempre segue um padrão linear a medida que a taxa de bits é acrescida. É possível ver que em determinadas sequências, como a *Airplane*, a *Knight* e a *Skydiving* a maior diferença de qualidade entre as codificações se situa em taxas de bits intermediárias, variando de acordo com a sequência: *Airplane* (316 Kbps), *Knight* (305 Kbps), *Skydiving* (61 Kbps).

5. AGRADECIMENTOS

À CAPES, pelos recursos e bolsa cedidos no projeto RH-TVD de No. 231/2008, através do programa de formação de recursos humanos em televisão digital.

6. REFERENCIAS

[1] Associação Brasileira de Normas Técnicas, Rio de Janeiro. *NBR 15601: Televisão digital terrestre - Sistema de transmissão*, nov. 2007.

[2] Associação Brasileira de Normas Técnicas, Rio de Janeiro. *NBR 15602-1: Televisão digital terrestre - Codificação de vídeo, áudio e multiplexação*, nov. 2007.

[3] A. Bradski, G.; Kaehler. *Learning OpenCV*. O'Reilly Media, Inc, first edition edition, 2008.

[4] C. Connolly. Stereoscopic imaging. In *Sensor Review*, volume Vol. 26 Iss: 4, pages 266–271. Emerald Group Publishing Limited, 2006.

[5] DTV. História da tv digital no brasil. http://www.dtv.org.br/index.php/informacoes-tecnicas/historia-da-tv-digital-no-brasil, Acesso em jun. 2012.

[6] Institute of Electrical and Electronics Engineers. *ITU-T Recommendation H.222*, Sep. 1995.

[7] ITU. 3dtv moves into focus. http://www.itu.int-/newsroom/press_releases/2010/-01.html, Acesso em jun. 2012.

[8] D. Minoli. *3DTV content capture, encoding and transmission: building the transport infrastructure for commercial services*. John Wiley Sons, Inc., September 2010.

[9] O. Nemeié, S. Rimac-Drlje, and M. Vranjes. Multiview video coding extension of the h. 264/avc standard. *52nd International Symposium ELMAR*, Sep. 2010.

[10] O. B. Neto. Captura e modelagem de mãos com rastreamento de movimentos. Master's thesis, Universidade Federal do ABC, 2010.

[11] C. Peng, V. Ollikainen, and S. Tomperi. Stereoscopic 3-d tv webcast. In *Proceddings of the 9th international interactive conference on Interactive television*, EuroITV '11, pages 177–180, New York, NY, USA, 2011. ACM.

Uma Proposta de Adaptação Dinâmica para Sistemas Ubíquos Baseados em Espaço de Tuplas Distribuído

Mestrando: Benedito José de Almeida Neto*

Orientadora: Rossana M. C. Andrade‡

Coorientador: Marcio E. F. Maia

Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat)
Mestrado e Doutorado em Ciência da Computação (MDCC)
Universidade Federal do Ceará (UFC)
Fortaleza – CE – Brasil
{beneditoneto, marcio, rossana}@great.ufc.br

Nível: Mestrado
Ano de Ingresso: 2011
Previsão de Conclusão: Março de 2013

Resumo: *A evolução das tecnologias móveis e da forma como a informação é acessada favorece o surgimento de sistemas capazes de antever as necessidades do usuário e se adaptar às variações de seu contexto de forma imperceptível. Tais sistemas, denominados sistemas ubíquos, apresentam o problema de adaptação dinâmica em um cenário altamente distribuído, heterogêneo e volátil, no qual componentes devem cooperar entre si, de forma descentralizada, para realizar uma determinada atividade. Baseado no levantamento bibliográfico sobre computação ubíqua verificou-se a necessidade da adoção de estratégias de coordenação descentralizada, permitindo que os diversos dispositivos que compõem um sistema ubíquo possam interagir e cooperar de forma eficiente diante de uma estrutura de rede dinâmica e com constantes desconexões. Neste sentido, este trabalho propõe uma estratégia de coordenação com foco em redes ad hoc, baseada em espaço de tuplas distribuídos, que permite a reconfiguração dinâmica de seus componentes.*

Palavras-chave:

Computação pervasiva, computação ubíqua¸ coordenação, espaço de tuplas distribuído, sensibilidade ao contexto, adaptação dinâmica.

*Bolsista de mestrado (MDCC/UFC) financiado pela CAPES através do programa PROPAG da UFC (Programa REUNI de Orientação e Operacionalização da Pós-Graduação Articulada à Graduação).

‡Bolsista de Produtividade Desenvolvimento Técnico e Extensão Inovadora do CNPq – Nível 2.

1. CONTEXTO TEÓRICO

A computação móvel tem se expandido consideravelmente devido a fatores como a popularização dos dispositivos portáteis e a evolução das redes móveis. A forma como as informações são acessadas evolui continuamente, atualmente os dispositivos computacionais estão imersos no ambiente, e oferecem diferentes modalidades de interação (e.g., visual, gestual, auditiva). Assim, surge uma demanda por sistemas rápidos, confiáveis e acessíveis em qualquer lugar e a qualquer momento (ubíquo) [1].

Diante da complexidade do desenvolvimento de aplicações ubíquas, a literatura [3][8][9] propõe o uso de arquiteturas, middlewares e sistemas de suporte que facilitem a ubiquidade, como em [15], que apresenta uma arquitetura para o suporte de aplicações ubíquas em redes domésticas centradas em TV digital.

Um dos desafios desse tipo de sistema é garantir a cooperação entre seus componentes e a adaptação suave a mudanças de contexto. Cooperação necessária, por exemplo, em uma aplicação que realize a transferência automática de um vídeo de um celular para uma TV baseada no contexto do usuário.

1.1 Computação Ubíqua

Historicamente, a computação tem evoluído a forma como os usuários interagem com os computadores. Partindo dos mainframes, passando pelos computadores pessoais e portáteis, até o advento da computação ubíqua, no qual um único usuário possui diversos dispositivos (*smartphones*, *tablets*, *laptops*) voltados à suas necessidades. O principal objetivo dos sistemas ubíquos é fornecer acesso à informação a qualquer momento e onde quer que o usuário esteja [3].

Segundo Mark Weiser [2], a computação ubíqua tem o objetivo de melhorar o uso do computador, fazendo muitos computadores disponíveis em todo o lugar, mas tornando-os efetivamente invisíveis para o usuário. Essa definição é considerada visionária e norteou muitas outras pesquisas das quais resultaram novas definições. Diante do atual cenário tecnológico, Lima [4] define computação ubíqua como: "O uso de um conjunto de computadores dos mais variados tamanhos, formatos e funções, que de forma coordenada e autônoma, auxiliam as pessoas na realização das diversas tarefas cotidianas. Esse auxílio é realizado de tal forma que a infraestrutura computacional responsável fica escondida no ambiente". Assim, a autonomia e a coordenação entre os componentes do sistema tornam-se pontos fundamentais para a concretização da computação ubíqua.

Tais sistemas possuem características próprias que se relacionam à sensibilidade ao contexto, adaptabilidade e mobilidade. Em [4], é apresentada uma compilação dessas características, gerada a partir de uma ampla revisão literária, dentre as quais podemos citar a Captura de experiências e intenções; Comportamento adaptável; Descentralização; Descoberta automática de serviços; Heterogeneidade de dispositivos e serviços; Interoperabilidade espontânea; Mínima intervenção do usuário; Onipresença dos serviços; e Tolerância a falhas.

Diante dos desafios encontrados no desenvolvimento de sistemas ubíquos, esta pesquisa está direcionada ao comportamento adaptativo a partir da coordenação dos componentes que integram esses sistemas. O sistema deve estar apto a se adaptar diante da perda de conexão com algum dispositivo móvel, assim como ao surgimento repentino de vários dispositivos ao mesmo tempo. O principal objetivo da coordenação é prover a autonomia necessária para a adaptação dinâmica tanto diante de mudanças no contexto quanto diante de falhas no sistema [5].

1.2 Middleware Ubíquo

Middlewares ubíquos devem atender aos requisitos da computação ubíqua, fornecendo abstrações que facilitem o desenvolvimento de aplicações. Na literatura, são encontrados trabalhos como [6], [7], [8] e [16] que elencam tais requisitos. Eles devem prover, a qualquer momento, acesso a informações de contexto heterogêneas, distribuídas e imprevistas em uma escala global e para diferentes cenários, permitindo a descoberta de novos tipos de contexto e informações restritas a um ambiente, garantindo a interoperabilidade semântica de contexto [8].

Segundo Rocha [6], esses middlewares devem suportar importantes requisitos de ubiquidade como: heterogeneidade, descoberta de serviço, coordenação e interoperabilidade. Em [4], são definidos os requisitos básicos comuns a muitos sistemas ubíquos e que devem ser gerenciados pelo middleware, sendo eles: *Coordenação*, *Descoberta/Descrição de Serviços*, *Interoperabilidade*, *Sensibilidade ao Contexto e Adaptabilidade*, *Invisibilidade* e *Autonomia*. O requisito de sensibilidade ao contexto está diretamente relacionado ao requisito de *Adaptabilidade*, uma vez que o sistema deve se adaptar a variações de contexto.

1.3 SysSU

SysSU[1] (*System Support for Ubiquity*) [4], é uma infraestrutura de suporte a ubiquidade que tem o objetivo de atender aos principais requisitos do desenvolvimento de sistemas ubíquos. Atualmente, ele suporta satisfatoriamente os requisitos de coordenação, descrição e descoberta de serviços, interoperabilidade e sensibilidade ao contexto. É possível adaptar sua arquitetura para que seja executado totalmente em uma plataforma embarcada e ainda estendê-lo para funcionamento em uma rede *ad hoc*. Ele implementa um modelo de coordenação formado pela composição dos modelos Linda [10] e *Publish/Subscribe* [11] e ainda especifica uma sintaxe para troca de mensagens, independente de linguagem de programação ou plataforma de desenvolvimento. O SysSU é baseado em espaço de tuplas centralizado, como pode ser visto na Figura 1. Dessa forma não é possível que aplicações executem sem conexão com um servidor central.

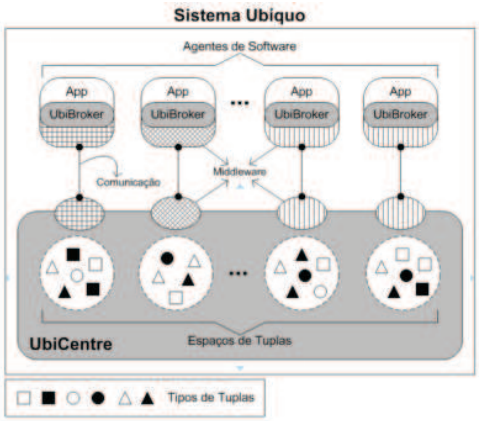


Figura 1. Arquitetura do SysSU [4].

[1]http://code.google.com/p/syssu/

As entidades principais do SysSU são *App*, *UbiBroker*, e *UbiCentre* (Figura 1). A *App* representa as aplicações ou serviços que compõem o sistema. O *UbiCentre* é a entidade que representa o espaço de tuplas centralizado, ele permite que as tuplas sejam acessadas concorrentemente pelas *Apps* através de um *UbiBroker*.

2. IDENTIFICAÇÃO DO PROBLEMA

Os sistemas ubíquos devem estar preparados para operar em, pelo menos, dois tipos de ambientes: (i) ambientes voláteis, no qual dispositivos móveis aparecem e desaparecem de maneira repentina; e (ii) ambientes dinâmicos, no qual a estrutura da rede, seus componentes e serviços estão em constante mudança. Para prover um suporte adequado ao desenvolvimento de aplicações ubíquas, surge a necessidade de infraestruturas de software que ofereçam coordenação, interoperabilidade, mobilidade, sensibilidade ao contexto e autonomia [7] [9].

Em um cenário em que há interação entre diversos dispositivos móveis e diferentes provedores de serviços, e em que o usuário pode se mover por diversos contextos e utilizar diferentes dispositivos com diferentes recursos, uma abordagem de coordenação distribuída e desacoplada se torna necessária para garantir adaptação dinâmica [14]. Tratando-se de dispositivos móveis, o desafio a ser enfrentado é a coordenação descentralizada das ações de cada dispositivo para que alcancem um objetivo global.

3. OBJETIVO

O objetivo deste trabalho é propor um sistema que forneça suporte a computação ubíqua, permitindo a comunicação *ad hoc* e adaptação dinâmica ao contexto de seus componentes. Este trabalho propõe uma extensão do SysSU, através do uso de espaço de tuplas distribuído e da implementação de uma camada de comunicação adaptativa que permita acesso a informações contextuais disponíveis em outros dispositivos.

4. CONTRIBUIÇÕES ESPERADAS

Espera-se que, com a utilização de espaço de tuplas distribuído, as aplicações ubíquas construídas sobre o SysSU possam ter mais mobilidade e independência de uma infraestrutura de software centralizada, aumentando assim, a disponibilidade do sistema. Espera-se também que os dispositivos móveis possam se comunicar e se coordenar de forma *ad hoc*, adaptando-se dinamicamente às tecnologias de redes disponíveis.

As tuplas presentes no espaço de tuplas do sistema servem a diferentes propósitos. Elas podem representar informações contextuais, relativas às aplicações e ao próprio sistema de coordenação. Tuplas também representam componentes e serviços disponíveis no sistema, como sensores ou gerenciadores de acesso. As mensagens de coordenação também serão criadas através de tuplas, permitindo que uma aplicação possa ser notificada caso ocorra alguma mudança do espaço de tuplas.

Aplicações embarcadas em um dispositivo poderão ter acesso direto a tuplas criadas por outros dispositivos. Como pode ser visto na Figura 2, uma aplicação que possua acesso local apenas às tuplas T1 e T2, mas necessite de informações contextuais presentes nas tuplas T1, T2, T3 e T4, poderá ter acesso direto às tuplas que estão disponíveis em outro dispositivo, sem a necessidade de um intermediador centralizado.

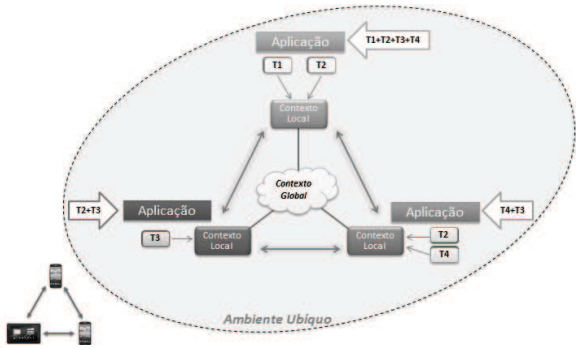


Figura 2. Contexto distribuído.

Com a adoção do modelo de espaço de tuplas distribuído será possível acessar as informações de contexto presentes em vários dispositivos. Isso permitirá a formação de um contexto global (Figura 2), composto por informações contextuais capturadas de forma independente e propagadas de acordo com regras definidas na própria tupla [13], considerando o tempo de vida e o alcance (número de saltos) de cada tupla.

Este trabalho faz parte de um projeto maior chamado LoCCAM (*Loosely Coupled Context Acquisition Middleware*), que está sendo desenvolvido por pesquisadores do GREat[2] (Grupo de Redes, Engenharia de Software e Sistemas). O LoCCAM é um middleware extensível, configurável, com baixo acoplamento para o gerenciamento de contexto. O LoCCAM é direcionado para dispositivos Android e utiliza o SysSU em sua arquitetura. As contribuições deste trabalho permitirão que o LoCCAM gerencie informações de contexto distribuídas e troque mensagens entre dispositivos móveis em redes *ad hoc*.

5. METODOLOGIA ADOTADA

Os esforços de estudo e pesquisa deste trabalho visam contemplar as seguintes metas: Revisar a bibliografia relativa à computação ubíqua e aos desafios da coordenação distribuída; Revisar as soluções de ubiquidade oferecidas pelo SysSU; Portar o SysSU para um dispositivo móvel (*smartphone*), permitindo que o *UbiCentre*, com seu espaço de tuplas, opere na plataforma Android (requisito do projeto LoCCAM); Implementar uma camada de comunicação que permita troca de mensagens entre os dispositivos móveis em redes *ad hoc*; Desenvolver um mecanismo de acesso local ao espaço de tupla que funcione como um *UbiBroker* interno; Definir um modelo de representação contextual através de tuplas empregando a especificação semântica padrão do SysSU (requisito do projeto LoCCAM); Definir as regras de propagação das tuplas e regras de inferência baseadas no contexto global; Habilitar o LoCCAM a realizar gerenciamento de contexto distribuído; e Fornecer subsídios para elaboração de trabalhos futuros relacionados ao tema pesquisado.

A metodologia de pesquisa consiste na leitura de artigos; estudo e testes da arquitetura e da implementação do SysSU em plataformas móveis (Android); estudo, implementação e testes de mecanismos de comunicação e descoberta de serviços em redes *ad hoc*; e estudo de técnicas de distribuição de informações de contexto e gerenciamento de contexto global.

[2]Rossana Andrade, Windson Viana, Marcio Maia, Lincoln Rocha, Benedito Neto, André Fonteles e Rômulo Gadelha.

Para avaliar essa proposta será desenvolvida uma aplicação que exercite as funcionalidades oferecidas pelo SysSU em uma rede *ad hoc*. Será verificado o comportamento do sistema em um cenário de mobilidade, desconexão, compartilhamento de recursos remotos e adaptação a mudanças no contexto. Será realizada uma avaliação de desempenho a partir do tempo gasto para acessar informações contextuais distribuídas traçando um comparativo com a abordagem centralizada.

6. ESTADO ATUAL DO TRABALHO

Atualmente o SysSU foi migrado com sucesso para a plataforma Android, tarefa realizada em conjunto com pesquisadores do projeto LoCCAM. Já é possível realizar testes de acesso direto a um espaço de tupla embarcado em um *smartphone*. A partir daí foi construído um *UbiBroker* interno para acesso local ao espaço de tupla. O próximo passo é a construção de uma camada de comunicação adaptável que permitirá conexão via *Bluetooth* e via *WiFi* de forma *ad hoc*, de acordo com o contexto do sistema. Serão definidas ainda, as políticas de sincronização e propagação nos espaços de tuplas distribuídos.

7. COMPARAÇÃO COM TRABALHOS RELACIONADOS

Os trabalhos listados nessa seção apresentam modelos de coordenação adaptados do modelo *Linda* e baseados em espaço de tuplas distribuído. Eles Introduzem a ideia de reação, baseada em mecanismos de evento, reagindo a mudanças no espaço de tuplas. Foram projetados para sistemas móveis e sensíveis ao contexto, e suportam operações em redes *ad hoc*. Adequando-se, assim, aos principais requisitos da computação ubíqua.

Lime (*Linda in a Mobile Environment*) [12], middleware que aprimora o modelo Linda para ser utilizado em ambientes móveis. Adota uma abordagem de espaço de tuplas descentralizado compartilhando as tuplas de acordo com a conectividade de seus componentes móveis. Dessa forma o espaço de tuplas não fica associado a um dispositivo específico. Cada processo possui seu próprio espaço de tuplas, e este é compartilhado com os demais processos locais ou acessíveis através de uma rede de comunicação. O acesso aos espaços de tuplas remotos é feito de forma transparente, sendo que a operação de leitura de uma tupla é distribuída, enquanto a escrita é local.

TOTA (*Tuples On The Air*) [13], middleware projetado com foco na comunicação desacoplada entre componentes de aplicações distribuídas. Uma versão do TOTA é embarcada em cada dispositivo e as tuplas são propagadas pela rede de acordo com regras de propagação presentes na própria tupla. Ao contrário do Lime, a operação de leitura no espaço de tupla é feita de forma local e a escrita é distribuída. O TOTA monitora constantemente a composição da rede, disponibilizando a informação de quem entra e quem sai da rede.

Os trabalhos citados se assemelham a esta proposta, porém não oferecem simultaneamente a interoperabilidade, o desacoplamento e a expressividade semântica existentes no SysSU. Por outro lado, o SysSU não oferece suporte a operações em redes *ad hoc*, uma importante funcionalidade para sistemas ubíquos [12] [13] [14]. Assim, pode-se perceber a relevância de habilitar o funcionamento do SysSU em redes *ad hoc*, agregando estratégias para coordenação em espaço de tuplas distribuído.

8. AGRADECIMENTOS

Este trabalho é um resultado parcial do projeto UbiStructure, financiado pelo CNPq (MCT/CNPq 14/2011 - Universal - 481417/2011-7) e do projeto Learning While Moving (LWM) financiado pelo Ministério de Educação Superior Francês e Ministério da Educação Brasileiro sob o programa de cooperação científica chamado STIC-AmSud. Agradecimentos especiais para Rossana Andrade, Marcio Maia, Windson Viana, Lincoln Rocha, André Fonteles e Rômulo Gadelha.

9. REFERÊNCIAS

[1] Corradi, A., Lodolo, E., & Monti, S. (2009). Dynamic reconfiguration of middleware for ubiquitous computing. Dependable mobile ubiquitous, 7-12.

[2] Weiser, M. (1991). The computer for the 21st century. Scientific American, 265(3), 94–104. New York.

[3] Hansmann, U., et al. (2003). Pervasive Computing. Springer-Verlag, ISBN 3-540-00218-9, Germany

[4] Lima, F. F. P. (2011). SysSU - Um Sistema de Suporte para Computação Ubíqua. Dissertação de mestrado, UFC.

[5] Sykes, D., et al. (2008). From goals to components: a combined approach to self-management. Proceedings of the international workshop on Software engineering for adaptive and self-managing systems (pp. 1–8). ACM.

[6] Rocha, R. C. A., Endler, M. (2009). Context Management for Distributed and Dynamic Context-Aware Computing. Program. PhD Thesis, PUC-Rio.

[7] Maia, M., Rocha, L., & Andrade, R. M. C. (2009). Requirements and challenges for building service-oriented pervasive middleware. Conference on Pervasive services.

[8] Rocha, R. C. A., Endler, M., & Siqueira, T. S. (2008). Middleware for ubiquitous context-awareness. Proceedings of the 6th international workshop on Middleware for pervasive and ad-hoc computing (pp. 43–48). ACM.

[9] Lima, F. F. P., Rocha, L. S., Maia, P. H. M., & Andrade, R. M. C. (2011). Uma Arquitetura Desacoplada e Interoperável para Coordenação em Sistemas Ubíquos. SBCARS

[10] Carriero, N. and Gelernter, D. (1989). Linda in context. Communications of ACM, vol. 32, no. 4, pp. 444–458.

[11] P. T. Eugster, P. a. Felber, R. Guerraoui, and A.-M. Kermarrec. (2003). The many faces of publish/subscribe. ACM Computing Surveys, vol. 35, no. 2, pp. 114–131.

[12] Murphy, A., & Picco, G. (2006). LIME: A coordination model and middleware supporting mobility of hosts and agents. *ACM Transactions on Software*, *15*(3), 279-328.

[13] Mamei, M. (2009). Programming pervasive and mobile computing applications: The TOTA approach. *ACM Transactions on Software Engineering*, 1-53.

[14] Souza, R. S. D., et al. (2012). Um Modelo de Coordenação Escalável e Proativo para Aplicações Ubíquas. *IV SBCUP*.

[15] Freitas, G. B. D., & Teixeira, C. A. C. (2010). Uma Arquitetura de Serviços para Aplicações Ubíquas em Redes Domésticas Centrada em TV Digital. WebMedia'10.

[16] da Costa, C. A., Yamin, A. C., & Geyer, C. F. R. (2008). Toward a general software infrastructure for ubiquitous computing. *Pervasive Computing, IEEE*, *7*(1), 64–73.

Parte II

XI Workshop de Ferramentas e Aplicações (WFA)

Prefácio

O Workskop de Ferramentas e Aplicações é um dos tradicionais eventos promovidos durante a realização do Simpósio Brasileiro de Sistemas Multimídia e Web (Webmedia). Este workshop é um fórum importante onde membros da academia e da indústria podem apresentar suas ferramentas e/ou protótipos. A principal função deste evento é promover o compartilhamento de conhecimento técnico-científico por meio da apresentação e discussão de ferramentas e aplicações com potencial para auxiliar o desenvolvimento de sistemas multimídia e Web, além de viabilizar a inovação de produtos da área em diversos domínios de aplicação.

Em sua décima primeira edição, o evento recebeu um total de 16 submissões, cada uma delas avaliadas por pelo menos dois revisores. Após o processo de avaliação, foram selecionados 8 artigos, resultando em uma taxa de aceitação de 50%. As ferramentas selecionadas relacionam-se a temas bastante atuais e relevantes tais como: processamento de linguagem natural, Web semântica, TV digital, dispositivos móveis, segmentação de vídeo, além de outros.

O sucesso deste evento é baseado na qualidade dos trabalhos submetidos. Sendo assim, agradecemos a todos os autores que submeteram seus trabalhos para este Workshop. Muitos trabalhos bons não puderam ser aceitos devido as limitações de tempo e espaço físico do evento. Agradecemos também a todos os membros do comitê de programa por sua dedicação, pelas valiosas revisões dos artigos e pela ativa participação no oferecimento de críticas construtivas que são fundamentais para o aperfeiçoamento do texto e da ferramenta desenvolvida.

Agradecemos também aos organizadores do evento pelo convite e confiança ao nos designar para a coordenação deste workshop.

São Paulo, Outubro, 2012
Anarosa Alves Franco Brandão - EPUSP
Seiji Isotani - ICMC-USP

Organização

Coordenação do WFA

Anarosa Alves Franco Brandão	EPUSP
Seiji Isotani	ICMC-USP

Comitê de Programa do WFA

Barbosa, Ellen Francine	ICMC/USP
Barbosa, Simone Diniz Junqueira	PUC-Rio
Bittencourt, Ig Ibert	UFAL
Fernandes, Clovis	ITA
Fileto, Renato	UFSC
Gomes, Alex Sandro	UFPE
Manzato, Marcelo	USP
Nakamura, Ricardo	USP
Nunes, Fátima	USP
Oliveira, Elaine H.T.	UFRGS
Pimentel, Edson	UFABC
Tori, Romero	USP
Trinta, Fernando	UFC

aNa: API for NCL Authoring

Joel A. F. dos Santos
Laboratório MídiaCom, Instituto de Computação, Universidade Federal Fluminense
Niterói, RJ, Brazil
joel@midiacom.uff.br

Julia V. da Silva
Laboratório MídiaCom, Instituto de Computação, Universidade Federal Fluminense
Niterói, RJ, Brazil
julia@midiacom.uff.br

Renan R. Vasconcelos
Programa de Engenharia de Sistemas e Computação, COPPE, Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
renanrv@cos.ufrj.br

Wagner Schau
Programa de Engenharia de Sistemas e Computação, COPPE, Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
schau@cos.ufrj.br

Cláudia Werner
Programa de Engenharia de Sistemas e Computação, COPPE, Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
werner@cos.ufrj.br

Débora C. M. Saade
Laboratório MídiaCom, Instituto de Computação, Universidade Federal Fluminense
Niterói, RJ, Brazil
debora@midiacom.uff.br

ABSTRACT

There are several available NCL (Nested Context Language) authoring and formatting tools using their own metamodel to represent the code they are working on. This paper presents an API that implements a metamodel specifically created to represent NCL documents. This API helps the creation of tools to manipulate NCL documents and brings some benefits to code reuse to the Digital TV Systems development context. The API here presented is called aNa, an acronym for API for NCL Authoring. aNa is available for free download and open for contributions. aNa has already been used for the development of some NCL authoring and analysis tools.

1. INTRODUCTION

Nested Context Language (NCL) is the standard declarative language of the Brazilian Digital TV system [1] and ITU standard for IPTV services [8]. The growth in the use of NCL for the creation of interactive content shall increase the need for tools to help interactive application creation. Those tools can be authoring tools, analysis tools and even presentation tools.

Usually, each tool that has been created to manipulate NCL code implements its own metamodel to represent the code it is working on. If one does not create a model to represent the document, it is common to use available XML parsing tools, like DOM (Document Object Model) [4] or SAX (Simple API for XML)[1]. Although those parsers produce a good result and help the tool developer to manipulate NCL code, their purpose is generic, allowing the parsing of any XML document.

This paper presents a metamodel specifically created to represent NCL documents in a model-oriented environment and help the creation of tools to manipulate NCL code. This metamodel is implemented in Java by an API called aNa, an acronym for API for NCL Authoring. aNa is available for free download and is open for contributions. aNa has already been used for the development of some NCL authoring and analysis tools.

The remaining of this paper is structured as follows. Section 2 summarizes available tools to help NCL authoring. Section 3 presents aNa structure. Section 4 discusses the API implementation and its main characteristics that facilitate the creation of NCL tools, like code reuse. Section 5 presents some tools that already use aNa and Section 6 concludes the paper and presents future work.

2. NCL AUTHORING TOOLS

Currently there are some available tools for helping the authoring of NCL documents. Those tools present different capabilities, focusing on different user profiles. The following paragraphs present those NCL authoring tools.

Composer 3 [9] provides a single authoring environment suitable for different user profiles, from home users to content producers. Composer 3 proposes the basis for building an integrated environment that can adapt itself to various profiles and support non-functional requirements.

NCL-Eclipse [2] is an authoring tool, available as an Eclipse plugin, which helps the author creating an NCL document. It presents some facilities to help coding as: code completing, code highlighting and errors and warnings highlighting.

NCL-Inspector [7] is a tool based on other tools for code quality critique, which supports the authoring of NCL applications. It supports the author in terms of code quality.

XTemplate 3.0 [6] defines a language and tools for the authoring of NCL documents using composite templates. Composite templates define generic structures of nodes and links that can be reused in different document compositions, facilitating the authoring of interactive applications in Dig-

[1] http://www.saxproject.org/

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WebMedia '12, October 15-18, 2012, São Paulo, SP, Brazil
Copyright 2012 ACM X-XXXXX-XX-X/XX/XX ...$10.00.

ital TV systems.

Berimbau iTV Author [3] is a graphical authoring tool for digital TV applications aimed at media professionals who have no programming knowledge. The tool provides a simple and intuitive interface and a media repository to be used while creating the application.

Notice that each one of those tools uses its own metamodel, in an informal manner, to represent NCL documents. If there were an NCL reusable API that implemented an unified metamodel available to model and represent NCL documents, the programming effort to develop NCL tools would be much smaller. This paper proposes such API.

3. ANA METAMODEL

aNa was created to represent an NCL document as a model. Its structure is optimized so the author of NCL tools that manipulate XML code does not need to worry about the language representation, but with the model itself. Every NCL element is represented as a class, which will be called element class. An element class contains the same attributes of the NCL element it represents. Every element class in aNa inherits from the basic type *NCLElement*.

The element classes follow the same hierarchy of the NCL elements, therefore, an element class will be associated to all element classes that represent its child elements. The cardinality of those associations is defined following the NCL language specification. By representing parent-child relations as associations, aNa makes it possible to navigate from an element to its children and the opposite as well. Figure 1 presents the related representation in aNa of the element <*ncl*>, represented by class NCLDoc, which has attributes *id*, *title* and *xmlns* and the elements *head* and *body* as children.

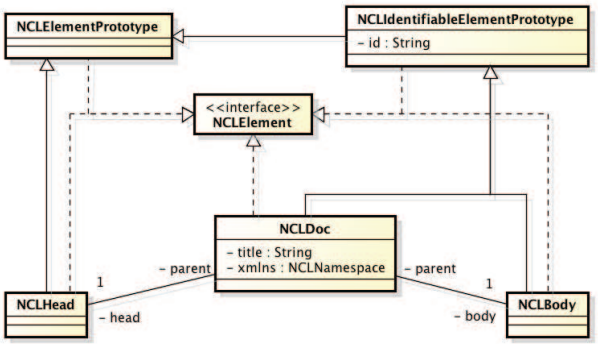


Figura 1: NCLDoc class representation

NCL also defines attributes that refer to other elements. This reference is usually done by defining the attribute value equal to the referred element id (more common) or another attribute of the referred element. In order to improve navigation over element classes, aNa represents those references as associations between them. Listing 1 presents an example of reference between elements and Figure 2 presents how those elements are represented in aNa.

Listing 1: Element reference example

```
1 <regionBase>
2     <region id="reg1"/>
3 </regionBase>
4 <descriptorBase>
5     <descriptor id="desc1" region="reg1"/>
6 </descriptorBase>
```

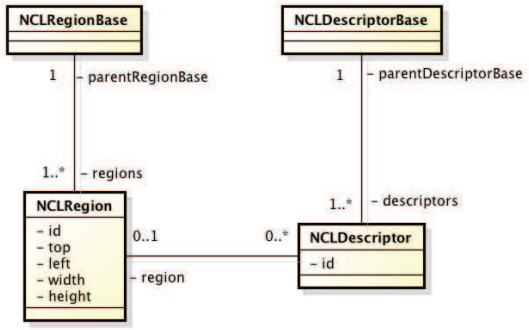


Figura 2: Element reference representation

Listing 1 presents an NCL document part. The example defines a <*regionBase*> element that has a <*region*> element as child. The <*region*> element *id* attribute is a String that represents its identification. The example also defines an element <*descriptorBase*>, that has a <*descriptor*> element as child. The <*descriptor*> defines two attributes, *id* and *region*. The *id* attribute represents its identification and the *region* attribute is a String that represents the identification of the <*region*> used by that <*descriptor*>. In Figure 2, it is possible to observe that the *region* attribute of the <*descriptor*> is represented as an association between the *NCLDescriptor* and *NCLRegion* classes.

Some element attributes may have a value from a specific value set, like the *xmlns* attribute. In those cases, aNa defines the attribute type as an Enumeration with all the possible values for that attribute (see *xmlns* in Figure 1). Sometimes an attribute value can have more than one type. For example, consider the elements presented in Listing 2.

Listing 2: Attribute value examples

```
1 <region id="reg1" top="10" left="10"/>
2 <region id="reg3" top="10.5%" left="10%"/>
```

Notice that the element <*region*> has attributes that may be an integer without a percent sign (%) or a integer or a double with a percent sign. NCL also defines other elements whose attributes can be numbers or strings, as the *max* attribute of a connector condition, where it can be a positive integer or the string "unbounded"; elements whose attributes can be a value (like a string, a number, etc) or another element representing a parameter, as the *delay* attribute of a connector action, where it can be a double or a reference to a connector parameter element.

In aNa, those attributes are defined with type *Object*. So, they can receive any of the possible values the attribute demands. When receiving a new value, the API tests if its type is correct. If not, aNa raises an error indicating the values the attribute may receive. In those cases, if the value received is an string, aNa tries to parse the value to the format used by NCL.

Figure 3 presents a fragment of the API structure, representing relations among elements of the document body. In order to simplify the diagram, the figure presents only class names and associations between them.

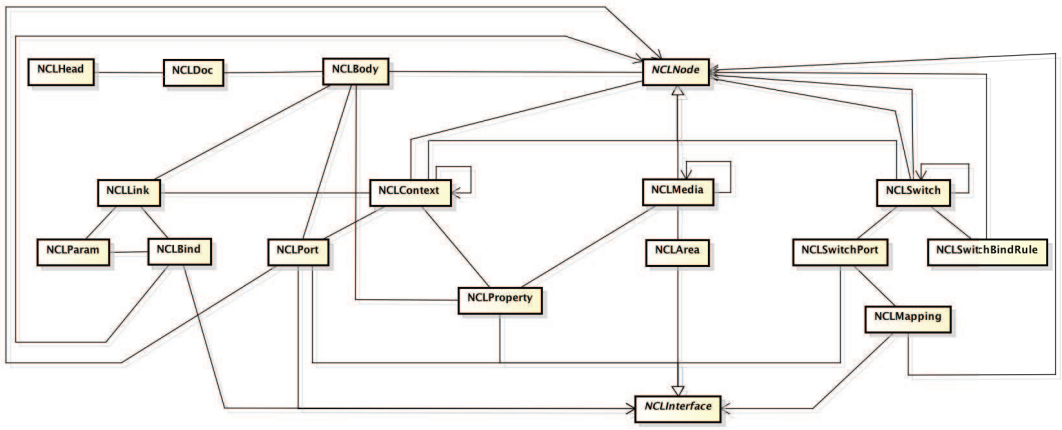


Figura 3: aNa document body structure

4. ANA IMPLEMENTATION

aNa is implemented in Java, providing portability in different platforms. It presents methods to get and to change the value of an element class attribute, to navigate through the NCL document, to create Java objects from an NCL document and to write an NCL document from those objects.

aNa provides facilities to increase reusability in the development of Digital TV applications. NCL itself already allows code reuse, since the same elements can be used in different contexts, without the need to redefine them. Because of the Java implementation, the benefits of object oriented design, such as allowing reuse by class inheritance or object composition [12], help increasing the application development productivity in comparison to direct NCL programming.

The document parsing is done using DOM [4]. aNa walks through the DOM tree gathering information about the NCL elements and creating Java objects that represent them. During that object creation, aNa already creates references between objects. For example, if aNa finds the value "reg1" in the attribute *region* of a <*descriptor*> element, it will search for a <*region*> element in the region base with that *id* in order to create this reference and it will raise an error if no <*region*> with that *id* is found.

The search for a referred element is done only in the attribute scope. That is, if an attribute indicates the *id* of an element inside the same context, aNa will search for that element only inside the context. For example, suppose a <*port*> element that defines *component* and *interface* attributes. aNa will search for an NCL node with the *id* defined in the *component* attribute inside the port parent context. Once that element is found, aNa will search for an interface with the *id* defined in the *interface* attribute inside that node.

During parsing, aNa gathers from the DOM representation of an NCL element only the information that makes sense to it, that is, the attributes and child elements defined in the language specification. Also, when reading an NCL document, DOM already verifies if the XML document is well written, that is, all the XML tags are opened and closed correctly. So, after the document parsing, aNa will have a consistent document representation. If a wrong definition is found in the NCL document, aNa will raise errors. Listing 3 presents an error example. Errors always present the whole path from the root document element to the element where the error occurred. It also shows a message that informs the error found to the author.

Listing 3: Parsing error example

```
1 Error parsing Head > ConnectorBase >
      CausalConnector(onKeySelectionStop) >
      SimpleCondition
2 Could not find a param in connector with name: tecla
```

The opposite way, that is, create an NCL document from the aNa representation is done by getting, from each Java object, its XML representation. The method that implements it returns a String with the XML element representation. It is worth to highlight that the code returned is indented, making the document reading easier.

Once aNa is developed to be used by tools that manipulate NCL code, it is able to notify the tool that uses it about a modification in an element class. This notification can help the tool maintaining a consistent document representation. For example, suppose a tool that is built to graphically show all document regions. Every time a modification occurs in the position of any region, the tool is notified, so it is able to apply the necessary changes in the graphical position of the modified region. The API has a *ModificationNotifier* which may have one or more *ModificationListeners*. The notifier will send notifications when the value of an attribute is set and a child element is added or removed. As this feature may not be necessary for all kinds of tools, the tool is not obliged to implement the *ModificationListener*.

Also, in order to maintain a consistent document representation, once element references are represented as associations, when removing an element from its parent, aNa verifies if such element is used in a reference. If so, aNa indicates to the author the elements that make reference to it and disable its removal until the references are removed.

Another characteristic of aNa is that it is implemented using parameterized classes, which is done using the Generics Java language feature[2]. Using that feature, aNa element classes extensions are simpler, requiring less coding effort.

[2]more information available in http://docs.oracle.com/javase/tutorial/java/generics

5. ANA USE CASES

As mentioned before, aNa was developed to help tools modeling and manipulating NCL code. One use case is a graphical editor that was built for helping users creating NCL connectors [10]. The editor uses aNa to read an NCL document and extract the <*connectorBase*> element. This element represents a base containing all connectors that may be used in <*link*> elements. Once the editor has the object, created by aNa, representing the connector base, it can use aNa methods to get necessary information from the element and to graphically show the connector to the author. The editor also allows the author to perform creations, removals and modifications on connector child elements. All these modifications are performed through aNa methods. Moreover, the editor allows the author to create a new connector base document. The final NCL document code is also created by aNa.

Another tool that uses aNa is called NEXT (NCL Editor supporting XTemplate) [11]. It is a graphical authoring tool developed to facilitate the creation of digital TV applications using the NCL language and supporting the use of composite templates. Its architecture is based on a core that is able to communicate with plugins and is completely independent from them. NEXT plugins are allowed to manipulate the document and, when a plugin makes any modification on it, NEXT notifies other plugins about the modification. In order to obtain knowledge about the change, NEXT uses aNa's feature that notifies about changes occurred in the document. Moreover, NEXT uses aNa objects to create a tree model representing the nesting structure of the NCL document. aNa also enables NEXT to open and to edit any standard NCL document.

aNa is also used as the basis for the development of an NCL document validation tool. That tool, called aNaa - API for NCL Authoring and Analysis, extends aNa by adding methods that allow the validation of the NCL document being authored [5]. It uses aNa for reading an NCL document and gathering information about document elements. After that, aNaa creates different representations of the NCL document, which allow the tool to investigate some document properties and verify its temporal consistency.

6. CONCLUSION

In general, when developing tools to manipulate NCL code, developers have to implement their own metamodel to represent the code to be manipulated. Sometimes no metamodel is created at all. Since those tools need to read an NCL document, it is common to use available XML parsing tools, like DOM or SAX. Although those parsers produce a good result and help the tool developer to manipulate NCL code, their metamodel is generic for any XML document.

This paper presented aNa, an API that provides a metamodel created specifically for representing NCL code and helping the creation of tools that manipulate NCL documents. aNa considers NCL attribute types and verifies if the document follows NCL syntactic and reference rules as defined in the Brazilian standard [1]. Besides API specificities, with a common core that represents an NCL document, aNa makes it possible to exchange object-oriented data among different tools without the need to generate XML code.

Currently, aNa is being used as a basis for the creation of authoring and validation tools. Its code is available for free download[3] and the tool is also open for contributions. We intend to continuously improve aNa, based on feedback from the NCL developers community.

In the current version of the API, error messages are presented in English. A future work is the creation of aNa error messages in different languages. Another future work is to improve the API to receive NCL live editing commands and to produce the necessary modifications in the document.

7. ACKNOWLEDGEMENTS

This work was partially supported by CNPq, FAPERJ and CAPES.

8. REFERÊNCIAS

[1] ABNT. Digital terrestrial television - Data coding and transmission specification for digital broadcasting - Part 2: Ginga-NCL for fixed and mobile receivers - XML application language for application coding, 2011.

[2] R. G. A. Azevedo, M. M. Teixeira, and C. S. S. Neto. NCL Eclipse: Ambiente Integrado para o Desenvolvimento de Aplicações para TV Digital Interativa em Nested Context Language. In *Salão de ferramentas - SBRC*, 2009.

[3] Batuque TV digital. Berimbau iTV Author. http://www.batuque.tv/, 2011.

[4] W. W. W. Consortium. Document Object Model (DOM) Level 2 Core Specification, 2000. W3C Recommendation DOM-Level-2-Core-20001113.

[5] J. A. F. dos Santos. Multimedia and hypermedia document validation and verification using a model-driven approach. Master's thesis, Universidade Federal Fluminense, 2012.

[6] J. A. F. dos Santos and D. C. Muchaluat-Saade. XTemplate 3.0: spatio-temporal semantics and structure reuse for hypermedia compositions. *Multimedia Tools and Applications*, 2011.

[7] G. S. C. Honorato and S. D. J. Barbosa. NCL-Inspector: Towards Improving NCL Code. In *ACM SAC*, pages 1946–1947, 2010.

[8] ITU. Nested Context Language (NCL) and Ginga-NCL for IPTV services. http://www.itu.int/rec/T-REC-H.761-200904-P, 2009. ITU-T Recommendation H.761.

[9] B. Lima, R. Azevedo, M. Moreno, and L. Soares. Composer 3: Ambiente de autoria extensível, adaptável e multiplataforma. In *WebMedia – Workshop de TV Digital Interativa (WTVDI)*, 2010.

[10] J. V. Silva and D. C. Muchaluat-Saade. Editor Gráfico de Conectores Hipermídia para Linguagem NCL 3.0. In *WebMedia*, 2011.

[11] J. V. Silva and D. C. Muchaluat-Saade. NEXT - Editor Gráfico para Autoria de Documentos NCL com Suporte a Templates de Composição. In *WebMedia*, 2012.

[12] S. Srinivasan and J. Vergo. Object Oriented Reuse: Experience in Developing a Framework for Speech Recognition Applications. pages 322–330, 1998.

[3] https://github.com/joeldossantos/aNa

Web-Based Virtual Lab for Taxonomic Description

Alessandra Gomes, André Santanchè, and Fabiani de Souza
Institute of Computing, University of Campinas
Campinas, SP, Brazil
ra075936@students.ic.unicamp.br, santanche@ic.unicamp.br,
ra108171@students.ic.unicamp.br

ABSTRACT
The taxonomic description of a specimen is an essential task carried out by biologists aimed to identify and study living beings. The usual approach involves analysing and describing a given specimen in a physical laboratory. Nevertheless, several tasks are being virtualized. Images, sounds, and videos of living beings are being digitalized; records are stored in spreadsheets and databases and the description task itself are being supported by specialized software. This work investigates a step beyond, where the laboratory itself becames virtual.

Categories and Subject Descriptors
D [**Software**]: [Miscellaneous]; J.3 [**Life and Medical Sciences**]: [Biology and genetics]

General Terms
Experimentation

Keywords
virtual lab, software component, taxonomic description, web

1. INTRODUCTION
Virtual laboratories simulate physical equipments and the infrastructure of a physical laboratory by using computational techniques. They can represent experiments by graphical interfaces and offer interactive simulations.

Virtual labs can be tooled to afford learning experiences comprising exercices, theorical explanations, and interactive assistants that explain experiments step by step. In many cases, they can be used any time and from anywhere. This kind of laboratories are also known as simulated laboratories or e-laboratories [4].

By handling and combining visual software components, users can describe specimens in a virtual laboratory. This paper presents our work of such a tool involving the description of living beings. In this work we investigate a specific kind of Biology virtual laboratory to support taxonomic description of specimens, in which the basic lab elements are virtualized as visual software components.

This is an ongoing work and, in order to validate our proposal, we have implemented a preliminary prototype with components to describe monitor lizards of the genus Varanus. The lab is based on a system called Varan-ID.

Varan-ID is an online determination system for monitor lizards. It is based on a morphological knowledge base of a group on carnivorous lizards, the genus Varanus. The system is based on the idea that not only experts are involved with monitor lizards. Students, curious, breeders, keepers are either interested in this subject, but may not have the necessary knowledge to work with the specimen. The process of identification in the Varan-ID system is based on descriptors.

In this paper we present a prototype of our virtual web laboratory to describe and to identify living beings. It is based on visual components handled by direct manipulation, which play roles of building blocks for descriptions and lab tools. Therefore, when a user inserts a component that represents a tail in the composition, he/she will add a related tail descriptor in the lizard description. The entire lab runs over the web on top of the *Componere* authoring environment[8], which explores the Rich Internet Application (RIA) approach to provide an interactive interface.

The remaining of the paper is organized as follows: Section 2 presents a taxonomic description model of the context of the developed tool. Section 3 presents implementation details of the tool. Section 4 presents future works and conclusions.

2. TAXONOMIC DESCRIPTION MODEL
The starting point for designing our lab was the software Xper2 (http://lis-upmc.snv.jussieu.fr/lis/?q=en/resources/software/xper2/). This tool supports the identification and description of specimens. It follows the character/character state (C,CS) [5] approach, organized in three phases:

(i) to define descriptors and possible states;
(ii) to relate descriptors/states to species;
(iii) to identify a given specimen by recognizing values for each descriptor.

The Varan-ID system was developed over the Xper2. Its descriptors are organized in two distinct groups, easy-to-see descriptors and expert descriptors. Table 1 shows an example of some easy-to-see descriptors and their respective states.

Table 1: Easy-to-see Descriptors

Lizard Part	Descriptor	States
Tail	transversal section of the tail	roundish or laterally compressed
Head	position of nostrils between eyes and tip of snout	same distance from eyes than from tip of snout or nearer the tip of snout than the eyes or nearer the eyes than the tip of snout
Tongue	tongue coloration	red, light pink or whitish or blue, purple or black

Our tool is able to interact with Xper2 by accessing SDD files it can export. The Structure Descriptive Data format (SDD) is an open standard endorsed by the TDWG (Taxonomic Database Working Group) and DELTA (Descriptive Language for Taxonomy) for representing taxonomic descriptions in a XML format (http://wiki.tdwg.org/SDD/).

Figure 1 shows a diagram representing a fragment of a SDD file containing data to describe Varanus lizards. The hexagons represent elements, the rectangles represent texts and the ovals represent attributes. The CategoricalCharacter element defines a descriptor and possible states. In this example it defines the tongue coloration element and three possible states: red, light pink or whitish and blue, purple or black.

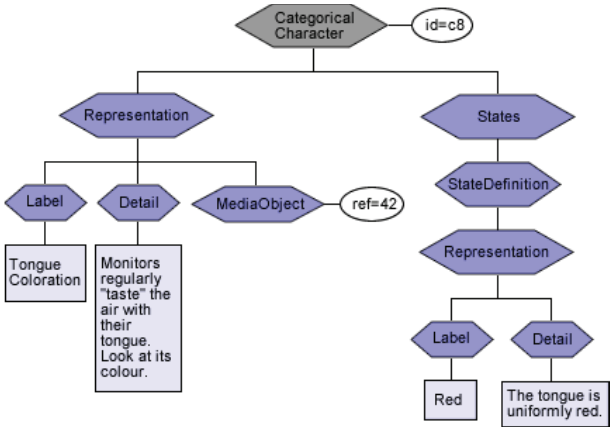


Figure 1: A Fragment of a SDD file of Varanus Lizards

The States element aggregates all possible states: StateDefinition elements. As can be seen in the diagram, the Representation element can be applied in many levels of the schema, containing textual and multimedia descriptions. This element is formed by a label, a detailed description (Detail) and references to multimedia resources (MediaObject)

In our tool we map these SDD description blocks in the following way:

(i) each CategoricalCharacter becomes a description component;
(ii) the set of states that the CategoricalCharacter can assume is transformed in a set of possible states that the component can assume;
(iii) every time the description component assumes a state it provides a visual feedback.

These description components transform the task of describing specimens in selecting and customizing components, which are combined in compositions.

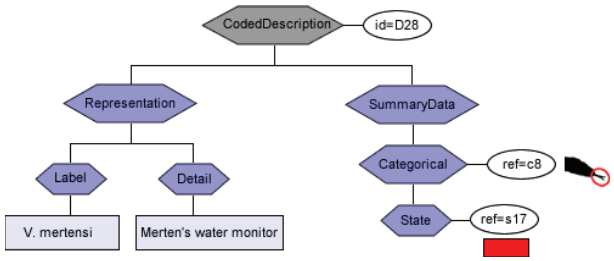


Figure 2: The Structure of Lizard SDD Base

Our resulting composition reflects another part of the SDD representation, illustrated in Figure 2. Besides the Representation element, described before, the diagram shows the summary data of the lizard. This element contains the element Categorical that represents the description of a given specimen (a lizard in the example), specifying Categorical Characters – referenced by the Categorical element – and setting values to it, through the State element. In this example, the categorical character "tongue coloration" assumes the state "red". Since each composition specifies the description of a given specimen, the resulting composition can be mapped to a SDD structure presented in Figure 2 and vice-versa.

In order to automatically derive SDD Categorical Characters to description components, a process was created to get all information on the SDD file and use it to fetch the respective description components, customizing them with the respective values. Figure 3 illustrates the creation process of lizard components and their use during a composition. A proxy reader component accesses a SDD file (step 1). This information is delivered to a component generator that creates all the description components (step 2 and 3) – lizard description components in the example. Therefore, the categorical element will be used to identify a specimen characterization – a lizard in the example – to generate its respective composition. The State element will generate the value that the parameter assume.

The author customizes and combines components, building a composition, whose parameters are recorded by the Settings component (step 4). When the author clicks in the Candidate button it triggers the Result component (step 5), which in turn retrieves the parameters from the Setting component (step 6), and uses the proxy component to fetch

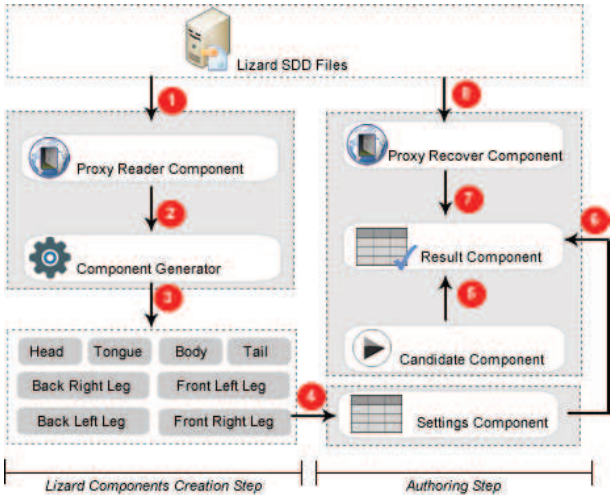


Figure 3: The Creation of Lizard Components and Composition Process

data from XML files in the base (step 7); which uses the HTTPXMLRequest API (step 8).

In our proposal, a description component is one type of the available components. Another type comprises tool components, a group that will be responsible to support the authoring process. The classification based on this two types of components is illustrated in Figure 4.

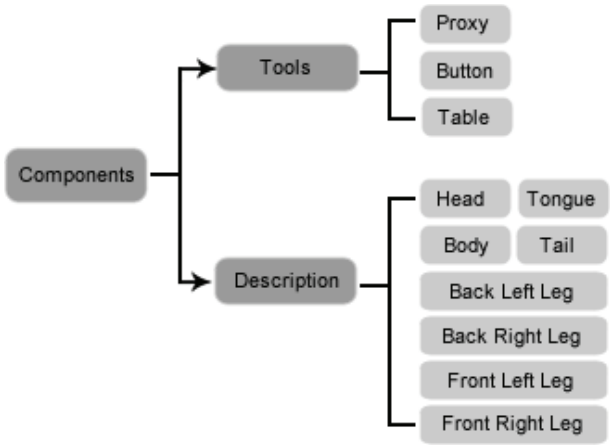


Figure 4: The Component Classification for the Lizard Lab

Our lab is designed to afford any kind of description component for living beings. However, in our prototype we have produced only description components representing each part of the Varanus lizard. They are visual components derived from Varan-ID easy-to-see descriptors. The tool components can be visual or not. For example, the button is a visual component that starts the execution process. The table is a visual component that contains data organized as rows and cols. The proxy is a non visual component that brings data stored in a database to the composition.

As mentioned before, our laboratory is built over the *Componere* environment. In the original *Componere* authoring environment all components play the role of building blocks. Our lab, on the other hand, introduces this new kind of component - the tool component - to assist the authoring task itself.

3. IMPLEMENTATION

The implementation of the proposed laboratory involved two steps: the construction of description components and the construction of the laboratory on *Componere*. As mentioned before, this first prototype is focused on a specific practical scenario involving the identification of Varanus lizards, based on the Varan-ID database.

3.1 Construction of Lizard Components

The Varan-ID base is composed by 7 knowledge bases: the Main base, V. indicus-group base, V. prasinus-group base, V. timorensis-group base, V. gouldii-group base, V. salvator-group base, and the Australian spiny-tailed base. Each base can be exported as an SDD file. To build the components, we analysed the available descriptors in these bases.

A component builder engine was developed to extract information from the exported database and to use them to build each one of the lizard components.

There are two ways to build *Componere* compositions. The first is by using a javascript code to instantiate and to connect components. The second is by embeding compositions in HTML pages through microformats based specifications [8]. During the authoring process the laboratory uses the first dynamic approach. Resulting compositions can be further materialized as HTML embedded compositions.

3.2 The Laboratory on Componere

As mentioned before, *Componere* is a framework based on javascript components that works over web browsers.

Thus, the Lizard Lab is an environment totally based on javascript, mainly directed to beginners in the monitor lizard identification process. An overview of the system is illustrated in Figure 5.

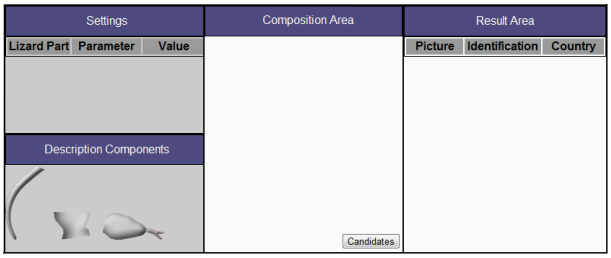


Figure 5: The Model of the Lizard Lab

The environment is organized in four areas: description components, settings, composition and result area. The description components area is where the components representing parts of the lizard are placed. Each one is independent and has its own set of parameters to be configured.

In order to produce a description the author drags description components to the Composition Area where they are

customized and connected. Each description component has two basic main actions: close and configure. The first will remove the component of the composition. The second will open a dialog box with the parameters to be configured. Whenever a component is changed, the Settings area (displayed in Figure 5) stores and shows a log of the values assigned to descriptors. Figure 3 illustrates this relation in the process.

During the description/identification process the author can access the base containing available descriptions of existing species – lizard in this case – whose descriptor/states match with those already assigned in the lab. For example, if the author assigns a specific tongue color and tail shape, the system will record these settings in the Settings area; whenever the author clicks in the "Candidates" button, available in the environment (see Figure 5), the system fetches and presents all lizards in the base which have the informed tongue color and tail shape. The steps to execute this process is illustrated in Figure 3. This technique to present progressive candidates is based in Xper2 approach to describe specimens.

The prototype is available at http://fluidweb.sourceforge.net. The page has many experiments using different kind of software components. To access the work proposed in this paper, click on the link "Lizard Prototype".

4. RELATED WORK

Our work combines two approaches: virtual laboratories and tools to describe and identify specimens.

According to [4, 6] the laboratories can be classified in three categories: real, remote and virtual. Real labs are physical rooms, having concrete equipments and infrastructure. Remote labs enable access to physical resources of real labs by networks – as the Internet – through a simulation software, which replicates the remote environment. Virtual labs have the goal of offering a simulation environment to support virtual experiments. Our proposal can be considered a mixture of the three contexts, since it is a virtual lab that grabs data from the real world and is built over the web.

Laboratories usually offer specialized resources according to the context they are inserted. [1] proposes an educational environment for electronics and electrical engineering. [2] presents an environment for genetics learning. [3] presents a virtual lab of Chemical Vapor Deposition aimed to complement a physical laboratory in the undergraduate course curriculum. [7] proposes a virtual laboratory for medical digital analysis based on grids.

The tool to describe specimens – as Xper2, detailed before, and Lucid (http://www.lucidcentral.org) – are designed for specialists and does not follow a laboratory approach. As far as we know, there is no such a tool combining the characteristics of virtual laboratories and biology description/identification tools.

5. CONCLUSIONS

In this paper we present our virtual laboratory based on virtualized visual components. In other to validate it, we have implemented a prototype of the virtual lab to describe monitor lizards of the genus Varanus based on a system called Varan-ID. The environment allows direct manipulation of visual components that work as basic elements of an authoring process, to support the identification and description of living beings. These basic elements derives from descriptions present on the Varan-ID database. The proposed tool runs over the web on top of the *Componere* framework.

The main contribution of this paper is our unified approach to produce a virtual lab for taxonomic description, combining the perspective of tools to describe specimens with the virtual laboratory model. It involved the design of a new component based description approach, in which components work as basic descriptive building blocks.

Future works include to expand the laboratory features, enabling it to better integrate with real world resources, i.e., fetching images and other kinds of media of real world specimens, including them in the description process. We are also working to generalize the process of building description components, enabling smooth expansion to new descriptors and other domains.

6. ACKNOWLEDGMENTS

This work was partially funded by CAPES, CNPq, FAPESP, CAPES-COFECUB (AMIB project) and INCT in Web Science (CNPq 557.128/2009-9).

References

[1] M. Duarte and B. P. Butz. An intelligent universal virtual laboratory (uvl). *IEEE Transactions on Education*, 51(1):2 – 9, 2008.

[2] K. M. B. et al. Genetics education. *Genetics*, 179:1151 – 1155, 2008.

[3] M. K. et al. Enhancement of student learning in experimental design using a virtual laboratory. *IEEE Transactions on Education*, 51:76–85, 2008.

[4] J. Ma and J. V. Nickerson. Hands-on, simulated, and remote laboratories: A comparative literature review. *ACM Comput. Survey*, 38(3):1–24, 2006.

[5] P. Mabee, M. Ashburner, Q. Cronk, G. V. Gkoutos, M. Haendel, E. Segerdell, C. Mungall, and M. Westerfield. Phenotype ontologies: the bridge between genomics and evolution. *Trends in Ecology and Evolution*, 22(7):345 – 350, 2007.

[6] Z. Nedic, J. Machotkd, and A. Najhlsk. Remote laboratories versus virtual and real laboratories. *Frontiers in Education, 2003. FIE 2003. 33rd Annual*, 1:1 – 6, 2003.

[7] T. G. S. Olabarriaga and P. de Boer. A virtual laboratory for medical image analysis. *IEEE Transactions on Information Technology in Biomedicine (TITB)*, 14(4):979–985, 2012.

[8] A. Santanche, M. Mota, D. Costa, N. Oliveira, and C. O. Dalforno. Componere autoria na web baseada em componentes. *WebMedia*, pages 91–98, 2009.

SWRL Editor: Uma ferramenta Web para visualização e edição de regras SWRL

João Paulo Orlando, Adriano Rívolli, Kleberson J. Serique, Dilvan A. Moreira

Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
Av. Trabalhador Sancarlense, 400 – São Carlos – SP
orlando@icmc.usp.br, rivolli@icmc.usp.br, serique@icmc.usp.br, dilvan@icmc.usp.br

ABSTRACT

The Semantic Web is a way to associate explicit meaning to the content of web documents to allow them to be processed directly by machines. To allow this processing, computers need to have access to structured information collections and rule sets to reason about these content. The Semantic Web Rule Language (SWRL) allows the combination of rules and ontology terms, defined using the Web Ontology Language (OWL), to increase the expressiveness of both. However, as rule sets grow, they become difficult to understand and error prone, especially when used and maintained by more than one person. If SWRL is to become a true web standard, it has to be able to handle big rule sets. To find answers to this problem, we first surveyed business rule systems and found the key features and interfaces they used and then, based on our finds, we proposed techniques and tools that use new visual representations to edit rules in a web application. They allow error detection, rule similarity analysis, rule clustering visualization and atom reuse between rules. These tools are implemented in the SWRL Editor, an open source plug-in for Web-Protégé (a web-based ontology editor) that leverages Web-Protégé's collaborative tools to allow groups of users to not only view and edit rules but also comment and discuss about them. We evaluated our solution comparing it to the only three SWRL editor implementations available in the literature and showed that it implements more of the key features present in traditional rule edition systems.

Categories and Subject Descriptors

H.3.4 [**Information storage and retrieval**]: Systems and Software – Semantic Web.

General Terms

Algorithms, Management.

Keywords

SWRL rules; Rule composition; Rule visualization; Data annotation; Semantic Web;

1. INTRODUÇÃO

A Web Semântica é uma maneira de explorar a associação de significados explícitos aos conteúdos de documentos presentes na Web, para que esses possam ser processados diretamente ou indiretamente por máquinas [1]. Para possibilitar esse processamento, os computadores necessitam ter acesso a coleções estruturadas de informações (dados e metadados) e a conjuntos de regras de inferência sobre esses conteúdos (que ajudem no processo de dedução automática) para que seja possível o raciocínio automatizado sobre os mesmos [1].

A Web Semântica renovou e aumentou o interesse em sistemas baseados em regras e seu desenvolvimento [2]. A SWRL (*Semantic Web Rule Language*) é a linguagem padrão para regras da Web Semântica, muitas áreas de estudo na computação estão usando SWRL, entre elas podemos citar: Serviços sensíveis ao contexto, gestão de energia em ambientes domésticos, sistemas de e-learning, gerenciamento de SLA (*Service Level Agreement*) para serviços de IPTV (*Internet Protocol Television*), cálculos de redes de co-autoria em redes sociais, sensores em ambientes inteligentes, gerenciamento inteligente de fotos digitais, extração de características (Ex.: de frequência cardíaca) no acompanhamento contínuo de eletrocardiogramas, etc.

Um dos principais problemas no uso de SWRL é que a medida que o conjunto de regras cresce, os desenvolvedores começam a ter problemas no seu gerenciamento. Um conjunto grande de regras se torna difícil de compreender e sujeito a erros, especialmente quando mantido por mais de uma pessoa (colaborativamente). Apesar disso, o SWRL carece de uma ferramenta para a manipulação de regras que seja mais do que um simples editor de texto. Os editores existentes para SWRL, encontramos apenas 3 na literatura, não utilizam as ferramentas e interfaces normalmente encontradas em editores de regras usados em outras áreas [2], notadamente os editores para regras de negócios. Eles carecem de uma quantidade maior de recursos para os seus usuários, como árvores de decisão e agrupamento de regras, facilidades que existem a muito tempo em outras áreas. Essa é a lacuna que este trabalho pretende preencher.

Por essa razão, o objetivo central deste trabalho é desenvolver soluções para implementar um editor de regras SWRL que incorporem as melhores técnicas em Edição/Visualização de outros domínios de regras (ex. Domínio de regras de negócio) para obter, ao final, uma ferramenta mais completa do que as atuais para a Edição/Visualização de regras SWRL.

Para alcançar esse objetivo, o Web-Protégé foi escolhido como base para esse editor por ser a versão Web do Protégé, um editor de ontologias consolidado e de código aberto, que facilita a colaboração entre desenvolvedores de ontologias.

2. SWRL

As regras são usadas para inferir novos conhecimentos a partir de bases de conhecimento em OWL [5]. Regras SWRL são divididas em duas partes: antecedente e consequente. Cada regra é uma implicação entre o antecedente e o consequente, que pode ser entendida como: se todas as condições do antecedente são

verdadeiras, então as condições do consequente são executadas. Ambas as partes consistem em uma conjunção de zero ou mais átomos, não permitindo disjunções ou negação.

Cada átomo é composto por um predicado e uma ou mais variáveis (o número de variáveis é determinado pelo tipo do predicado). Os seis tipos de predicados são [6]: *Class*, *Object property*, *Data valued property*, *Data range*, *Same/different*, *Built-in*.

3. TRABALHOS RELACIONADOS

Na literatura, apenas são encontradas as seguintes ferramentas para Edição/Visualização de regras SWRL: o SWRLTab, Axiomé e o ACE View. As duas primeiras plug-ins do Protégé 3 e a ultima um plug-in do Protégé 4.

- A ferramenta SWRL Tab é um ambiente de desenvolvimento para regras SWRL que usa a API Protégé-OWL (OWL 1.1). O SWRL Tab tem suporte a edição, execução, sugestões de termos (autocomplete) e identificação de erros sintáticos em regras SWRL.
- O Axiomé usa a mesma API do SWRL Tab. Para o usuário o Axiomé amplia a quantidade de recursos voltados a organização, visualização e edição das regras SWRL [6].
- O ACE View [7] utiliza o *Attempto Controlled English* (ACE) para visualizar e editar ontologias (OWL) e para regras (SWRL) é apenas suportada a visualização. ACE é um subconjunto do Inglês (linguagem natural controlada), responsável por representar o conhecimento de forma inequívoca, ou seja, cada frase representa uma única forma lógica.

Além disso, foram estudados motores de inferência para regras de negócios, pois essas encontram-se em desenvolvimento e uso a muito mais tempo que a própria existência da Web. Isso tornou a investigação dessas ferramentas um passo importante para o levantamento de requisitos da nova ferramenta. A análise destas ferramentas foi conduzida a partir da revisão de [2]. As ferramentas analisadas foram: BRS Rule Track, CLIPS, Drools, Ferramentas ORM (*Object Role Modeling*), Fico Blaze Advisor, IBM ILOG, LibRT Rule Management System (RMS), OpenLexicon, Oracle Business Rules, RuleXpress, SAP NetWeaver Business Rules Management ou QuickRules Editor, Translator (*TRANSlator from LAnguage TO Rules*), Visual Rules.

4. SWRL EDITOR

O SWRL Editor foi desenvolvido como um plug-in para o Web-Protégé, a versão web do editor de ontologias Protégé 3 [8]. Protégé 3 é um editor de ontologia gratuito, extensível e de código aberto. Ele foi desenvolvido pelo *Stanford Center of Biomedical Informatics Research* (*Stanford University*). Já por sua vez, o Web-Protégé foi desenvolvido em GWT (*Google Web Toolkit*) para fornecer um ambiente próximo ao desktop. No lado do servidor ele usa a mesma API do Protégé 3 (Protege-OWL).

Uma das características do Web-Protégé é sua arquitetura de plug-ins, pois permite que recursos novos sejam adicionados gradualmente. Com isso, ele é uma plataforma flexível, mas ao mesmo tempo sólida para o desenvolvimento de tecnologias da Web Semântica. Como um plug-in, o SWRL Editor implementa uma *tab* e uma por *portlet* (componentes para adição de novos plug-ins no Web-Protégé) e através de RPCs comunica com o servidor.

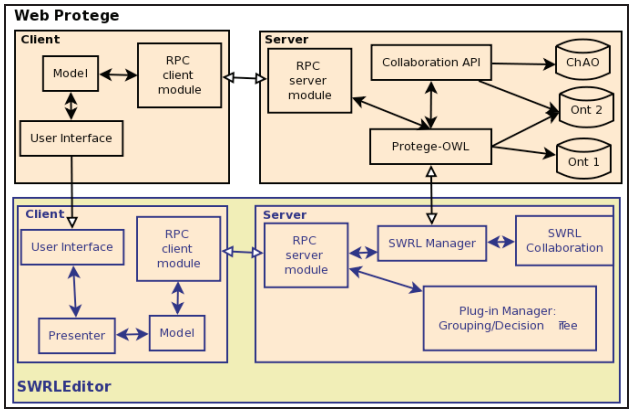


Figura 1.Integração entre o SWRL Editor e o Web-Protégé.

A Figura 1 mostra a integração das arquiteturas do SWRL Editor e Web-Protégé. No lado do Web-Protégé, a Protégé-OWL permite a manipulação das ontologias e a *Collaboration API* apóia os serviços de colaboração, tais como, anotação e controle de alterações em termo da ontologia. No lado do Cliente do Web-Protégé, existe uma copia parcial da ontologia (*Model*) e um módulo de RPC (para comunicar com o servidor).

Já a arquitetura do SWRL Editor está dividida em Servidor (Seção 4.1) e Cliente (Seção 4.2). O Servidor é responsável por modificar e armazenar os arquivos da ontologia. Já no Cliente estão as interfaces gráficas para: representação, edição e busca de regras, entre outras funções.

4.1 Servidor

O Servidor centraliza as operações com as regras, por exemplo, ontologias e regras são carregadas de seus arquivos apenas uma vez e ficam disponíveis para todos os clientes através de chamadas RPC. Além disso, a centralização das operações é um ponto chave para que a ferramenta possa funcionar colaborativamente, já que o servidor atua como nó central de distribuição das alterações feitas por cada usuário.

4.1.1 Gerenciador

O gerenciador (*SWRL Manager* da Figura 1) foi criado para intermediar todas as consultas ou modificações no conjunto de regras da ontologia com a API Protégé-OWL. Ele converte os elementos da ontologia, passando do formato da API Protégé-OWL para uma representação do SWRL Editor e na edição de uma regra fazem o processo inverso. Por esse fato, o gerenciador torna-se importante, dado que a ferramenta fica menos dependente arquitetura do Web-Protégé. Esse requisito foi considerado importante, pois a versão 4 do Protégé utiliza novas APIs não compatíveis com as utilizadas no Web-Protégé. Para facilitar uma futura migração para essas novas APIs, foi estabelecida uma representação interna do SWRL Editor para regras.

4.1.2 Gerenciador de atualizações

O Web-Protégé, por tratar-se de uma ferramenta de edição colaborativa de ontologias, possui um gerenciador de mudanças na ontologia (*Collaboration API*) [8]. Porém, esse gerenciador de atualizações não controla alterações nas regras SWRL. Então, a partir da metodologia empregada na *Collaboration API*, foi criado um gerenciador de atualizações para as regras.

Nesse novo gerenciador (*SWRL Collaboration* da Figura 1), a cada nova alteração nas regras é incrementado um número de versão contido no Servidor. Também é mantida uma lista de alterações, contendo o tipo de alteração e o que foi modificado. Já no cliente é mantido um conjunto de regras com seu número de versão. Em um intervalo definido nas configurações do Web-Protégé (por padrão 5 segundos), o cliente faz uma chamada RPC em que envia o número da versão local das regras para o servidor. Já do servidor, é retornada uma lista de alterações, que vão desde a versão do cliente até a versão do servidor. Quando o cliente recebe as alterações, ele atualiza o seu conjunto de regras e automaticamente os recursos de interfaces as mostram.

Só o gerenciador de atualizações modifica o conjunto de regras do cliente. As operações que modificam os conjuntos de regras (inserir, editar e excluir) enviam suas alterações para o servidor e, caso essas alterações sejam aceitas e implementadas no servidor, é retornado ao cliente o aviso de sucesso na operação e esse força o gerenciador de atualizações a buscar imediatamente pelas novas alterações.

4.1.3 Sistemas de carregamento automático de algoritmos

Um ponto bastante discutido neste trabalho foi a questão de quanto um algoritmo de visualização [15] é importante para várias ontologias ou vários usuários. Mais de um algoritmo foi desenvolvido para criar essas visualizações, porém mesmo assim não é possível garantir que eles sempre sejam úteis para todos os usuários. Por esse motivo, no SWRL Editor foi implementado um sistema para carregamento automático de algoritmos para criação de agrupamentos [4] e árvores de decisão [9].

O sistema de carregamento automático de algoritmos tornou-se uma ótima estratégia para usuários mais avançados que tenham interesse em desenvolver seus próprios algoritmos. Ele facilita essa operação já que para inserir um novo algoritmo é apenas necessário gerar um arquivo JAR com as implementações de interfaces Java e o colocá-lo na pasta em que está o servidor do SWRL Editor.

4.2 Cliente

Já no Cliente, as principais funcionalidades implementadas foram: utilização de rdfs:Label na visualização e edição de regras SWRL (Seção 4.2.1) e melhorias na composição de regras (Seção 4.2.2).

4.2.1 rdfs:Label

O SWRL Editor permite aos usuários trabalharem com os rdfs:Labels. O rdfs:Label é uma descrição para o termo da ontologia (representado pelo seu identificador único rdf:ID). Os rdfs:Labels podem tornar mais fácil o entendimento por não serem apenas um ID. Usando a ontologia para categorizar os fenótipos de autismo [3], na Tabela 1 é mostrado alguns rdf:ID de termos da ontologia com o seu respectivo rdfs:Label. Fica evidente que os rdfs:Labels podem tornar mais fácil o entendimento e a edição das regras.

Tabela 1. rdf:ID versus rdfs:Label

rdf:ID	rdfs:Label
ados1:AUTISMC000004	Autism Diagnostic Observation Schedule
ados4:AUTISMC000001	Autism Diagnostic Observation Schedule
Autism-core:AUTISMC100000	Phenotype Record

Como rdfs:Labels não são únicos, podem ocorrer redundância na edição, pois um mesmo rdfs:Label (descrição) pode estar em mais de um termo da ontologia. Um exemplo está na Tabela 1 em que os termos `ados1:AUTISMC000004` e `ados4:AUTISMC000001` possuem uma mesma descrição. A solução para isso está apresentada na próxima seção.

4.2.2 Composição

O processo de criação de regras disponibiliza dois modos de trabalho [4]:

- Editor – Utiliza a representação hierárquica para apresentar e organizar os átomos da regra e um formulário para editar;
- SWRL – Editor SWRL com *Highlight*, no qual o usuário escreve a regra utilizando a sintaxe SWRL;

As contribuições deste trabalho para a composição foram a sugestão de termos (autocompletar) e o uso dos rdfs:Labels na edição das regras. A primeira funcionalidade consiste em sugerir termos à medida que o nome de algum elemento da ontologia é digitado. Como ontologias podem ser muito grandes, por padrão, cada vez que o usuário acrescenta mais uma letra o autocompletar sugere no máximo 8 termos da ontologia (a medida que o usuário digita mais letras um desses 8 termos vai ser o que ele procura).

Na Figura 2 (A) é mostrada a sugestão de termos por meio do autocompletar. O usuário digitou "Au" e o autocompletar sugeriu 4 termos. Nesse exemplo, os rdfs:Labels (descrições do identificadores únicos da ontologia) estão sendo usados como nomes das classes OWL. Além de rdfs:Labels terem nomes mais intuitivos, geralmente com poucas letras já se consegue uma lista de poucos termos.

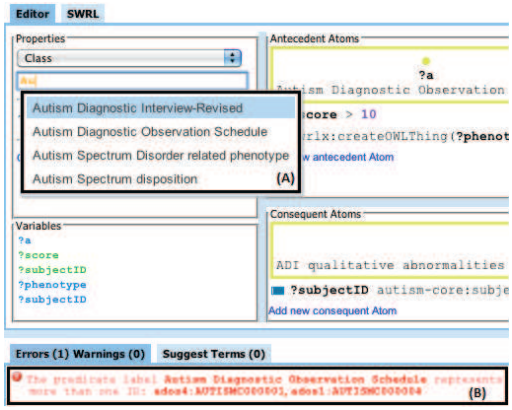


Figura 2. Composição – (A) Autocompletar; (B) Mensagem de erro de redundância de rdfs:Labels.

Foi também implementado o tratamento de erros de redundância que os rdfs:Labels podem gerar. Na Figura 2 – (B) é mostrada a seguinte mensagem de erro: "`The predicate label` **`Autism Diagnostic Observation Schedule`** `represents more than one ID:` **`ados4:AUTISMC000001, ados1:AUTISMC000004`**". O usuário poderá então usar um dos dois rdf:IDs para representar o elemento e assim evitar o erro.

5. COMPARAÇÃO DE FERRAMENTAS

Nesta seção, é apresentada uma comparação entre recursos/interfaces das ferramentas de manipulação de SWRL que são encontradas na literatura. São comparadas ao SWRL Editor as seguintes ferramentas: SWRL Tab, Axiomé e ACE View. Uma

vantagem sobre as demais é o fato de que seus recursos estão disponíveis na Web e, por essa razão, podem beneficiar-se dos recursos de colaboração que a Web trás e, mais especificamente, dos recursos que o Web-Protégé trás para colaboração.

Para fazer a comparação, utilizaremos os principais recursos/interfaces encontrados em sistemas de regras de negócios e regras SWRL [2]. A Tabela 2 mostra a comparação entre as quatros ferramentas (para o SWRL Editor são incluídos os recursos desenvolvidos em [4] e [9]):

Tabela 2. Comparação das características entre as principais ferramentas para edição de regras SWRL

Recurso/ Interface	SWRL Editor	SWRL Tab	Axiomé	ACE View
Plataforma	Web	Desktop	Desktop	Desktop
Tabela de decisão				
Árvore de decisão	✓			
Diagrama gráfico			✓	
Editor de texto	✓	✓	✓	
Editor de texto com *Highlight*	✓			
Formulário/ *Template*	✓		✓	
Linguagem natural	Parcial		Parcial	✓
Representação visual	✓		✓	
Agrupamento	✓		✓	
Sugestão de termos	✓	✓		
Mecanismos de busca	✓	✓	✓	✓
Customização	✓			
Adição de novos algoritmos	✓			
Edição de Regras com rdfs:Label	✓			
Detecção de Erros	✓	✓	✓	
Execução das regras	✓	✓	✓	

6. CONCLUSÃO

A linguagem SWRL é muito útil para desenvolvedores de ontologias, pois suas regras podem fazer inferências de novos conhecimentos sobre indivíduos de uma ontologia (OWL). Um dos principais problemas no uso de SWRL é que grandes conjuntos de regras são mantidos de forma colaborativa e, à medida que crescem, os desenvolvedores começam a ter problemas no seu gerenciamento. Para auxiliar no desenvolvimento desses conjuntos de regras, foi desenvolvido um Editor/Visualizador colaborativo de regras SWRL. Para isso foram incorporadas as melhores técnica em Edição/Visualização de outros domínios de regras (ex. Regras de negócio). Ampliando o conjunto de recursos desenvolvidos em [4] obteve-se o SWRL Editor, o Editor/Visualizador de código aberto mais completo para regras SWRL disponível no momento.

Após seu desenvolvimento, foi feita uma comparação com as outras três ferramentas de Edição/Visualização de SWRL disponíveis na literatura. Nessa comparação foi possível constatar que o SWRL Editor tem vantagem sobre elas, pois ele possui a maioria dos recursos citados em [2]. Além disso, até mesmo recursos que já estão em outras ferramentas para SWRL foram melhorados.

Outra contribuição foi mostrar que é possível fazer uma ferramenta Web para a edição de regras SWRL que tem os recursos de diversas ferramentas desktop. Isso só é possível agora devido a novas tecnologias da Internet, como HTML 5. Além disso, por estar na Web, foi adicionado o recurso de colaboração para os desenvolvedores de regras SWRL.

Como trabalhos futuros, podem ser citados: Detecção de erros semânticos; Detecção de relevância e relações semânticas entre as Regras e Exploração de técnicas que visualizem a localização das regras dentro do contexto da ontologia.

7. AGRADECIMENTOS

Esse trabalho contou com o financiamento do CNPq.

8. REFERÊNCIAS

[1] Berners-Lee, T., Fischetti, M. 2008. Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web. HarperSanFrancisco, ISBN: 978-0062515872.

[2] Zacharias, V. 2008. Development and verification of rule based systems – a survey of developers. Rule Representation, Interchange and Reasoning on the Web: International Symposium, Orlando, Florida, USA, (5321) 6-16. DOI=10.1007/978-3-540-88808-6_4.

[3] Hassanpour, S., O'connor, M. J., Das, A. K. 2011. Evaluation of Semantic-Based Information Retrieval Methods in the Autism Phenotype Domain. AMIA Annual Symposium, Washington, DC, 569–577.

[4] Silva, A. R. 2012. Aprimorando a visualização e composição de regras SWRL na Web. Dissertação - Instituto de Ciências Matemáticas e de Computação (USP), acesso julho 2012.

[5] O'Connor, M. J., Knublauch, H., Tu, S., Grosof, B., Dean, M., Grosso, W., Musen, M. 2005. Supporting Rule System Interoperability on the Semantic Web with SWRL. Fourth International Semantic Web Conference (ISWC2005), Galway, Ireland, (2005). Disponível em: http://bmir.stanford.edu/file_asset/index.php/1157/BMIR-2005-1080.pdf, acesso em Maio de 2012.

[6] Hassanpour, S., O'Connor, M. J., Das, A. K. 2009. Exploration of SWRL Rule Bases through Visualization, Paraphrasing, and Categorization of Rules. Proceedings of the 2009 International Symposium on Rule Interchange and Applications (RuleML 2009), Las Vegas, Nevada, 246–261. DOI=10.1007/978-3-642-04985-9_23.

[7] Kaljurand, K. 2008. ACE View – an ontology and rule editor based on Attempto Controlled English. Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions (OWLED 2008), (432) 1-12. DOI=10.5167/uzh-8822.

[8] Tudorache, T., Vendetti, J. e Noy, N. F. 2008. Web-Protégé: A Lightweight OWL Ontology Editor for the Web. OWLED 2008, Karlsruhe, Germany.

[9] Orlando, J. P. 2012. Usando aplicações ricas para internet na criação de um ambiente para visualização e edição de regras SWRL. Dissertação - Instituto de Ciências Matemáticas e de Computação (USP), acesso julho 2012.

Mconf-Mobile: videoconferência BigBlueButton no Android

Felipe Cecagno, Valter Roesler
Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre, Rio Grande do Sul, Brasil
{fcecagno, roesler}@inf.ufrgs.br

RESUMO

Este artigo apresenta um dos resultados desenvolvidos no projeto Mconf. O Mconf-Mobile é um aplicativo de código aberto para dispositivos móveis com sistema operacional Android. Ele permite interação entre usuários de smartphones e tablets e usuários de computadores de mesa através do sistema de webconferência BigBlueButton. Serão apresentados alguns objetivos do projeto, a arquitetura da solução e suas principais funcionalidades.

ABSTRACT

This article presents a result of the Mconf project. The Mconf-Mobile is an application for mobile devices with Android operational system. It makes possible to interact with other users in a BigBlueButton videoconference. We present some goals of the project, the solution's architecture and its main functionalities.

Categories and Subject Descriptors

C.3 [**Special-purpose and application-based systems:**]: Real-time and embedded systems; H.4 [**Information systems applications**]: Communications Applications: Computer conferencing, teleconferencing, and videoconferencing

Keywords

Mconf, Webconferência, Android, BigBlueButton, Código aberto

1. INTRODUÇÃO

Dispositivos móveis inteligentes vêm ganhando a cada dia mais espaço e popularidade. A troca de telefones celulares "burros" por smartphones acontece por diversos motivos, e um deles é a grande variedade de aplicativos disponíveis. Dentre os oferecidos na loja oficial de aplicativos do Android, os mais populares são os aplicativos de entretenimento [3], categoria que inclui os jogos, que aproveitam a capacidade de processamento dos dispositivos e oferecem diversão aos usuários.

Além dos jogos, aplicações de vídeo em tempo real vêm crescendo no mercado de aplicativos. Clientes VoIP[1] como o Sipdroid e aplicativos populares por sua versão para desktop como o Skype já incluem a funcionalidade de videochamada nos seus aplicativos para dispositivos móveis. Além destes, o Fring permite chamada em grupos de até quatro pessoas. Porém, tanto o Skype quanto o Fring são softwares proprietários, ou seja, o código fonte dos sistemas não está disponível, e não utilizam protocolos padrão conhecidos e/ou abertos, o que impossibilita a interoperabilidade com outros sistemas. Além disso, os dois sistemas usados como exemplo têm restrições quanto a funcionalidades: o Skype permite vídeo chamada apenas entre dois pontos - entre mais pontos, existe o recurso de chamada de voz, ou então um dos usuários deve ter uma conta Premium do sistema; e o Fring possibilita comunicação entre quatro participantes, não sendo possível videoconferência entre muitos participantes.

Os clientes VoIP são mais genéricos e utilizam protocolos conhecidos - o Sipdroid, por exemplo, utiliza o protocolo padrão SIP. Entretanto, tais sistemas possuem conhecidos problemas de usabilidade: são complicados de configurar e de utilizar, o que muitas vezes impede o uso por usuários leigos.

O principal objetivo do projeto Mconf é proporcionar uma solução de código aberto para webconferência com foco na facilidade de uso e integração com dispositivos móveis [5]. A solução integra o sistema de webconferência BigBlueButton [1] com o portal Mconf-Web [4] (baseado no Global Plaza[2]), e prevê interação com dispositivos móveis através do Mconf-Mobile. O Mconf-Web é a aplicação web utilizada no portal `https://mconf.org`.

O projeto Mconf é desenvolvido pelo Laboratório de Projetos em Áudio e Vídeo da Universidade Federal do Rio Grande do Sul, teve início em novembro de 2010 e é financiado pela Rede Nacional de Ensino e Pesquisa no âmbito do programa Grupos de Trabalho da área de Pesquisa, Desenvolvimento e Inovação da RNP.

O código-fonte do Mconf-Mobile está disponível em um repositório público[3] e é licenciado sob a *GNU General Public License*. O aplicativo está disponível gratuitamente para *download* na loja oficial de aplicativos do Android[4].

2. ARQUITETURA

Conforme descrito anteriormente, o Mconf integra dois sistemas existentes de código aberto voltados para videoconferência: o BigBlueButton e o Global Plaza. Entretanto, o

[1] *Voice over IP*, ou Voz sobre IP

[2] `http://www.global-project.eu/`

[3] `https://github.com/mconf/mconf-mobile`

[4] `https://play.google.com/store/apps/details?id=org.mconf.android.mconfmobile`

Mconf-Mobile não se limita a essa integração e pode ser utilizado junto a qualquer outro *front end* integrado ao BigBlueButton, como por exemplo o Moodle, ferramenta de apoio à aprendizagem, ou o Wordpress, plataforma para criação de blogs e sites.

A possibilidade de integração a diversos *front ends* se deve à API do tipo REST que o BigBlueButton oferece, e a um recurso do Android chamado de *Intent Filter*. O *Intent Filter* é utilizado para definir que qualquer URL acessada no Android via navegador Web que contenha o prefixo **bigbluebutton://** ao invés do tradicional **http://** direcione a chamada para a aplicação Mconf-Mobile que é executada automaticamente ao clicar no link. O Mconf-Mobile, por sua vez, utiliza a URL com o prefixo modificado para entrar na sala de videoconferência. A integração do Mconf-Mobile com o portal Mconf-Web será apresentada na subseção 3.4.

O BigBlueButton é um sistema de código aberto para webconferência voltado principalmente para ensino à distância. Ele oferece funcionalidades como interação por áudio, vídeo e bate-papo (público e privado), além de compartilhamento da tela do computador e apresentação síncrona de slides e documentos. Além disso, o BigBlueButton possui uma grande comunidade de desenvolvedores, com mais de 1200 membros[5].

O cliente web para desktop é desenvolvido na plataforma Adobe Flash e toda a comunicação com o servidor é feita através do protocolo RTMP[6]. Já o servidor do BigBlueButton utiliza o Red5[7], que é uma implementação livre do Adobe Flash Media Server. Além da comunicação por vídeo, realizada de forma transparente entre o cliente Flash e o servidor Red5, são utilizados *Remote Shared Objects* (RSO) e *Remote Procedure Calls* (RPC) para as demais trocas de mensagens, como bate-papo e gerenciamento de status de participantes. Tanto RSO quanto RPC são recursos presentes na especificação do protocolo RTMP [2].

Para áudio, o sistema utiliza um servidor VoIP, que pode ser tanto o FreeSWITCH quanto o Asterisk. No cliente web é utilizada uma implementação de telefone SIP[8] em Flash.

Apesar do Android oferecer suporte à plataforma Flash a partir da sua versão 2.2, optou-se por desenvolver o Mconf-Mobile como um aplicativo nativo através da SDK padrão do Android para permitir compatibilidade com as versões anteriores do Android e, principalmente, para permitir captura e transmissão de áudio e vídeo através do dispositivo móvel, recursos indisponíveis no Adobe Flash Player para Android. A arquitetura do aplicativo será detalhada nas subseções seguintes.

2.1 Flazr

Toda a comunicação RTMP entre o aplicativo e o servidor Red5 é feita através da biblioteca Flazr, uma implementação livre em Java de protocolos de *streaming* multimídia[9]. A biblioteca implementa procedimentos fundamentais do protocolo RTMP, como por exemplo o *handshake* inicial e multiplexação/demultiplexação de mensagens de controle, áudio e vídeo, comandos de RPC e controle de fluxos de dados.

Entretanto, a biblioteca não oferece suporte a RSO, recurso indispensável para realização plena da comunicação com o servidor BigBlueButton. Por exemplo, informações de entrada e saída de participantes, presença de vídeo de um determinado participante ou pedido de atenção através do recurso "levantar a mão" são ações informadas através do objeto **participantsSO**. Outro exemplo é a troca de mensagens públicas de bate-papo, que é realizada através do objeto **chatSO**.

Por isso, o suporte a RSO teve de ser adicionado à biblioteca. A implementação de gerência de objetos compartilhados e da multiplexação/demultiplexação de mensagens teve como base o código do Red5 e a especificação do protocolo RTMP [2].

2.2 BBB-Java

O BBB-Java é a biblioteca responsável pela interação entre um aplicativo genérico em Java e um servidor BigBlueButton. Essa biblioteca foi desenvolvida pelo projeto Mconf para ser utilizada no Mconf-Mobile, mas buscou-se manter um baixo acoplamento entre a biblioteca e o aplicativo Android para possibilitar o desenvolvimento de novos aplicativos não-web integrados ao sistema de webconferência. Além disso, a biblioteca pode ser utilizada na criação de aplicativos robôs que auxiliem na realização de testes funcionais e de carga.

Dentre os recursos oferecidos pela biblioteca estão: acesso a salas de videoconferência, atualização do status dos participantes da sessão, troca de mensagens de bate-papo (público e privado) e recepção e envio de dados de vídeo.

O acesso a salas de conferência pode acontecer de duas formas:

- Através da tela inicial da aplicação: ao executar o Mconf-Mobile, o usuário é convidado a entrar com suas credenciais de acesso do portal `mconf.org` (como pode ser visto na figura 1, e a partir dessa tela ele poderá acessar sua sala de videoconferência pessoal e as salas de comunidades das quais o usuário faz parte.

- Através de um portal Web: a lógica por trás da API do BigBlueButton pode ser implementada por um portal Web, como por exemplo o Mconf-Web ou o Moodle. Ao clicar em um link para entrar numa sala de videoconferência no navegador do Android, a URL de JOIN com o prefixo modificado **bigbluebutton://** fará com que o Mconf-Mobile seja executado automaticamente, e com essa URL ele será capaz de entrar na sala pretendida pelo usuário.

2.3 Mconf-Mobile

[5] `https://groups.google.com/group/bigbluebutton-dev/about`
[6] *Real-Time Messaging Protocol*
[7] `http://www.red5.org/`
[8] *Session Initiation Protocol*
[9] `http://flazr.com`

O Mconf-Mobile é o aplicativo Android que utiliza o BBB-Java para interação com o servidor BigBlueButton. Esse aplicativo foi escrito principalmente em Java, utilizando o *Software Development Kit* (SDK) padrão do Android, e uma pequena parte crítica em desempenho (relacionada à codificação e decodificação de vídeo) foi desenvolvida em C/C++ com o *Native Development Kit* (NDK).

Considerando que o servidor BigBlueButton utiliza um servidor VoIP para lidar com os fluxos de áudio, para o Mconf-Mobile adotou-se a estratégia de integrar ao aplicativo uma solução existente de VoIP de código aberto. Optou-se pelo aplicativo Sipdroid[10], um dos telefones SIP mais populares da loja de aplicativos oficial do Android, adicionando assim a funcionalidade de interação por áudio. Dentre os codecs suportados pelo Sipdroid está o Speex, codec padrão de áudio do BigBlueButton, implementado na linguagem C e compilado através do NDK.

O codec de vídeo padrão utilizado no BigBlueButton é o H.263, e a estratégia de solução foi a mesma do Sipdroid. Optou-se por utilizar uma compilação otimizada da biblioteca FFmpeg[11] para Android, gerada através do NDK, que tivesse apenas esse codec habilitado. O FFmpeg é uma biblioteca escrita na linguagem C que implementa diversos codificadores e decodificadores de áudio e vídeo.

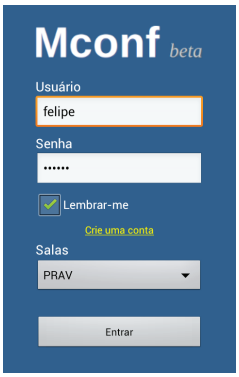


Figura 1: Tela de entrada do Mconf-Mobile

3. PRINCIPAIS FUNCIONALIDADES

A figura 2 ilustra a tela principal do aplicativo, onde é exibida a lista dos participantes da sessão, indicativos de status dos participantes (por exemplo, se o participante está transmitindo vídeo), o vídeo de um participante remoto sobrepondo a interface e os controles de áudio.

O menu principal do aplicativo, acessível a partir do botão **Menu** do dispositivo móvel, exibe opções de início da interação por áudio e vídeo e o recurso de levantar a mão. As principais funcionalidades da aplicação serão descritas a seguir.

3.1 Bate-papo

Assim como o cliente web do BigBlueButton, o aplicativo possui a funcionalidade de bate-papo, que pode ser público

[10] https://play.google.com/store/apps/details?id=org.sipdroid.sipua
[11] http://www.ffmpeg.org/

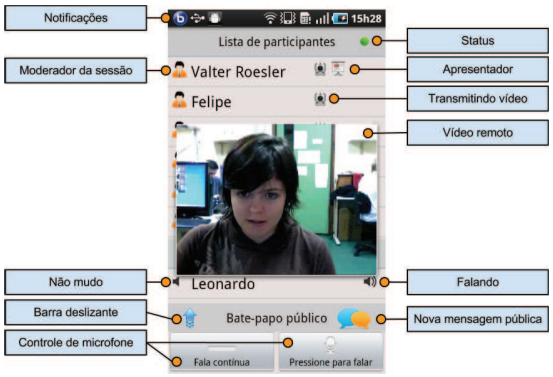


Figura 2: Tela principal do Mconf-Mobile

Figura 3: À esquerda, menu principal do aplicativo; à direita, menu de toque longo sobre o nome de um participante

ou privado. O bate-papo público pode ser observado na figura 2 como uma barra deslizante, que com um toque do usuário encobre a lista de participantes e deixa visíveis todas as mensagens trocadas (juntamente com o originador da mensagem e horário) e uma caixa de texto para envio de novas mensagens.

Já o bate-papo privado é acessado com um toque do usuário sobre o nome do participante com o qual ele deseja iniciar uma conversa. Uma nova tela é aberta, com as mesmas características do bate-papo público, mas com a particularidade de que somente aquele usuário terá acesso à conversação.

3.2 Áudio

O áudio é parte fundamental da interação em um ambiente de videoconferência, logo essa importância é refletida na interface principal do aplicativo. Ao habilitar a conferência de áudio através do botão **Menu** do aparelho (como mostra a figura 3), dois botões aparecem na parte inferior da interface (ver figura 2: "Fala contínua" e "Pressione para falar").

Os botões de controle de áudio são indispensáveis para uma interação de qualidade, pois experimentalmente verificou-se que o microfone de dispositivos móveis é muito sensível a ruído. Além disso, a utilização do botão "Pressione para falar" atenua problemas de eco acústico que podem ocorrer se o usuário não utilizar fones de ouvido.

Através do menu também é possível habilitar o alto-falante do dispositivo, ajustar controle de volume e ganho do mi-

crofone, e também encerrar a conferência de áudio. Estas novas opções são oferecidas somente depois que o usuário selecionar a opção **Iniciar voz**, e por isso não aparecem na figura 3.

3.3 Vídeo

A tela principal da aplicação indica, entre outras informações, quais são os usuários que possuem o recurso de vídeo habilitado. Isso pode ser visto na figura 2, onde ao lado do nome de alguns participantes aparece o ícone de uma webcam. Ao efetuar um toque longo sobre o nome de qualquer usuário, a aplicação exibe um menu de ações possíveis para com aquele usuário, como mostra a figura 3. Para os usuários que tiverem o recurso de vídeo habilitado, a opção "Exibir vídeo" será listada, e permitirá visualizar o vídeo do participante remoto, como ilustra a figura 2.

A janela de vídeo do participante remoto é flutuante e sobrepõe a lista de usuários. Ao colocar o dispositivo na posição horizontal, o vídeo é maximizado ocupando assim toda a tela. A qualquer momento, pressionando o botão **Voltar** do dispositivo, o vídeo é fechado.

Além de visualizar o vídeo de um participante remoto, também é possível transmitir para a videoconferência a imagem capturada da câmera frontal do dispositivo. Esse recurso é acessível através do menu principal do aplicativo, como pode ser visto na figura 3, opção "Iniciar vídeo".

3.4 Integração com portais Web

Conforme descrito anteriormente, o Mconf-Mobile pode ser facilmente integrado a portais Web que servem de *front end* para o serviço de webconferência do BigBlueButton. Na figura 4 é exemplificado o mecanismo de integração do Mconf-Mobile com o portal `mconf.org`.

Figura 4: Modelo de integração com o `mconf.org`

Quando o usuário entra no portal `mconf.org` através do navegador Web do seu dispositivo móvel e coloca suas credenciais de acesso, é exibida na página inicial informações sobre a sua sala pessoal de webconferência. Essa sala possui um link fixo no formato `https://mconf.org/webconf/<id_do_usuário>`, recurso que facilita o convite de pessoas para uma sessão de webconferência.

Como pode ser visto à esquerda na figura 4, um ícone com a forma de um telefone celular é apresentado para que o usuário acesse sua sala através de um dispositivo móvel. Ao clicar nesse ícone, sempre através do navegador Web do Android, é exibida a tela exemplificada à direita na figura 4, que contém um link para acesso à sala (redireciona para uma URL de prefixo **bigbluebutton://**) e um QR-Code, que da mesma forma dá acesso à sala de webconferência quando escaneado por um dispositivo móvel. O mesmo modelo é usado para as salas de webconferência das comunidades de usuários.

Vale ressaltar que a integração realizada no portal `mconf.org` pode ser facilmente reproduzida em qualquer outro portal Web, desde que este seja integrado ao sistema BigBlueButton.

4. CONCLUSÃO

Este artigo apresentou o Mconf-Mobile, um aplicativo para videoconferência em dispositivos móveis com sistema operacional Android. A principal contribuição do projeto Mconf é o desenvolvimento em código aberto de uma solução completa de webconferência que integra desktops e dispositivos móveis. O desenvolvimento modular, conforme apresentado na seção 2, permite que as partes de software produzidas sejam utilizadas em outros projetos de forma independente.

Novas funcionalidades estão previstas para o futuro, como por exemplo a possibilidade de visualização de vários vídeos ao mesmo tempo, integração do módulo de apresentação do BigBlueButton ao aplicativo e desenvolvimento de um novo módulo que permita edição cooperativa de texto através de um bloco de notas. Além disso, será desenvolvido um mecanismo de priorização do fluxo de dados de áudio sobre os demais fluxos a fim de manter a qualidade da conferência de voz em dispositivos com limitação de processamento ou largura de banda de rede.

A aplicação Mconf-Mobile está disponível para download gratuito na loja oficial de aplicativos do Android, e o portal `mconf.org` é aberto e acessível por qualquer pessoa.

5. REFERÊNCIAS

[1] BigBlueButton. Disponível em `http://bigbluebutton.org`. Acessado em julho de 2012.

[2] Real-Time Messaging Protocol (RTMP) Specification. Disponível em `http://www.adobe.com/devnet/rtmp.html`. Acessado em maio de 2012.

[3] Top categories on the Android Market. Disponível em `http://www.appbrain.com/stats/android-market-app-categories`. Acessado em julho de 2012.

[4] F. Bottin, L. C. Daronco, and V. Roesler. Mconf-web: uma ferramenta para gerência de webconferências. In *WEBMEDIA Brazilian Symposium on Multimedia and the Web - Workshop of Tools and Applications, Florianopolis, SC*, 2011.

[5] V. Roesler, F. Cecagno, L. C. Daronco, and F. Dixon. *Mconf: An Open Source Multiconference System for Web and Mobile Devices, Multimedia - A Multidisciplinary Approach to Complex Issues*. ISBN: 978-953-51-0216-8, InTech, 2012.

mCMS: A Content Management System Adapter Architecture for Mobile Devices

Mark Joselli
MediaLab, IC-UFF
http://www.ic.uff.br/~medialab
Niteroi, RJ
mjoselli@ic.uff.br

Eduardo Soluri
Nullpointer Tecnologia
http://www.nullpointer.com.br
Rio de Janeiro- RJ
esoluri@nullpointer.com.br

Jose Ricardo Silva Junior
MediaLab, IC-UFF
http://www.ic.uff.br/~medialab
Niteroi, RJ
jricardo@ic.uff.br

Marcelo Zamith
MediaLab, IC-UFF
http://www.ic.uff.br/~medialab
Niteroi, RJ
mzamith@ic.uff.br

Esteban Clua
MediaLab, IC-UFF
http://www.ic.uff.br/~medialab
Niteroi, RJ
esteban@ic.uff.br

ABSTRACT

Most of the content on the World-Wide Web is designed for desktop computers. Nowadays, with the evolution of mobile phones, smartphones, tablets and Digital TV, this content must be adapted to these different devices, where each have specific characteristics and constraints, like screen dimensions. This adaptation task can consume time and computationl resources, since most solutions resolve it by creating different content versions for different devices. A CMS (Content Management System) is a software that keeps track of every piece of content that is used by websites and portals. A CMS can be adapted for others consumers, like mobile devices. This adaptation can be done by developing plugins for each different device, or creating different resources for each different device, because most of the adaptation needs to be done for each device. In this work, a novel CMS adapter architecture is presented, which is made to adapt the content on CMSs for different devices, through the use of templates that describes how the content must be transformed. Also, since most applications need some form of offline content and normally the cost of network is high, the architecture also provides cache management layer on the client side, in order to provide offline cached content, a data compression mechanism for keeping the data exchange in the network low, and a content version control. This way, the architecture avoid higher data transfers throught the network which in some cases can be slow and expensive over mobile networks.

Categories and Subject Descriptors

D.2.11 [[**SOFTWARE ENGINEERING**]]: Software Architecture

General Terms

Architecture, CMS

Keywords

CMS, Content Management Systems, Content Adaptation, Multimedia, Mobile Devices, Software Architectures

1. INTRODUCTION

Mobile devices, like smartphone, tablets, and Digital TVs have many constraints [6], when compared to PC. For example: hardware constraints (processing power and screen size); user input, (buttons, voice, touch screen and accelerometers); and different operating systems, like Android, Blackberry OS, iPhone OS, Symbian and Windows Mobile. These different characteristics must be taken into account when developing content for this kind of device [5]. Hence, the design and adaptation of content for all these platforms and devices is a tedious task. The presented architecture provides a Content Management System Adapter Architecture, which can be used to adapt all the content for mobile devices.

In general, the content adaptation envelops not only the adaptation of format and types, but also different styles, dimensions, data compression and specifications [4] [3], since quality of experience of the user can suffer from a not adapted (or poor adapted) media [1]. Also, different contents and information can be available for specific devices or systems, requiring a custom adaptation or generation of the content, which the CMS Adapter Architecture (the mCMS) can provide.

Most of the World-Wide Web content is manageable though CMSs (content management system). The CMSs provides a better content organization, increased access to resources and a greater organizational effectiveness as some of its' many advantages. CMSs can be used, natively or with plugins, for others devices, like TVs and Mobile devices, which can require the input of new contents. The mCMS provides an adaptation of the CMSs web content for the different device with the use of templates, providing a generic non-intrusive content adaptation. Also, natively, CMSs does not provide cache mechanism, compression of content, and version control (for the content on the client) like the mCMS

provides.

The mCMS was developed to be used together with popular CMSs like JOOMLA [7] and DRUPAL [2], and also with proprietary CMSs, CRMs (Customer relationship management), ERPs (Enterprise resource planning), and legacy systems. This work shows a test case of the architecture with a mobile application accessing the adapted content provided by the mCMS, which were created from resources made for web sites provided by a proprietary CRM and a JOOMLA CMS.

Mobile devices, normally, do not have connectivity all the time, needing some offline data in order to function when there is no network available. Also, the connectivity can still be very slow and expensive, depending on the network connection and the network carrier, requiring some form of data compression and caches techniques to reduce the data transfers. The mCMS provides a cache mechanism in order to fulfill these requirements. A version control of the content is also provided, so that the content transferred between the server and the client is only the difference between the data that is on the device and the new data from the server.

The mCMS consists in two parts, a server side and a thin client. The server is responsible for adapting the content from different sources throught the uses of templates, compressing the adapted content in order to delivery to the device and controlling the different contents versions. The thin client is an native application installed on the device, which is able to connect to mCMS Server, check for content updates, download the content package and save it locally on the mobile device. Also, using mobile devices can provide lots of features such as camera, navigation (location based features), social networks integration, statistics data, and much more. The thin client is designed to be a hybrid application, based on web patterns (HTML5, Cascading Style Sheets and Javascript) to provide the graphics interface, and a native code used to access the available devices capabilities.

This work is divided as follows, section 2 presents the mCMS architecture and thin client. Section 3 shows the test case in order to validate the presented architecture and finally section 4 presents the conclusions and future works.

2. THE MCMS SERVER

This architecture is build as a web service that adapts and creates content from CMS based on templates. An overview of the mCMS server can be seen on figure 1.

In this figure, the main modules of the architecture can be seen. The organization module is shown, that is composed by the resources provides, which can be the CMSs or even an web portals. These content are gathered by the controller and saved on a database. The controller is the service that connects all the different ones, and make them work together. Afterwards, the resources are adapted though the use of templates and them saved on a proprietary server or on a cloud server, like the amazon S3 and cloud front. Also, resources that come from secure services can be delivered and adapted, using the identity server. The Push notification services that are normally on the brand server, are accessed by a web service, and triggered by the controller

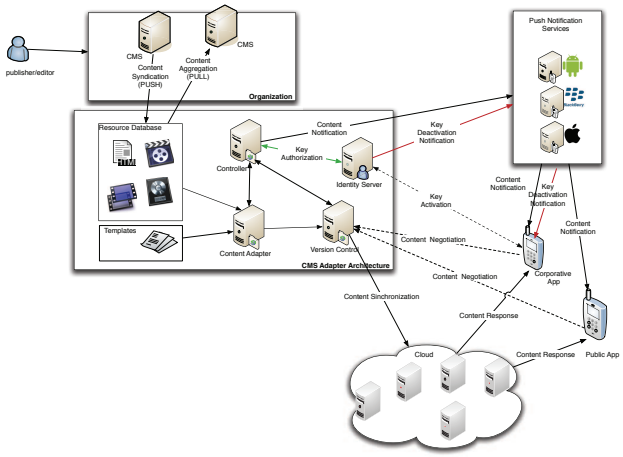


Figure 1: Overview of the mCMS server.

when new content arrives. This process is better illustrated on the Figure 2.

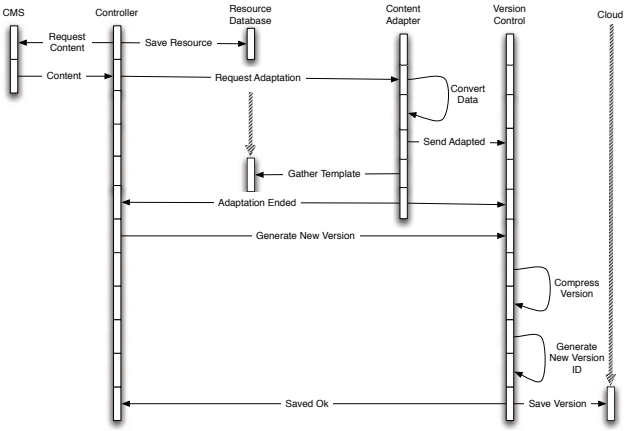


Figure 2: mCMs Execution Flow.

The templates for customization and configuration of the mCMS have to be registered in the controller by a developer/publisher/editor. These templates are built upon an XML based language, and they are the baseline of how the mCMS adapts the contents. Some contents can be specific for some device, like icons and logos, in this case they can be placed on the CMS server or inside the mCMS.

The architecture is built upon components. Based on experiments using this architecture in different application domains, a variety of technical implementations were used to abstract and create a general framework which different modules or plugins can be achieved for context adaption, personalization and contextualization.

The Identity Service is based on an AAA pattern (Authentication, Authorization and Accounting). It is responsible for generate certificates and key tokens used to implement a security connection between devices and the mCMS Server. Using this approach, we are able to guarantee access to special encrypted content and log all communication between

source and client. An extended feature of this enabling component is gather all statistic data available on the server and client sides an compile them on a report database.

The content version control is responsible for keeping each version of the adapted content. Every time the CMS Adapter Architecture gathers content from the source or when templates has changed, it generates a new version on its repository. This enabling component is built using a balk design pattern that only executes an action on an object when the object is on a particular state. In this case it only updates and generates a new version of the content, when all the content gathering and adaptation has finished doing its work.

2.1 Content Adaptation

The content adaptation is responsible for gathering the content from the CMS and adapting it to the required devices. This component normally uses a XML for the configuration of the adapter, and a series of XMLs describing how each of the content should be adapted. The process is simple, it uses a XML to gather information about the CMS service that provides the content, and how it should gather and adapt these content.

The component can gather the information in two ways, by push a notification generated by the CMS service, or in an automatic way, pooling from time to time (scheduler). The push notification requires that this service is implemented in the CMS, but it has the advantage that the content can be adapted as soon as published. In the automatic manner, this implementation is not required, but the content is only published for the media when the service runs.

This enabling component can support different content types and formats, like sound, music, documents, video and HTMLs. The adaptations of these content are made by templates, which are described by a XML document. These templates are implemented using Abstract Factory pattern, which defines the methods available for the content adaptation. The XML document describes how the component gathers the information, and which transformation need to be made for each platform.

The image adaptation is normally done using ImageMagick library, which is an free open source library, that the mCMS uses to change the image properties, like dimension, format and qualities. For the adaptation of audio or video, the architecture uses the FFMEPG library, which is an open source library, in order to convert the video/audio files, this way the adapter can change its properties, like the frame rate, format type, quality and dimensions. The configuration for each common device (mobile device) are registered in the application.

2.2 Thin Client

The thin client is responsible for: gather the data from the mCMS server, provide the server with the characteristics of the device, implement a cache system for the gathered data for offline use and gather of user data for statistics reports.

The client is developed as a framework, in order to be reused in others projects. It is mainly developed in object-C and Java, in order to work with the apple IOS and Google Android platforms. Others platforms are being developed, like for Window mobile, Blackberries and J2ME. This client can show different resources like, HTMLs, images, audio and video. It also can process and show a specify data structure for maps, which are used in the validation. Also, It implements components for localization system, cache management, statistics data gathering and integration with social network, like Facebook, Twitter and Google+.

This client was implemented mainly as a hybrid application, which is a mix of web app with native apps. Web apps are web sites, which all the content and logic are made exclusively for each device, but it cannot access some of the capabilities of the device. Native apps are implemented on the device, with a higher cost of implementation, but it can access all the functionalities of the devices that can be used by developers. Hybrid applications are a mix of both web and native apps, it uses mostly web for the interface, but it is still a native app, so it can use all the resources of the device, similar to the native app. Mostly of the customization of the application is done by XMLs.

The client uses extensively the cache for its contents. It uses this cache for providing the end-user with offline content and also minimize the data transfer between updates. This cache can also be cryptographic or compressed if needed by the application. Also, the cache consistency can be a problem in some devices, since the network data exchange can fail. This client only updates its version after verifying its hash code between the file that should be downloaded and the last updeted version. This client also uses a service to save the sessions' data, in order to gather statistics of the application use, if needed by the server.

In order to update the data in the client, the following workflow is used: first the device, if registered gets and push notification of new content availability; then when the user opens the application, the device authenticates and it gathers from the server the needed resources for its update; in the case the user has not an available connection, the application starts without this update process, by accessing the cached resources.

3. VALIDATION

In order to validate the architecture, the mCMS was used in a commercial project for Shoppings Centers. The application has all its data and contents gathered from different CMSs. This application aims to provide a channel of communication with relevant information about the mall, like the news, the sales, the attractions and also a map system for location of stores and parking.

One of the main problems of this kind application, without the use of the mCMS, is that the content comes from different services and different formats. With that, the mCMS need to have different configuration for gathering the content, but it can still show the same result for the different content providers. The CMSs used in this application comes from two different kinds, a Joomla! CMSs and a proprietary CRM. These services were already being used for the mall web sites, so the same content that was used for the sites is also used for the mobile application. All the content is gathered as pull service, running five times a day. The use

of mCMS architecture, has save some rework that needed to be done when customizing difference platforms for releasing different contents.

The server was developed in PHP and is composed of a linux with apache and MySQL. The thin client used for this application was adapted in order to work on iPhones and iPads and it was developed natively for these devices using the Cocoa framework. This client show the following contents: images, htmls, videos, maps and integration with social media (from Facebook and Twitter). This application also gathers some statistics data that is sent from time to time to the server to analyze the use of the software by the user. These information can be used latter for delivery of promotion, sales and events of the malls. One example of the mCMS that come from a JOOMLA CMSs that also provides the content for the web site can be seen on figure 3.

4. CONCLUSIONS

The devices nowadays have many difference characteristics and constraints, requiring specific content. With that, the devices need the publishing of specify content of the CMS or the adaptation of already published content. This work has presented a new adapter for CMS content adaptation, that provides a layer between the CMS and the devices.

Moreover, devices can have connection constraints, like availability and also this data can be very expensive. This presented architecture also provide an version control system and a cache system in order to provide offline data and also keep the data communication to a minimum.

5. REFERENCES

[1] F. Agboma and A. Liotta. Quality of experience management in mobile content delivery systems. *Telecommunication Systems*, 49(1):85–98, 2010.

[2] D. Buytaert. Drupa cms, June 2012.

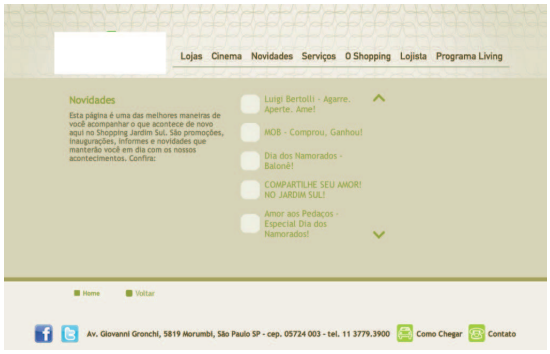
[3] D. Carvalho and F. Trinta. Content adaptation for multiplatform applications. In *Proceedings of the XV Brazilian Symposium on Multimedia and the Web*, WebMedia '09, pages 41:1–41:4, New York, NY, USA, 2009. ACM.

[4] T. Chaari, F. Laforest, and A. Celentano. Adaptation in Context-Aware Pervasive Information Systems: The SECAS Project. *Int. Journal on Pervasive Computing and Communications(IJPCC)*, 3(4):400–425, Dec. 2007.

[5] A. Hildebrand, T. C. Schmidt, and M. Engelhardt. Mobile elearning content on demand. *Information Sciences*, 5(2):94 – 103, 2007.

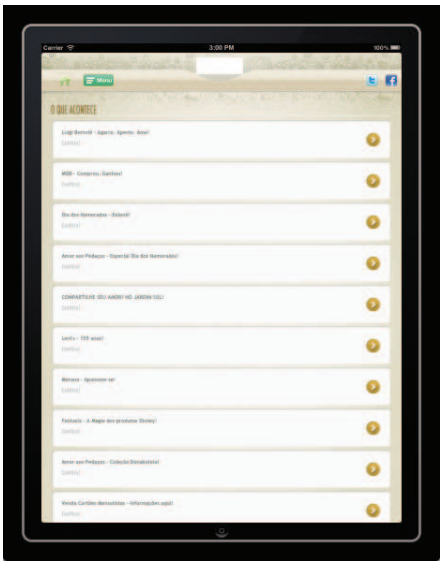
[6] M. Joselli and E. Clua. grmobile: A framework for touch and accelerometer gesture recognition for mobile games. In *Proceedings of the 2009 VIII Brazilian Symposium on Games and Digital Entertainment*, SBGAMES '09, pages 141–150, Washington, DC, USA, 2009. IEEE Computer Society.

[7] I. Open Source Matters. Joomla cms, June 2012.



(a) A list on the web site

(b) A list on a smartphone



(c) a list on a tablet

Figure 3: The mCMS in action adapting html content and image. The images were edited in order to remove the mark of the client (the white boxes).

aNa: API for NCL Authoring

Joel A. F. dos Santos
Laboratório MídiaCom,
Instituto de Computação,
Universidade Federal
Fluminense
Niterói, RJ, Brazil
joel@midiacom.uff.br

Julia V. da Silva
Laboratório MídiaCom,
Instituto de Computação,
Universidade Federal
Fluminense
Niterói, RJ, Brazil
julia@midiacom.uff.br

Renan R. Vasconcelos
Programa de Engenharia de
Sistemas e Computação,
COPPE, Universidade Federal
do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
renanrv@cos.ufrj.br

Wagner Schau
Programa de Engenharia de
Sistemas e Computação,
COPPE, Universidade Federal
do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
schau@cos.ufrj.br

Cláudia Werner
Programa de Engenharia de
Sistemas e Computação,
COPPE, Universidade Federal
do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
werner@cos.ufrj.br

Débora C. M. Saade
Laboratório MídiaCom,
Instituto de Computação,
Universidade Federal
Fluminense
Niterói, RJ, Brazil
debora@midiacom.uff.br

ABSTRACT

There are several available NCL (Nested Context Language) authoring and formatting tools using their own metamodel to represent the code they are working on. This paper presents an API that implements a metamodel specifically created to represent NCL documents. This API helps the creation of tools to manipulate NCL documents and brings some benefits to code reuse to the Digital TV Systems development context. The API here presented is called aNa, an acronym for API for NCL Authoring. aNa is available for free download and open for contributions. aNa has already been used for the development of some NCL authoring and analysis tools.

1. INTRODUCTION

Nested Context Language (NCL) is the standard declarative language of the Brazilian Digital TV system [1] and ITU standard for IPTV services [8]. The growth in the use of NCL for the creation of interactive content shall increase the need for tools to help interactive application creation. Those tools can be authoring tools, analysis tools and even presentation tools.

Usually, each tool that has been created to manipulate NCL code implements its own metamodel to represent the code it is working on. If one does not create a model to represent the document, it is common to use available XML parsing tools, like DOM (Document Object Model) [4] or SAX (Simple API for XML)[1]. Although those parsers produce a good result and help the tool developer to manipulate NCL code, their purpose is generic, allowing the parsing of any XML document.

This paper presents a metamodel specifically created to represent NCL documents in a model-oriented environment and help the creation of tools to manipulate NCL code. This metamodel is implemented in Java by an API called aNa, an acronym for API for NCL Authoring. aNa is available for free download and is open for contributions. aNa has already been used for the development of some NCL authoring and analysis tools.

The remaining of this paper is structured as follows. Section 2 summarizes available tools to help NCL authoring. Section 3 presents aNa structure. Section 4 discusses the API implementation and its main characteristics that facilitate the creation of NCL tools, like code reuse. Section 5 presents some tools that already use aNa and Section 6 concludes the paper and presents future work.

2. NCL AUTHORING TOOLS

Currently there are some available tools for helping the authoring of NCL documents. Those tools present different capabilities, focusing on different user profiles. The following paragraphs present those NCL authoring tools.

Composer 3 [9] provides a single authoring environment suitable for different user profiles, from home users to content producers. Composer 3 proposes the basis for building an integrated environment that can adapt itself to various profiles and support non-functional requirements.

NCL-Eclipse [2] is an authoring tool, available as an Eclipse plugin, which helps the author creating an NCL document. It presents some facilities to help coding as: code completing, code highlighting and errors and warnings highlighting.

NCL-Inspector [7] is a tool based on other tools for code quality critique, which supports the authoring of NCL applications. It supports the author in terms of code quality.

XTemplate 3.0 [6] defines a language and tools for the authoring of NCL documents using composite templates. Composite templates define generic structures of nodes and links that can be reused in different document compositions, facilitating the authoring of interactive applications in Dig-

[1]http://www.saxproject.org/

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WebMedia '12, October 15-18, 2012, São Paulo, SP, Brazil
Copyright 2012 ACM X-XXXXX-XX-X/XX/XX ...$10.00.

ital TV systems.

Berimbau iTV Author [3] is a graphical authoring tool for digital TV applications aimed at media professionals who have no programming knowledge. The tool provides a simple and intuitive interface and a media repository to be used while creating the application.

Notice that each one of those tools uses its own metamodel, in an informal manner, to represent NCL documents. If there were an NCL reusable API that implemented an unified metamodel available to model and represent NCL documents, the programming effort to develop NCL tools would be much smaller. This paper proposes such API.

3. ANA METAMODEL

aNa was created to represent an NCL document as a model. Its structure is optimized so the author of NCL tools that manipulate XML code does not need to worry about the language representation, but with the model itself. Every NCL element is represented as a class, which will be called element class. An element class contains the same attributes of the NCL element it represents. Every element class in aNa inherits from the basic type *NCLElement*.

The element classes follow the same hierarchy of the NCL elements, therefore, an element class will be associated to all element classes that represent its child elements. The cardinality of those associations is defined following the NCL language specification. By representing parent-child relations as associations, aNa makes it possible to navigate from an element to its children and the opposite as well. Figure 1 presents the related representation in aNa of the element <*ncl*>, represented by class NCLDoc, which has attributes *id*, *title* and *xmlns* and the elements *head* and *body* as children.

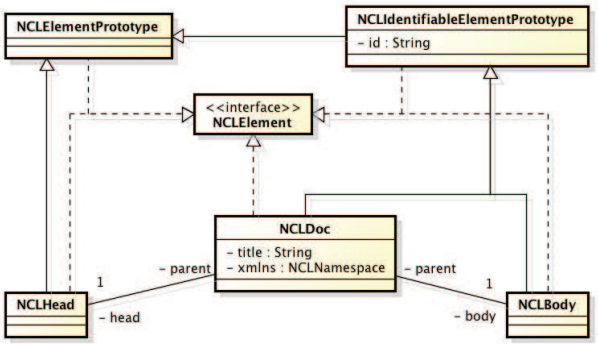


Figura 1: NCLDoc class representation

NCL also defines attributes that refer to other elements. This reference is usually done by defining the attribute value equal to the referred element id (more common) or another attribute of the referred element. In order to improve navigation over element classes, aNa represents those references as associations between them. Listing 1 presents an example of reference between elements and Figure 2 presents how those elements are represented in aNa.

Listing 1: Element reference example

```
1 <regionBase>
2     <region id="reg1"/>
3 </regionBase>
4 <descriptorBase>
5     <descriptor id="desc1" region="reg1"/>
6 </descriptorBase>
```

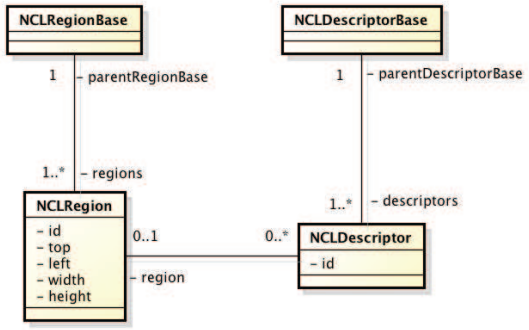


Figura 2: Element reference representation

Listing 1 presents an NCL document part. The example defines a <*regionBase*> element that has a <*region*> element as child. The <*region*> element *id* attribute is a String that represents its identification. The example also defines an element <*descriptorBase*>, that has a <*descriptor*> element as child. The <*descriptor*> defines two attributes, *id* and *region*. The *id* attribute represents its identification and the *region* attribute is a String that represents the identification of the <*region*> used by that <*descriptor*>. In Figure 2, it is possible to observe that the *region* attribute of the <*descriptor*> is represented as an association between the *NCLDescriptor* and *NCLRegion* classes.

Some element attributes may have a value from a specific value set, like the *xmlns* attribute. In those cases, aNa defines the attribute type as an Enumeration with all the possible values for that attribute (see *xmlns* in Figure 1). Sometimes an attribute value can have more than one type. For example, consider the elements presented in Listing 2.

Listing 2: Attribute value examples

```
1 <region id="reg1" top="10" left="10"/>
2 <region id="reg3" top="10.5%" left="10%"/>
```

Notice that the element <*region*> has attributes that may be an integer without a percent sign (%) or a integer or a double with a percent sign. NCL also defines other elements whose attributes can be numbers or strings, as the *max* attribute of a connector condition, where it can be a positive integer or the string "unbounded"; elements whose attributes can be a value (like a string, a number, etc) or another element representing a parameter, as the *delay* attribute of a connector action, where it can be a double or a reference to a connector parameter element.

In aNa, those attributes are defined with type *Object*. So, they can receive any of the possible values the attribute demands. When receiving a new value, the API tests if its type is correct. If not, aNa raises an error indicating the values the attribute may receive. In those cases, if the value received is an string, aNa tries to parse the value to the format used by NCL.

Figure 3 presents a fragment of the API structure, representing relations among elements of the document body. In order to simplify the diagram, the figure presents only class names and associations between them.

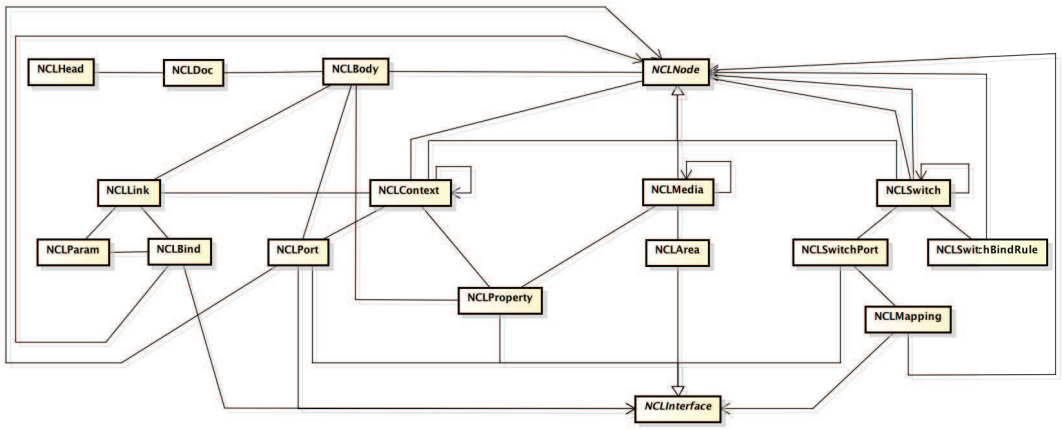


Figura 3: aNa document body structure

4. ANA IMPLEMENTATION

aNa is implemented in Java, providing portability in different platforms. It presents methods to get and to change the value of an element class attribute, to navigate through the NCL document, to create Java objects from an NCL document and to write an NCL document from those objects.

aNa provides facilities to increase reusability in the development of Digital TV applications. NCL itself already allows code reuse, since the same elements can be used in different contexts, without the need to redefine them. Because of the Java implementation, the benefits of object oriented design, such as allowing reuse by class inheritance or object composition [12], help increasing the application development productivity in comparison to direct NCL programming.

The document parsing is done using DOM [4]. aNa walks through the DOM tree gathering information about the NCL elements and creating Java objects that represent them. During that object creation, aNa already creates references between objects. For example, if aNa finds the value "reg1" in the attribute *region* of a <*descriptor*> element, it will search for a <*region*> element in the region base with that *id* in order to create this reference and it will raise an error if no <*region*> with that *id* is found.

The search for a referred element is done only in the attribute scope. That is, if an attribute indicates the *id* of an element inside the same context, aNa will search for that element only inside the context. For example, suppose a <*port*> element that defines *component* and *interface* attributes. aNa will search for an NCL node with the *id* defined in the *component* attribute inside the port parent context. Once that element is found, aNa will search for an interface with the *id* defined in the *interface* attribute inside that node.

During parsing, aNa gathers from the DOM representation of an NCL element only the information that makes sense to it, that is, the attributes and child elements defined in the language specification. Also, when reading an NCL document, DOM already verifies if the XML document is well written, that is, all the XML tags are opened and closed correctly. So, after the document parsing, aNa will have a consistent document representation. If a wrong definition is found in the NCL document, aNa will raise errors. Listing 3 presents an error example. Errors always present the whole path from the root document element to the element where the error occurred. It also shows a message that informs the error found to the author.

Listing 3: Parsing error example

```
1 Error parsing Head > ConnectorBase >
      CausalConnector(onKeySelectionStop) >
      SimpleCondition
2 Could not find a param in connector with name: tecla
```

The opposite way, that is, create an NCL document from the aNa representation is done by getting, from each Java object, its XML representation. The method that implements it returns a String with the XML element representation. It is worth to highlight that the code returned is indented, making the document reading easier.

Once aNa is developed to be used by tools that manipulate NCL code, it is able to notify the tool that uses it about a modification in an element class. This notification can help the tool maintaining a consistent document representation. For example, suppose a tool that is built to graphically show all document regions. Every time a modification occurs in the position of any region, the tool is notified, so it is able to apply the necessary changes in the graphical position of the modified region. The API has a *ModificationNotifier* which may have one or more *ModificationListeners*. The notifier will send notifications when the value of an attribute is set and a child element is added or removed. As this feature may not be necessary for all kinds of tools, the tool is not obliged to implement the *ModificationListener*.

Also, in order to maintain a consistent document representation, once element references are represented as associations, when removing an element from its parent, aNa verifies if such element is used in a reference. If so, aNa indicates to the author the elements that make reference to it and disable its removal until the references are removed.

Another characteristic of aNa is that it is implemented using parameterized classes, which is done using the Generics Java language feature[2]. Using that feature, aNa element classes extensions are simpler, requiring less coding effort.

[2]more information available in http://docs.oracle.com/javase/tutorial/java/generics

5. ANA USE CASES

As mentioned before, aNa was developed to help tools modeling and manipulating NCL code. One use case is a graphical editor that was built for helping users creating NCL connectors [10]. The editor uses aNa to read an NCL document and extract the <*connectorBase*> element. This element represents a base containing all connectors that may be used in <*link*> elements. Once the editor has the object, created by aNa, representing the connector base, it can use aNa methods to get necessary information from the element and to graphically show the connector to the author. The editor also allows the author to perform creations, removals and modifications on connector child elements. All these modifications are performed through aNa methods. Moreover, the editor allows the author to create a new connector base document. The final NCL document code is also created by aNa.

Another tool that uses aNa is called NEXT (NCL Editor supporting XTemplate) [11]. It is a graphical authoring tool developed to facilitate the creation of digital TV applications using the NCL language and supporting the use of composite templates. Its architecture is based on a core that is able to communicate with plugins and is completely independent from them. NEXT plugins are allowed to manipulate the document and, when a plugin makes any modification on it, NEXT notifies other plugins about the modification. In order to obtain knowledge about the change, NEXT uses aNa's feature that notifies about changes occurred in the document. Moreover, NEXT uses aNa objects to create a tree model representing the nesting structure of the NCL document. aNa also enables NEXT to open and to edit any standard NCL document.

aNa is also used as the basis for the development of an NCL document validation tool. That tool, called aNaa - API for NCL Authoring and Analysis, extends aNa by adding methods that allow the validation of the NCL document being authored [5]. It uses aNa for reading an NCL document and gathering information about document elements. After that, aNaa creates different representations of the NCL document, which allow the tool to investigate some document properties and verify its temporal consistency.

6. CONCLUSION

In general, when developing tools to manipulate NCL code, developers have to implement their own metamodel to represent the code to be manipulated. Sometimes no metamodel is created at all. Since those tools need to read an NCL document, it is common to use available XML parsing tools, like DOM or SAX. Although those parsers produce a good result and help the tool developer to manipulate NCL code, their metamodel is generic for any XML document.

This paper presented aNa, an API that provides a metamodel created specifically for representing NCL code and helping the creation of tools that manipulate NCL documents. aNa considers NCL attribute types and verifies if the document follows NCL syntactic and reference rules as defined in the Brazilian standard [1]. Besides API specificities, with a common core that represents an NCL document, aNa makes it possible to exchange object-oriented data among different tools without the need to generate XML code.

Currently, aNa is being used as a basis for the creation of authoring and validation tools. Its code is available for free download[3] and the tool is also open for contributions. We intend to continuously improve aNa, based on feedback from the NCL developers community.

In the current version of the API, error messages are presented in English. A future work is the creation of aNa error messages in different languages. Another future work is to improve the API to receive NCL live editing commands and to produce the necessary modifications in the document.

7. ACKNOWLEDGEMENTS

This work was partially supported by CNPq, FAPERJ and CAPES.

8. REFERÊNCIAS

[1] ABNT. Digital terrestrial television - Data coding and transmission specification for digital broadcasting - Part 2: Ginga-NCL for fixed and mobile receivers - XML application language for application coding, 2011.

[2] R. G. A. Azevedo, M. M. Teixeira, and C. S. S. Neto. NCL Eclipse: Ambiente Integrado para o Desenvolvimento de Aplicações para TV Digital Interativa em Nested Context Language. In *Salão de ferramentas - SBRC*, 2009.

[3] Batuque TV digital. Berimbau iTV Author. http://www.batuque.tv/, 2011.

[4] W. W. W. Consortium. Document Object Model (DOM) Level 2 Core Specification, 2000. W3C Recommendation DOM-Level-2-Core-20001113.

[5] J. A. F. dos Santos. Multimedia and hypermedia document validation and verification using a model-driven approach. Master's thesis, Universidade Federal Fluminense, 2012.

[6] J. A. F. dos Santos and D. C. Muchaluat-Saade. XTemplate 3.0: spatio-temporal semantics and structure reuse for hypermedia compositions. *Multimedia Tools and Applications*, 2011.

[7] G. S. C. Honorato and S. D. J. Barbosa. NCL-Inspector: Towards Improving NCL Code. In *ACM SAC*, pages 1946–1947, 2010.

[8] ITU. Nested Context Language (NCL) and Ginga-NCL for IPTV services. http://www.itu.int/rec/T-REC-H.761-200904-P, 2009. ITU-T Recommendation H.761.

[9] B. Lima, R. Azevedo, M. Moreno, and L. Soares. Composer 3: Ambiente de autoria extensível, adaptável e multiplataforma. In *WebMedia – Workshop de TV Digital Interativa (WTVDI)*, 2010.

[10] J. V. Silva and D. C. Muchaluat-Saade. Editor Gráfico de Conectores Hipermídia para Linguagem NCL 3.0. In *WebMedia*, 2011.

[11] J. V. Silva and D. C. Muchaluat-Saade. NEXT - Editor Gráfico para Autoria de Documentos NCL com Suporte a Templates de Composição. In *WebMedia*, 2012.

[12] S. Srinivasan and J. Vergo. Object Oriented Reuse: Experience in Developing a Framework for Speech Recognition Applications. pages 322–330, 1998.

[3] https://github.com/joeldossantos/aNa

NCLFORMS: Uma API para desenvolvimento de GUIs em aplicações interativas para TV Digital

Renan Ribeiro de Vasconcelos
Universidade Federal do Rio de Janeiro - UFRJ
Centro de Tecnologia, bloco H, sala H-217
Rio de Janeiro, RJ
renanrv@cos.ufrj.br

Cláudia Maria Lima Werner
Universidade Federal do Rio de Janeiro - UFRJ
Centro de Tecnologia, bloco H, sala H-217
Rio de Janeiro, RJ
werner@cos.ufrj.br

Wagner Schau
Universidade Federal do Rio de Janeiro - UFRJ
Centro de Tecnologia, bloco H, sala H-217
Rio de Janeiro, RJ
schau@cos.ufrj.br

Débora Christina Muchaluat Saade
Universidade Federal Fluminense - UFF
Instituto de Computação - IC
Laboratório MídiaCom
Niterói, RJ
debora@midiacom.uff.br

Glauco Fiorott Amorim
Centro Federal Celso Suckow da Fonseca - Cefet/RJ
Coordenação de Telecomunicações/TV Digital
UnED Petrópolis, RJ
glauco.amorim@gmail.com

ABSTRACT

Ginga-NCL is the Brazilian declarative middleware that provides support for the development of interactive TV applications. Ginga-NCL indicates the use of NCL (Nested Context Language), which is a declarative language for building interactive multimedia applications, and Lua, which is a script language that can be used together with NCL. NCL itself does not provide facilities to build forms, as HTML does. However, form components can be implemented in Lua and included in NCL documents. This work proposes an API, named NCLForms, to provide the creation of Lua form components inside NCL programs. Using NCLForms, it is easy to create and manipulate forms inside an NCL program, without the need to program in Lua.

Categories and Subject Descriptors

H.5 [**Information Interfaces and Presentation**]: Design Tools and Techniques; H.5.2 [**User Interfaces**]: Graphical user interfaces (GUI)—*developement, interaction styles*

General Terms

Application

Keywords

Aplicações de TV Digital, Interface Gráfica, TV Digital, Reutilização de Software, Padrões de Projeto

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...$10.00.

1. INTRODUÇÃO

Com a chamada TV Digital interativa, aplicações podem ser executadas diretamente na televisão. Do ponto de vista do desenvolvimento, novos paradigmas devem ser seguidos ao se pensar em uma aplicação para TV, além de aspectos como domínios de aplicações, tipos de usuários, interface de interação, dentre outros.

Uma forma de apoiar o desenvolvimento de aplicações é por meio de bibliotecas de código. Tal alternativa permite colocar em prática técnicas de reúso a fim de acelerar a construção de tais aplicações. Outro caminho muito comum em desenvolvimento de software diz respeito aos *design patterns*, que fornecem orientações na forma de melhores práticas para a implementação de projetos em domínios específicos.

Assim como em qualquer área de software, aplicações interativas de TV Digital também podem contar com bibliotecas e *patterns*, porém ainda necessitam de novas propostas e principalmente da aplicação de tais propostas no cenário brasileiro, pelo fato do middleware brasileiro Ginga [1] utilizar tecnologias inovadoras e ainda não muito comuns no mercado de desenvolvimento de software .

O presente trabalho, ao identificar uma dificuldade em inserir elementos de interface gráfica em aplicações desenvolvidas para o ambiente declarativo Ginga-NCL, apresenta uma biblioteca de código que propõe uma *Application Programming Interface* (API) para desenvolvimento de interfaces de usuário com elementos gráficos em aplicações com componentes de formulários. Tal proposta é colocada sob a ótica da reutilização de software no contexto do ambiente Ginga-NCL, através dos recursos fornecidos pela integração entre as linguagens NCL e Lua. Com isso, o estudo em questão busca colaborar no desenvolvimento de interfaces gráficas em aplicações de TV Digital no cenário brasileiro.

Dessa forma, o texto é organizado em cinco seções, con-

tando com esta introdução. A Seção 2 apresenta uma descrição dos tipos de aplicações interativas para TV Digital, além de destacar as principais características do modelo brasileiro para interatividade. A Seção 3 faz uma revisão sobre APIs em termos de reutilização de software, abordando também características de elementos de interface gráfica, além de ferramentas voltadas para a construção de *Graphical User Interfaces* (GUIs). Na Seção 4, é apresentada a API desenvolvida, chamada de NCLForms, e é proposta a sua aplicação na implementação de soluções em uma linguagem de *patterns* estudada. A Seção 5 conclui o trabalho, apresentando as considerações finais.

2. APLICAÇÕES INTERATIVAS DE TV DIGITAL

A fim de guiar o progresso da pesquisa, alguns pontos devem ser estabelecidos, de forma que o conceito de aplicações interativas para TV Digital não seja interpretado erroneamente. Nesse contexto, algumas características podem ser ressaltadas, como independência de conteúdo (o que diferencia aplicações interativas para TV de TV interativa ou programas de TV interativos) e tipo de aplicação, e estar relacionado à televisão digital. Uma simples definição que abrange esses tópicos é a que aponta aplicações interativas de TV como serviços avançados ou interativos com TV Digital [3].

Um dos equipamentos fundamentais em ambientes de TV Digital é o receptor digital (*set-top box* ou a própria TV). O software incluído no receptor possui alguns componentes, sendo o *middleware* o componente central, funcionando como uma camada de interface entre o hardware e o software básico do dispositivo e as aplicações. O *middleware* determina as linguagens de programação que podem ser utilizadas no desenvolvimento de aplicações interativas.

O modelo brasileiro de TV Digital é baseado no *middleware* Ginga [1]. Tal sistema tem suporte a aplicações declarativas e a aplicações procedurais. O suporte a esses recursos é dividido em dois ambientes principais: um ambiente de apresentação, conhecido como Ginga-NCL, e um ambiente de execução, denominado Ginga-J. No entanto, aplicações híbridas são igualmente possíveis, fazendo uso de recursos dos dois ambientes, além da possibilidade de usar uma linguagem procedural, como Lua, em conjunto com aplicações declarativas na máquina de apresentação Ginga-NCL. O módulo Ginga-NCL faz uso da linguagem declarativa NCL (*Nested Context Language*) e suas características podem ser encontradas em [7] e [8].

Scripts Lua podem ser adotados no subsistema Ginga-NCL a fim de implementar objetos procedurais em documentos NCL. Conforme destaca [7], a fim de se adaptar ao cenário de TV Digital, a linguagem Lua foi estendida com novas funcionalidades, adicionando os seguintes módulos à biblioteca-padrão da linguagem: *event*, *canvas*, *settings* e *persistent*.

3. REUTILIZAÇÃO E DESENVOLVIMENTO DE *GRAPHICAL USER INTERFACES*

A adoção de bibliotecas de código no processo de desenvolvimento de software é uma atividade comum e deve ser cada vez mais incentivada, dados os benefícios da reutilização neste contexto. Grandes sistemas são baseados em coleções reutilizáveis de funcionalidades implementadas na forma de bibliotecas [2].

No entanto, tais funcionalidades necessitam de uma interface que forneça o serviço esperado de tudo o que foi implementado na coleção em questão. A fim de exercer essa função, as APIs (*Application Programming Interfaces*) permitem aos desenvolvedores acessar bibliotecas e aplicar os serviços providos pelas mesmas. Dentre suas principais características, as APIs fornecem abstrações em alto nível, suportam reutilização de código (sua essência), e colaboram de forma a simplificar a fase de programação, uniformizando algumas tarefas [6].

As alternativas visando o reúso de software são aplicáveis a diferentes áreas, desde aplicações tradicionais em ambientes *desktop* até aplicações para TV Digital. No que diz respeito às aplicações interativas de TV, um campo que merece destaque é o de elementos de interface gráfica.

Em um ambiente como o da TV Digital, cuja interação se dá basicamente pelo controle remoto, não tendo à disposição um teclado ou um mouse, o projeto da interface gráfica deve levar diversos aspectos em consideração, a fim de produzir uma aplicação intuitiva. Nesse sentido, o desenvolvimento de interfaces para o usuário deve ter como preocupação quais os dispositivos em que tais interfaces serão visualizadas, os diferentes grupos de usuários, e os variados ambientes de uso [12].

4. NCLFORMS - APRESENTAÇÃO E CONTEXTO DE USO

Utilizando os conceitos de bibliotecas de código e elementos de interface gráfica, juntamente com as características próprias das aplicações interativas para TV Digital, mais propriamente dentro do modelo brasileiro, foi desenvolvida neste trabalho uma API para a construção de *Graphical User Interfaces* (GUIs), cujo nome é NCLForms.

A implementação da API NCLForms tem como objetivo fornecer um conjunto de elementos de interface gráfica, aqui tratados como *widgets*, ao desenvolvedor de aplicações NCL para o *middleware* Ginga, de modo a facilitar a adição de formulários em aplicações interativas para TV Digital.

Apesar de ser possível inserir formulários em aplicações NCL por meio de documentos HTML, não é possível uma customização tão profunda, a ponto de escolher aspectos relacionados à aparência dos elementos e, até mesmo, uma integração mais eficiente com os demais componentes da aplicação.

É importante lembrar que a exibição de formulários e elementos gráficos em HTML/CSS é apenas suportada pelo padrão brasileiro. Ou seja, dependendo da implementação do *middleware*, podem ocorrer incompatibilidades e variações na exibição desses elementos. Como a implementação dos *widgets* na API foi feita usando o padrão brasileiro NCLua, pode-se contar com a padronização da sua exibição e do seu comportamento, assim como é possível prevenir problemas de incompatibilidade entre diferentes versões do *middleware*. Além disso, a criação de documentos NCL que utilizam a API ocorre de maneira bem simplificada, cabendo ao desenvolvedor cuidar apenas da parte de tratamento dos dados relacionados ao formulário, pois a parte de interface já se encontra estruturada pela API.

A Figura 1 exibe um protótipo de aplicação, apresentando um formulário interativo, fazendo uso da API para a cons-

trução de uma GUI.

Figura 1: Exemplo de uma aplicação desenvolvida com a API NCLForms.

Durante o desenvolvimento da API foram utilizadas como ferramentas a máquina virtual Ginga-NCL Virtual Set-top Box v.0.12.3 [5], para emular o *middleware* Ginga-NCL, o software de virtualização VMware Player v.3.1.4 [11], para executar a máquina virtual, e o ambiente de desenvolvimento Eclipse [10], com *plug-in* NCL Eclipse [4] para validação de arquivos NCL e *plug-in* akdebugger [9] para edição de arquivos Lua.

4.1 *Widgets*

Visando o desenvolvimento de um único elemento NCLForms, todos os *widgets* existentes em um formulário foram implementados de modo uniforme e integrado.

De forma a ilustrar a relação entre os elementos propostos, a Figura 2 apresenta um modelo conceitual na forma de um diagrama de classes de todos os *widgets* oferecidos pela API NCL Forms.

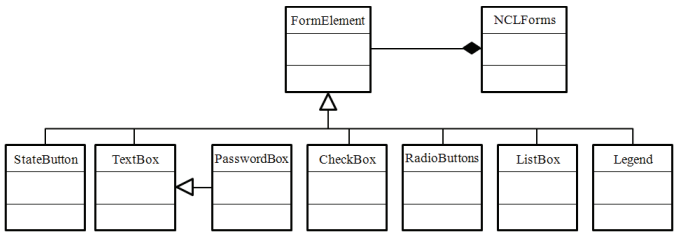


Figura 2: Modelo Conceitual das Widgets.

Cada *widget* conta com um conjunto de parâmetros que permite a sua customização.

Os parâmetros devem ser passados em uma ordem específica no documento NCL para que o *script* Lua possa interpretá-los corretamente. A fim de simplificar a relação dos mesmos, X e Y representam as coordenadas X e Y, respectivamente, do canto superior esquerdo de um *widget* e dX e dY representam o espaçamento nos eixos horizontal e vertical, respectivamente, entre um rótulo e uma caixa para entrada de caracteres em certos elementos. Os *widgets* e seus parâmetros são listados a seguir:

- **statebutton:** ordem de foco, rótulo, X, Y, largura, altura, cor de fundo, tamanho da fonte e cor da fonte;
- **textbox:** ordem de foco, rótulo, X, Y, dX, dY, largura, altura, cor de fundo, cor de borda, tamanho da fonte, cor da fonte e comprimento máximo;
- **passwordbox:** ordem de foco, rótulo, X, Y, dX, dY, largura, altura, cor de fundo, cor de borda, tamanho da fonte, cor da fonte e comprimento máximo;
- **checkbox:** ordem de foco, número de opções, rótulo, X, Y, largura, altura, cor de fundo, tamanho da fonte, cor da fonte e rótulo das opções;
- **radiobuttons:** ordem de foco, número de opções, rótulo, X, Y, largura, altura, cor de fundo, tamanho da fonte, cor da fonte e rótulo das opções;
- **listbox:** ordem de foco, número de opções, rótulo, X, Y, largura, altura, cor de fundo, número de opções visíveis, cor da fonte e rótulo das opções;
- **legend:** ordem de foco, número de opções, X, Y, largura, altura, cor de fundo, tamanho da fonte, cor da fonte e teclas e texto das dicas;

4.2 Utilização da API

Nesta seção, são apresentadas a forma correta de uso dos elementos oferecidos pela API NCLForms e a forma adequada de incorporar uma GUI desenvolvida com a API em um documento NCL.

A Figura 3 exibe um elemento NCL `<media>` fazendo referência ao *script* Lua que implementa a API NCLForms. Esse trecho de código informa à aplicação NCL qual o objeto NCLua deverá ser executado quando for solicitado. Deve-se destacar a importância de se definir um elemento `<property>` denominado *values* para o objeto de mídia. Tal elemento será importante na passagem de parâmetros para a API. Um elemento `<property>` denominado *result* também deve ser incluído no objeto de mídia. Esse elemento poderá armazenar, posteriormente, os resultados das seleções feitas em cada um dos *widgets* do formulário. O *script* NCLForms.lua é o arquivo principal da API, sendo responsável por exibir os elementos do formulário na aplicação.

```
<media id="forms" src="NCLForms.lua" descriptor="dsForms">
      <property name="values"/>
      <property name="result"/>
</media>
```

Figura 3: Adição da função principal da API no documento NCL

Após inserir o elemento `<media>` com a indicação do arquivo Lua a ser incorporado, deve-se informar quais os parâmetros de formulário para a API, a fim de saber quais os *widgets* que serão exibidos, bem como a forma deles. Isso é realizado através do elemento NCL `<link>`, atribuindo os parâmetros à interface de nome *values*. A Figura 4 ilustra esse passo.

```
<link xconnector="onBeginSetN">
   <bind role="onBegin" component="forms"/>
   <bind role="set" component="forms" interface="values">
      <bindParam name="var"
       value="textbox(1,AGENCIA:,420,210,70,0,350,16,white,black,12,black,10),
       textbox(2,CONTA:,420,250,70,0,350,16,white,black,12,black,10),
       passwordbox(3,SENHA:,420,290,70,0,350,16,white,black,12,black,10),
       radiobuttons(4,5,OPCOES:,420,350,220,150,white,12,black,Brasil,Cert.
                    Digital,Def.Visual,Nao Correntista,Exterior),
       radiobuttons(5,4,Titularidade,670,350,220,150,white,12,black,
                    1o. Titular,2o. Titular,3o. Titular,4o. Titular),
       statebutton(6,ENVIAR,600,520,105,35,silver,24,black),
       legend(1,3,1050,610,210,60,white,9,black,  OK,ENTER,Apagar,RED,Próximo,CURSOR_RIGHT),
       legend(2,3,1050,610,210,60,white,9,black,  OK,ENTER,Apagar,RED,Próximo,CURSOR_RIGHT),
       legend(3,3,1050,610,210,60,white,9,black,  OK,ENTER,Apagar,RED,Próximo,CURSOR_RIGHT),
       legend(4,5,1050,610,210,60,white,9,black,  OK,ENTER,Up,CURSOR_UP,Down,CURSOR_DOWN,Next,
              CURSOR_RIGHT,Previous,CURSOR_LEFT),
       legend(5,5,1050,610,210,60,white,9,black,  OK,ENTER,Up,CURSOR_UP,Down,CURSOR_DOWN,Next,
              CURSOR_RIGHT,Previous,CURSOR_LEFT),
       legend(6,2,1050,610,210,60,white,9,black,  OK,ENTER,Anterior,CURSOR_LEFT)"/>
   </bind>
</link>
```

Figura 4: Passagem de parâmetros para o script Lua

Após a passagem de parâmetros no documento NCL, a API irá interpretar esses dados e desenhar na tela os componentes indicados na interface *values*.

A comunicação entre o código NCL e o script Lua é feita através de uma função denominada *nclHandler*, presente no arquivo NCLForms.lua, contido no pacote que compõe a API e que foi registrada para tratar eventos de atribuição. Tal função é responsável por processar eventos voltados para a digitação de texto após a exibição dos componentes na tela, eventos de nome *"text"*, evento de atribuição de valores dos parâmetros e eventos de nome *"values"* reconhecendo quais primitivas gráficas foram informadas. Foi uma decisão de implementação, utilizar um único nó Lua para implementar todos os *widgets* de um mesmo formulário, passando todos os parâmetros necessários através da propriedade *"values"*. Isto facilita a gerência do foco do usuário na interface dentro do próprio código Lua, facilitando a especificação do código NCL pelo autor do documento.

A função *nclHandler* faz chamadas à função *formParser*, que recebe como parâmetros o valor da *string* com todos os parâmetros e uma *string* indicando qual seu *widget* correspondente. A função *formParser* irá criar efetivamente as metatabelas para cada elemento, separando os parâmetros específicos de cada um.

Após serem criadas as metatabelas de cada elemento, os métodos de cada estrutura podem ser executados pelos componentes de controle e desenho da API, a fim de exibir o resultado na tela. Cada elemento tem sua própria estrutura, com métodos próprios que farão uso dos parâmetros informados ainda no documento NCL.

É importante, ainda, destacar a função *redraw* do arquivo NCLForms.lua, responsável por executar o método *redraw* da metatabela de cada elemento. Esse método é utilizado para atualizar a tela com as interações provocadas pelo usuário como: mudança de um elemento para o outro ou escolha de um item do elemento que está em foco.

As soluções desenvolvidas não ficam restritas ao uso apenas como biblioteca de código em aplicações interativas com formulários. Os elementos de interface gráfica que compõem a API NCLForms (i.e., primitivas gráficas) podem ser aplicados na extensão da linguagem de *patterns* proposta por [3]. Pode-se fazer o download da API acessando a página http://www.midiacom.uff.br/nclforms.

5. CONCLUSÃO

A partir do estudo inicial das linguagens NCL e NCLua, a dificuldade de implementar elementos de interface gráfica de modo simplificado inspirou a decisão da proposta de uma API voltada para o desenvolvimento de GUIs.

A API NCLForms, desenvolvida para o ambiente Ginga-NCL, que viabiliza a construção de GUIs para aplicações interativas em TV Digital, representa a principal contribuição do presente trabalho.

Dentro do cenário de aplicações interativas de TV Digital, levando em consideração as orientações de *design patterns* de interação, a API NCLForms é uma proposta para incentivar a reutilização de software no ambiente Ginga-NCL. Em termos de trabalhos futuros, tal proposta é passível de adição de novas funcionalidades, como a implementação de um teclado virtual a ser exibido na tela e o desenvolvimento de novos *widgets*. Do ponto de vista do que já foi implementado, também seria possível pensar em outro formato para os parâmetros, como diminuir o número de parâmetros obrigatórios ou incluir valores *default* para algumas variáveis.

6. REFERÊNCIAS

[1] ABNT. *Data coding and transmission specification for digital broadcasting - Part 2: Ginga-NCL for fixed and mobile receivers - XML application language for application coding*. ABNT NBR 15606-2:2011 standard, 2011.

[2] D. Kawrykow and M. P. Robillard. Improving API Usage through Automatic Detection of Redundant Code. In *Proceedings of the 2009 IEEE/ACM International Conference on Automated Software Engineering*, pages 111–122. ASE, 2009.

[3] T. Kunert. *User-Centered Interaction Design Patterns for Interactive Digital Television Applications*. Springer-Verlag, London, 2009.

[4] NCLEclipse. NCL Eclipse. Disponível em: <http://laws.deinf.ufma.br/ncleclipse>, 2012. Acesso em: 07 jan 2012, 16:00:00.

[5] Portal. Portal do software público brasileiro - Ginga. Disponível em: <http://www.softwarepublico.gov.br>, 2012. Acesso em: 07 jan 2012, 17:00:00.

[6] M. P. Robillard. What Makes APIs Hard to Learn? Answers from Developers. *IEEE Software*, 26(6):27–34, November 2009.

[7] L. F. G. Soares and S. D. J. Barbosa. *Programando em NCL 3.0: Desenvolvimento de Aplicações para o Middleware Ginga, TV Digital e Web*. Elsevier Editora, Rio de Janeiro, 2009.

[8] L. F. G. Soares, R. F. Rodrigues, and M. F. Moreno. Ginga-NCL: The Declarative Environment of the Brazilian Digital TV System. *Journal of the Brazilian Computer Society*, 12(4):37–46, 2007.

[9] SourceForge. Akdebugger. Disponível em: <http://sourceforge.net/projects/akdebugger/>, 2012. Acesso em: 07 jan 2012, 16:00:00.

[10] TheEclipseFoundation. Eclipse. Disponível em: <http://www.eclipse.org/>, 2012. Acesso em: 07 jan 2012, 16:00:00.

[11] VmwarePlayer. Vmware Player. Disponível em <http://www.vmware.com/products/player/>, 2012. Acesso em: 07 jan 2012, 16:00:00.

[12] L. Wang, A. Sajeev, and L. Inchaiwong. A Formal Specification of Interaction Widgets Hierarchy Framework. *ITNG'06*, pages 658–664, 2006.

AVSA: An automatic video segmentation application

Tiago H. Trojahn
Institute of Computer Sciences and Computational Mathematics
Av. Trabalhador São-Carlense 400
São Carlos, São Paulo
ttrojahn@icmc.usp.br

Rudinei Goularte
Institute of Computer Sciences and Computational Mathematics
Av. Trabalhador São-Carlense 400
São Carlos, São Paulo
rudinei@icmc.usp.br

ABSTRACT

Segmentation is an important preprocessing step for a number of current multimedia applications using video, like recommendation, personalization or indexing. Since manual segmentation is prone to interpretation error and are time demanding, researchers concentrate efforts in developing automatic segmentation methods.

Automatic techniques need a number of technical input parameters, requiring specialists to be operated. Moreover, these techniques need the specialist to give a threshold, used to decide when a shot or scene transition occurs. Obtaining an adequate threshold is time consuming and mostly an empirical process. The precision is greatly affected by particularities of the input video, so, an inadequate threshold can lead to over or under-segmentation.

To addresses these problems, this paper presents a friendly user application developed in Java which has two main contributions: perform video segmentation using both an automatic method for calculate the needed threshold and a heuristic to overcome some gradual shot transitions issues. The application, named AVSA, uses the video histogram intersection or histogram absolute differences to perform the segmentation. Furthermore, performance tests are presented in order to testify the precision and the recall of the application when segmenting newscast videos.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Shot Detection

General Terms

Algorithms

Keywords

Shot detection, video segmentation, application

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WEBMEDIA '12 São Paulo, Brasil
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$10.00.

1. INTRODUCTION

In recent years there has been a huge development of video content authoring, due to lower prices for cameras, laptops and other devices which can create videos. This resulted in a large amount of video data, available in a wide range of storages, like video streams sites as YouTube[1] and vimeo[2].

That situation leads to a problem: How, in this vast amount of data, can we find a specific video segment of interest? One approach to answer that question is to segment the video in smaller pieces which can be addressed and indexed through some directives, like the presence of specific actors, the action being recorded and so on. Segmentation is a key step for important domains like multimedia information retrieval, content-based video retrieval, summarization and personalization [1].

A traditional way to segment is using experts to manually segment the video. They watch the video at least one time, possible dozens of times, and manually perform the indexing using the boundaries of the desired video segment. Unfortunately, the manual segmentation is labor intensive, making it a non-pratical solution [12].

To avoid these problems, an automatic video segmentation technique can be used. The literature contains a number of these techniques [2, 4, 8, 10]. In general, these methods focuses in segmenting the video in shots, defined as an "unbroken sequence of frames taken from one camera" [7], or "scenes", which do not have a consensual definition. Among these techniques, the most usual method to perform video segmentation is the frame-to-frame histogram comparison [3]. The histogram comparison stands out for being simple, invariant to camera movement and has low computational cost [11].

Unfortunately, little effort [6, 12] was done to develop tools which can be used by non-specialists users. Most of the implementations is not available for general use, have a confusing interface requiring the user to set dozens of parameters, or even need some adjusts in the source code and a posterior compilation process. Our tool, on the other hand, was developed to avoid these needs and thereby facilitating its use.

One of these input parameters is the threshold. The threshold value is of the utmost importance: an incorrect value can lead the technique to perform poorly, giving many false positives and negatives. Then, to free the user to input the threshold value, can be used a fixed threshold for all possible video inputs. That approach can lead to poor results

[1]http://www.youtube.com/
[2]http://vimeo.com/

because the appropriate threshold varies from domain to domain of the input video. Even videos of the same domain can require a specific threshold to achieve at least acceptable results. An adequate threshold is normally discovered through several manual empirical tests, which demands time and a prior ground truth. Our tool, however, can generate a good approximate threshold through an automatic adaptive threshold method for a given input video.

This work presents the AVSA (stands for *Automatic Video Segmentation Application*), which can be effectively used by users to perform an automatic video segmentation. The program can segment a large number of input video formats in terms of shots, returning to the user a set of possible outputs, like a XML description of the shot boundaries or the frames for each shot.

This paper is organized as follows: the **Section 2** presents the application and his features. The **Section 3** presents some details of the application architecture. Finally, in **Section 4**, are presented our conclusions and some future works.

2. THE APPLICATION FEATURES

The AVSA fundamental feature and main goal is to provide a simple method to segment a video file in shots using an automatic threshold.

To operate, the application requires only a video input file. Most of the video formats and video coding procedures are supported by the application, being limited to formats and CODECS which are installed in the users computer and which are supported by the OpenCV[3] library. Since the expected output may vary from user to user, AVSA offers:

- A XML description of the video input showing the transitions boundaries.
- All frames of the video, grouped in shots, using the lossy JPEG or the lossless PNG formats.
- A video sequence for each shot of the input video.

The AVSA most basic mode of operation consists in 1) select the desired output, 2) select the segmentation procedure (intersection and/or differences), 3) select the input video and 4) click on the "Process" button. To relieve the user of repeating the steps 1 and 2, the program comes with default settings loaded at run-time.

The user can also adjust some parameters of the AVSA segmentation procedure. The options relate to the threshold used in both segmentation methods: it can be used a specific threshold or a technique which automatically calculates the threshold. For default, both intersection and differences threshold method are set to automatic with default boundaries defined. Also, the gradual transitions heuristics is enabled by default. More details of the automatic threshold and the gradual transitions heuristics are presented at **Subsection 3.2.1** and **Subsection 3.2.2**, respectively.

Other feature of AVSA application is the Comparing tool, available at the "Tools" menu. The Comparing is a simple and useful tool for developers allowing to compare a segmentation file against a ground truth. The tool requires a ground truth input file and the file to be compared, the latter being provided by the AVSA itself, as a result of a video segmentation. The results are show in terms of True Positives, False Positives, False Negatives, Precision, Recall and F1-Score [5].

Another available tool at the "Tools" menu are the Manual Creating tool. With that tool, the user can manually insert the boundaries of the shots and export it to a valid XML, which can be used in the Comparing tool as a ground truth, for example. The aim of this tool is to ease the need to manually create a XML or develop a separate application only for that.

Both the AVSA and his tools uses the TRECVID[4] *shotBoundaryReferenceSegmentation* Document Type Definition (DTD). Additionally, the AVSA XML provides more informations like threshold used and heuristics strength value, useful for testing purposes, for example.

3. THE PROPOSED APPROACH

This section presents the AVSA implementation at **Subsection 3.1**. At **Subsection 3.2** is presented the segmentation technique as well as the tests performed in order to measure the tool efficiency.

3.1 Implementation details

The application is divided into six distinct classes which performs different functionalities.

The main class, named *GUI*, was developed using the well-know Java Swing API, being responsible to present the user interface responsible to manage users' interactions. The other classes are instantiated in appropriate moments to perform specific tasks.

The *ConfigsHandler* class is responsible to read and write the configuration files and adjust the graphic user interface according to them. This class is called when an user saves the actual configuration, load the default configuration or even when the application is launched.

The *Segment* class is responsible to, given an input video, creates a shot transitions index of the video. The class was developed with the JavaCV[5] library, which provides the OpenCV functionalities to Java language. After the processing is done, another class, named *OutputHandler*, is called to create and organize the desired output to the user (the XML description, shots frames and/or the shot video sequence).

The *Comparing* class represents the Comparing tool and is responsible to open two XML description files and to compare them: the test file against the ground truth file.

The Manual Creating tool, represented by the class *ManualCreating*, simply receives a set of pairs of integer values and creates a XML description file which can be read by the Comparing tool, for example.

The main processing of the application, the segmentation technique, is described in the **Subsection 3.2**.

3.2 Segmentation technique

The main processing of the application is segmenting videos in shots performed by the *Segment* class.

First, the video is opened and a frame grabber is created. For each frame decoded is calculated it's HSV histogram [6] using 8 bins for H, 4 for S and 4 for V [9]. The resulting histogram is stored and the frame is deallocated. Was used the HSV histogram because it's presents higher precision

[3]http://opencv.willowgarage.com/

[4]http://trecvid.nist.gov/

[5]http://code.google.com/p/javacv/

values at the same recall when comparing with the RGB color space [6] and the 8:4:4 quantization was adopted to follow the basis technique [9]. It's important to mention that the technique itself isn't limited by most video details like resolution, frame rate or video length.

After decoding all frames with corresponding histograms extraction, AVSA performs comparisons with adjacent histograms using histograms intersection and/or histograms absolute differences, generating a histogram differences array. This array is then normalized in order to contain values in the range [0,1], in case of histogram intersection, or [0,2] in case of histogram differences. In the first case, 1.0 means that the two histograms are equal to each other and in the second case 2.0 means the two histograms does not have any similarity.

The shot detection is then performed analyzing the differences values calculated and stored in an array with a desired threshold. If the k-value, a value in the array, is lower than the threshold, in the case of histogram intersection, or higher than the threshold, in case of histogram differences, the k and $k+1$ frames are detected as possessing a transition between then. If consecutive transitions are detected, the algorithm merge then, forming a gradual transition.

This shot segmentation technique is based on a technique know as Backward Shot Coherence (BSC) [9], where the authors use the shot segmentation as initial step towards the scene segmentation. Our implementation, however, presents two major contributions: the automatic threshold calculation, presented in the **Subsection 3.2.1**, and the gradual transitions heuristics, presented in **Subsection 3.2.2**. Furthermore, the segmentation based in the histogram absolute differences were not used in the BSC technique.

3.2.1 The automatic threshold calculation method

The automatic threshold calculation method is defined in **Equation 1**. We defined two values, called lower bound (LB) and upper bound (UP), which are used as references to calculate the average values between then. The V_i represents the i-value of the differences array and K the number of frames which have values between LB and UP.

$$Threshold = \frac{\sum_i^K V_i}{K} \qquad \forall i \mid LB \leq V_i \leq UP \qquad (1)$$

If the K value are 0, i.e. no value was been found between LB and UP, the threshold used is equal to $LB * 1.2$.

The **Equation 1**, in other words, calculates the average values needed to detect gradual transitions which values falls between the LB and UP. The LB value must be small enough to be considered as the minimum similarity between two adjacent frames of the same shot, in case of histogram intersection, or the minimum dissimilarity of two adjacent frames of different shots, in case of histogram absolute differences. Furthermore: if the i-value is lower than LB, the i and $i+1$ frames are automatically considered as being of different shots, in case of histogram intersection, or as being of the same shot, in case of histogram absolute differences.

The UP value, on the other hand, has different meanings to histogram intersection and to histogram absolute differences. In the first case, the value must be high enough to represent all possible transitions, i.e., no shot transition can have value above UP. In the second case, the UP value must be high enough to represent gradual transitions, but can not represent abrupt transitions. That arises from the fact of the abrupt transitions often have overly high values, which in turn would rise the threshold value, ignoring the gradual transitions which values are closer to the LB value.

Adequate values for LB and UP were determined through several performance tests in the newscast domain. The default LB and UP values for the intersection are 0.5 and 0.9 respectively. For the absolute differences, the default values are 0.3 and 0.9, respectively. The obtained precision and recall of the automatic threshold calculation method using these values is presented in **Subsection 3.2.3**.

The LB and UP are the only needed parameters for video segmentation in AVSA. These values can be defined only once and, even so, the application comes with default LB and UP values, making them effectively transparent to the users. The default LB and UP values are adequate to newscast domain: adequate values can change to other domains.

3.2.2 The gradual transitions heuristics

Techniques which compare adjacent frames can detect changes comparing the similarity/dissimilarity against a threshold. If the threshold is exceeded, a transition, classified as an abrupt transition, are flagged. However, modern digital videos often presents gradual transitions: in these transitions, the similarity decrease and the dissimilarity increases over time. These techniques, under such circumstances, can over-segment the video, detecting several abrupt transitions when there is actually a single gradual transition.

One way around this problem is merging consecutive abrupt transitions in a single and larger gradual transition. However, some complex gradual transitions can shows subtle changes spanning over several frames, resulting in a set of gradual and abrupt transitions when there is a single gradual transition. To avoid these outliers, we created a heuristics to detect such gradual transitions.

When a transition is detected, we search the N (N is the size of the search window) previous histogram absolute differences or histogram intersections to see if a shot transition was detected. If there is a shot transition in those frames, all frames from the last transition up to the actual frame are considered as belonging to the same gradual transition. The default number of searched frames, called "strength", is 3. Although a high "strength" value does not influence in the abrupt detection, if a shot length are smaller than the window size, the shot will not be indexed properly.

3.2.3 Evaluation tests

For the evaluation tests, we used a set of videos obtained through direct TV signal capture. The video domain were newscast transmitted at different channels. Some video details, like name, duration, number of frames and number of transitions, are presented in **Table 1**. The video set was digitalized using 720x480 progressive resolution at 30 fps. The commercial breaks were removed.

Table 1: Some details for each video used in the evaluation tests

Video Name	Duration	# frames	# transitions
NewsA	24:09	43470	317
NewsB	23:07	41610	278
NewsC	1:04:51	116730	808

In terms of shot transition type, the video set presents

a larger amount of abrupt transitions when compared with gradual transitions. The NewsC, however, presents about the same rate of abrupt and gradual shot transitions, reflecting a slight lower precision, recall and F1 score than with the NewsA and NewsB videos.

First, we built a ground truth base for the shot transitions. Then the results of the segmentations presented by our application was compared with the ground truth, resulting in the precision, recall and F1 values presented in **Table 2** and **Table 3**. The tests results were obtained through the Comparing tool, using plus and minus 3 frames as frame error tolerance for gradual transitions. In these tests, the gradual heuristics strength was set to 3.

The **Table 2** presents the results of the segmentation using the histograms intersection with the automatic threshold method presented in **Equation 1** with lower and upper bounds values at 0.5 and 0.9, respectively.

Table 2: Precision, recall and F1 score obtained with histogram intersection

	NewsA	NewsB	NewsC
Precision	89.57%	85.76%	71.28%
Recall	92.11%	86.69%	81.43%
F1	90.82%	86.22%	76.02%

The **Table 3** presents the results of the segmentation using the histograms absolute differences with the automatic threshold method presented in **Equation 1** with lower and upper bounds values at 0.3 and 0.9, respectively.

Table 3: Precision, recall and F1 score obtained with histogram absolute differences

	NewsA	NewsB	NewsC
Precision	97.30%	89.25%	84.43%
Recall	68.45%	80.93%	65.09%
F1	80.37%	84.90%	73.51%

The results show a higher precision when using the histogram absolute differences when comparing with the histogram intersection obtained results. The histogram intersection, on the other hand, presented a significantly higher recall value.

4. CONCLUSIONS AND FUTURE WORKS

In this paper we presented the AVSA application, which can be used both by casual users and also by developers or experts to easily segment videos in shots. The features and the architecture of the application was presented. Also, the techniques used by the application was described, presenting also a set of precision and recall tests.

As future works, we intent to add support to other comparison techniques, like Euclidean Distance, in order to measure similarities/dissimilarities. We also plan to apply AVSA to other video domains.

Is also planned add a help assistant in the application. Useful information like the explanation of the lower and upper bounds and the constraints of the application could ease the AVSA usage and improve the results.

Another possible future work is the development of tools which can simplify the burden of specific tasks such as XML editing or even to visualize the results.

5. ACKNOWLEDGMENTS

We would like to thanks to the FAPESP for the financial support of this work.

6. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17:734–749, June 2005.

[2] E. Bruno and D. Pellerin. Video shot detection based on linear prediction of motion. In *Multimedia and Expo, 2002. ICME '02. Proceedings. 2002 IEEE International Conference on*, volume 1, pages 289 – 292 vol.1, 2002.

[3] P. Geetha and V. Narayanan. A Survey of Content-Based Video Retrieval. *Journal of Computer Science*, 4(6):474–486, June 2008.

[4] A. Hampapur, T. Weymouth, and R. Jain. Digital video segmentation. In *Proceedings of the second ACM international conference on Multimedia*, MULTIMEDIA '94, pages 357–364, New York, NY, USA, 1994. ACM.

[5] X.-S. Hua, D. Zhang, M. Li, and H.-J. Zhang. Performance evaluation protocol for video scene detection algorithms. In *Workshop on Multimedia Information Retrieval, in conjunction with 10th ACM Multimedia*, 2002.

[6] S. Jeong. Histogram-based color image retrieval. Technical report psych221/ee362, Stanford university, Mar. 2001.

[7] I. Koprinska and S. Carrato. Temporal video segmentation: A survey. *Signal Processing: Image Communication*, 16(5):477–500, 2001.

[8] S. Manjunath, D. S. Guru, M. G. Suraj, and B. S. Harish. A non parametric shot boundary detection: an eigen gap based approach. In *Proceedings of the Fourth Annual ACM Bangalore Conference*, volume 1 of *COMPUTE '11*, pages 14:1–14:7, New York, NY, USA, 2011. ACM.

[9] Z. Rasheed and M. Shah. Scene detection in hollywood movies and tv shows. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II – 343–8 vol.2, june 2003.

[10] K. Sawai, T. Takahashi, D. Deguchi, I. Ide, and H. Murase. Scene segmentation of wedding party videos by scenario-based matching with example videos. In *Proceedings of the 19th ACM international conference on Multimedia*, volume 1 of *MM '11*, pages 1545–1548, New York, NY, USA, 2011. ACM.

[11] M. Swain. Interactive indexing into image databases. In *In Storage and Retrieval for Image and Video Databases*, pages 95–103, 1993.

[12] H. Zhong, L. Wenyin, and S. Li. Interactive tracker - a semi-automatic video object tracking and segmentation system. In *Multimedia and Expo, 2001. ICME 2001. IEEE International Conference on*, pages 1167 –1170, aug. 2001.

Parte III

IX Workshop de Trabalhos de Iniciação Científica (WTIC)

Prefácio

O IX Workshop de Trabalhos de Iniciação Científica (WTIC) é um dos eventos integrantes do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia) 2012 e constitui um importante fórum, dirigido a estudantes de graduação, dedicado à apresentação e discussão de pesquisas nas áreas de Multimídia, Hipermídia e Web. A meta principal do WTIC é estimular a integração de estudantes de graduação com membros da academia e da indústria, potencializando a formação de futuros pesquisadores.

Neste ano, o WTIC recebeu um total de 20 submissões de artigos técnicos de diferentes instituições, do Brasil e do exterior, contemplando trabalhos de áreas diversas. O evento contou com a ajuda de um Comitê de Programa (CP) de alto nível, sendo cada artigo avaliado por três membros do CP, tendo contado com a colaboração de revisores externos.

Dos 20 trabalhos de iniciação científica, 13 foram selecionados para apresentação e publicação nos anais do evento, resultando numa taxa de aceitação de 65%. Os trabalhos aceitos mostram um bom engajamento destes alunos em atividades de pesquisa.

Agradecemos a todos os autores que submeteram artigos para o Workshop, contribuindo para o seu enriquecimento. Agradecemos também a todos os membros do comitê de programa e aos avaliadores ad hoc por sua dedicação, pelas valiosas revisões dos artigos e pelo comprometimento com os prazos estabelecidos. Agradecimento especial às Professoras Maria da Graça Pimentel (ICMC-USP) e Graça Bressan (POLI-USP), pelo convite para coordenarmos o WTIC e pela cooperação durante todo o processo. E ao Professor André Santanchè (Unicamp) pela colaboração no período de revisão.

Por fim, desejamos sucesso a todos os participantes do WTIC 2012.

São Paulo, Outubro, 2012

Luciana Zaina - UFSCar/Sorocaba
Leônidas de Oliveira Brandão - IME-USP

Organização

Coordenação do WTIC

Luciana Zaina	UFSCar/Sorocaba
Leônidas de Oliveira Brandão	IME-USP

Comitê de Programa do WTIC

Alexandre Alvaro	UFSCar - Sorocaba
Alfredo Goldman	USP
Ana de Melo	USP
Antonio Francisco Prado	UFSCar
Carlos Ferraz	UFPE
Carlos Hitoshi Morimoto	USP
Diana Francisca Adamatti	UFRG
Elisa Tomoe Moriya Schlünzen	UNESP
Fábio dos Santos	UEA
Fernando Trinta	UFC
Gustavo Vieira	UFSCar - Sorocaba
Ig Ibert Bittencourt	UFAL
Jussara Almeida	UFMG
Klaus Schlünzen	UNESP
Kelly Rosa Braghetto	USP
Lucia Filgueiras	USP
Mario Teixeira	UFMA
Sérgio Roberto P. da Silva	UEM

Modelagem de interfaces ricas cientes de contexto na Web 2.0 para plataforma *Android*

Andre L. Bonfatti
Univ. Federal de São Carlos
Rdv João Leme dos Santos,
Km 110 - CEP 18052-780 –
Sorocaba – SP – Brasil
andrebnf@gmail.com

Diego H. Quintale
Univ. Federal de São Carlos
Rdv João Leme dos Santos,
Km 110 - CEP 18052-780 –
Sorocaba – SP – Brasil
diegohquintale@gmail.com

Luciana A. M. Zaina
Univ. Federal de São Carlos
Rdv João Leme dos Santos,
Km 110 - CEP 18052-780 –
Sorocaba – SP – Brasil
lzaina@ufscar.br

Fábio L. Verdi
Univ. Federal de São Carlos
Rdv João Leme dos Santos,
Km 110 - CEP 18052-780 –
Sorocaba – SP – Brasil
verdi@ufscar.br

RESUMO

A Computação Ubíqua tem trazido novos desafios para a Engenharia de Software. Um desses desafios tem sido o desenvolvimento para diversos dispositivos móveis. Dentro desse escopo a plataforma *Android* tem sido alvo de diferentes experiências exploratórias. Esse artigo apresenta uma ferramenta que permite modelar interfaces do usuário para páginas da web 2.0 para plataforma *Android* que sejam adaptáveis de acordo com os dispositivos e as preferências do usuário. Uma aplicação de *upload* e *download* de arquivos foi desenvolvida e testada em diferentes dispositivos por meio de simuladores.

Categorias

D.2.2 [**Design Tools and Techniques**]: user interfaces

Termos Gerais

Design

Palavras-chave

Adaptação de interfaces, *Android*, interface rica, Web 2.0.

1. INTRODUÇÃO

A área da Computação Ubíqua tem motivado a área de Engenharia de Software a produzir recursos que dêem suporte a diversos ramos da produção e manutenção de software. Um dos aspectos críticos é como desenvolver aplicações que se adéquem às mais díspares capacidades dos dispositivos existentes, as denominadas aplicações ubíquas[17].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Workshop de Trabalhos de Iniciação Cientifica- Webmedia 2012, 15 a 18 de Outubro, 2012, São Paulo, São Paulo, Brazil.

Um dos aspectos importantes dentro do desenvolvimento de aplicações ubíquas é a ciência de contexto. Nas aplicações ubíquas um contexto pode representar as restrições de um ambiente, já que muitos dispositivos podem ter poucos recursos disponíveis. Outro fator de contexto está relacionado ao perfil do usuário que acessa a aplicação, sua localização geográfica, a língua mais adequada para ele, entre outros. A ciência de contexto também está relacionada ao desenvolvimento de interface com o usuário. Um dos aspectos críticos é como desenvolver aplicações que sejam adequadas as diferentes capacidades dos dispositivos existentes e que ao mesmo tempo possam atender as preferências do usuário. Essa necessidade de conhecer os elementos que envolvem a interação e adaptar a aplicação segundo esses elementos é denominada de ciência de contexto. De maneira genérica, pode-se definir contexto como um conjunto de condições relevantes e influências que possibilitam a compreensão de uma situação[12, 16].

O objetivo deste artigo é apresentar uma ferramenta denominada *DroidRichWeb*, cujo objetivo é modelar interfaces ricas na Web para a plataforma Android, que sejam adaptáveis de acordo com o contexto da interação do usuário. Para validar a proposta foi desenvolvida uma aplicação de *upload* e *download* de arquivos, onde a interface dessa aplicação foi modelada através do *DroidRichWeb*. Foram utilizados simuladores de diferentes dispositivos para validação da aplicação e da adaptação. O restante do artigo está estruturado da seguinte forma: a seção 2 apresenta sucintamente uma discussão sobre adaptação em dispositivos móveis e um estudo de tecnologias para interface de desenvolvimento Web e de trabalhos relacionados. A ferramenta proposta é apresentada na seção 3. A seção 4 descreve a validação da proposta e a seção 5 realiza as considerações finais.

2. FUNDAMENTOS E TECNOLOGIAS

2.1 Conceitos

Há alguns anos são exploradas metodologias que provêm mecanismos de desenvolvimento de interfaces adaptáveis dependendo do usuário e seu dispositivo. Com o crescimento do uso de dispositivos móveis esta demanda tem sido cada

vez maior. O desenvolvimento de interfaces para dispositivos móveis, no entanto, apresenta muitos desafios [16]: utilização racional do espaço de tela, mecanismos de interação e desenvolvimento de interfaces genéricas. A necessidade de criar diferentes versões de interfaces que sejam aderentes aos diversos tipos de dispositivos móveis, acarreta um tempo de dedicação maior ao desenvolvimento. O mais adequado é que parte da interface seja projetada durante o processo de desenvolvimento e outra parte possa ser construída dinamicamente em tempo de execução da interface[9].

Estudos demonstram que a satisfação do usuário na utilização de interfaces em dispositivos com tela de tamanho pequeno está diretamente relacionada a capacidade de adaptação da interface. A adaptação deve procurar exibir uma interface que seja mais precisa considerando as limitações de tamanho da tela [14].

2.2 Tecnologias

Além do desenvolvimento de aplicações web na plataforma *Android*, outras tecnologias foram estudadas durante este trabalho. O navegador padrão do *Android* 2.2[15] foi testado a fim de indicar quais funcionalidades da Web 2.0 era capaz de executar. Ao final dos testes foi concluído que as *tags* de organização do HTML5[3] (como `header`, `nav` etc) são interpretadas corretamente além da maioria dos eventos e efeitos implementados pela biblioteca *Javascript JQuery*[6]. Com base nesse teste a ferramenta foi implementada dando as opções de inclusão, modificação e exclusão de elementos básicos HTML (imagem, link etc) além de outros recursos como efeitos implementados pelo *JQuery* e organização do HTML5. Também foi estudada a propriedade `pattern` do HTML 5. A sua função é bloquear o botão de submissão de um formulário quando algum campo de texto não está no formato exigido pelo desenvolvedor. Essa propriedade, porém, não é interpretada pelo navegador nativo do *Android* e, portanto, a validação de campos de texto foi feita utilizando a biblioteca *JQuery*. O *JQuery* é usado adicionando-se a biblioteca ao código HTML e fazendo a chamada de suas funções. Como a ferramenta utiliza-se de arquivo XML[2] foi adotada a biblioteca JDOM[4] para facilitar sua manipulação. Outro teste foi realizado e este procurava validar algumas funcionalidades CSS3[1] no navegador padrão do *Android*. Foram testadas as propriedades: `face-font`, `nth-child`, `rgba`, `text-shadow`, `border-radius`, `border-image`, `stroke`, `gradient`, `column`, `transformation`; bem como as funcionalidades de animação e caixas (`box`) flexíveis. Todos os casos de teste foram satisfatórios. Quanto à adaptabilidade, algumas técnicas de modelagem para formulários e campos de texto[20] foram estudadas a fim de que estes objetos fossem corretamente estruturados e, após sua adaptação, o resultado fosse satisfatório.

2.3 Trabalhos Relacionados

Alguns trabalhos já desenvolvidos dentro do escopo de desenvolvimento de interfaces adaptáveis foram estudados. O ambiente *XMobile* permite a geração de aplicações adaptativas baseadas em formulários para dispositivos móveis[19]. Outra ferramenta, a *Semantic Transformer*, permite realizar a transformação automática de páginas *Web* desenvolvidas para a plataforma *desktop* em páginas Web adequadas para dispositivos móveis[18]. Algumas outras ferramentas foram consultadas, como a *qooxdoo*[7], *Jo HTML5 Mobile App Framework*[5], entre outras; porém a maioria delas têm foco no desenvolvimento de interfaces que se adaptam tanto para ambientes *desktop* quanto para *mobile* e isso faz com que a adaptabilidade em função do perfil do usuário e/ou do dispositivo não seja tão eficaz. Uma ferramenta explorada foi a *DroidDraw*[13], que permite a modelagem de interfaces para aplicações para plataforma *Android*.

Billsus et al [10] reporta um trabalho que realiza a apresentação de interfaces adaptáveis para acesso ubíquo por meio da web. A adaptação da interface é baseada na aprendizagem das escolhas do usuário. Cada decisão que o usuário toma é essencial para a exibição das informações. São realizados testes com exibição de menus o conteúdo é adaptado, mantendo as informações das quais o usuário demonstra mais interesse acima das demais. Os exemplos realizados foram específicos para alguns domínios de aplicação. Além disso, a ferramenta não utiliza informações do dispositivo como base para as adaptações.

3. FERRAMENTA DROIDRICHWEB

O desenvolvimento do *DroidRichWeb* foi baseado na ferramenta *DroidDraw*[13], tanto na interface gráfica quanto na arquitetura usada. A *DroidRichWeb* permite que o desenvolvedor crie modelos de interfaces ricas, gerando modelos genéricos de interfaces. A partir da definição dos modelos genéricos o desenvolvedor irá especificar quais pontos da interface possuem variabilidades definindo quais critérios serão considerados para a adaptação daquele ponto. A ferramenta gera a estrutura da interface do usuário no formato XML (*Extensible Markup Language*)[2]. Também cria um arquivo XML com regras de adaptação das interfaces ricas que será utilizado para a adaptação da interface em tempo de execução da aplicação. São considerados dois aspectos contextuais: o perfil do dispositivo e o perfil do usuário. Durante a modelagem é possível selecionar que características relativas ao perfil dispositivo e ao perfil do usuário possuem impacto na adaptação da interface.

3.1 Arquitetura e Funcionalidades

Na Figura 1 são apresentados dois diagramas de classes, contendo classes centrais da ferramenta. A classe `Component` é a especialização da classe `Element` do JDOM[4] (representando cada componente HTML), já a classe `PropertyCapability` estende a classe `Attribute` do JDOM (representando os atributos de uma tag).

Na Figura 1 (a) podem ser observadas as classes que representam os componentes da ferramenta: a classe `Component` é a superclasse de todos os componentes. A classe `Generic` representa os componentes genéricos, que são especificados na inicialização do objeto. Já as classes `Html`, `Img`, `Link`, `Text`, `Form` e `Input` representam os componentes mais elaborados que precisam de métodos especiais para sua manipulação. Já na Figura 1 (b) é apresentada a manipulação do componente `Adaptation` pela ferramenta. As classes `Profile` e `Match` fazem parte da classe `Adaptation` e implementam funções que facilitam a montagem desse elemento. Por fim a classe `PropertyCapability` estende a classe `Property` com algumas funções que automatizam a leitura e alteração dos atributos das tags do componente `Adaptation`.

A classe `Adaptation` define um ponto de adaptação da interface. A classe `Profile` define o perfil da adaptação, cada perfil tem vários `Matchs` que armazena o valor do elemento que satisfaz o perfil; a próxima subseção explica em detalhes como a adaptação é feita.

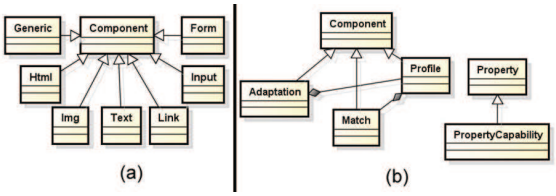


Figure 1: Principais classes da ferramenta.

Existem outras classes, que implementam a interface gráfica e realizam os detalhes das ligações entre as classes. A estrutura da ferramenta apresentada aqui é baseada na arquitetura utilizada pelo *DroidDraw*[13].

A ferramenta exibe de forma simplificada a estrutura da página por meio de modelos, ou seja, não é exibido exatamente como será a página e sim onde ficam os seus componentes. Na Figura 2 é ilustrada essa situação: o componente *body* está selecionado (2) (portanto a visão exibida é desse elemento) e possui os componentes *header*, *nav*, *section* e *footer* (1).A inserção de componentes é feita com o clique sobre o elemento no menu *Components* (4) da ferramenta. Ao clicar-se em algum elemento (por exemplo, *header*) a visão passa a ser desse elemento e é exibido um menu com suas propriedades (5), permitindo sua alteração. Também é permitida a exclusão desse elemento. Através do componente *link* (7) a ferramenta permite que sejam usados elementos pertencentes à Web 2.0[11]. A ferramenta também facilita a geração do código CSS para estilo da página, essa funcionalidade é acessada por meio da aba CSS (6) no painel superior direito.

É possível também inserir na interface formulários e campos de texto (disponíveis no menu *Components*) com a formatação definida pelo desenvolvedor. A ferramenta tem três formatações pré-definidas e são elas: alfanumérico, somente números e data. Essas regras são adicionadas a um arquivo XML e para adicionar uma validação basta editar o mesmo.

A geração de código é feita por meio do botão *Generate* (3), onde o código é gerado no painel inferior: código HTML (9) e código CSS (10). O código HTML produz um código XML em arquivo que representa o modelo da página. Esse modelo é interpretado por adaptador, cujo protótipo foi elaborado nessa pesquisa com a finalidade de validar a adaptação.

Para acessar a funcionalidade de adaptação é necessário inserir o componente *Adaptation* (8). Por meio de sua inserção é determinado que naquele ponto haverá uma adaptação. Ao criar esse componente é necessário informar quais serão os critérios de adaptação de acordo com o dispositivo e o usuário. O componente *Adaptation* permite escolher o tipo de perfil no qual se deseja incluir um elemento de adaptação. Para cada Adaptation inserido no modelo é possível adicionar vários critérios de adaptação. Cada um destes critérios será traduzido para uma regra que será executada durante o processo de adaptação.

A execução da adaptação é apresentada na Figura 3. A partir da chamada efetuada em um dispositivo com *Android*, o servidor Web efetua a requisição para respectiva página (1), esta gerada pelo *DroidRichWeb*. Quando um ponto de adaptação é encontrado na página é acionado o adaptador (2), este referencia as regras de adaptação da página que estão associadas ao ponto de variabilidade em questão (3),

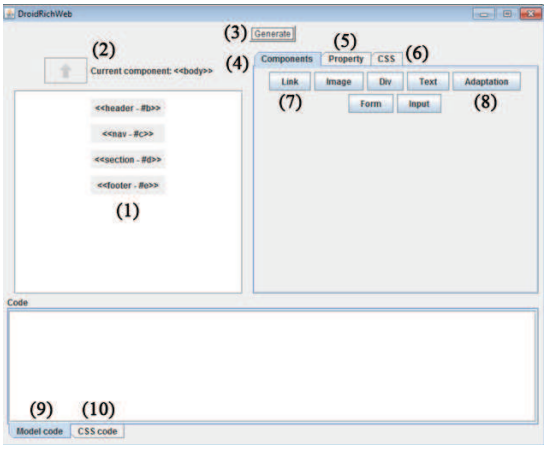


Figure 2: Visão Geral da Ferramenta.

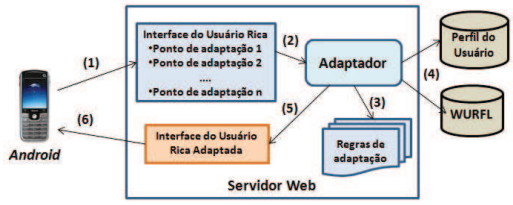


Figure 3: Execução da adaptação.

consultando os dados necessários dos perfis do usuário e do dispositivo (4). O adaptador então realiza as alterações na página (5). Este procedimento é realizado por toda a página requisitada. Ao final das adaptações a nova versão é enviada para o dispositivo (6). Como a ferramenta gera um código XML, este precisa ser transformado em um HTML válido e essa conversão é feita pelo adaptador. O adaptador é um *servlet* que recebe o arquivo XML como entrada e inicia sua interpretação. O adaptador utiliza o *WURFL* (*Wireless Universal Resource File*)[8] , que se trata de um arquivo XML que contém informações de milhares de dispositivos móveis. Assim o adaptador apenas copia as tags HTML ordinárias, e os pontos adaptáveis são analisados. Ao final um código HTML válido e adaptado é gerado.

4. VALIDAÇÃO DA FERRAMENTA

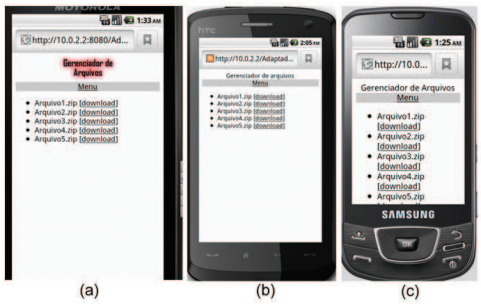


Figure 4: Interfaces adaptadas de acordo com a largura do display.

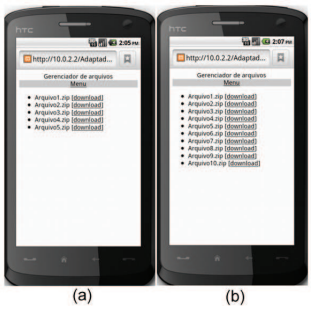


Figure 5: Interfaces adaptadas de acordo com a largura de banda.

Para validação da ferramenta foi desenvolvida uma aplicação de *download* e *upload* de arquivos, cuja interface adaptável foi modelada através do *DroidRichWeb*. A aplicação tem a finalidade de testar o modelo adaptável da interface de acordo com dois critérios: largura da tela do dispositivo (`max_image_width`) e largura de banda do usuário (`bandwidth`). A aplicação conta com um menu que é implementado usando os efeitos do componente *link* (efeitos da Web 2.0). Existem dois modelos de cabeçalho, que é adaptado de acordo com o dispositivo: um com imagem e um em texto puro. Caso a largura da tela do dispositivo seja menor que `350px` o texto puro é exibido, caso a largura seja maior ou igual, a imagem é mostrada. A exibição dos arquivos no servidor é adaptável de acordo com o usuário: se a largura de banda for maior que `127 kbps` serão mostrados os últimos 20 arquivos enviados, se for menor serão exibidos os últimos cinco.

No teste utilizamos três dispositivos: Motorola Droid (Milestone) de display `854 x 480`; Samsung i7500 de display `480 x 320` e HTC Touch de display `320 x 240`. Os dois primeiros foram testados em uma conexão de `64 kbps`, o último foi testado em uma conexão de `64 kbps` e em uma de `1 Mbps`.

Na Figura 4 podem ser observados três dispositivos conectados a uma rede de `64 kbps`. Na Figura 4(a) é apresentado o resultado no dispositivo Motorola Droid; na Figura 4(b), no HTC Touch e na Figura 4(c), no Samsung i7500. Já na Figura 5(a), temos o HTC Touch com conexão de `64 kbps` e na Figura 5(b), o HTC Touch com conexão de `1 Mbps`.

5. CONCLUSÃO

Este artigo apresentou a ferramenta *DroidRichWeb* que automatiza a criação interfaces ricas para *Android*, implementando componentes HTML 5 e de Web 2.0. A ferramenta permite além da modelagem da página web a especificação de pontos variáveis que deverão ser adaptados durante a execução da página de acordo com o perfil do dispositivo e do usuário.

O desenvolvimento da ferramenta já foi finalizado e nesta última versão, há funcionalidades de adaptação para os componentes *link*, imagens, *divs*, textos, formulários e campos de texto. Também foi desenvolvido um adaptador para o processamento da interface adaptável de acordo com os perfis de usuário e dispositivo.

6. AGRADECIMENTOS

Ao CNPq pelo apoio financeiro.

7. BIBLIOGRAFIA

[1] *Cascading Style Sheets 3 (CSS3)*. 2011. Disponível em: http://www.w3schools.com/css3/. Acesso em 14/10/2011.

[2] *Extensible Markup Language (XML)*. 2011. Disponível em: http://www.w3.org/XML/. Acesso em 05/05/2011.

[3] *HyperText Markup Language 5 (HTML5)*. 2011. http://www.w3.org/TR/2011/WD-html5-20110525/. Acesso em: 31/03/2011.

[4] *JDOM*. 2011. Disponível em: http://www.jdom.org/. Acesso em 05/05/2011.

[5] *Jo HTML5 Mobile App Framework*. 2011. Disponível em: http://www.joapp.com/. Acesso em 06/05/2011.

[6] *JQuery*. 2011. Disponível em: http://jquery.com. Acesso em: 31/03/2011.

[7] *Qooxdoo*. 2011. Disponível em: http://www.qooxdoo.org/. Acesso em 06/05/2011.

[8] Wurfl. 2011. Disponível em: http://wurfl.sourceforge.net/. Acesso em: 02/04/2011.

[9] S. Ceri, F. Daniel, M. Matera, and F. M. Facca. Model-driven development of context-aware web applications. *ACM Transactions on Internet Technology*, 7(1), 2007. Article 2.

[10] C. E. B. G. Daniel Billsus, Clifford A. Brunk and M. Pazzani. Adaptive interfaces for ubiquitous web access. pages 34, 38, 2002.

[11] P. J. Deitel and H. M. Deitel. *AJAX, Rich Internet Applications, and Web Development for Programmer*. Prentice Hall, first edition, 2008.

[12] A. K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.

[13] DroidDraw. *Projeto DroidDraw*. 2011. Disponível em: http://www.droiddraw.org//. Acessado em: 05/05/2011.

[14] L. Findlater and J. McGrenere. Impact of screen size on performance, awareness, and user satisfaction with adaptive graphical user interfaces. pages 1257, 1256, 2008.

[15] Google. *Plataforma Android*. 2011. Disponível em: http://code.google.com/android/. Acessado em: 15/04/2011.

[16] E. G. Nilsson. Design patterns for user interface for mobile applications. *Advances in Engineering Software*, 40(12):1318, 1328, 2009.

[17] O. Pastor and F. Valverde. *Facing the Technological Challenges of Web 2.0: A RIA Model-Driven Engineering Approach*, volume 5802 of *Lecture Notes in Computer Science*, pages 131, 144. 2009.

[18] F. Paternò, C. Santoro, and A. Scorcia. Automatically adapting web sites for mobile access through logical descriptions and dynamic analysis of interaction resources. *ACM Proceedings of the Working Conference on Advanced Visual Interfaces*, pages 260–267, 2008.

[19] W. Viana and R. M. C. Andrade. Xmobile: a mb-uid environment for semi-automatic generation of adaptive applications for mobile devices. *Journal of Systems and Software*, 81(3):382–394, 2008.

[20] L. Wroblewski. *Web Form Design: Filling in the Blanks*. Rosenfled Media, 2008.

Validação de Consistência em Aplicações Replicadas

Vinícius Lopes da Silva
DComp — CCTS — UFSCar
Sorocaba, São Paulo
vinicius.lopes.silva89@gmail.com

Gustavo Maciel Dias Vieira
DComp — CCTS — UFSCar
Sorocaba, São Paulo
gdvieira@ufscar.br

RESUMO

Esta pesquisa de iniciação científica visou melhorar a confiabilidade da biblioteca de replicação Treplica, no contexto do desenvolvimento de aplicações web. Com esse objetivo foi criada uma ferramenta de validação de consistência de dados e a mesma foi usada para fazer uma avaliação comparativa da confiabilidade de Treplica. A base desta comparação foi a substituição do algoritmo de consenso Paxos usado por Treplica pelo mecanismo de ordenação baseado em sincronia virtual JGroups. Como resultado foi possível averiguar que a confiabilidade da implementação atual de Treplica/Paxos é superior ao JGroups.

Palavras-chave

Middleware, Replicação, Paxos, Tolerância a falhas, Ordenação Total, Sistemas distribuídos

1. INTRODUÇÃO

A web tornou-se ferramenta essencial para as empresas que desejam manter-se no mercado. Muitas delas disponibilizam serviços que servem para entreter e facilitar o dia a dia dos usuários, como por exemplo transferências de dinheiro entre contas bancárias, Massively Multiplayer Online Games (MMOG), armazenamento de e-mail, entre outros. Por serem aplicações interativas, esses sistemas exigem tempo de resposta reduzido, servem um grande número de clientes e o seu mal funcionamento pode parar completamente processos de negócios vitais. Mesmo assim, falhas são frequentes. Por várias razões, sistemas web são implementados como sistemas distribuídos. Umas das razões é dividir a carga gerada por um parque muito grande de usuários entre servidores especializados em atividades distintas. Daí vem a conhecida arquitetura em três camadas de aplicações web, com um servidor HTTP frontal que repassa requisições a um servidor de aplicações que faz uso de um sistema gerenciador de banco de dados. Desta forma, no núcleo destas aplicações reside um componente de banco de dados, responsável por armazenar o seu estado. Normalmente, esse componente é implementado por um sistema gerenciador de banco de dados (SGBD) centralizado, tornando a armazenagem persistente de dados um gargalo de desempenho e um ponto único de falhas. A replicação de dados pode ser usada para resolver os problemas causados por esta centralização, mas esta é uma solução raramente usada, exceto em aplicações muito específicas. Por trás da baixa adoção da replicação reside o fato de que é usualmente muito difícil replicar dados de forma consistente e ao mesmo tempo preservar o desempenho de um sistema centralizado [1].

As mesmas mudanças que nos tornaram tão dependentes de aplicações web também trouxeram novas oportunidades a serem exploradas. *Hardware* de prateleira começou a ser organizado em poderosos aglomerados de estações de trabalho [2]. Esses aglomerados são construídos com partes facilmente obtidas e quando software livre, revelam-se um ambiente de custo atraente para a execução de aplicações paralelas e replicadas de alto desempenho. O desafio que se apresenta agora é combinar aglomerados computacionais e replicação de dados para aumentar a disponibilidade de aplicações web, igualando e melhorando o desempenho de sistemas centralizados e não confiáveis.

A biblioteca de replicação Treplica [3, 4] é uma abordagem para resolver esse problema. Treplica simplifica o desenvolvimento de aplicações altamente disponíveis ao tornar transparente a complexidade de se lidar com replicação e persistência de dados. Esta complexidade não é desprezível [5], e Treplica define uma forma atraente de fatorar esses requisitos em uma interface de programação simples de entender. Treplica se situa a meio caminho entre a flexibilidade de baixo nível de um sistema de comunicação em grupo [6] baseado em sincronia virtual [7, 8] e os vastos recursos de processamento de dados de um SGBD. A principal característica de Treplica é a ideia de apresentar ao programador web uma abstração de programação unificada para replicação e persistência, propondo o uso de consenso [9] como a fundação para a construção desta ferramenta unificada.

Esta pesquisa de iniciação científica visou melhorar a confiabilidade da biblioteca de replicação Treplica, no contexto do desenvolvimento de aplicações web. Com esse objetivo foi criada uma ferramenta de validação de consistência de dados e a mesma foi usada para fazer uma avaliação comparativa da confiabilidade de Treplica. A base desta comparação foi a substituição do algoritmo de consenso Paxos [11, 12] usado por Treplica pelo mecanismo de ordenação

baseado em sincronia virtual JGroups [14]. Como resultado foi possível averiguar que a confiabilidade da implementação atual de Treplica/Paxos é superior ao JGroups. Mesmo apresentando confiabilidade superior, a estratégia de avaliação adotada permitiu identificar e corrigir erros latentes na implementação atual de Treplica/Paxos.

2. TREPLICA

Treplica foi projetada para prover uma forma simples e orientada a objetos de se construir aplicações altamente confiáveis. Estas aplicações podem se estender ao sistema inteiro ou se restringir à subsistemas onde o desempenho, consistência e confiabilidade são cruciais. Para alcançar esse objetivo, Treplica decompõe o problema de se implementar replicação em componentes com interfaces simples e bem definidas. Desta forma, um desenvolvedor que deseja implementar uma aplicação distribuída não precisa pensar em termos de processos, mensagens e falhas. Ao invés, ele pensa sobre a execução das operações da aplicação, transições de uma máquina de estados replicada, que são disparadas por eventos disponibilizados através de uma fila persistente assíncrona [4].

A principal decisão de projeto por trás do Treplica é que o programador pode considerar a sua aplicação como não tendo estado, ficando a durabilidade da mesma garantida pela biblioteca. Esta decisão é suportada pela observação que os mesmos requisitos de replicação ativa podem ser usados para prover um mecanismo simples e poderoso de persistência. A replicação ativa exige que a aplicação execute ações que modificam o seu estado de forma determinista. Estas ações são então propagadas, na mesma ordem, para todas as réplicas de um serviço que as reexecutam localmente. Dentro desta organização, as ações não são apenas enviadas para as outras réplicas, mas arquivadas em memória estável [10]. Desta forma, é possível recuperar de uma falha reexecutando o histórico de ações. O determinismo garante que após cada recuperação a aplicação reiniciará com o mesmo estado que possuía antes da falha.

Para suportar a replicação ativa, Treplica se concentra em algoritmos de ordenação total baseado em consenso [9] para o modelo de falhas falha-e-recuperação. O algoritmo Paxos [11] e suas variantes [12] são especialmente aptos para esse uso, pois foram criados especificamente para suportar replicação ativa. No entanto, qualquer mecanismo de troca de mensagens totalmente ordenadas pode ser usado como base para a construção de suas abstrações básicas.

3. JGROUPS

A difusão totalmente ordenada usada por Treplica é comumente associada a ferramentas de comunicação em grupos. A ferramenta Isis [6] introduziu muitas ideias que influenciaram outras ferramentas, incluindo a noção de sincronia virtual [7,8]. Implementações mais recentes destas ideias podem ser encontrados em Ensemble [13], JGroups [14], Spread [15] e Appia [16]. Neste projeto, a ferramenta escolhida para ser integrada ao Treplica e substituir o algoritmo Paxos foi o JGroups. Apesar de não ter sido a única possibilidade de ferramenta para substituir o Paxos, o JGroups foi escolhido devido a sua adoção em grandes projetos como o JBoss e também por apresentar uma extensa documentação, além de um grupo de discussões ativo.

O JGroups é um *middleware* que garante a comunicação em grupo de maneira confiável [14]. Os membros (nós) são processos que fazem parte de um grupo (*cluster*). O JGroups utiliza um paradigma de comunicação orientado a troca de mensagens, permitindo que um membro possa enviar mensagens para todos os outros do mesmo grupo ou enviar mensagens diretamente para um único membro. O JGroups monitora todos os membros que participam de um grupo, notifica todos os membros do grupo quando um novo membro entra ou sai do grupo e também quando um membro recebe uma nova mensagem. Para que uma aplicação possa enviar e receber mensagens usando o JGroups, ela precisa criar um *channel* (cujo conceito é similar ao de um socket). Esse *channel* será conectado a um grupo e possui um endereço único.

4. VALIDAÇÃO DE CONSISTÊNCIA

A validação formal da implementação de um *middleware* como o Treplica é uma tarefa difícil e que não possui ferramentas e procedimentos gerais definidos. Por isso, na iniciação científica foi adotada uma forma de validação mais simples, que consiste em criar um teste de longa duração que avalie a consistência de uma aplicação construída usando o Treplica.

Uma aplicação distribuída que use replicação é dita consistente se, após um certo tempo de execução, todas as réplicas do serviço possuem o mesmo estado. A consistência desta aplicação é resultado direto da correção do sistema de replicação subjacente. Desta forma, a criação de uma ferramenta de testes de longa duração envolve o desenvolvimento de uma aplicação onde é fácil determinar uma violação de consistência e a criação de um ambiente de execução que seja capaz de executar esta aplicação de forma controlada. A ideia adotada desde o início do projeto era a de criar uma ferramenta autônoma e simples de usar, que possa ser empregada sempre que Treplica for adaptada a um novo ambiente ou modificações sejam realizadas em sua implementação.

5. RESULTADOS

Diversos testes usando a ferramenta desenvolvida foram realizados para o JGroups e o Treplica. Nesta seção serão apresentadas a ferramenta de testes e os experimentos que foram relevantes para se concluir algo a respeito do funcionamento de ambos os sistemas através da aplicação criada. Na nomenclatura de comunicação em grupo, os membros de um grupo podem ser chamados de nós, entretanto, nesta seção eles serão chamados de réplicas, já que sua função é justamente a de manter o estado replicado.

5.1 Implementação da Ferramenta de Testes

A ferramenta desenvolvida consiste em trocas de mensagens em um ambiente de comunicação em grupo, com um limite fixo N na quantidade de membros que podem fazer parte do grupo. Portanto, o grupo não pode ter um número de membros maior do que N, entretanto, é permitido que haja um número de membros inferior a N, desde que esse número não seja inferior a $N/2 + 1$, caso seja, a aplicação interrompe o envio de mensagens e fica aguardando um número de membros que seja maior ou igual a $N/2 + 1$ para que volte a trocar mensagens entre as réplicas do grupo. É importante que a aplicação não continue executando caso haja

menos de N/2 + 1 nós. Caso essa regra seja violada, é possível que haja inconsistência sobre o estado mantido entre os diferentes nós e não haveria como contornar tal situação de maneira elegante. Garantindo que sempre haja mais da metade dos nós em execução, é possível reverter situações onde uma minoria deles armazenam um estado diferente da maioria.

As mensagens enviadas para os membros do grupo são geradas aleatoriamente e tem um tamanho fixo de três caracteres, é importante ressaltar que esse tamanho foi definido arbitrariamente e não implica em perda de desempenho ou comprometimento da consistência do estado, podendo ser alterado conforme as necessidades do desenvolvedor. As réplicas mantém um estado que contém o histórico de todas as mensagens que foram recebidas, as mensagens enviadas por uma réplica também são enviadas para a réplica que enviou a mensagem em questão, como pode ser observado na Figura 1.

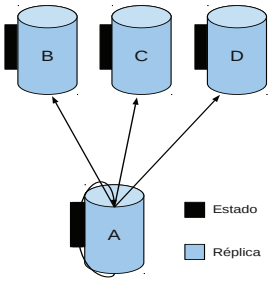


Figura 1: Esquema representando um ambiente de comunicação em grupo onde a réplica A envia mensagem para todas as outras réplicas do grupo e todas as réplicas possuem o mesmo estado

Devido ao fato desse histórico de mensagens crescer muito rapidamente, foi utilizada uma abordagem para reduzir a quantidade de memória usada pela aplicação. Para conseguir isso, foi utilizado o CRC-32 que gera um hash do histórico de mensagens com um tamanho fixo de 32 bits. Desta forma, sempre que uma mensagem é recebida, é calculado um novo hash para o histórico de mensagens recebidos, se o hash de todas as réplicas for igual, então é dito que a aplicação está consistente. Sempre que a aplicação é executada, deve-se informar a frequência com que as mensagens serão enviadas e o tempo em segundos pelo qual a aplicação executará. Ao término da execução, as réplicas informam o hash final e o número de operações que foram realizadas. O tempo de duração de um experimento pode ser grande, portanto, deseja-se saber se a consistência do estado foi comprometida antes de chegar ao fim do experimento. Por isso, durante a sua execução, as réplicas exibem o seu hash a cada 100 mensagens recebidas, permitindo saber se a consistência foi violada ou não. Caso seja necessário, é possível abortar o experimento e fazer as correções necessárias.

5.2 Validação de JGroups

Foram feitas três réplicas para a execução dos experimentos com o JGroups, sendo duas delas em uma máquina A e uma na máquina B. As três réplicas executariam por 18000s (5h), sendo que todas as réplicas possuíam um mesmo vazão de 700 msg/s. Nesse experimento o JGroups durou aproximadamente 2h, sem a introdução de falhas. Após esse tempo começaram a surgir mensagens de erro informando que não havia como retransmitir as mensagens para os outros nós devido a ausência de memória. Isso mostra que haviam limitações de *hardware*, portanto, o vazão foi reduzido e as réplicas passaram a ter um vazão de 300 msg/s. Após 1h foi verificado que as réplicas continuavam consistentes, então começou-se a introduzir falhas no sistema. Especificamente, matava-se uma das réplicas em execução de forma a manter o grupo em execução. Após um certo tempo do grupo em execução com um membro a menos, inseria-se esse novo membro no grupo. Esse então obtinha o estado atual do grupo e executava normalmente com os outros membros, mantendo assim a consistência do estado em todas as réplicas.

Entretanto, quando foi introduzida a falha em que o cabo de rede é retirado da máquina física a consistência foi violada. A réplica sai do grupo ao qual pertencia e cria um novo grupo onde participarão apenas as réplicas que estão na mesma máquina física, caso o número de membros desse novo grupo seja maior ou igual a N/2 + 1. Esse grupo então vai executar como se nada de errado tivesse ocorrido, ou seja, eles executarão como se estivessem em rede ainda. Teoricamente, quando o cabo de rede fosse colocado novamente na máquina física, as réplicas deveriam voltar a formar o mesmo grupo de maneira a tornar o estado consistente, mas isso não ocorreu. Foram pesquisadas informações no grupo de discussões do JGroups, mas a resposta obtida do autor do programa foi a de que deve ser um bug do próprio JGroups. A pilha de protocolos, montada em XML e que possui uma camada que garante a ordenação total e outra que garante a sincronia virtual, não havia sido testada apropriadamente e correspondia a uma configuração não suportada. Ou seja, para replicação usando ordenação total de mensagens o JGroups não oferece a confiabilidade mínima necessária.

5.3 Validação de Treplica/Paxos

O Treplica utiliza um paradigma de programação orientado a estados, diferente do que ocorre no JGroups, o que pode causar um pouco de confusão para desenvolvedores que não estão acostumados com esse paradigma. Os experimentos desenvolvidos para o Treplica possuíam o mesmo objetivo daqueles que foram feitos para o JGroups. O ambiente de testes foi o mesmo, mas diferente dos experimentos usando o JGroups onde o vazão era de mais de 100 msg/s, nos experimentos feitos usando o Treplica, o vazão foi reduzido para 10 msg/s nas três réplicas. Isto ocorre pois, usando Paxos, o Treplica não chega a enviar mais de 100 mensagens por segundo, independente do valor de frequência que for informado.

Os experimentos executaram normalmente, mantendo a consistência do estado entre todas as réplicas por duas horas sem a introdução de falhas e também ao introduzir a falha de matar uma das réplicas e colocá-la novamente no grupo. Ao introduzir a falha de retirar o cabo de rede, novamente foi observado um problema. Ao perder a conexão a réplica interrompia sua execução, mas ao colocar novamente o cabo de rede na máquina física a réplica não identificava que a má-

quina física havia sido reconectada. Foi identificada então uma limitação do Treplica em relação a sua confiabilidade.

Foi necessário fazer uma correção no Treplica, para resolver o problema identificado. A abordagem anterior adotada pelo Treplica era a de lançar uma exceção para qualquer erro relacionado ao envio de mensagem. A modificação feita permite tratar a exceção e, quando ocorre uma exceção relacionada a E/S ao enviar uma mensagem, o Treplica tenta reenviá-la, caso o cabo de rede seja conectado a tempo, então o Treplica envia a mensagem para os membros do grupo e a réplica que havia sido interrompida recebe o novo estado do grupo, mantendo assim a consistência do estado entre todas as réplicas. Essa abordagem é muito importante pois torna transparente para usuários e aplicações possíveis falhas externas que são comuns em um ambiente de rede.

6. CONCLUSÃO

A abstração utilizada pelo JGroups é baixa se comparada ao Treplica. No JGroups é necessário dar um nome para o grupo criado e é necessário conectar o membro a esse grupo em questão através do *channel* ou de um bloco de construção, consequentemente, os mesmos devem ser gerenciados pela própria aplicação. Ao desenvolver a aplicação para o Treplica, mudou-se o paradigma de desenvolvimento. Enquanto no JGroups utiliza-se um paradigma orientado a troca de mensagens, o Treplica utiliza um paradigma orientado ao estado, o que pode levar mais facilmente a erros conceituais que acarretam na inconsistência do estado. Por outro lado, o Treplica garante mais transparência. Além disso, para a aplicação desenvolvida, o JGroups apresentou bugs, não permitindo que ele suportasse as possíveis falhas introduzidas no sistema. Teoricamente isso deveria ser suportado modificando a pilha de protocolos, mas a pilha montada não é suportada pelo JGroups. O Treplica por outro lado consegue tratar as falhas apresentadas de maneira transparente, permitindo que o desenvolvedor não se preocupe com possíveis falhas externas, aumentando sua produtividade. A ferramenta de testes desenvolvida serviu para validar os ambientes de desenvolvimento tanto do Treplica quanto do JGroups, mas ela pode ser utilizada para validar outras ferramentas que se propõe a garantir consistência em aplicações distribuídas.

7. BIBLIOGRAFIA

[1] J. Gray, P. Helland, P. O'Neil, and D. Shasha. The dangers of replication and a solution. In SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data, pages 173–182, New York, NY, USA, 1996. ACM Press.

[2] T. Sterling, D. J. Becker, J. E. Dorband, D. Savarese, U. A. Ranawake,and C. V. Packer. BEOWULF: A parallel workstation for scientific computation. In Proceedings of the 24th International Conference on Parallel Processing, pages I:11–14, 1995.

[3] G. M. D. Vieira and L. E. Buzato. Treplica: Ubiquitous replication. In SBRC '08: Proc. of the 26th Brazilian Symposium on Computer Networks and Distributed Systems, Rio de Janeiro, Brasil, May 2008.

[4] G. M. D. Vieira and L. E. Buzato. Implementation of an object-oriented specification for active replication using consensus. Technical Report IC-10-26, Institute of Computing, University of Campinas, Aug. 2010.

[5] T. D. Chandra, R. Griesemer, and J. Redstone. Paxos made live: an engineering perspective. In PODC '07: Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing, pages 398–407, New York, NY, USA, 2007. ACM Press.

[6] K. P. Birman. The process group approach to reliable distributed computing. Commun. ACM, 36(12):37–53, 1993.

[7] K. P. Birman and T. A. Joseph. Reliable communication in the presence of failures. ACM Trans. Comput. Syst., 5(1):47–76, 1987.

[8] R. Friedman and R. van Renesse. Strong and weak virtual synchrony in Horus. In SRDS '96: Proceedings of the 15th Symposium on Reliable Distributed Systems (SRDS '96), page 140, Washington, DC, USA, 1996. IEEE Computer Society.

[9] M. Barborak, A. Dahbura, and M. Malek. The consensus problem in fault- tolerant computing. ACM Comput. Surv., 25:171–220, June 1993.

[10] A. D. Birrell, M. B. Jones, and E. P. Wobber. A simple and efficient implementation of a small database. In SOSP '87: Proceedings of the eleventh ACM Symposium on Operating systems principles, pages 149–154, New York, NY, USA, 1987. ACM Press.

[11] L. Lamport. The part-time parliament. ACM Trans. Comput. Syst., 16(2):133–169, 1998.

[12] L. Lamport. Lower bounds for asynchronous consensus. Distributed Computing, 19(2):104–125, June 2006.

[13] R. van Renesse, K. P. Birman, M. Hayden, A. Vaysburd, and D. Karr. Building adaptive systems using Ensemble. Softw. Pract. Exper., 28(9):963–979, 1998.

[14] B. Ban. Design and implementation of a reliable group communication toolkit for java. Technical report, Cornell University, 1998.

[15] Y. Amir, C. Danilov, and J. R. Stanton. A low latency, loss tolerant architecture and protocol for wide area group communication. In DSN '00: Proceedings of the 2000 International Conference on Dependable Systems and Networks, pages 327–336, Washington, DC, USA, 2000. IEEE Computer Society.

[16] H. Miranda, A. Pinto, and L. Rodrigues. Appia: A flexible protocol kernel supporting multiple coordinated channels. In ICDCS '01: Proceedings of the The 21st International Conference on Distributed Computing Systems, pages 707–710, Washington, DC, USA, Apr. 2001. IEEE Computer Society.

Aprendizagem em Redes Socias: uma Análise de Dados do Twitter

Guilherme M. Torres
Univ. Federal De São Carlos Rdv João Leme Dos Santos, Km 110 – CEP 18052-780 – Sorocaba – SP - Brasil
guimato@gmail.com

Luciana A. M. Zaina
Univ. Federal De São Carlos Rdv João Leme Dos Santos, Km 110 – CEP 18052-780 – Sorocaba – SP - Brasil
lzaina@ufscar.br

Tiago A. de Almeida
Univ. Federal De São Carlos Rdv João Leme Dos Santos, Km 110 – CEP 18052-780 – Sorocaba – SP - Brasil
talmeida@ufscar.br

Resumo

A Web 2.0 fez com que o uso de aplicações relacionadas a redes sociais tenham crescido. O *Twitter* tem se destacado por ser um meio de colaboração, comunicação e de troca de ideias de pessoas com interesses em comum. Este artigo apresenta um algoritmo que tem como objetivo buscar padrões de intersse em mensagens do *Twitter*, a partir de um conjunto de palavras-chave em um ambiente de aprendizagem. Um experimento foi realizado durante o segundo semestre de 2011 com um grupo de alunos que seguiam docentes da área de Computação. Foi realizada uma análise das mensagens coletadas.

Categorias

H.1.2 [**User/Machine Systems**]: Human information processing

Termos Gerais

Experimentation

Palavras-chaves

Tokenização, minerção de dados, redes sociais, aprendizagem colaborativa.

1. INTRODUÇÃO

O surgimento da Web 2.0 fez com que aplicações relacionadas a redes sociais fossem amplamente utilizadas. Isto porque suas características favorecem a expressão e socialização por meio de ferramentas de comunicação e colaboração como *blogs*, *wikis* e redes sociais em geral [1]. As redes sociais online sociais surgiram com diferentes propósitos sendo alguns exemplos: rede de profissionais, *LinkedIn*, redes para compartilhamento de mensagens curtas, *Twitter*, compartilhamento de vídeos, *Youtube*, redes de amigos, *Facebook* [2].

As redes sociais tem fomentado a utilização de seus ambientes para a aprendizagem. Nelas os usuários têm a possibilidade de expressar seu conhecimento e compartilhar este com diferentes pessoas que possuem os mesmos interesses [3]. Dentre as diversas ferramentas de redes sociais o *Twitter* tem se destacado. O *Twitter* pode ser definido como um serviço de *microblogging*, que utiliza mensagens curtas (140 caracteres ou menos) para transmitir uma informação, podendo ser executado em diferentes dispositivos. Diariamente vinte milhões de usuários geram mais de cinquenta milhões de mensagens[4]. Analisar as mensagens postadas no *Twitter* pode auxiliar no levantamento de algumas informações sobre os usuários. Além de verificações simplesmente numéricas, como o número de seguidores que certa pessoa tem, é possível analisar as mensagens postadas através de técnicas de mineração de textos e análise de padrões. A mineração de textos tem sido utilizada na área de aprendizagem eletrônica com o objetivo de auxiliar não só o professor na identificação de sucessos e insucessos durante o processo de ensino-aprendizagem, como também para identificar, ou mesmo recomendar assuntos que possam contribuir com a aprendizagem do aluno [5]

O objetivo deste trabalho é propor um algoritmo com o objetivo de buscar padrões de interesse em mensagens do *Twitter*, a partir de um conjunto de palavras-chaves. Para isto foram coletadas e analisadas mensagens do *Twitter,* postadas por docentes que estejam sendo utilizadas com a finalidade de aprendizagem. Para validar o algoritmo foi realizada uma experiência, onde foi analisado se os alunos que seguiam os docentes reencaminhavam as mensagens ou mesmo, postavam novas mensagens sobre o mesmo tema. Um questionário foi aplicado aos alunos que participaram da experiência com o objetivo de analisar os resultados obtidos a partir do algoritmo.

O restante do artigo está organizado da seguinte forma: Seção 2 faz uma breve discussão dos fundamentos teóricos do trabalho. Seção 3 relata alguns trabalhos relacionados. Seção 4 é proposto o algoritmo para análise dos dados. Seção 5 é feita a experimentação e análise dos resultados. Seção 6 as considerações finais sobre o trabalho.

2. FUNDAMENTAÇÃO TEÓRICA E TECNOLÓGICA

Para que o trabalho pudesse ser desenvolvido alguns conceitos sobre mineração de textos foram estudados, conforme descrito a seguir.

A mineração de textos pode ser definida como a procura por padrões em um texto em linguagem natural através de um processo de análise do texto, buscando extrair informações deste texto para um propósito em particular. A mineração de textos é dividida em etapas: identificação do problema, pré-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Workshop de Trabalhos de Iniciação Científica'12, 15 a 18 de Outubro, 2012, São Paulo, São Paulo, Brazil.

processamento, extração de padrões (*data mining*), pós-processamento e utilização do conhecimento [5].

O pré-processamento é a etapa de preparação dos textos que irão ser utilizados na mineração. Um dos problemas na mineração de textos é que os dados não se encontram estruturados implicando em uma limitação na utilização dos algoritmos de aprendizagem de máquina. Uma das maneiras de se estruturar os dados é transformando-os em uma matriz de atributo-valor na qual a frequência das palavras, independentemente do seu contexto, é contado. Para produzir uma tabela com atributo-valor relevante ao problema, é realizada a tokenização, que examina um texto não estruturado e identifica suas características importantes separando o texto em *tokens* (palavras). O processo de tokenização pode ser definido como um processo de análise léxica, que analisa uma entrada de linhas de caracteres e gera uma sequência de símbolos. Durante a tokenização é necessário remover alguns caracteres indesejados, como sinais de pontuação, separação silábica, marcações especiais e números, os quais, isoladamente fornecem pouca informação [7]. Outro conceito estudado está relacionado as *stopwords, que* são palavras que devem ser desconsideradas, pois não agregam significado para o algoritmo de aprendizagem. Também foram estudadas expressões regulares, que são métodos de identificar um padrão em um texto [6].

Também foi necessário estudar a API do *Twitter*[1] para poder recuperar as mensagens postadas. A API é dividida em três partes, duas API REST que permite os desenvolvedores a acessar todos os dados do *Twitter* incluindo atualizações, status dos dados e informações de um usuário.

3. TRABALHOS RELACIONADOS

Alguns trabalhos relacionados ao uso de redes sociais como ambientes de aprendizagem foram estudados [9],[10],[11], [9], [10].

O *Twitter* foi introduzido como uma ferramenta para compartilhar ideias em um curso, em que os alunos acompanharam os docentes. Um dos resultados observados pelos autores foi que os alunos conseguiam escrever de maneira concisa as novas descobertas que tinham sobre um determinado tema, compartilhando essas ideias com outros colegas [9].

Experimentos com alunos chineses foram realizados em um curso de língua inglesa. Após os experimentos um questionário foi aplicado aos alunos onde 62% deles relataram que gostaram da experiência com o *Twitter*. Também foram feitas pesquisas com o *Twitter* para educação em saúde, utilizando como ferramenta de feedback para artigos científicos [10].

O aumento da comunicação entre os alunos e docentes através do uso do *Twitter* foi relatado em [11]. Docentes recebem mensagens diretas dos estudantes sobre o curso. Também foi analisada as mensagens trocadas entre eles no *Twitter* para poder identificar as questões que mais lhe interessam.

Um estudo comparando as mensagens postadas no *Twitter* durante um determinado período e os termos mais citados naquele período é apresentado em [9], relacionando estes com o perfil das pessoas que postaram as mensagens (idade, sexo, localização geográfica, etc). Para analisar as mensagens e identificar os perfis foi utilizado o algoritmo de *Kohonem* para auto-organização de mapas através do aplicativo Viscovery SOMine[2]. Os autores concluíram que os termos mais citados estão diretamente relacionados a alguma característica do perfil de quem postou a mensagem.

Outro estudo sobre a relevância do *Twitter* como ferramenta para disseminação de informações durante um desastre natural é discutido em [10]. Observou-se as mensagens postadas durante dois desastres naturais diferentes. Foram que coletadas mensagens do *Twitter* que possuiam as *tags* relevantes a localização geográfica do desastre natural. As mensagens coletadas foram analisadas através do e-Data Viewer[3], buscando observar a formação de grupos que caracterizassem uma categoria relevante ao fenômeno natural. Concluiu-se que as categorias estão relacionadas ao fenômeno natural ocorrido.

4. ALGORITMO PROPOSTO

A partir da fundamentação teórica e dos trabalhos relacionados este trabalho propõe um algoritmo com o objetivo de buscar padrões de interesse em mensagens do *Twitter*, a partir de um conjunto de palavras-chaves. Para isso, é utilizada uma etapa de tokenização das mensagens e depois são criadas matrizes que permitem verificar a ocorrência e frequência de palavras nas mensagens. Para execução do algoritmo deve-se considerar um conjunto de mensagens postadas no *Twitter*.

A partir das listas de *tokens* extraídas de cada mensagem avaliada, é criada uma matriz de ocorrência **D(|T|,|M|)**, onde **T** é a lista dos *token*s e **|T|** sua respectiva cardinalidade; **M** o conjunto das mensagens extraídas do *Twitter* e **|M|** a cardinalidade. Seja **m**, cada mensagem extraída do conjunto de mensagens, onde **m** pode ser descrito como uma matriz de *tokens*, **m(t1t2t3t4...tn)**, para cada mensagem é utilizado a função de tokenização, ***ftoken*(m)** que retorna uma lista de *tokens* extraídos da mensagem. Em (1) é apresentado o trecho do algoritmo que retrata o processo de criação das matriz **D**:

```
j <- 0
  for each m∈M
    ftoken(m)
      for each t_k ⊂ m
        i <- search(T, t_k)
          if i != 0                                   (1)
             D(i,j) <- 1
          else
             T  <- T∪t_k
             D(|T|+1,j) <- 1
  j++
```

[1] http://dev.twitter.com/doc.

[2] http://www.viscovery.net/somine/

[3] http://www.cs.colorado.edu/~starbird/e-dataviewer.html

A função ***ftoken*(m)**, descrita no algoritmo representado em (2.1), tem o objetivo de realizar a análise léxica da mensagem. Dois conjuntos de delimitadores são usados para identificar os *tokens*. Isto é necessário porque nas mensagens do *Twitter* é muito comum existirem *links* dentro das mensagens, e estes *links* podem ser relevantes a análise. O primeiro conjunto (**Dlin**) especifica apenas delimitadores mais gerais como quebra de linha, espaço em branco e tabulação. O segundo (**DlinLink**) utiliza delimitadores mais comuns na escrita de textos como pontos e vírgulas, dois pontos, travessão, entre outros. A função de tokenização é executada em duas etapas. Primeiro, é lido caractere por caractere da mensagem, armazenando-os em **a**. Durante a leitura são ignorados os delimitadores mais gerais. Quando é encontrado um caractere que não é um delimitador geral, se inicia a etapa de criação do *token*, onde são concatenado os caracteres lidos em **a,** até que um delimitador geral seja novamente encontrado.

for each m∈M

while a == Dlin

a++ **(2.1)**

while a != Dlin

t_k <- $t_k \cup$ a

a++

Após a etapa (2.1) o algoritmo de tokenização, verifica se o *token* extraído não é um *link*. Para esta verificação é empregada uma expressão regular. Caso o *token* considerado não seja um *link*, um outro processo de tokenização, apresentado em (2.2) é executado utilizando o segundo conjunto de delimitadores.

if t_k != Link

While a == DlinLink

a++

While a != DlinLink

t_{k2} = $t_{k2} \cup$ a

a++ **(2.2)**

T <- T $\cup$ t_{k2}

else

T <- T $\cup$ t_k

Após a matriz de ocorrência ser criada, deve-se construir uma matriz de frequência. Seja **F(|T|)**, a matriz de frequência, onde **T** é a matriz que contém os rótulos dos *tokens*. Cada linha da matriz de frequência **F** possui correspondência com a linha que possui o respectivo rótulo do *token* da matriz **T**. Para cada $t_k \in$ **T**, percorre-se todas as mensagens em **D**, somando a frequência que em que t_k aparece nas mensagens. A construção de **F** é representada por (3):

$$\sum_{j=0}^{|M|} \mathbf{D}(t_k,j), \qquad (3)$$

A partir da definição de uma lista de termos considerados relevantes para o domínio, **R**, e considerando a matriz de frequência **F**, é construída uma matriz de frequência relevante **Y(|R|)**. O algoritmo que reporta a construção de **Y** é apresentado em (4):

for each cada t_k∈F

i = search(t_k,R)

if i != 0 **(4)**

j <- search(t_k,T)

Y(i) <- F(j)

São construídas duas matrizes de termos relevantes, uma considerando o docente e outra considerando os alunos. Após a definição do matriz de frequência relevante **Y**, para alunos e docentes, é realizada a intersecção desses conjuntos para verificar com que frequência os alunos e docentes utilizam os mesmos termos (5).

$$\mathbf{Y}_{\mathbf{teacher}} \cap \mathbf{Y}_{\mathbf{student}} \qquad (5)$$

5. EXPERIMENTAÇÃO E ANÁLISE DOS RESULTADOS

Para validação do algoritmo utilizou-se MatLab[4] na sua implementação. A escolha deste ambiente de desenvolvimento foi devido a seu desempenho com matrizes e com um grande volume de dados. O Matlab é um ambiente de programação para desenvolvimento de algoritmos, análise de dados, visualização e cálculo numérico de alto desempenho, integrando cálculo com matrizes, processamento de sinais e construção de gráficos.

Para a avaliação do algoritmo proposto foi realizado um experimento, considerando as mensagens postadas no *Twitter* de dois docentes que atuam nas áreas de desenvolvimento para Web, engenharia de software e empreendedorismo; e dos alunos que seguiam esses docentes. Durante o segundo semestre de 2011 foram recuperadas, a cada quinze dias, durante cinco meses, mensagens postadas pelos docentes. Foram também recuperadas as mensagens de 52 alunos que seguiam esses docentes. Criando uma base de dados para ser utilizada na avaliação contendo 1794 mensagens, sendo 118 dos professores, e 1676 dos alunos. É importante destacar que os docentes não fizeram nenhuma recomendação aos alunos que os seguiam. Isto porque se desejava observar o comportamento destes alunos em uma rede social, sem um direcionamento por parte do docente.

Considerando a base de dados das mensagens coletadas, foram criadas as matrizes de frequência dos termos relevantes dos alunos e dos professores utilizando os algoritmos propostos na Seção 3. Os docentes indicaram os termos que deveriam ser considerados na matriz de termos relevantes, termos estes relacionados às disciplinas ministradas pelos docentes. Através da execução do algoritmo observou-se que os alunos que acompanhavam os docentes no *Twitter* realizavam com baixa frequência o reenvio de mensagens ou mesmo a postagem de mensagens que contivessem os termos relevantes. Dos 36 termos relevantes considerados pelos docentes constatou-se que apenas 2 termos foram mencionados pelos alunos em novas postagens ou

[4] http://www.mathworks.com/products/matlab/

reenvios. Para fundamentar as conclusões sobre o uso do *Twitter* com os alunos, neste experimento, foi elaborado um questionário com oito perguntas que buscavam identificar quais eram os termos que mais interessava os alunos, a frequência que eles usavam o *Twitter* e se as mensagens dos professores estavam contribuindo para a aprendizagem deles. O objetivo era verificar se os alunos eram agentes apenas receptores. Ou seja, os alunos acompanharam as mensagens postadas, mas não as repassava aos seus seguidores.

Dos 52 alunos que tiveram suas mensagens coletas, 38 responderam o questionário, ou seja, 73%. Alguns dados importantes foram levantados a partir das respostas dos alunos. Para a pergunta sobre a frequência com que os alunos repassavam as mensagens sobre os termos relevantes, praticamente 80% dos alunos responderam que a frequência com que eles repassam é inferior a dois.

Dentre os termos que os alunos relataram no questionário, os que mais se destacaram foram: "Internet das coisas", "Web", "Android", "Dispositivos móveis", "Google" e "*smathphones"*. Diferindo assim com os dados obtidos pela mineração das mensagens já que os alunos apenas reenviaram mensagens sobre "jobs" e "google". Cerca de 70% dos alunos relataram que acessam os links contidos nas mensagens; e 87% afirmaram que acessam os links relacionados aos termos considerados relevantes. Após ler alguma notícia sobre esses termos, 68% dos alunos relataram que buscam mais informações sobre esse tema na web. Porém, todos os alunos (100%) responderam que as mensagens que eles receberam contribuíram para adquirir novas informações.

Após a comparação dos resultados obtidos com a execução do algoritmo proposto e com a aplicação do questionário, conclui-se que os alunos são agentes receptores de informações no ambiente *Twitter*. Através desta conclusão observa-se a necessidade de que os ambientes de redes sociais necessitam de funcionalidades diferenciadas para a aprendizagem.

6. CONSIDERAÇÕES FINAIS

Este trabalho teve como objetivo propor um algoritmo para buscar padrões de interesse em mensagens do *Twitter*, a partir de um conjunto de palavras-chaves. Para validar o algoritmo foram coletadas e analisadas mensagens de docentes que estavam sendo utilizadas com a finalidade de aprendizagem. Um questionário foi aplicado aos alunos que participaram da experiência, buscando comparar os resultados obtidos com o algoritmo e a opinião direta dos alunos.

A partir dos resultados obtidos com a execução do algoritmo proposto e com a aplicação do questionário, conclui-se que os alunos são agentes receptores de informações no ambiente *Twitter*. Através desta conclusão observa-se a necessidade de que os ambientes de redes sociais necessitam de funcionalidades diferenciadas para a aprendizagem. Como futuro trabalho está sendo planejada uma comparação dos links existentes nas mensagens com os termos relevantes considerados na experimentação. O objetivo é tentar analisar se os alunos reenviam os links a outras pessoas através do *Twitter*.

7. AGRADECIMENTOS

Agradecemos a FAPESP pelo apoio financeiro.

8. REREFÊNCIAS

[1] Recuero, Raquel (2009). Redes Sociais Na Internet. Editora Sulina, 2009.

[2] Nielsen Online (2009). "Social networks & blogs now 4th most popular online activity, ahead of personal email". Disponível em: http://www.nielsen-online.com/pr/pr_090309.pdf. Acessado em: 20/01/2011.

[3] Dabbagh, Nada; Reo, Rick (2010). "Back to the future: tracing the roots and learning affordances of social software". In: Lee, Mark J.W; McLoughlin, Catherine, Web 2.0-Based E-Learning: Applying Social Informatics for Tertiary Teaching, 2010.

[4] Cormode, Graham; Krishnamurthy, Balachander; Willinger, Walter (2010). "A Manifesto for Modeling and Measurement in Social Media". First Monday (Online), Vol. 15, N. 9, 2010. Disponível em: http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/3072. Acessado em: 19/10/2010.

[5] Machado, Aydano; Ferreira, Rafael; Bittencourt, Ig Ibert; Elias, Endhe ; Brito, Patrick ; Costa, Evandro de Barros (2010). "Mineração de Texto em Redes Sociais aplicada à Educação a Distância". Colabor@ (Curitiba), Vol. 6, N. 23, p. 1-21, 2010.

[6] Dunlap, Joanna C. ; Lowenthal, Patrick R. (2009). Tweeting the night away: Using Twitter to enhance social presence. Journal of Information Systems Education, Vol. 20, N. 2, pp. 129-136, 2009.

[7] Borau, Kerstin; Ullrich, Carsten; Feng, Jinjin; Shen, Ruimin (2009). "Microblogging for Language Learning: Using Twitter to Train Communicative and Cultural Competence". ICWL 2009, LNCS 5686, pp. 78–87, 2009.

[8] JSOnline (2009). "Professors experiment with Twitter as teaching tool". Disponível em: http://www.jsonline.com/news/education/43747152.html. Acessado em: 27/06/2012.

[9] Marc Cheong and Vincent Lee. 2009. Integrating web-based intelligence retrieval and decision-making from the twitter trends knowledge base. In Proceedings of the 2nd ACM workshop on Social web search and mining (SWSM '09).

[10] Sarah Vieweg, Amanda L. Hughes, Kate Starbird, and Leysia Palen. 2010. Microbloggingduring two natural hazards events: what twitter may contribute to situational awareness. In Proceedings of the 28th international conference on Human factors in computing systems (CHI '10).

Wizard para Autoria Gráfica de Documentos NCL com Templates

Douglas Paulo de Mattos, Júlia Varanda da Silva, Débora Christina Muchaluat Saade
Departamento de Ciência da Computação/Instituto de Computação, Laboratório MídiaCom
Universidade Federal Fluminense - UFF
R. Passo da Pátria, 156 – São Domingos – Niterói – RJ - 24210240
(21) 2629-5690
{douglas, julia, debora}@midiacom.uff.br

ABSTRACT
Using the declarative approach, interactive applications for the Brazilian digital TV system are developed using NCL (Nested Context Language). Although declarative languages facilitate building interactive TV applications, the author needs to have at least a basic knowledge of NCL. This paper proposes a wizard tool to allow the authoring of NCL documents for those with no knowledge about NCL. The proposed tool uses composite templates built with the XTemplate 3.0 language, which specify generic structures for NCL programs. The wizard presents a graphical interface which has drag-and-drop support for filling template components with specific media content nodes chosen by the author, making the creation of interactive TV applications easier. To do so, such media nodes are stored in a repository that organizes them by type. Furthermore, the graphical editing of content anchors for continuous media nodes is also provided. This wizard tool is a new version of a plugin of the graphical authoring environment for digital TV programs called NEXT (NCL Editor Supporting XTemplate).

Categories and Subject Descriptors
H.5.2 **[Information Interfaces and Presentation]**: User Interfaces – *graphical user interfaces (GUI), interaction styles*. D.2.2 **[Software Engineering]**: Programming environments – *graphical environments, integrated environments.*

General Terms
Design, Languages.

Keywords
TV Digital Interativa, NCL, Ginga, XTemplate, template de composição hipermídia, wizard, ferramenta de autoria, drag and drop, editor gráfico.

1. INTRODUÇÃO
O padrão para o sistema de televisão digital interativa no Brasil utiliza a tecnologia de transmissão do padrão japonês e acrescenta inovações em algumas camadas do sistema de TV digital [1]. Na camada de middleware é apresentado o diferencial do padrão brasileiro, onde é especificado o GINGA [1], que pode utilizar linguagem declarativa ou procedural. No ambiente declarativo, utilizam-se as linguagens NCL (Nested Context Language) [7] e Lua [3] para o desenvolvimento de aplicações interativas, ambas desenvolvidas por universidades brasileiras.

A linguagem NCL é baseada no modelo conceitual hipermídia NCM (Nested Context Model) [8] que permite estruturar logicamente o documento através de contextos (conjuntos de nós e elos) e faz uso do paradigma baseado em eventos para relacionar os objetos de mídia temporalmente e espacialmente. O modelo baseado em eventos é bastante expressivo para autoria de documentos multimídia interativos. Assim, a sincronização dos objetos de mídia (nós de contexto, vídeo, áudio, imagem, texto, programas Lua, etc.) é feita através de elos, cuja semântica da relação é definida por conectores hipermídia. Quando documentos NCL têm muitos componentes e relacionamentos entre eles, a definição de elos pode se tornar complicada para autores que não são especialistas em NCL.

Sendo assim, o propósito deste artigo é apresentar uma ferramenta do tipo wizard para facilitar a autoria de documentos NCL, ou seja, facilitando a criação de conteúdo interativo para o sistema brasileiro de TV digital, até mesmo para autores sem conhecimento da linguagem.

A fim de atender esta necessidade, a ferramenta apresentada neste artigo usa estruturas genéricas de contextos, chamadas de templates de composição, especificados com a linguagem XTemplate 3.0 [5], que podem ser reutilizados em diversos programas com conteúdo distinto, porém com uma estrutura de sincronização similar. Desta maneira, os relacionamentos de sincronização podem ser gerados automaticamente pelo módulo de autoria, sem necessitar que o autor conheça profundamente a linguagem.

A ferramenta, implementada em Java, possui uma interface bastante amigável permitindo aos autores construírem suas aplicações através do recurso *drag–and-drop* (arrastar e largar) para preencher as telas dos templates que serão utilizados para criar a aplicação, de forma bem prática e simples. Para isso, a ferramenta oferece um repositório que armazena as mídias escolhidas pelo autor e as organiza de acordo com seu tipo em uma árvore de navegação, permitindo uma melhor visualização das mídias a serem utilizadas na aplicação NCL. Além disso, as âncoras de conteúdo (trechos de mídia) dos nós de mídia contínua também são definidas de forma intuitiva através do editor de âncoras oferecido pela ferramenta.

Esta ferramenta do tipo wizard é uma nova versão do plugin para utilização de templates de composição do editor gráfico de autoria para TV digital chamado NEXT [6], que oferece um conjunto de plugins para edição de um documento NCL.

A estrutura deste artigo é como se segue. Na Seção 2, são comentados alguns trabalhos relacionados. Na Seção 3, é apresentado o repositório de mídias, oferecido como um recurso auxiliar da ferramenta wizard. A Seção 4 apresenta a ferramenta wizard, detalhando as modificações realizadas em relação ao plugin original do NEXT. Finalmente, na Seção 5, as conclusões são apresentadas e os trabalhos futuros são apontados.

2. TRABALHOS RELACIONADOS

O trabalho desenvolvido por [9] apresenta um conjunto de editores gráficos de âncoras para objetos de mídia imagem, vídeo, áudio e texto. Esses editores tornam mais rápida e precisa a autoria de âncoras, principalmente em mídias contínuas que necessitam ser reproduzidas para melhor definir o intervalo de tempo desejado para cada âncora da mídia.

Outra ferramenta de autoria gráfica que também facilita o desenvolvimento de aplicações NCL é o Berimbau iTV [2], direcionada para profissionais que não possuem conhecimentos de programação e desejam desenvolver aplicações NCL para TV digital. A ferramenta possui uma interface prática e intuitiva que visa tornar a construção de documentos NCL bastante amigável para o autor e também possui um repositório de mídias para facilitar a busca das mídias que irão compor o documento. Entretanto, ela não oferece o uso de templates para auxiliar na construção das aplicações.

A ferramenta Composer 3 [4] é um ambiente de autoria integrado (IDE) que se adapta a diversos perfis de autores e com suporte a requisitos não funcionais. A arquitetura é baseada em um micronúcleo que permite a troca de informações entre os diferentes módulos que complementam o ambiente. Além do micronúcleo, um modelo central e extensões compõem a arquitetura. Suas extensões são construídas através de plugins, que são programas que interagem com o micronúcleo e acrescentam novas funcionalidades à ferramenta. Apesar do Composer facilitar e agilizar a criação de aplicações para TV digital, ele também não permite o uso de templates.

3. REPOSITÓRIO DE MÍDIAS

O repositório de mídias é uma ferramenta auxiliar que permite o armazenamento e visualização das mídias a serem utilizadas em uma aplicação NCL. As mídias são representadas graficamente no repositório como ícones para tornar mais fácil o processo de identificação do conteúdo desejado. Ou seja, para a mídia do tipo imagem, o ícone é representado pelo *thumbnail* (versão reduzida da imagem) da mídia; para mídia vídeo, o primeiro quadro do vídeo compõe o ícone; para os tipos de mídia áudio, texto e outros, são usados símbolos para representá-los. Na Figura 1, é ilustrada a interface gráfica do repositório de mídias.

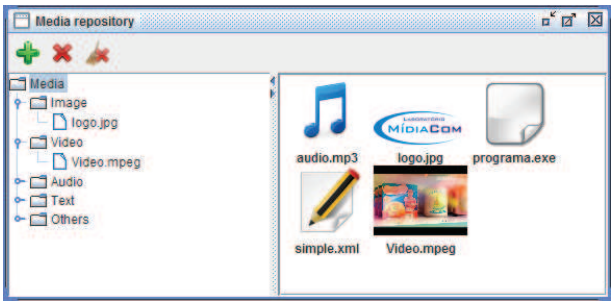


Figura 1. Interface gráfica do repositório de mídias.

Além disso, para facilitar a organização do conteúdo do repositório, é utilizada uma estrutura em árvore. A árvore de mídias possui cinco nós sendo que cada um armazena um tipo de mídia, ou seja, imagem, vídeo, áudio, texto e outros, e é apresentada do lado esquerdo na janela. Quando um dos nós é selecionado na árvore, são apresentadas as mídias que correspondem ao tipo desse nó. Também é possível exibir todas as mídias simultaneamente, selecionando-se o nó raiz (Media).

Existem três botões na ferramenta: o primeiro corresponde à inclusão de mídias no repositório, o segundo exclui a mídia selecionada e o último apaga todo o conteúdo do repositório. As mídias do repositório podem ser arrastadas para outras aplicações que suportam o recurso *drag-and-drop*. Este recurso é bastante utilizado na ferramenta proposta neste artigo.

As principais classes criadas para implementação do repositório são: *Repository, Tree, List, ButtonPanel, TransferList* e *Media*. A classe *Repository* é responsável por apresentar a árvore de mídias, a barra de botões e por exibir a lista de mídias. A classe *ButtonPanel* cria a barra de botões e implementa as ações destes. A classe *Tree* constrói a árvore de mídias e é responsável por mantê-la consistente conforme alterações feitas na lista de mídias. Ela também realiza a ação que deve ser tomada quando um de seus nós é selecionado. A *List* é responsável por guardar as mídias importadas e exibir as representações gráficas de cada tipo de mídia, as quais são criadas pela classe *Media*. A classe *TransferList* permite que as mídias do repositório sejam arrastadas para os plugins do editor gráfico NEXT. A Figura 2 mostra o diagrama de classes implementado.

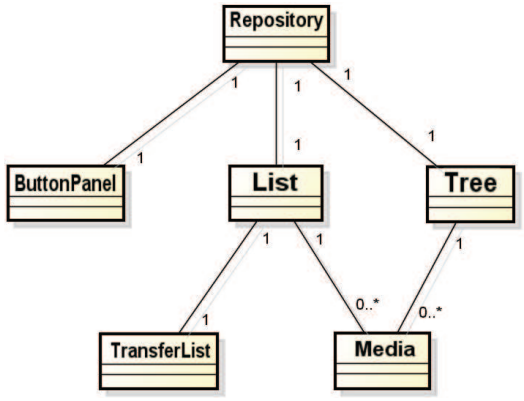


Figura 2. Diagrama de classes do repositório de mídias.

4. A FERRAMENTA WIZARD

O plugin para utilização de templates de composição do NEXT [6] facilita a autoria de aplicações para TV digital através do uso de templates especificados em XTemplate 3.0 [5]. Ele permite que autores possam construir suas aplicações em NCL indicando somente as mídias que irão compor o documento, sem precisar especificar a estrutura e os relacionamentos entre os nós do documento NCL. Além disso, ele agiliza a criação de aplicações que possuem um grande número de linhas de código e que muitas vezes são até repetitivas. Visando exemplificar essa última facilidade, considere um vídeo que possua 100 legendas. Para sincronizá-las com trechos do vídeo é necessária a criação de uma âncora para cada legenda e dos elos para relacioná-las. Assim, a digitação do código dessa aplicação se torna uma tarefa árdua e muito propícia a erros, porém utilizando o plugin de template, todo código é gerado automaticamente apenas com a definição das mídias (vídeo e legendas) pelo usuário.

O plugin utiliza um template como entrada e através da especificação do mesmo suas telas são geradas, indicando graficamente como os nós de mídia da aplicação serão apresentados simultaneamente na TV. A representação gráfica das telas do template indica o seguinte: os retângulos azuis representam os componentes da tela que devem ser preenchidos pelo autor, os vermelhos já são definidos pelo próprio template e as notas musicais representam um componente do tipo áudio que

pode ser azul ou vermelho, seguindo a mesma ideia dos retângulos. Esses componentes são preenchidos através de uma tabela de acordo com a tela atualmente selecionada, como ilustrado na Figura 3.

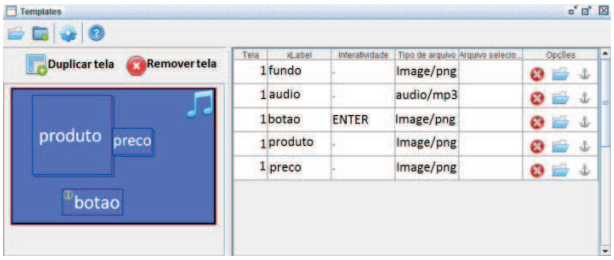


Figura 3. Interface gráfica do plugin original.

Em [6] foram realizados testes de usabilidade com 10 alunos de computação que não conheciam a linguagem NCL e dois alunos que já a conheciam, a fim de avaliar a funcionalidade e praticidade do plugin de templates. Após os testes, os mesmos deram sugestões para o aprimoramento do plugin. As principais sugestões dadas foram: arrastar mídias de um repositório diretamente para a tela do template, exibir a imagem da mídia escolhida no lugar dos retângulos e melhorar a indicação de quando uma âncora pode ser criada na mídia.

A ferramenta wizard, proposta neste artigo, visa facilitar o preenchimento dos componentes de um template oferecendo praticidade e rapidez. Assim, tal preenchimento deixou de ser feito através de tabelas e passou a ser feito na própria tela do template. Ou seja, quando o autor seleciona uma das telas do template na lista de telas, no lugar da tabela é exibida a própria tela selecionada, em tamanho maior, para que seja possível o preenchimento através do recurso *drag-and-drop*. Para isto, foi usado como ferramenta auxiliar, o repositório de mídias. A Figura 4 mostra como o repositório e a ferramenta wizard são usados em conjunto.

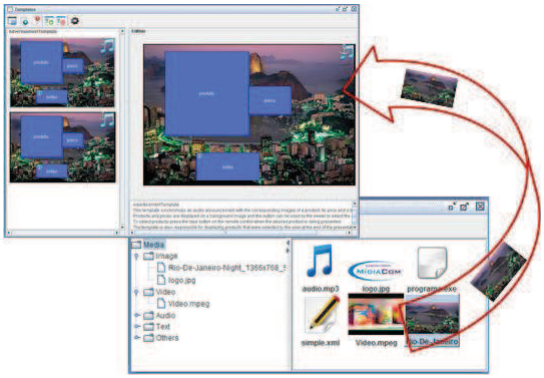


Figura 4. Drag and Drop.

Quando o autor deseja preencher algum componente da tela de um template, ele abre o repositório, seleciona a mídia desejada e a arrasta para dentro do componente a ser preenchido na tela do template. Para um componente (retângulo azul) que receba alguma mídia imagem, ele passa a exibir a imagem desta mídia; caso seja um vídeo, ele exibe o primeiro quadro do vídeo; caso a mídia seja do tipo texto, o retângulo passa a ser de cor verde e o conteúdo textual é exibido no centro do retângulo; e para o tipo outros, é exibido um ícone indicando seu preenchimento. No preenchimento de componente áudio, o símbolo que o representa é alterado para cor verde. A Figura 5 mostra como cada tipo de componente da tela do template é modificado após a escolha da mídia. Os componentes que definem um nó NCL específico, já pré-definido no template, e cujo conteúdo não poderá ser redefinido, deixam de ser representados por um retângulo vermelho e passam, nesta nova versão, a exibir seu conteúdo conforme regras já ditas.

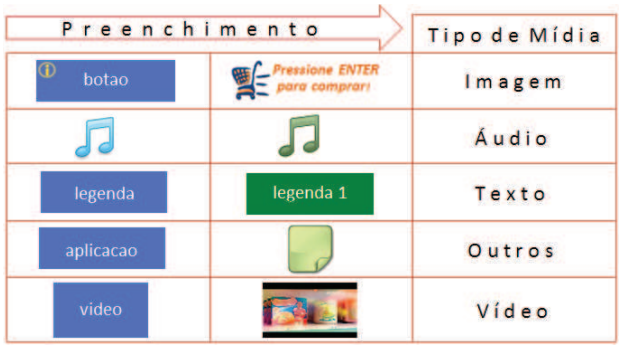


Figura 5. Alteração dos componentes após o preenchimento

Para melhor explicar a funcionalidade específica de um template, a ele é associado um arquivo RDF (*Resource Description Framework*) explicativo. Na primeira versão do plugin, somente a funcionalidade geral do template era exibida ao usuário. No wizard, a partir do documento RDF, além de exibir a explicação geral da função do template, para cada componente da tela do template, também é exibida sua descrição quando o cursor do mouse é posicionado sobre o mesmo, facilitando ainda mais o entendimento do template pelo usuário do wizard.

A criação de âncoras em um componente, na primeira versão do plugin, é feita clicando-se no botão que representa a âncora, localizado na tabela de componentes (vide Figura 3). Porém, este botão está presente em todos os componentes, mesmo quando estes não podem ter âncoras, o que pode confundir o autor. Deste modo, na nova versão, o ícone de âncora é apresentado sobre o desenho somente dos componentes que podem ter âncoras. E, para sua criação ou edição, basta clicar sobre o ícone. A Figura 6 apresenta esta nova indicação de âncora.

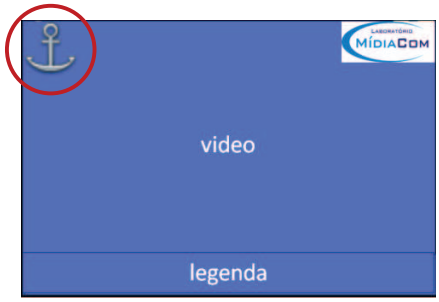


Figura 6. Representação de âncora em um componente

A janela de criação e edição de âncoras de mídias contínuas possui um player que foi implementado utilizando a biblioteca JMF (Java Media Framewok) para reprodução da mídia áudio ou vídeo. Assim, é possível determinar o trecho (início e fim) que define uma âncora da mídia usando a linha do tempo de reprodução da mídia no player. Essa mesma tela exibe ainda a lista de âncoras já criadas para esse componente. Nessa nova versão, a edição de âncoras se tornou mais intuitiva e prática. A Figura 7 mostra a nova interface da janela de criação de âncoras.

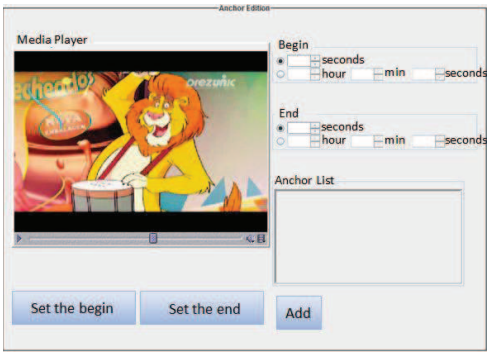


Figura 7. Tela de edição de âncoras.

A implementação desta ferramenta se dá com as seguintes classes principais: *StartPlugin, PluginInfo, MyLayeredPane, MyList, Screen, Template, TempScreen, VocabularyComp, Picture, MediaPanel* e *PlayerEventHandler.* A classe *StartPlugin* é responsável por iniciar o plugin que será chamado pelo NEXT quando o autor desejar utilizá-lo. *PluginInfo* fornece ao NEXT informações sobre o plugin, como o seu apelido e sua lista de elementos NCL de interesse. A classe *MyLayeredPane* tem como função armazenar todas as telas do template, que são apresentadas pela *MyList,* em um panel especial, o qual permite apresentar a tela selecionada na lista em tamanho maior. A classe *Template* armazena as informações do template de composição e a ela está associada a classe *TempScreen*, que armazena todas as telas do template. Já a classe *Screen* corresponde a apenas uma tela do template. As telas são guardadas pela *MyList* e *MyLayeredPane.*

Para armazenar as informações de cada componente do template, foi criada a classe *VocabularyComp,* a qual permite saber o tipo de conteúdo do componente, se este possui interface e se ele é editável ou se foi definido pelo próprio template. Cada componente tem uma representação gráfica associada que será usada para desenhar as telas do template, tal representação é dada pela classe *Picture*. Já as classes *MediaPanel e PlayerEventHandler* trabalham juntas para permitir que se edite ou crie âncoras através do player oferecido pela API JMF. O diagrama de classes é apresentado na Figura 8.

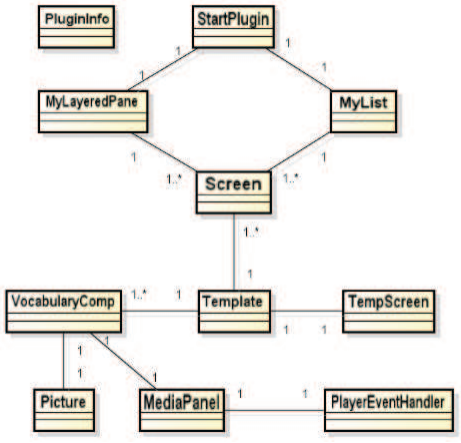


Figura 8. Diagrama de classe da ferramenta wizard.

5. CONCLUSÃO

Este trabalho teve como principal objetivo a construção de uma ferramenta wizard, como uma nova versão do plugin de templates do NEXT, para melhorar o preenchimento das telas dos templates de composição, tornando-o mais prático e rápido. Para isso foi utilizado o recurso *drag-and-drop* para preencher as telas do template, que são exibidas em tamanho maior, cujos componentes apresentam o conteúdo da mídia arrastada ao invés de utilizar uma tabela de componentes. A indicação de âncora passou a ser bem mais clara, visto que somente componentes que podem ter âncoras apresentam o ícone da mesma sobre sua representação. E ainda, para tornar mais precisa e rápida a edição de âncoras, utilizou-se a API JMF, permitindo a reprodução da mídia, áudio ou vídeo, para determinar o intervalo de tempo da âncora do componente.

Como trabalho futuro, será desenvolvido um novo plugin para o NEXT, oferecendo a visão temporal de documentos NCL e permitindo a edição da cadeia temporal da aplicação. Ele ainda terá uma estratégia para tratar a interatividade das aplicações, visto que isso torna a exibição temporal uma tarefa não trivial. O objetivo é tornar mais fácil a alteração da ordem de apresentação das mídias de uma aplicação NCL através de uma interface gráfica amigável, para que autores sem grande conhecimento em NCL possam editar a ordem temporal das mídias em suas aplicações.

6. AGRADECIMENTOS

Este trabalho foi parcialmente financiado pelo CNPq, CAPES e FAPERJ.

7. REFERÊNCIAS

[1] ABNT NBR 15606-2:2007. Televisão digital terrestre - Codificação de dados e especificações de transmissão para radiodifusão digital – Parte 2: Ginga-NCL para receptores fixos e móveis – Linguagem de aplicação XML para codificação de aplicações. Setembro 2007.

[2] Berimbau iTV Author, http://www.batuque.tv/, acessado em novembro de 2011.

[3] Ierusalimschy, R.; Figueiredo, L.H.; Celes, Waldemar. “Lua 5.1 Reference Manual.” Lua.org, Agosto de 2006. ISBN 85-903798-3-3.

[4] Lima, B. S., Azevedo, R. G., Moreno, M. F. e Soares, L. F. – Composer 3: Ambiente de autoria extensível, adaptável e multiplataforma. WebMedia – Workshop de TV Digital Interativa (WTVDI), 2010.

[5] Santos, J. A. F. and Muchaluat Saade, D. C. Linguagem XTemplate 3.0: Facilitando a Autoria de Programas NCL para TV Digital Interativa. In Anais do XV Simpósio Brasileiro de Sistemas Multimídia e Web, 2009.

[6] Silva, J.V e Muchaluat Saade, D. C. NEXT – Editor Gráfico para Autoria de Documentos NCL com Suporte a Templates de Composição, In Anais do XVIII Simpósio Brasileiro de Sistemas Multimídia e Web, 2012.

[7] Soares, L.F.G.; Barbosa, S.D.J. Programando em NCL 3.0 – Desenvolvimento de Aplicações para o Middleware Ginga, TV Digital e Web. Elsevier. 2009.

[8] Soares, L.F.G., Rodrigues, R.F. Nested context model 3.0 part 1 – ncm core. Technical report, PUC-Rio, Rio de Janeiro, 2005.

[9] Vernot, Lucas ; Amorim, T.; Muchaluat Saade, D. C . Editores de Âncoras para Objetos Multimídia. In: Simpósio Brasileiro de Sistemas Multimídia e Web/Workshop de Iniciação Científica, 2005.

Uma Proposta para Estruturação e Visualização Semântica de Resultados de Busca Exploratória

Lucas Pupulin Nanni
Departamento de Informática
Universidade Estadual de Maringá
Av. Colombo, 5790, zona 07, Maringá – PR, 87020-900
lucasnanni@gmail.com

Sérgio Roberto P. da Silva
Departamento de Informática
Universidade Estadual de Maringá
Av. Colombo, 5790, zona 07, Maringá – PR, 87020-900
+55 44 3011 4076
sergio.r.dasilva@gmail.com

RESUMO

A tarefa de identificar informações relevantes sobre um determinado assunto na *Web* tem sido dificultada devido ao grande volume de informação disponível atualmente, tornando-se particularmente mais complexa à medida que o assunto em questão não é de total conhecimento do usuário. O fato dos mecanismos de busca atuais não fornecerem uma organização semântica dos resultados recuperados acarreta dificuldades no entendimento e julgamento dos mesmos quando o usuário realiza buscas exploratórias. Desta forma, este trabalho tem como objetivo obter uma proposta para estruturação e visualização de resultados de busca exploratória baseada em grafos que leve em conta o agrupamento semântico dos resultados recuperados em mecanismos convencionais de busca, promovendo, assim, a facilitação da visualização e identificação dos resultados relevantes aos usuários.

Palavras-chave

Visualização, clusterização, busca exploratória.

1. INTRODUÇÃO

O grande volume de informação atualmente disponível para os usuários da *Web* tem dificultado a tarefa de identificar informações relevantes sobre um determinado assunto. Esta tarefa torna-se particularmente mais complexa à medida que o assunto em questão não é de total conhecimento do usuário, ou seja, quando ele está realizando uma busca exploratória no início de um projeto de pesquisa ou trabalho de graduação, por exemplo. Um problema enfrentado pelos usuários quando realizam buscas exploratórias nos mecanismos de busca atuais é decorrente do fato destes apresentarem seus resultados em forma de lista, empregando muitas vezes uma abordagem puramente léxica, não considerando a ordenação ou agrupamento semântico dos resultados. Esta desconsideração da semântica é particularmente problemática na busca exploratória, pois os usuários não têm conhecimento do jargão léxico mais adequado para a realização da busca, o que é essencial para se escolher as palavras-chave adequadas para a criação de uma *query* de qualidade, isto é, uma *query* que retorne documentos de alta relevância para o usuário. Em geral, a criação de uma *query* de qualidade vai além da escolha das palavras-chave, pois a forma como estas palavras são combinadas na *query* resulta em um número muito variável de resultados ordenados de forma diferenciada.

Um dos aspectos considerados importantes nos mecanismos de busca exploratória é a sua interação com os usuários [4], afetando a forma com que os mesmos constroem suas *queries* e avaliam os resultados da busca [1]. Uma das grandes apostas em técnicas de visualização da informação resultante de buscas é a utilização de grafos [2], já que os mesmos facilitam o agrupamento dos resultados por meio de elementos semânticos, auxiliando o usuário a relacionar e determinar a relevância dos documentos.

O objetivo deste trabalho é obter uma proposta para estruturação e visualização semântica de resultados de busca exploratória baseada em grafos, considerando o agrupamento dos resultados recuperados a partir de um sistema de buscas convencional, a fim de sugerir um protótipo de interface que possa ser avaliado em termos da facilitação da visualização e identificação dos resultados relevantes aos usuários.

Este artigo está organizado da seguinte forma. Na Seção 2 discutimos o problema de busca exploratória. Na Seção 3 falamos sobre o problema da representação visual. Na Seção 4 discutimos nossa proposta de visualização. Finalmente na Seção 5 apresentamos nossas conclusões e trabalhos futuros.

2. A BUSCA EXPLORATÓRIA

Segundo Marchionini [4], existem três tipos de atividades realizadas na busca pela informação. A primeira delas, conhecida como busca *lookup*, é a mais básica delas e tem sido o foco dos sistemas gerenciadores de banco de dados e dos motores de busca na *Web* atuais. Geralmente, as buscas *lookup* são adequadas em estratégias de busca analítica, na qual *queries* bem especificadas geram resultados precisos, sem a necessidade de verificação e comparação dos itens recuperados.

Entretanto, a *Web* tem se tornado fonte primária para aquisição de conhecimento exigindo que os sistemas de busca *lookup* sejam superados. Aliadas ao crescimento do volume de informação disponível surgiram duas outras atividades de busca, desta vez, focadas no aprendizado do usuário. A busca por aprendizagem e por investigação, como são conhecidas, envolvem diversas iterações de busca, requerendo esforço cognitivo desempenhado pelo usuário em análises, comparações e julgamento dos documentos recuperados. Esta atividade de busca e refinamento desenvolvida pelos usuários que se engajam em satisfazer sua necessidade de informação tem sido cunhada como "Busca Exploratória", a qual é caracterizada pela incerteza sobre o contexto da pesquisa e pela própria natureza do problema [4], [8].

O crescimento do interesse pela informação, aliada a grande disponibilidade da mesma, sugere que os problemas associados com a busca exploratória e seus usuários devam ser tratados com maior atenção. Deste modo, surgem ferramentas que, associadas à busca exploratória, visam auxiliar o usuário no entendimento do contexto da sua pesquisa, bem como na aquisição de conhecimento e habilidades. Ultimamente, a utilização de artefatos visuais e interfaces diferenciadas têm se tornado grandes apostas de apoio à busca exploratória [1], [3], [2].

3. A REPRESENTAÇÃO VISUAL

Inúmeras técnicas de visualização vêm sendo estudadas para propor melhorias na organização e apresentação dos resultados de buscas na *Web*. Quando aplicadas no contexto da busca exploratória, estas técnicas são ainda mais valorizadas por auxiliarem o usuário a identificar e relacionar os resultados, permitindo que o processo de compreensão do domínio da busca seja melhorado. Em seu trabalho, Sallaberry *et al*. [8] apresentam um sistema de visualização interativa para análise de conteúdo de resultados de busca. O sistema combina diversos algoritmos para apresentar um leiaute que auxilia os usuários a navegarem através de uma coleção de páginas *Web*.

Para este trabalho foi estudada e aplicada uma técnica de visualização por grafos que, como visto, é considerada uma das grandes apostas para exibição de resultados de buscas. A fim de gerar uma visualização de resultados que auxilie o usuário de busca exploratória a compreender o domínio de interesse, é importante estruturá-la de forma que forneça suporte natural a um componente visual. O problema do agrupamento dos resultados em núcleos (vértices) de acordo com sua semelhança pode ser visto como um problema de clusterização, que geralmente emprega aprendizagem de máquina não supervisionada sobre modelos espaciais para criar grupos (*clusters*) e decidir seus elementos (pontos) pertinentes. Segundo Zamir e Etzioni [7], este agrupamento auxilia os usuários a localizar documentos de interesse e obter uma visão geral do domínio dos documentos recuperados.

Em geral, as técnicas de clusterização de resultados se baseiam em duas abordagens. A primeira, e mais comum, consiste em aplicar um algoritmo clássico de clusterização sobre um conjunto de documentos e então "etiquetar" cada *cluster* gerado com o auxílio de técnicas de identificação de tópicos. Já a segunda utiliza o conceito inverso, descobrindo primeiramente termos representativos e, então, realizando a clusterização a partir deles. Algoritmos como *STC* [10], *Vivisimo* (*http://vivisimo.com*), SHOC [11] e *Lingo*© [6] aplicam esta abordagem utilizando diferentes técnicas de descoberta de termos e de agrupamento. Essa abordagem, empregada neste trabalho, se mostra mais promissora devido à criação dos *clusters* ser restringida aos termos representativos eleitos, gerando desta forma "*clusters* bem-descritos.

4. UMA PROPOSTA DE VISUALIZAÇÃO

O conceito base de nossa proposta de visualização é fornecer uma organização visual inicial dos resultados recuperados a partir de uma consulta realizada em um mecanismo convencional de busca. Para este trabalho, o mecanismo de busca considerado foi o *Google*®, devido a sua popularidade e qualidade reconhecidas. Entretanto, a proposta não é alterada ao ser considerado outro mecanismo, permitindo sua aplicação nas diversas ferramentas de busca disponíveis atualmente.

A organização visual fornecida é considerada inicial, pois sustentará o ponto de partida para uma visualização de resultados estendida que será abordada por um projeto futuro, no qual o usuário poderá externalizar seu modelo conceitual do domínio de busca ao adicionar, alterar e remover os elementos que estruturam a visualização gerada. Junto a esse conceito, outro que também poderá ser abordado é a aplicação da visualização de resultados estendida junto à visualização de histórico de busca, criando um ambiente visual ainda mais enriquecido. O projeto responsável pela visualização de histórico de buscas já está sendo desenvolvido pelo Grupo de Sistemas Inteligentes Iterativos (GSII) (*http://www.din.uem.br/gsii*) e possuirá implementação pareada com a proposta de extensão deste trabalho.

A construção do protótipo de visualização contou com a utilização do *framework* de clusterização *Carrot*^{2} (*http://project.carrot2.org*), da biblioteca de manipulação de documentos *D3.js* (*http://d3js.org*) e da *Google*® *Custom Search API* (*http://developers.google.com/custom-search*). Também foram criados *scripts*, utilizando a linguagem *Python* (*http://www.python.org*) para o processamento e redirecionamento do fluxo de dados entre as ferramentas utilizadas. O processamento dos resultados exigiu a utilização da interface à ontologia léxica *WordNet*© (*http://wordnet.princeton.edu*) fornecida pela ferramenta de processamento de linguagem natural *NLTK* (*http://nltk.org*). As ferramentas apresentadas estão disponibilizadas de forma gratuita e são mantidas atualizadas pelos seus desenvolvedores, o que favoreceu a escolha das mesmas.

A fim de esclarecer as atividades envolvidas no processo de visualização dos resultados, foi sugerida uma arquitetura geral cujo diagrama é ilustrado pela Figura 1.

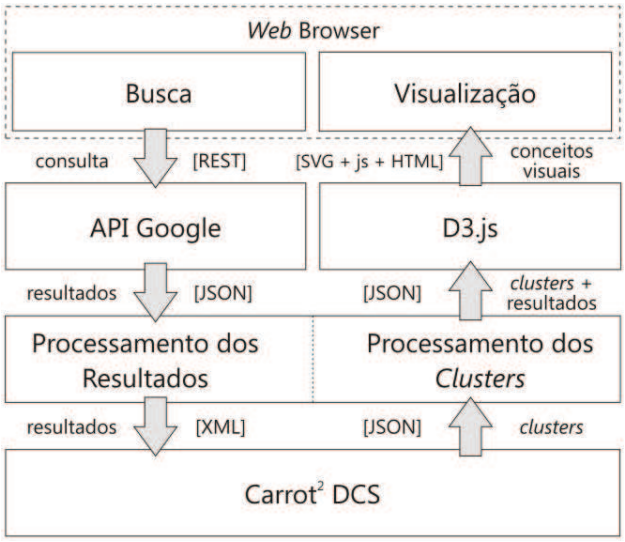


Figura 1 – Arquitetura da visualização proposta.

Por meio do diagrama apresentado é possível identificar que a busca do usuário é redirecionada à *API* de busca do *Google*® retornando um documento no formato *JSON* dos resultados recuperados. O documento retornado é processado e reestruturado a fim de ser submetido ao *Document Clustering Server* (*DCS*), um servidor de clusterização previamente configurado com o algoritmo *Lingo*© e disponibilizado pelo *framework Carrot*^{2}. Os *clusters* criados pelo *DCS* são registrados em um documento *JSON*, o qual especifica os resultados e anotações pertinentes a cada *cluster*. Por fim, o documento contendo o registro dos *clusters* é reestruturado para se adequar as especificações visuais estabelecidas. Com auxilio da biblioteca *D3.js* a visualização final dos resultados é criada junto a uma página *HTML* que será exibida ao usuário como resposta à busca realizada. As setas que envolvem as transações dos dados representam os *scripts* criados para o redirecionamento das entradas e saídas dos componentes utilizados.

4.1 Recuperação e processamento dos resultados

Atualmente existem duas técnicas principais para a recuperação dos resultados de busca na *Web*. A técnica mais primitiva é o emprego de *scraping* sobre as páginas *HTML* retornadas pelo mecanismo de busca, extraindo os resultados das mesmas. A outra técnica, utilizada neste trabalho, consiste em acessar diretamente os resultados por meio de *APIs* oferecidas pelos mecanismos de busca, como a *Custom Search API* do *Google*®, permitindo recuperar a estrutura completa dos resultados.

Após a recuperação dos resultados, estes são processados a fim de normalizá-los semanticamente e reestruturá-los de forma que possam ser clusterizados. A normalização semântica consistiu em inferir o sentido dos termos presentes no título e no resumo de cada resultado e, então, substituí-los pelo lema do sentido associado. O processo de inferir o sentido de um termo envolve buscá-lo em uma ontologia léxica, recuperar um conjunto de conceitos associados a ele e então aplicar alguma técnica de desambiguação de sentidos para determinar o conceito que deve ser associado ao termo. Para a construção e validação do protótipo foram utilizadas a ontologia léxica *WordNet*© e a técnica de desambiguação proposta por McCarthy *et al*. [5]. Com a substituição dos termos pelos lemas correspondentes, espera-se que termos lexicamente distintos, mas semanticamente similares, sejam fundidos e representados por um único termo, auxiliando o processo seguinte a clusterizar os resultados de forma mais concisa.

4.2 Configuração do serviço de clusterização

A escolha e configuração do algoritmo de clusterização foram baseadas nas características discutidas pelas técnicas de clusterização apresentadas. Dentre elas, foi definido que os *clusters* gerados deveriam estabelecer relações entre si, permitindo assim, serem associados visualmente. Além disso, os *clusters* deveriam ser anotados com termos significativos ao usuário, permitindo sua fácil identificação.

Este trabalho considerou a utilização do algoritmo *Lingo*© uma vez que o mesmo provê *clusters* com documentos compartilhados, permitindo estabelecer uma relação simples entre os grupos de documentos. Outro fator que corroborou para a escolha do *Lingo*© foi a possibilidade de configurá-lo de forma que os *clusters* pudessem ser anotados com termos compostos, ou frases, acrescendo na qualidade de descrição dos mesmos. Novamente, a escolha de outras técnicas de clusterização não é restringida pela proposta discutida, cabendo apenas a validação da mesma em relação às características exigidas para a construção da visualização.

Para que o *Lingo*© se adequasse às exigências estabelecidas, o mesmo sofreu alguns ajustes, possibilitando o aperfeiçoamento dos *clusters* gerados. Tais ajustes foram realizados com auxílio do *benchmark* disponibilizado pelo *framework Carrot*[2] e consistiu na modificação de alguns parâmetros como o *"Cluster count base"*, o qual permite ao algoritmo estimar o número aproximado de *clusters* que devem ser gerados. Neste caso, foi determinado que deveriam ser gerados em torno de 9 *clusters*, por este ser um número razoável de elementos passíveis de serem identificados com clareza pelo usuário. Outros parâmetros como *"Phrase Label Boost"* (9) e *"Phrase length penalty-start/stop"* (5) privilegiaram o uso de frases, ou palavras compostas, para a anotação dos *clusters*, fornecendo uma descrição mais aprimorada dos mesmos.

O recurso léxico de *stop words* utilizado pelo algoritmo também foi modificado com a inserção e remoção de regras léxicas, permitindo ao *Lingo*© processar documentos com termos mais abrangentes como *"information"*, e ao mesmo tempo ignorar frases pré-definidas, como *"Wikipedia, the free encyclopedia"*.

A configuração dos parâmetros do algoritmo visou, principalmente, reduzir o número de *clusters* gerados e priorizar a utilização de termos compostos para a anotação dos *clusters*, demandando menos esforço cognitivo por parte do usuário na identificação dos mesmos quando estruturados na visualização proposta.

4.3 Construção do protótipo de visualização inicial

Para a criação da visualização, primeiramente os *clusters* são recuperados fornecendo ao DCS os resultados já processados. Um documento *JSON* com o registro do conjunto de *clusters* é então gerado, no qual um *cluster C* pode ser definido pela tripla (*l*, *R*, *w*), em que *l* é a anotação gerada para o *cluster*, *R* é o conjunto de resultados contidos em *C*, e *w* é a pontuação associada ao *cluster*. O documento que especifica os *clusters* gerados é então processado para que componha um documento final que possa ser utilizado na produção da visualização. O processamento consiste em criar um documento *JSON* estendido que identifique as relações entre os *clusters* e que compreenda a estrutura original dos resultados, como metadados, formatações *HTML* e miniaturas. O relacionamento entre os *clusters* é estabelecido a partir da pertinência mútua dos resultados contidos neles. Um *cluster* C_i está relacionado a outro *cluster* C_j se existe um resultado *r* que pertença tanto ao *cluster* C_i quanto ao *cluster* C_j.

É requerido da representação visual que ela apresente um conceito visual simples e interativo, ao mesmo passo que permita ao usuário identificar de forma clara o significado dos conceitos aplicados. Para tanto, o protótipo de visualização foi construído utilizando o conceito de grafos não direcionados, no qual os vértices da estrutura consistem dos *clusters* gerados e as arestas representam as relações obtidas entre os *clusters*. Além do grafo, um painel lateral complementa a visualização com a listagem dos resultados contidos em um *cluster* selecionado pelo usuário.

Os vértices foram concebidos como círculos dimensionados de acordo com a pontuação do *cluster* associado e, então, coloridos aleatoriamente a partir de uma palheta de 15 cores distintas. Já as arestas, representando as relações entre os *clusters*, foram idealizadas como linhas retas que conectam os centros dos *clusters* relacionados. Para o desenho e interação da visualização foi utilizada a biblioteca *D3.js*, a qual permitiu, além de estruturar de forma adequada os componentes da visualização, manipular a estrutura da página *HTML* que a comportava. A Figura 2 ilustra a visualização gerada para a busca "*information retrieval"*.

A interação provida pela visualização permite ao usuário focalizar um grupo de *clusters* relacionados, bastando para isso selecionar qualquer *cluster* pertencente ao grupo. Uma vez focalizado um grupo de *clusters*, um *cluster* pertencente a ele pode ser analisado ao também ser selecionado. O foco é desfeito ao selecionar um *cluster* que não pertença ao grupo já focalizado, retornando a visualização ao seu estado inicial. O termo seleção foi empregado de forma genérica, não restringido o meio pelo qual se realiza a seleção, como um clique ou um toque. A Figura 3 ilustra a

situação em que o cluster *"Information Retrieval and Web"* foi selecionado e um grupo de 6 *clusters* relacionados foi focalizado.

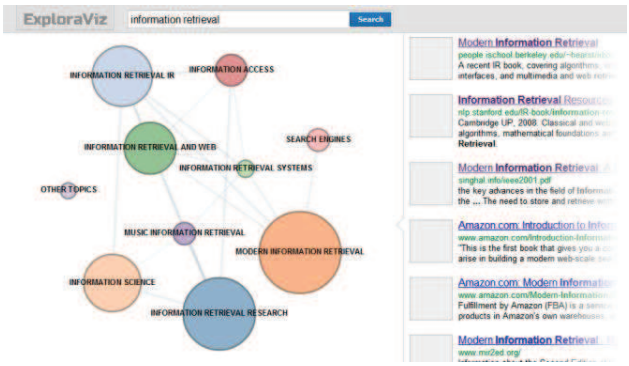


Figura 2 – Visualização gerada para a busca "*information retrieval*".

A visualização permite que os *clusters* sejam reposicionados espacialmente, fornecendo ao usuário a liberdade de organizar a informação apresentada. O posicionamento inicial dos *clusters* é automaticamente realizado pelo algoritmo de disposição de grafos dirigido por forças de atração e repulsão.

O painel de resultados lateral é inicialmente posicionado de forma recolhida, permitindo que o grafo ocupe a área central da visualização e forneça espaço suficiente para complementos visuais futuros, como expansão da proposta inicial. A utilização de miniaturas sugere, ainda que de forma parcial, a organização visual dos documentos recuperados pela busca, complementado a concepção do resultado textual.

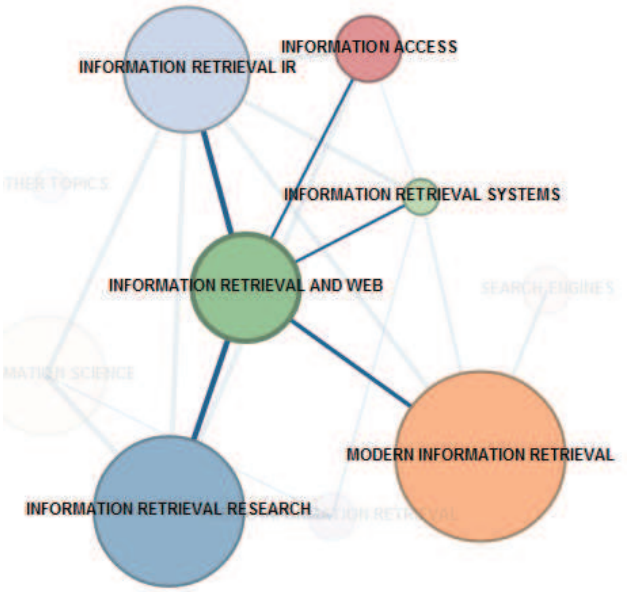


Figura 3 – Grupo de clusters focalizado após seleção do cluster *"Information Retrieval and Web"*.

5. RESULTADOS E CONCLUSÕES

A partir de testes iniciais realizados com o protótipo construído, foi verificado que o processo de clusterização se comportou de forma esperada, gerando um número adequado de *clusters* bem anotados. Acredita-se que o número reduzido de *clusters*, entre 8 e 10, permitirá ao usuário interpretar a visualização gerada com maior facilidade.

Em relação à visualização propriamente dita, esta se mostrou uma proposta viável para que futuramente novos atributos visuais e funcionalidades sejam adicionados a ela. Os resultados obtidos ainda são preliminares e necessitam ser complementados com avaliações qualitativas e quantitativas da proposta visual.

Como discutido anteriormente, este projeto condiciona trabalhos futuros a abordarem a extensão da visualização proposta a fim de permitir ao usuário modificar os elementos visuais apresentados, possibilitando, assim, a externalização de seu conceito de busca. Também poderá ser abordada, futuramente, a integração deste trabalho com a proposta de visualização de histórico de busca, a qual está sendo desenvolvida pelo GSII e estabelecerá um ambiente enriquecido de busca visual.

6. REFERÊNCIAS

[1] Alhenshiri, A. et al. 2010. Augmenting the visual presentation of Web search results. *2010 Fifth International Conference on Digital Information Management (ICDIM)* (Thunder Bay, Jul. 2010), 101–107.

[2] Angelaccio, M. et al. 2007. Graph Use to Visualize Web Search Results: MyWish 3.0. *Information Visualization 2007 IV 07 11th International Conference* (Roma, 2007), 245–250.

[3] Kajinmi, T. et al. 2008. Application of keyword map to decision support through exploratory search. *2008 IEEE International Conference on Systems, Man and Cybernetics* (Oct. 2008), 2177–2181.

[4] Marchionini, G. 2006. Exploratory search. *Communications of the ACM*. 49, 4 (Apr. 2006), 41.

[5] McCarthy, D. et al. 2007. Unsupervised Acquisition of Predominant Word Senses. *Computational Linguistics*. 33, 4 (Dec. 2007), 553–590.

[6] Osinsk, S. 2003. *An Algorithm for Clustering of Web Search Results*. Poznan University of Tecnology.

[7] Sallaberry, A. et al. 2010. Interactive visualization and navigation of web search results revealing community structures and bridges. *Proceedings of Graphics*. (2010), 105–112.

[8] White, R.W. et al. 2005. Exploratory Search Interfaces : Categorization , Clustering and Beyond. *SIGIR Forum*. 39, 2 (2005), 52–56.

[9] Zamir, O. and Etzioni, O. 1999. Grouper : A Dynamic Clustering Interface to Web Search Results. *Work*. 31, 11-16 (1999), 1361–1374.

[10] Zamir, O. and Etzioni, O. 1998. Web document clustering. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '98* (New York, New York, USA, 1998), 46–54.

[11] Zhang, D. and Dong, Y. 2004. Semantic, hierarchical, online clustering of web search results. *Advanced Web Technologies and Applications*. 3007, (2004), 69–78.

Classificação de revisões para construção de perfis em sistemas de recomendação

Thiago Fujisaka Tanaka, Marcelo G. Manzato
Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo
Av. Trabalhador Sancarlense, 400, Caixa Postal 668 – 13560-970
São Carlos, SP – Brazil
tftanaka@grad.icmc.usp.br, mmanzato@icmc.usp.br

RESUMO

Este artigo apresenta uma avaliação de diferentes algoritmos de análise de sentimentos baseados em revisões de filmes criadas por usuários. O principal objetivo é identificar as vantagens e limitações de cada técnica, buscando auxiliar a escolha de uma abordagem a ser utilizada em um projeto de maior escala, que, em síntese, busca construir um mecanismo de criação de perfis de usuários baseado em anotações, como comentários, etiquetas e revisões. Esses perfis irão conter as preferências de usuários sobre diferentes tópicos, e serão utilizados como base para permitir a recomendação de conteúdo adicional de acordo com seus principais interesses.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*

General Terms

Design, Algorithms

Keywords

classificação de sentimentos, construção de perfis, sistemas de recomendação.

1. INTRODUÇÃO

Devido à intensificação no uso da rede mundial de computadores durante os últimos anos, observou-se um grande aumento na quantidade de conteúdo multimídia disponível. Consequentemente, devido a essa sobrecarga de informações, tornou-se necessário um modo no qual usuários possam recuperar os dados conforme seus interesses.

Uma maneira de se realizar a recuperação de conteúdo multimídia é por meio de sistemas de recomendação. A recomendação seleciona automaticamente itens multimídia sem a intervenção do usuário, baseando-se apenas em dados apreciados anteriormente pelo indivíduo, oferecendo outros conteúdos semelhantes. Isso é feito com base em um perfil de interesses que contém o histórico de itens avaliados anteriormente, de modo a se obter uma lista de preferências do usuário [7].

Dois desafios da recomendação multimídia são a indexação multimídia e a construção do perfil de interesses. No primeiro caso, a dificuldade está relacionada em manter todo o conteúdo indexado por palavras-chave relevantes, que não só representem seu título ou o descreva de maneira genérica, mas que envolva também elementos nele presentes, como locais, objetos, pessoas e situações. Essa indexação pode ser feita de maneira automática ou manual. A indexação automática, que possui a vantagem de ser executada de modo ágil e em grande escala, possui domínios restritos, pois detalhes presentes no conteúdo são inerentes à modalidade multimídia e em quais ambientes está presente. Já a indexação manual apresenta maior custo, pois especialistas são necessários para anotar grandes quantidades de itens multimídia disponíveis a cada dia.

Em relação à construção do perfil de interesses, duas metodologias são propostas pela literatura [10]: as coletas explícita e implícita de dados. Na coleta explícita, o próprio usuário fornece informações, como dados demográficos e avaliação de itens visitados anteriormente (notas e comentários). Uma desvantagem é a imposição de esforço adicional, que além de causar desinteresse, permite o fornecimento de informações incorretas. Já a coleta implícita obtém dados sem sua intervenção, exigindo desenvolvimento e manutenção de ferramentas que possam monitorar comportamentos, gerando altos custos e limitando o usuário à utilização de poucos tipos de dispositivos.

Por outro lado, recentes avanços relacionados à Web 2.0 [14] possibilitaram novas maneiras de se obter descrições sobre o conteúdo e usuário. O termo Web 2.0 define uma plataforma em que usuários produzem conteúdo e anotações, possibilitando a utilização de informações produzidas por eles na indexação desse conteúdo, e também, na construção do perfil de interesses. Assim, tem-se uma alternativa à indexação automática ou manual, eliminando problemas como a restrição de domínios e o alto custo, pois as anotações são colaborativas e não há necessidade de consultas a especialistas [12]. Adicionalmente, o ato de o usuário criar uma anotação relacionada a um conteúdo, como comentários, etiquetas, revisões e notas, indica o surgimento de uma reação (positiva ou negativa) nele [20]. Desse modo, o uso de anotações pode auxiliar a construção de seu perfil de interesses [7], e consequentemente, a recomendação de novos conteúdos.

Apesar de existirem vários trabalhos que exploram ano-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WebMedia'12, October 15–18, 2012, São Paulo/SP, Brazil.
Copyright 2012 ACM 978-1-4503-1706-1/12/10 ...$5.00.

tações para criar descrições sobre conteúdo multimídia [2, 9, 8], verifica-se ainda uma lacuna em como determinar a relevância das anotações em relação às preferências do usuário, uma vez que eles podem apresentar opiniões parciais e/ou diversas. Consequentemente, a classificação de sentimento do autor da anotação é um importante mecanismo para possibilitar a construção de perfis baseados em anotações mais eficientes. Como exemplo, dado um comentário razoavelmente completo do usuário sobre um filme, é possível extrair seus sentimentos em relação a diversos tópicos, como atuação do ator principal, assunto geral do filme relacionado ao gênero, cenas de interesse, etc. Essas informações, por sua vez, poderiam ser agregadas em um perfil para possibilitar o fornecimento de conteúdos relacionados.

O presente projeto de iniciação científica visa a extração de sentimentos relacionados a diversos tópicos a partir de anotações para possibilitar a construção de perfis de usuários. Entretanto, um passo anterior necessário para atingir esse objetivo é a classificação geral da anotação em positiva ou negativa. Este artigo tem como objetivo analisar diferentes técnicas de classificação de sentimento utilizando revisões de filmes criadas por usuários. Nesta análise, foram consideradas algumas estratégias de melhoria dos algoritmos existentes, de modo a quantificar o ganho obtido pela incorporação das funcionalidades. Uma dessas estratégias é a utilização da ferramenta SentiWordNet[1], que possibilita a mineração de opiniões pessoais a partir de termos de busca provenientes da base léxica WordNet[2].

Este artigo está organizado da seguinte maneira. Na Seção 2 são apresentados e discutidos alguns trabalhos relacionados. Na Seção 3 as abordagens selecionadas são detalhadas. A Seção 4 apresenta a avaliação dos métodos, comparando os resultados a partir de uma base de revisões única. Por fim, as Seções 5 e 6 descrevem as conclusões e agradecimentos deste trabalho.

2. TRABALHOS RELACIONADOS

Esta seção apresenta uma revisão dos trabalhos relacionados ao processo de classificação de sentimentos. Geralmente, as abordagens exploram diferentes características dos dados com o objetivo de melhorar os resultados de um classificador treinado a partir das informações extraídas.

Um dos trabalhos mais antigos da área foi proposto por Pang et al. [17], que desenvolveram um algoritmo de aprendizado supervisionado que utiliza *bag-of-words* em conjunto com máquinas de vetor de suporte (do Inglês, *support vector machines* – SVM) e unigramas.

Análises mais complexas que consideram partes do discurso (do Inglês, *part of speech* – POS) foram feitas em alguns trabalhos [22, 11, 6, 18], visando melhorar os resultados da classificação de sentimento em diferentes conjuntos de documentos. Além das informações contidas em partes de discurso, Turney [21] e Dave et al. [5] também propõem métodos para detectar e atribuir pesos a padrões de escrita a fim de se obter características dos dados a serem usados na classificação de sentimento em revisões de produtos.

Outras melhorias também foram obtidas por meio da identificação de sentenças objetivas e subjetivas [15], identificação de expressões coloquiais [23], e análise de aspectos do estilo da escrita [23, 1], tendo como objetivo a determinação de opiniões específicas.

Ohana & Tierney [13] exploram a base SentiWordNet para extrair características do texto que são usadas em uma máquina de vetor de suporte para realizar a classificação de polaridade.

Alguns dos trabalhos mencionados e discutidos acima foram selecionados neste artigo para serem avaliados. A próxima seção descreve detalhadamente os algoritmos escolhidos, os quais serão posteriormente avaliados a partir de uma mesma base de dados para possibilitar a comparação das abordagens.

3. CLASSIFICAÇÃO DE REVISÕES

Um algoritmo de fácil entendimento e que apresenta um bom desempenho na área de classificação de revisões com análise de sentimentos é o classificador Naïve Bayes, construído com base na aplicação do teorema de Bayes. Uma abordagem didática deste algoritmo é apresentada pelo curso *online* de Processamento de Linguagem Natural da *Stanford University*[3]. Devido à sua simplicidade, é possível utilizá-lo como algoritmo de patamar para comparação com resultados obtidos de outras abordagens, inclusive aquelas que utilizam a base SentiWordNet.

Um classificador Naïve Bayes é alimentado com um conjunto de dados de treinamento para a realização de inferências em um conjunto de teste. Dado um documento a ser classificado e suas palavras, obtêm-se as probabilidades de esse documento pertencer às classes positiva e negativa. Assim, escolhe-se aquela classe cuja probabilidade é maior. Inicialmente, define-se a probabilidade condicional como:

$$P(c_j|d) \approx P(c_j) \prod_i P(w_i|c_j) \quad , \tag{1}$$

onde $P(c_j)$ é a probabilidade *a priori* da classe c_j (sentimento positivo ou negativo), d o documento a ser analisado e w_i a palavra de índice i do documento. A probabilidade *a priori* é definida como:

$$P(c_j) = \frac{doccount(C = c_j)}{N_{doc}} \quad , \tag{2}$$

onde $doccount(C = c_j)$ é o número de documentos pertencentes à classe c_j e N_{doc} é o número total de documentos.

Em seguida, são calculadas as probabilidades de cada palavra do documento dada uma classe. Tal probabilidade é definida como:

$$P(w_i|c_j) = \frac{count(w_i, c_j) + 1}{\sum_{w \in V} count(w, c_j)) + |V|} \quad , \tag{3}$$

onde w_i representa a palavra de número i no documento, c_j a classe sendo avaliada e V o vocabulário de todo o conjunto de treinamento. Sobre esta fórmula é aplicada a técnica de suavização de Laplace [3], evitando-se que resultados desta probabilidade e da probabilidade condicional sejam zerados caso a palavra não ocorra no conjunto de treinamento.

Por fim, realiza-se o produto das probabilidades de cada classe, resultando em dois valores de probabilidade para um documento (probabilidade de o documento pertencer a uma inferência positiva ou negativa), sendo que o maior desses valores indica a provável classe a qual o documento pertence.

[1] http://sentiwordnet.isti.cnr.it/

[2] http://wordnet.princeton.edu/

[3] https://www.coursera.org/course/nlp/

Buscando aumentar a eficiência do algoritmo, são eliminadas as *stop words*, que são palavras que não possuem nenhum valor sentimental atrelado a elas e que podem influenciar negativamente no algoritmo, caso apareçam demasiadamente em uma das classes do conjunto de treinamento. Além disso, negações são tratadas de modo que, sempre que um termo de negação aparecer (como *"not"* e *"didn't"*), o prefixo *"not_"* seja adicionado a todas as palavras seguintes até a próxima pontuação [4, 17]. Desse modo, um documento que contém uma negação (como em *"that was not good"*) não terá seus aparentes termos positivos tratados de modo errôneo.

Uma outra melhoria na classificação [13] é utilizar a base léxica SentiWordNet, que contém uma mineração de opiniões pessoais a partir de termos de busca provenientes da base léxica WordNet. Em consultas a essa base léxica, além da própria palavra, é necessário saber sua classe gramatical (adjetivo, substantivo ou verbo). Assim, utiliza-se um etiquetador de partes de discurso [19], que adiciona a cada palavra um sufixo que descreve sua classe gramatical.

Tendo obtido as classes gramaticais é possível recuperar da base léxica os valores de positividade e negatividade associados aos termos. São desenvolvidas neste projeto abordagens que baseiam-se na soma destes valores, além da multiplicação dessa soma ao algoritmo de patamar (Naïve Bayes).

4. AVALIAÇÃO

Esta seção apresenta uma comparação dos algoritmos considerados, buscando-se, através da avaliação dos resultados, verificar como as abordagens se relacionam, inferindo possíveis vantagens e desvantagens de cada método.

Para esta avaliação, foi utilizada uma base de dados polarizada, introduzida por Pang & Lee [16], contendo 5331 revisões positivas sobre filmes e 5331 revisões negativas. Sobre esta base, foi aplicado o método de validação cruzada *10-fold*, evitando-se um possível sobreajuste de dados.

Através da contagem de verdadeiros positivos (VP), falsos positivos (FP), verdadeiros negativos (VN) e falsos negativos (FN) é possível calcular os índices de precisão e revocação (*precision e recall*). Em seguida, calcula-se a medida F1 que indica um valor único que combina ambas as métricas anteriores. Este processo é feito a cada particionamento da validação cruzada. Finalmente, as médias dos valores de precisão, revocação e F1 são calculadas para se obter os resultado finais.

Primeiramente, o algoritmo de patamar é aplicado por meio do classificador Naïve Bayes. Os resultados são apresentados na Tabela 1.

Tabela 1: Naïve Bayes.

Métrica/Classe	Positiva	Negativa
Precisão	0.7804	0.7790
Revocação	0.7786	0.7806
F1	0.7795	0.7798

Em seguida, são aplicadas a detecção de negações e a remoção de *stop words*. Em algumas iterações da validação cruzada este algoritmo se saiu melhor que o patamar, porém no geral foi influenciado por outras iterações nas quais apresentou resultados inferiores. Portanto, depois de calculada a média das métricas, a remoção de *stop words* e o tratamento de negações melhoraram somente a revocação da classe positiva. Isso indica que a estratégia é capaz apenas de reforçar aquelas palavras com conotação positiva que aparecem em documentos classificados como positivos. Por exemplo, nas frases *"that was good"* (classe positiva) e *"that was not good"* (classe negativa), a palavra *"good"* teria a mesma probabilidade em ambas as classes no algoritmo de patamar, ao passo que nesse algoritmo, *"good"* terá uma probabilidade maior de pertencer à classe positiva. Consequentemente, novos documentos com inferência positiva contendo a palavra *"good"* serão mais facilmente detectados, aumentando, assim, a revocação.

Tabela 2: Algoritmo de patamar com tratamento de negações e remoção de *stop words*.

Métrica/Classe	Positiva	Negativa
Precisão	0.7534	0.7785
Revocação	0.7895	0.7410
F1	0.7710	0.7592

Na Tabela 3 são apresentados os resultados obtidos para o algoritmo que, para cada palavra pertencente a um documento, resgata um valor sentimental da base léxica SentiWordNet. As somas dos valores de positividade e negatividade são realizadas, seguidas de uma comparação entre elas. O maior valor será a classe inferida àquele documento.

Tabela 3: Algoritmo de soma de valores obtidos da SentiWordNet.

Métrica/Classe	Positiva	Negativa
Precisão	0.5823	0.5995
Revocação	0.6775	0.4982
F1	0.6262	0.5441

Finalmente, na Tabela 4, são apresentados os resultados do algoritmo que combina o algoritmo de patamar com os valores da SentiWordNet. Após o produtório de todas as probabilidades de termos dada uma classe no classificador Naïve Bayes, a soma de todos os valores de positividade ou negatividade desse documento é aplicada à probabilidade resultante do algoritmo de patamar.

Tabela 4: Algoritmo de patamar multiplicado por SentiWordNet.

Métrica/Classe	Positiva	Negativa
Precisão	0.6873	0.6980
Revocação	0.7335	0.6486
F1	0.7094	0.6723

A melhor eficiência foi observada no algoritmo de patamar. Apesar disso, sabe-se que esse algoritmo é dependente do conjunto de treinamento, dependendo portanto de seu tamanho e do domínio envolvido. No cenário de estudo deste projeto, um conjunto de teste envolvendo revisões de filmes apresenta bons resultados. Porém, caso o domínio do conjunto de teste fosse alterado, os resultados poderiam ser inferiores, visto que termos técnicos de outros domínios não seriam reconhecidos, pois não apareceriam no conjunto de treinamento.

Por outro lado, a utilização da base SentiWordNet é interessante por ser um conjunto de dados bem definido e fixo, que não depende do domínio a ser aplicado e de nenhum conjunto de dados de treinamento. Essa independência, entretanto, também apresenta algumas desvantagens, pois em certos domínios algumas palavras podem apresentar valores sentimentais mais fortes do que em outros.

Todas as abordagens apresentaram vantagens e desvantagens, portanto, uma análise mais aprofundada das técnicas será realizada em trabalhos futuros, com a intenção de se

elaborar uma melhor combinação dos algoritmos que resolva ou reduza seus problemas de uma maneira mais eficiente se comparado ao algoritmo de combinação apresentado neste trabalho.

5. CONCLUSÃO

Este artigo apresentou uma avaliação de diferentes algoritmos de classificação de polaridade de revisões em filmes. Uma abordagem baseada em Naïve Bayes foi considerada como patamar, e a partir dela, melhorias foram adicionadas para que fosse quantificado o ganho obtido com essas alterações.

Conforme mencionado na introdução deste artigo, o presente estudo será importante no passo seguinte do projeto de iniciação científica, que é a construção de perfis de interesse para aplicação em sistemas de recomendação. Desse modo, como trabalhos futuros pretende-se investigar meios de se extrair múltiplas inferências de diferentes tópicos a partir de uma única revisão ou comentário. Espera-se como resultado que um perfil de preferências seja construído a partir de usuários colaboradores, oferencendo um mecanismo diferenciado de personalização de conteúdo.

6. AGRADECIMENTOS

Os autores gostariam de agradecer o apoio financeiro das agências PIBIC/CNPq e FAPESP, número de processo 2011/-17366-2.

7. REFERÊNCIAS

[1] A. Abbasi, H. Chen, and A. Salem. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Trans. Inf. Syst.*, 26(3):12:1–12:34, 2008.

[2] S. Angeletou, M. Sabou, and E. Motta. Folksonomy Enrichment and Search. In L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvonen, R. Mizoguchi, E. Oren, M. Sabou, and E. P. B. Simperl, editors, *6th. European Semantic Web Conference*, volume 5554 of *Lecture Notes in Computer Science*, pages 801–805. Springer, 2009.

[3] P. R. C. D. Manning and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[4] S. Das and M. Chen. Yahoo! for amazon: Extracting market sentiment from stock message boards. In *Proceedings of the Asia Pacific Finance Association Annual Conference (APFA)*, 2001.

[5] K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 519–528, New York, NY, USA, 2003. ACM.

[6] M. Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[7] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli. User Profiles for Personalized Information Access. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 54–89, 2007.

[8] H. Halpin, V. Robu, and H. Shepherd. The Complex Dynamics of Collaborative Tagging. In *Proceedings of the 16th International Conference on World Wide Web*, pages 211–220, Banff, Alberta, Canada, 2007.

[9] M. Heckner, T. Neubauer, and C. Wolff. Tree, funny, to_read, google: What are Tags Supposed to Achieve? A Comparative Analysis of User Keywords for Different Digital Resource Types. In *Proceeding of the 2008 ACM Workshop on Search in Social Media*, pages 3–10, Napa Valley, California, USA, 2008.

[10] D. Kelly and J. Teevan. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*, 37(2):18–28, Sept. 2003.

[11] A. Kennedy and D. Inkpen. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22(2):110–125, 2006.

[12] M. G. Manzato. *Uma arquitetura de personalização de conteúdo baseada em anotações do usuário*. PhD thesis, Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo, 2011.

[13] B. Ohana and B. Tierney. Sentiment classification of reviews using SentiWordNet. In *9th. IT & T Conference, Dublin Institute of Technology*, Dublin, Ireland, 2009.

[14] T. O'Reilly. What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *Communications & Strategies*, 1:17, 2007.

[15] B. Pang and L. Lee. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[16] B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124, 2005.

[17] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 79–86, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[18] F. Salvetti, S. Lewis, and C. Reichenbach. Automatic opinion polarity classification of movie reviews. *Colorado Research in Linguistics*, 17(1), 2004.

[19] K. Toutanova and C. D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, pages 63–70, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[20] K. H. L. Tso-Sutter, L. B. Marinho, and L. Schmidt-Thieme. Tag-aware Recommender Systems by Fusion of Collaborative Filtering Algorithms. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, pages 1995–1999, Fortaleza, CE, Brazil, 2008.

[21] P. D. Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 417–424, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[22] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[23] K. Yang, N. Yu, and H. Zhang. Widit in trec 2007 blog track: Combining lexicon-based methods to detect opinionated blogs. In E. M. Voorhees and L. P. Buckland, editors, *TREC*, volume Special Publication 500-274. National Institute of Standards and Technology (NIST), 2007.

Avaliando Técnicas de Cache para a Distribuição de Vídeo sob-Demanda na Internet

Marco Schulze, Josilene Aires Moreira
Universidade Federal da Paraíba
Cidade Universitária - João Pessoa - PB - Brasil - CEP: 58051-900
+55 (83) 3216-7093
marco.shulze@di.ufpb.br, josilene@ci.ufpb.br

RESUMO
A distribuição de vídeo na Internet tem crescido significativamente nos últimos anos. Porém os objetos de vídeos são bastante grandes e provocam congestionamentos, atrasos e alto consumo de banda na rede. As técnicas de cache têm sido propostas para minimizar estes impactos, incluindo abordagens de segmentação e de substituição de objetos. Este trabalho modifica técnicas já estabelecidas na literatura, melhorando o desempenho medido em acerto em bytes em até 65% para caches pequenas, enquanto também reduz o percentual de objetos com atraso inicial.

Categorias e Descritores
C.4 [**Measurement Techniques**]

Termos Gerais
Gerenciamento, Medição, Performance.

Palavras-chave.
Gerenciamento de Cache, Vídeo sob-Demanda, Distribuição de Vídeo na Internet

1. INTRODUÇÃO
O uso das aplicações de vídeo na Internet tem aumentado de modo muito significativo nos últimos anos. Particularmente, as aplicações de Vídeo sob-Demanda (VoD) têm apresentado um intenso crescimento: este tipo de aplicação foi responsável por 40% do tráfego na Internet em 2010 e, de acordo com recentes projeções, será responsável por 57% de todo o tráfego na Internet em 2014 [1]. Diversos autores reportam o fenômeno da criação e publicação na Web de vídeos criados pelos próprios usuários (*User-Generated Content - UGC*) como uma das principais causas deste crescimento [3][12]0.

YouTube [2] é o mais bem sucedido serviço de hospedagem e compartilhamento de vídeos UGC desde que foi criado em 2005, com 65.000 vídeos publicados diariamente e com um acervo de mais de 100 milhões de objetos. De modo semelhante, o serviço Daum UCC, o mais popular UGC na Coréia, serve dois milhões de visitantes e atende mais de 35.000 visualizações por semana, sendo conhecido por fornecer vídeos de alta qualidade, transmitidos a uma taxa de 800 Kb/s [3].

Entretanto, a distância dos servidores remotos para os clientes ainda é um obstáculo para a transmissão dos vídeos, que em geral são grandes e ocupam muita banda até chegar ao usuário final. Com a finalidade de reduzir o consumo de banda e também de oferecer uma maior qualidade, diversas técnicas têm sido propostas, podendo-se destacar o uso de Caches. As caches são servidores que replicam o conteúdo, disponibilizando-o em locais mais próximos dos usuários, e consequentemente reduzindo o caminho que os dados precisam percorrer até serem disponibilizados nos clientes [4][5].

As técnicas de cache têm sido amplamente utilizadas no armazenamento de páginas Web tradicionais; no entanto, como os objetos de vídeo têm características bem específicas, é necessário que abordagens mais apropriadas sejam desenvolvidas de modo a prover qualidade adequada e reduzir o consumo de banda. Os algoritmos de cache têm o objetivo de gerenciar o espaço reduzido reservado ao armazenamento dos objetos de modo a manter aqueles que estão sendo mais acessados e assim aumentar a taxa de acertos, consequentemente diminuindo o tráfego na rede e fornecendo melhor qualidade aos usuários finais [6].

Entretanto, esta não é uma tarefa fácil. Os algoritmos de cache trabalham com fatores como popularidade, número total de acessos e probabilidade de acessos futuros, entre outros, a fim de prover o melhor gerenciamento possível do espaço da cache. As estratégias procuram trabalhar com duas métricas principais, o acerto em bytes (*byte hit ratio*), que representa a economia de consumo de banda, e a redução do atraso inicial (*delayed start*). Um dos grandes desafios da área é proporcionar o equilíbrio das duas métricas, pois em geral não é possível melhorá-las ao mesmo tempo [7].

Este trabalho analisa variações e combinações de técnicas de cache bem conhecidas na literatura, utilizando traces reais de objetos do YouTube. Para alguns tamanhos de cache, verifica-se que é possível melhorar o acerto em bytes em 65%, modificando a estratégia de segmentação de uma técnica tradicional a fim de liberar espaços maiores na cache a cada objeto retirado. A pesquisa encontra-se em andamento e tem o objetivo maior de propor uma nova técnica que contribua de forma balanceada para aumentar a taxa de acertos em bytes e, concomitantemente, reduzir o atraso inicial.

2. TRABALHOS RELACIONADOS

As estratégias de cache para vídeo são em grande parte baseadas em um trabalho seminal onde Townsley et al. [8] propõem a técnica de particionar os objetos em duas partes, prefixo e restante do objeto, conservando na cache os prefixos dos objetos mais populares (*Prefix Caching*) Na próxima requisição para o mesmo objeto, a parte inicial é transmitida ao usuário enquanto o restante do vídeo é obtido do servidor principal. Grande parte das técnicas usadas até hoje estendem ou modificam a abordagem de prefixo.

Miao e Ortega [9] discutem as vantagens de armazenar outras partes importantes dos objetos, além dos segmentos iniciais, se houver espaço disponível na cache. A seleção dos segmentos considera o tamanho do buffer do usuário e as propriedades dos objetos de vídeo. Na abordagem conhecida por *Exponential Caching ou Pyramid* [10] os autores reservam uma área inicial da cache para os segmentos iniciais dos objetos e aplicam uma segmentação variável ao restante do vídeo. O tamanho do segmento aumenta exponencialmente, e o i-ésimo segmento tem o tamanho de 2^{i-1}. Esta técnica diminui o atraso inicial.

Uma importante abordagem conhecida como *Lazy* foi proposta por Chen et al. [11]. Neste método, todos os objetos são inseridos completamente no cache na primeira requisição, e a decisão sobre o tamanho dos segmentos é adiada para o momento em que um objeto precisa ser removido da cache. *Lazy* utiliza uma função para atribuir importância aos objetos, a qual tenta prever os acessos futuros aos objetos baseada na freqüência e duração dos acessos já realizados. Alcança um bom desempenho em termos de acertos em bytes (*byte hit*) e consequente diminuição do consumo de banda, e é uma das abordagens consideradas importantes para gerenciamento de caches. A função utilizada pelo algoritmo Lazy está descrita na Equação 1.

$$CP = \frac{n}{(T_c - T_r)} * MIN\left\{1, \frac{\frac{T_r - T_1}{n}}{T_c - T_r}\right\}$$

Equação 1 – Função de prioridade da abordagem Lazy

n = Número de acessos do objeto

T_c = Timestamp atual

T_1 = Timestamp do primeiro acesso ao objeto

T_r = Timestamp do último acesso ao objeto

$\frac{n}{(T_c - T_r)}$ = Número médio de acessos no intervalo de tempo

$MIN\left\{1, \frac{\frac{T_r - T_1}{n}}{T_c - T_r}\right\}$ = Probabilidade de acessos futuros

3. METODOLOGIA

Os dados reais usados nas simulações foram obtidos a partir de uma coleta de dados de acessos ao serviço YouTube realizada por 0, em um roteador localizado entre uma universidade e a Internet no período de Junho de 2007 a Maio de 2008. Os traces contém registros das requisições dos clientes e dos objetos de vídeo solicitados.

A fim de corroborar nossos resultados, utilizamos dois traces, o Trace A com um menor número de requisições (18.495) e o trace B com um número bem maior de requisições (60.318). O conjunto de objetos requisitados em cada trace também varia. A Tabela 1 mostra as características dos traces.

Tabela 1. Traces do YouTube usados nos experimentos

Dados YouTube	Trace A 012908.24	Trace B 031208
Número de requisições	18495	60318
Número de vídeos	13325	37245

Para analisar a efetividade dos algoritmos, foi escrito um simulador em Python. O simulador registra, a cada requisição, a quantidade de objetos atualmente no cache, e a quantidade total de espaço utilizada para guardar esses objetos, além de atributos dos objetos como a frequencia e o timestamp das requisições. A Figura 1 mostra a configuração utilizada nas simulações.

Considera-se que o servidor de cache (*Proxy cache*) encontra-se na mesma rede local que os clientes e, dessa forma, o acesso do clientes à cache é rápido e sujeito a poucas variações. Dessa

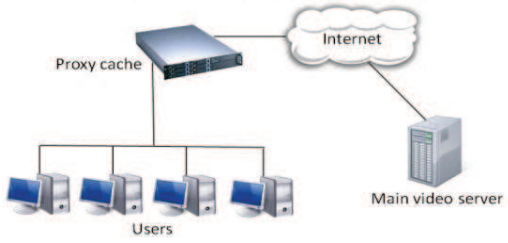


Figura 1 - Estrutura do simulador

forma, o atraso deste acesso local é desprezado.

Nas simulações considera-se que o servidor principal tem capacidade de atender a todas as requisições recebidas. Por outro lado, entre o servidor de cache e o servidor principal (*Main video server*) considera-se que existe uma latência considerável que pode gerar atrasos na transmissão, provocando perda na qualidade percebida pelos usuários. Assim, quando uma requisição chega ao servidor de cache e o vídeo não encontra-se armazenado ainda, contabiliza-se uma ocorrência de atraso inicial, pois será necessário que o usuário espere enquanto o conteúdo é buscado pela cache e armazenado no servidor principal.

Utilizam-se duas métricas principais para medir a eficiência das estratégias de cache, que são a taxa de acertos em bytes (*byte hit ratio*) e o atraso inicial (*start up delay*). A taxa de acertos em bytes é a razão da quantidade de bytes requisitada pela quantidade encontrada na cache. O atraso inicial contabiliza a quantidade de objetos que não foram encontrados na cache. Enquanto a taxa de acerto em bytes deve ser maximizada, consequentemente reduzindo o consumo de banda na rede, o atraso inicial precisa ser minimizado, melhorando a qualidade de experiência do usuário (QoE). Em geral, as técnicas de cache ou diminuem o consumo de banda ou melhoram a experiência do usuário, mas dificilmente atingem os dois objetivos simultaneamente.

As abordagens de cache são basicamente divididas em (a) estratégia de segmentação e (b) estratégia de substituição dos objetos. Nossos experimentos combinam as estratégias de (a) e (b) de diferentes abordagens visando obter melhores resultados. As estratégias de cache avaliadas nos nossos experimentos incluem

modificações e combinações das técnicas Lazy e Exponencial, descritas na seção 2.

Os experimentos medem a eficiência da técnica em função do tamanho da cache. O tamanho da cache é uma variação do tamanho total dos objetos presentes no trace, isto é, o tamanho total dos objetos que poderiam ser requisitados. Esta técnica de avaliação é frequentemente utilizada em experimentos de cache [7][8][9][10].

A técnica Lazy é bem sucedida em reduzir o consumo de banda, enquanto que a abordagem Exponencial reduz a quantidade de vídeos com atraso inicial Por questões de espaço, mostramos apenas as avaliações com a técnica Lazy, nas quais obtivemos os melhores resultados até o momento.

4. RESULTADOS PRELIMINARES

4.1 Avaliação do Acerto em Bytes

A Figura 2 mostra a taxa de acerto em bytes para o trace A. É apresentado o desempenho da estratégia de segmentação Lazy combinada com duas abordagens para a substituição de objetos na cache, a própria Lazy e a variação LRU.

A abordagem Lazy utiliza a fórmula mostrada em (1) para atribuir importância aos objetos que estão no cache, e os objetos menos importantes são excluídos primeiro. A abordagem LRU retira do cache os objetos referenciados há mais tempo, conservando na cache aqueles objetos requisitados mais recentemente.

É possível perceber que, para caches pequenas (até 10% do tamanho total dos objetos) a abordagem que utiliza LRU consegue melhorar o acerto em bytes. Quando a cache é bem pequena (2,5%), verifica-se que a segmentação Lazy combinada com a substituição LRU é capaz de melhorar o acerto em bytes em 36%. Isto representa uma grande economia de banda quando o volume de tráfego é alto, como no caso do tráfego de vídeos de serviços como o YouTube.

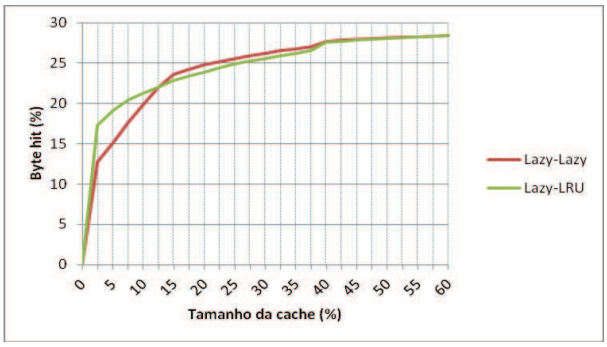


Figura 2 - Acerto em bytes trace A

Outro importante aspecto a destacar é a questão do melhor desempenho quando a cache é pequena. A variação do tamanho da cache nos experimentos é feita em relação ao tamanho total de objetos únicos do trace (metodologia adotada em inúmeros estudos de cache para vídeo). Sabe-se que a quantidade de objetos únicos nos traces do YouTube é muito grande, e que apenas cerca de 25% destes objetos são requisitados mais de uma vez. Dessa forma, uma cache pequena corresponde na maioria das vezes à realidade do tamanho das caches usadas na prática, já que é impossível variar o tamanho da cache em função do número de objetos de um serviço do YouTube. O dimensionamento inicial pode até ser feito com base no número de objetos, mas o crescimento da cache em geral não acompanha o crescimento do número de vídeos disponíveis no sistema. Assim, obter bons resultados para caches pequenas torna-se importante.

A Figura 3 mostra a mesma avaliação usando um trace com um número muito maior de requisições, o trace B (60.000). Fica claro que, quando o número de requisições aumenta, a diferença de desempenho entre as técnicas se torna maior. Para uma cache de tamanho 2.5% a técnica Lazy combinada com a LRU é capaz de obter um desempenho 65% melhor do que a técnica Lazy original.

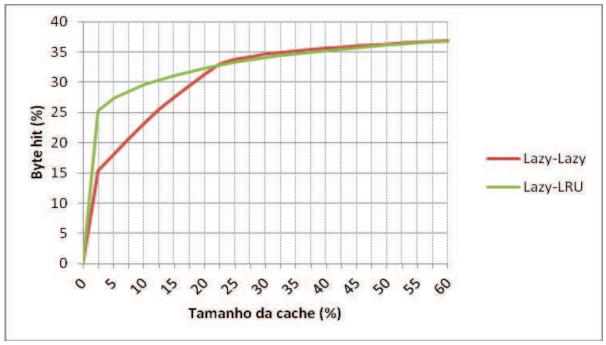


Figura 3 - Acerto em bytes trace B

4.2 Avaliação do Atraso Inicial

Como relatado em pesquisas anteriores, a técnica Lazy consegue obter um ótimo desempenho para o acerto em bytes; entretanto, esta técnica não é bem sucedida na redução do atraso inicial. Os nossos experimentos mostraram que a variação que combina a estratégia de segmentação Lazy combinada com a estratégia de substituição de objetos LRU é capaz de reduzir o percentual de objetos com atraso inicial para caches pequenas.

A Figura 4 mostra o desempenho medido pelo atraso inicial para as mesmas abordagens da seção anterior, para o trace A. Ressalta-se que o Eixo Y representa o percentual de vídeos com atraso inicial (aqueles que não foram encontrados na cache no momento da requisição), enquanto o Eixo X mostra a variação do tamanho da cache (Figuras 4 e 5).

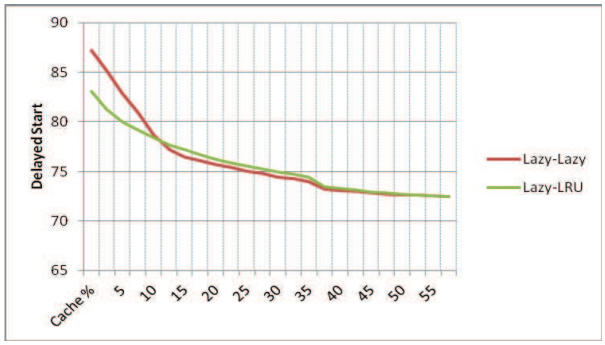


Figura 4 - Atraso inicial trace A

Observa-se que para tamanhos de cache de 2,5% e de 5%, a técnica combinada melhora o desempenho da técnica Lazy em 5%. Considerando-se que esta técnica é bastante complexa e foi muito trabalhada pelos autores, estimamos que este é um bom resultado, significando que um número menor de requisições irão sofrer atraso inicial.

A Figura 5 mostra o resultado da análise do atraso inicial para o trace B. É possível perceber que a técnica combinada Lazy-LRU é capaz de melhorar o desempenho da técnica Lazy para tamanhos

menores de cache. Verifica-se que para um número de requisições maior a diferença de desempenho entre as técnicas também é maior. Para um tamanho de cache de 2.5% a técnica combinada consegue obter um desempenho 9% melhor do que a técnica Lazy.

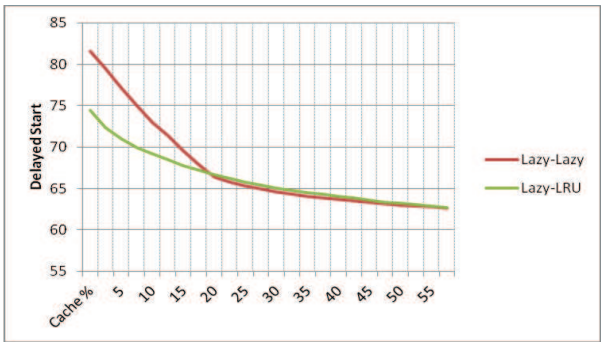


Figura 5 - Atraso inicial trace B

5. CONCLUSÕES

Uma dos principais objetivos do uso de caches é a redução do consumo de banda e do congestionamento na rede, através da colocação de servidores com réplicas de conteúdo em localizações estratégicas da rede. Os algoritmos de cache são extremamente importantes e estão diretamente relacionados com o desempenho das caches tanto em termos de consumo de banda como na redução dos atrasos.

O cenário deste trabalho é a utilização de um servidor de cache (*proxy cache*) na rede local a fim de interceptar as requisições de vídeo para serviços de distribuição de vídeo na Internet como o YouTube. Proxy caches já são utilizados na Internet; entretanto, ainda existe um número reduzido de análises específicas para a distribuição de vídeo.

Este trabalho analisa variações de técnicas de cache conhecidas que apresentam bom desempenho, a fim de melhorar ainda mais os seus resultados. A nossa principal contribuição é conseguir alterar a técnica Lazy, tradicionalmente com ótimo desempenho para o aumento do acerto em bytes, tornando-a ainda melhor para caches pequenas. Além de aumentar o acerto em bytes e, consequentemente, diminuir o consumo de banda, a variação que utiliza LRU como estratégia de substituição na cache é capaz também de reduzir o percentual de vídeos com atraso inicial, proporcionando uma melhoria da qualidade de experiência dos usuários (QoE).

O algoritmo modificado apresenta melhor desempenho para caches menores. Este achado é importante, visto que, na prática, torna-se difícil, redimensionar constantemente a cache em relação ao número de objetos disponível no sistema. Em geral, as caches permanecem com um tamanho fixo por um determinado período de tempo e, quando comparadas ao número de objetos disponíveis no YouTube, são quase sempre pequenas. Sendo assim, concluímos que a técnica pode contribuir significativamente para diminuir o consumo de banda na rede, em serviços como o YouTube, além de reduzir o atraso inicial.

Uma série de estudos estão sendo conduzidos como continuação deste trabalho. Ressalta-se os experimentos com traces sintéticos modelados para avaliar diversas condições de tráfego, nos quais se variam importantes parâmetros da carga de dados como a distribuição de popularidade, o tamanho dos objetos, a quantidade de requisições, a duração das sessões de usuários e o percentual de objetos *cacheable*. Além disso, pretende-se realizar experimentos com tamanho absoluto de cache, a fim de compreender melhor o comportamento dos algoritmos.

6. AGRADECIMENTOS

Nossos agradecimentos ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), por financiar este trabalho através de bolsa de iniciação científica (PIBIC).

7. REFERÊNCIAS

[1] CISCO, “Cisco Visual Networking Index: Forecast and Methodology, 2009-2014”. 2010, June. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf. Acessado em Novembro, 2010.

[2] YouTube. http://youtube.com/ Acessado in January, 2012.

[3] M. Cha , H. Kwak , P Rodriguez , Yong-Yeol Ahn , Sue Moon, “I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System”, Proc. of the 7th IMC 2007, San Diego, California, USA, October 2007.

[4] Hofmann, M., and Beaumont, L. R., “Content Networking: Architecture, Protocols, and Practice”. Morgan Kaufmann Publishers, San Francisco, CA, USA, 2005.

[5] Pathan, A. K., Buya, R., “A Taxonomy of CDNs", Content Delivery Networks, R. Buyya, M. Pathan, and A. Vakali (Eds.), Springer-Verlag, Germany, 2008.

[6] Jayarekha, P., P. C. Air, T. R. Gopalakrishnan, “A Rank Based Replacement Policy for Multimedia Server Cache Using Zipf-Like Law”, Journal of Computing Vol 2, Issue 3, March 2010.

[7] Romano, Sam, ElAarag, H. "Comparison of function based web proxy cache replacement strategies". In Proc. of 12th Intern. Conf. on Performance Eval. of Computer & Telecom. Systems (SPECTS'09), IEEE Press, NJ, USA, 2009.

[8] Sen, S., Rexford, J., and Towsley, D. "Proxy prefix caching for multimedia streams". In Proceedings of IEEE INFOCOM'99. IEEE Computer Society, 1999.

[9] Miao, Z. and Ortega, A. 1999. "Proxy Caching for Efficient Video Services over the Internet". In Proceedings of the 9th International Packet Video Workshop (PVW'99).

[10] Wu, K., Yu, P. S., and Wolf, J. L. “Segment-based proxy caching of multimedia streams”. In Proc. of the 10th Intern. Conference on WWW, Hong Kong, 2001.

[11] Chen, S., Wang, H., Zhang, X., Shen, B., and Wee, S., “Segment-Based Proxy Caching for Internet Streaming Media Delivery”. IEEE MultiMedia 12, July 2005.

[12] Zhou, R., Khemmarat, S., Gao,L., "The Impact of YouTube Recommendation System on Video Views", in Proc. of Internet Measurement Conference (IMC), 2010.

Zink, M., Suh, K., Gu, Y. and Kurose, J., "Watch Global Cache Local: YouTube Network Traces at a Campus Network - Measurements and Implications", IEEE MMCN, 2008.

Algoritmos de seleção de web services baseada em técnicas de mineração de dados aplicada em arquiteturas orientadas a serviços

Lucas Junqueira Adami
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
lucasadami@grad.icmc.usp.br

Julio Cezar Estrella
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
jcezar@icmc.usp.br

RESUMO

O objetivo deste artigo é apresentar os resultados oriundos de experimentos nos quais são utilizadas técnicas de mineração de dados, especificamente algoritmos de aprendizado de máquina, para realizar a seleção de serviços em arquiteturas orientadas a serviço. Tais técnicas são componentes de um módulo acoplado a um *Broker* de uma arquitetura denominada *WSARCH*. Para a mineração dos dados são considerados informações de acessos de clientes da arquitetura provenientes de um *LogServer*. O resultado do treinamento no *LogServer* é usado para classificação de futuras requisições selecionando provedores aptos a atender os diferentes clientes. Esta forma de classificar provedores de serviços pode ser útil em caso de falhas em registros de serviços como o *UDDI*. A análise dos resultados após as fases de planejamento de experimentos e análise de desempenho mostram a eficiência das técnicas de classificação acopladas à arquitetura.

1. INTRODUÇÃO

A interoperabilidade entre os sistemas computacionais pode, nos dias atuais, ser alcançada por meio do paradigma orientado a serviços utilizando para essa finalidade os chamados *web services*. No entanto, para garantir que a interoperabilidade seja adequada, a qualidade do serviço envolvida neste processo precisa ser melhor explorada. Para estudar e avaliar essa questão, a *WSARCH* (*Web Services Architecture*) [3] foi desenvolvida. Trata-se de uma arquitetura para prover *web services* com qualidade de serviço. A arquitetura *WSARCH* é uma arquitetura cuja meta é ser uma referência para tratar de questões de desempenho na integração de aplicações distribuídas utilizando o paradigma orientado a serviços. O propósito da arquitetura é sempre buscar o melhor desempenho possível na integração de aplicações presentes em empresas e órgãos governamentais. A arquitetura é separada em 5 módulos: aplicação cliente, provedores, *Broker*, *UDDI* e o *LogServer*. A interação entre eles é mostrada na Figura 1.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WEBMEDIA '12 São Paulo, São Paulo Brazil
Copyright 2012 ACM 27/07/12 ...$15.00.

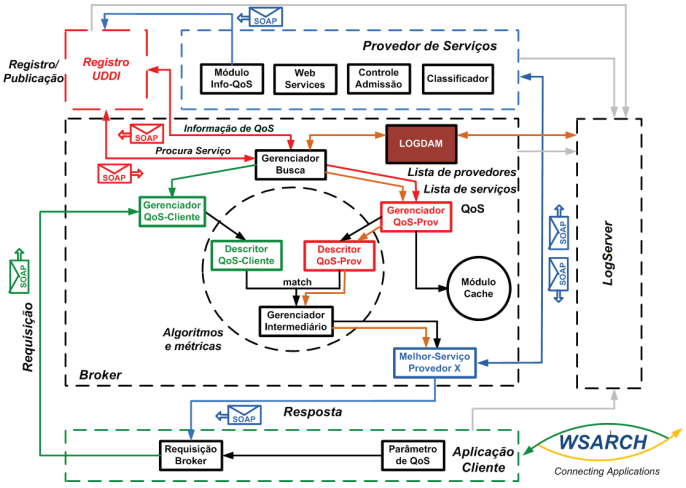


Figura 1: WSARCH - Modelo de interação [3]

Os clientes fazem requisições diretamente ao *Broker* da arquitetura. Eles especificam alguns parâmetros no envio da mensagem de requisição para que possam ser atendidos corretamente e receberem respostas adequadas. Os clientes podem ser desenvolvidos em qualquer linguagem de programação que se encaixe no padrão dos *web services*. Os provedores contém as aplicações que os clientes necessitam. Eles são servidores de aplicações que lidam com as requisições e respostas. Um provedor pode conter várias aplicações e cada aplicação pode apresentar várias operações. Assim como os clientes, os provedores de serviços podem ser codificados em qualquer linguagem de programação que esteja em conformidade com o padrão dos *web services*. O *Broker* é o elemento central da arquitetura. Ele é responsável pelo escalonamento das mensagens para os provedores de acordo com as exigências dos clientes. O *UDDI* armazena informações funcionais e não-funcionais dos serviços presentes nos provedores, tais como localização e especificações técnicas. O *Broker* da arquitetura faz requisições ao *UDDI* para selecionar o melhor provedor disponível para seus clientes. O *LogServer* armazena os registros de testes de desempenho realizados na arquitetura. Dentre esses registros, constam as informações de carga do *Broker* e do *UDDI* e as informações de acesso durante a requisição de um cliente. Os *logs* ficam armazenados em uma base de dados relacional do

LogServer. Esses dados contêm informações como horário do acesso, tempo de execução e endereço *IP* do provedor que atendeu a requisição [3].

Uma análise mais criteriosa e seletiva desses dados pode levantar questões quanto aos pontos críticos do sistema e servir de histórico para que decisões futuras possam ser tomadas com base no passado, permitindo realizar diversas abordagens. Nesse sentido, é proposto um módulo de análise de dados, o qual foi inicialmente descrito em [1], composto de algoritmos de seleção de *web services* que utilizam técnicas de aprendizado de máquina. Ele engloba um novo seletor na arquitetura *WSARCH* que utiliza dados locais para decidir qual provedor atenderá o cliente. O módulo entra em ação quando o seletor tradicional do sistema, que utiliza o *UDDI* para obter a lista de provedores disponíveis, não consegue buscar a informação de onde a requisição do cliente deve ser escalonada.

2. TRABALHOS RELACIONADOS

Guo et al. [4] definem um modelo de redes *Petri - WS_QPN* para processar os valores de *QoS* em composição de serviços. Uma vez encontrada que a solução para seleção de serviços com qualidade dirigida é NP-complexo, estratégias baseadas em algoritmos genéticos são mais adequados do que técnicas tradicionais de otimização. Foi usado um algoritmo evolutivo multiobjetivo, chamado *NSGA-II* para resolver este problema e avaliar o esquema de codificação e o objetivo das funções obtidas do *WS_QPN*. Shi [8] discute um modelo de *WSDL* estendido com o intuito de aumentar o detalhamento de *QoS*. Além disso, são propostos serviços com algoritmos baseados nesse modelo, e o algoritmo de seleção de *QoS* é baseado no modelo de programação inteira não-linear. O trabalho propõe o algoritmo de serviço correspondente com base neste modelo e do algoritmo de seleção *QoS* com base em NLIP. Depois de encontrar o solução do modelo avançado, o documento tem recebido as sequências de plano de serviços Web que cumpram as tarefas complexas de requisitos do usuário. Shen et al. [9] formalizam o problema de seleção *web services* como um processo de máquina de estados finitos. Propõem um algoritmo *backward* para criar a árvore de composição de *web services* (*WSCT - Web Services Composition Tree*), bem como um algoritmo heurístico para completar a seleção do *web service* com base no *WSCT*. Os resultados do experimento mostram que o algoritmo heurístico proposto é eficaz. Uma abordagem personalizada para prever *QoS* com base em filtragem colaborativa, para melhorar a eficácia é proposta em [7]. A ideia básica da abordagem é descobrir uma semelhança entre os consumidores, com os dados coletados de *QoS* e, em seguida, fazer a previsão para os serviços não utilizados, com base na similaridade. Os resultados experimentais mostram que a abordagem pode melhorar significativamente a eficácia da previsão de *QoS* em *web services*. Quando não há dados suficientes disponíveis para supor a semelhança, a abordagem ainda apresenta um bom potencial para prever a qualidade de serviço. Para selecionar um *web service* funcionalmente similar, os critérios não funcionais, tais como os de *QoS* são considerados em [5]. De acordo com o estudo, a maioria dos clientes não são experientes o suficiente para adquirir a melhor seleção de *web service* baseado em suas propriedades de *QoS*. Assim, propõem uma arquitetura baseada no *QoS Broker* o qual utiliza um mecanismo eficiente para encontrar o melhor *web service*.

3. MÓDULO DE ANÁLISE DE DADOS

Como é mostrado na Figura 2, o *Broker* é o responsável por receber requisições dos clientes e usar o seletor inteligente para escolher um provedor de serviços apto a atender o pedido do cliente. Originalmente, a arquitetura possui apenas um seletor, chamado de seletor tradicional. Esse seletor consulta o *UDDI* para obter a lista de provedores aptos a atender a requisição e utiliza a distância euclideana para escolher o melhor provedor para o atendimento. O seletor de serviços inteligente é um novo seletor do *Broker* da arquitetura. Ele utiliza o treinamento para realizar uma classificação usando alguns parâmetros obtidos do *LogServer*, retornando o *IP* do provedor que atenderá o cliente.

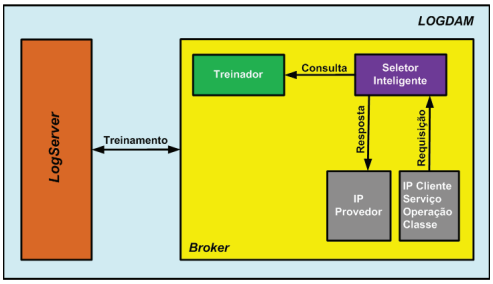


Figura 2: Módulo inteligente [1]

3.1 Desenvolvimento do treinador

Para realizar os treinamentos usando a base de dados, o treinador utiliza algoritmos oferecidos pelo *WEKA* [10]. O treinador é uma *thread* que executa junto ao *Broker*, ao qual são passados quatro parâmetros. O primeiro é o algoritmo que será responsável pela mineração. Há dois disponíveis: *Naïve Bayes* [2] e *IBk* [6]), implementados no *WEKA*. O segundo parâmetro é o número máximo de amostras recuperadas do banco de dados usadas no treinamento. Esse limite foi criado porque o número de acessos ao *Broker* pode ser muito grande e o treinamento para enormes quantidades de dados pode ficar inviável e lento. O terceiro parâmetro, a idade máxima das instâncias, permite ao treinador ignorar registros antigos que estejam presentes no *LogServer*. Registros antigos não são interessantes porque não refletem o comportamento atual da arquitetura. Caso o treinador utilize-os, o comportamento do seletor inteligente pode ficar inadequado e não obter um resultado adequado. O último parâmetro é o intervalo de treinamento. Não é necessário realizar um novo treinamento a todo instante. Por isso, essa variável foi criada para delimitar uma faixa de tempo em que o treinador fica inativo. Após a inicialização do treinador, sua *thread* treina um conjunto de dados que é recuperado na base de dados de acordo com os parâmetros passados. Os atributos utilizados para o treinamento são listados na Tabela 1.

Tabela 1: Atributos utilizados no treinamento

Atributos	
ServiceName	Nome do serviço
Operation	Operação associada ao serviço
ClientIP	IP do cliente que efetuou a requisição
Class	Classe do cliente (*gold*, *silver* ou *bronze*)
Provider	IP do provedor que atendeu a requisição

A classe a ser classificada de acordo com os atributos da Tabela 1 é o *Provider* e todos estão presentes em uma tabela

do *LogServer* denominada *AccessLog*. Para entender melhor como funciona o seletor inteligente, a Figura 3 mostra a interação entre as classes que compõem o módulo.

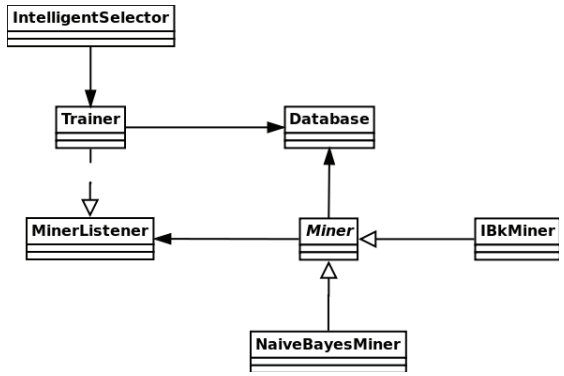


Figura 3: Diagrama de classes do módulo

A classe *Miner* é uma classe abstrata e pai de todas que utilizam os algoritmos de aprendizado de máquina. Ela é responsável por recuperar as instâncias para treinamento do *LogServer* e chamar o algoritmo de aprendizado implementado nas classes filhas *IBkMiner* e *NaiveBayesMiner*. Através da interface *MinerListener*, ela informa ao ouvinte o término do treinamento. A classe *Trainer* é o treinador do módulo. Ela é encarregada de administrar a conexão com o banco de dados através da classe *Database* e passá-la à classe mineradora. O treinador também carrega a classe mineradora especificada e chama-a periodicamente, sendo sua ouvinte. Além disso, a classe *Trainer* classifica novas requisições, pedido determinado pela classe *IntelligentSelector*.

4. PLANEJAMENTO DE EXPERIMENTOS

Para a execução dos testes dos algoritmos implementados, as seguintes configurações para o *UDDI*, *Broker* e provedores foram utilizadas de acordo com a Tabela 2.

Tabela 2: Infraestrutura

UDDI	
Processador	4 núcleos - Intel Core 2 Quad 8400
Memória	4GB - DDR3
HD	HD Virtual de 50GB
SO	Linux Ubuntu 11.10 Server
Qtde	1
Provedor	
Processador	1 núcleo - Intel Core 2 Quad 8400
Memória	6 com (512MB) e 6 com (1GB) - DDR3
HD	HD Virtual - 50GB
SO	Linux Ubuntu 11.10 Server
Qtde	12
Broker	
Processador	6 núcleos - AMD Phenom II X6 1090T
Memória	16GB - DDR3
HD	500GB
SO	Linux Ubuntu 11.10 Server
Qdte	1

Para avaliar os algoritmos usando o seletor inteligente e o seletor tradicional, dois cenários de avaliação de desempenho foram considerados conforme a Tabela 3.

No total, foram 1600 requisições para o primeiro cenário (16 clientes com 10 processos cada, repetindo 10 vezes) e 3200 requisições para o segundo (16 clientes com 10 processos cada, repetindo 20 vezes). Na primeira metade das requisições, o seletor tradicional com o algoritmo de distância euclidiana foi utilizado. Na segunda metade, o *Broker* utilizou o seletor inteligente com o algoritmo de classificação *Naïve Bayes* ou *IBk*.

Tabela 3: Cenários de avaliação de desempenho

Cenário 1	
Processos	10
Repetições	10
Qtde e tipo de cliente	8 bronze e 8 gold
Intervalo entre requisições	3 segundos
Cenário 2	
Processos	10
Repetições	20
Qtde e tipo de cliente	8 bronze e 8 gold
Intervalo entre requisições	3 segundos

5. RESULTADOS

Na avaliação dos resultados, três medidas foram usadas para analisar o desempenho do módulo e dos algoritmos de seleção: tempo de busca, tempo de serviço e tempo de requisição. O tempo de busca indica o intervalo que o seletor da requisição leva para conseguir uma lista de provedores capaz de fornecer o serviço adequado ao cliente. O tempo de serviço representa o intervalo da execução do serviço em uma requisição. O tempo de requisição indica o intervalo total da requisição. Ele é registrado no *LogServer* por meio da aplicação cliente.

Na Figura 4, cada segmento representa uma faixa de valores que contém os tempos de busca obtidos em cada item do experimento. Um item caracteriza-se pelo número de repetições (10 ou 20), o tipo do seletor (*IBk* ou tradicional) e o tipo do cliente (*gold* ou *bronze*). Observando a Figura, o tempo de busca para o seletor utilizando o algoritmo *IBk* é sempre próximo de zero, com uma faixa de valores pequena. Para o seletor tradicional, esse tempo varia, dependendo da carga imposta ao sistema. Essa diferença ocorre porque o seletor inteligente utiliza informações locais para classificar uma nova requisição enquanto o seletor tradicional busca no *UDDI* a lista de provedores aptos a atender o cliente.

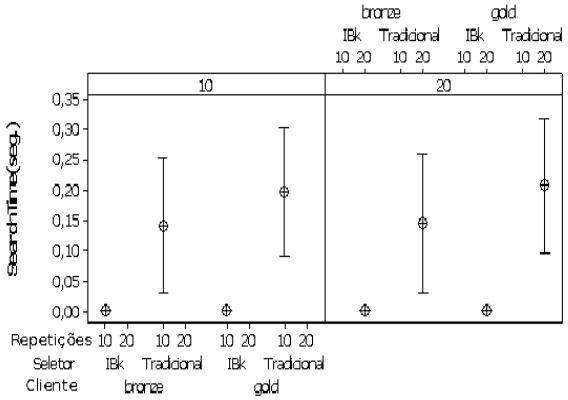


Figura 4: Intervalos de confiança do tempo de busca para o seletor *IBk* e o tradicional [1]

Na Figura 5, de modo similar à Figura 4, cada eixo vertical representa valores que representam os tempos de serviço do seletor inteligente utilizando o algoritmo *Naïve Bayes* e o *IBk* em cada item do experimento. O algoritmo *Naïve Bayes* é menos sensível à mudança da carga se comparado com o *IBk*. Esse comportamento é notado quando o tipo do cliente é *gold*.

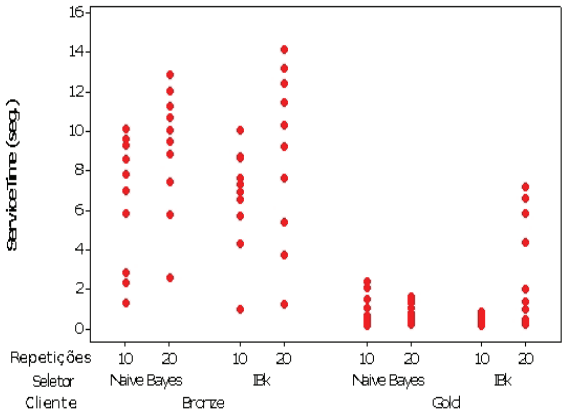


Figura 5: Concentração do tempo de serviço para os seletores *Naïve Bayes* e o *IBk*

Ao analisar a Figura 6 é possível observar que o algoritmo *Naïve Bayes* resulta em tempos que variam pouco na média quando a carga imposta ao sistema aumenta. Verifica-se também que os tempos para clientes *gold* e *bronze* são consistentes, uma vez que para o primeiro tipo, a duração da requisição é menor que para o segundo. Adicionalmente, o algoritmo observado possui um tempo de requisição menor na média se comparado com o seletor tradicional.

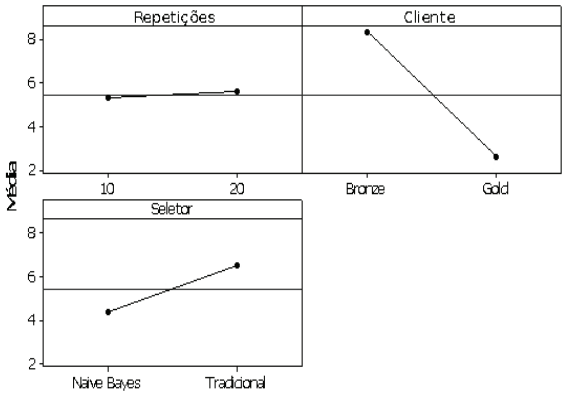


Figura 6: Efeitos sobre o tempo de requisição para o seletor *Naïve Bayes* [1]

6. CONCLUSÕES E TRABALHOS FUTUROS

Os resultados obtidos mostram que o objetivo de usar as informações do passado existente no *LogServer* para classificar novas requisições dos clientes por meio de aprendizado de máquina e atendendo aos padrões de qualidade de serviço foi alcançado. Como o método de seleção proposto não consulta o *UDDI* por provedores (o que é feito pelo seletor tradicional), mas se baseia na classificação do treinador, os resultados dos tempos de busca apresentados neste trabalho foram próximos de zero. Ainda, foi possível observar que o seletor inteligente foi melhor que o seletor tradicional para as configurações expostas. Se comparados os algoritmos de treinamento, ambos obtiveram resultados similares. O algoritmo *Naïve Bayes* foi menos sensível à mudança de carga nos testes. Adicionalmente, o trabalho mostra que é interessante pensar em um mecanismo onde o seletor tradicional possa consultar um *cache* local com informações provenientes do *UDDI* sem a necessidade de realizar um novo acesso a cada requisição de um cliente. Isso permitirá uma rápida consulta (tempo de busca), além de diminuir a carga no *UDDI*, melhorando assim o tempo de serviço e o tempo de resposta final. Como trabalhos futuros novos algoritmos, incluindo um mecanismo de inteligência artificial por meio de uma rede neural serão desenvolvidos, bem como um estudo comparativo entre as técnicas de mineração de dados e de reconhecimento de padrão/classificação.

7. REFERÊNCIAS

[1] L. J. Adami and J. C. Estrella. A data analyzer module for logs of service oriented architecture. In *III Escola Regional de Alto Desempenho - ERAD-SP*, jul. 2012.

[2] D. Aha and D. Kibler. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[3] J. C. Estrella, R. H. C. Santana, and M. J. Santana. *WSARCH: An Architecture for Web Services Provisioning with QoS Support - Performance Challenges*. Saarbrücken : VDM Verlag Dr. Müller GmbH & Co, 2011. http://www.amazon.com/WSARCH-Architecture-Provisioning-Performance-Challenges/dp/3639378245.

[4] F. Guo, L. Zhao, Y. Wang, and M. Zhang. Research on the web services selection problem. In *Education Technology and Computer Science (ETCS), 2010 Second International Workshop on*, volume 2, pages 284 –287, march 2010.

[5] P. Harshavardhanan, J. Akilandeswari, and R. Sarathkumar. Dynamic web services discovery and selection using qos-broker architecture. In *Computer Communication and Informatics (ICCCI), 2012 International Conference on*, pages 1 –5, jan. 2012.

[6] G. H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo, 1995. Morgan Kaufmann.

[7] H. Liu, F. Zhong, and B. OuYang. A web services selection approach based on personalized qos prediction. In *Parallel and Distributed Computing (ISPDC), 2011 10th International Symposium on*, pages 199 –206, july 2011.

[8] S. Na. The study of web services selection based on qos. In *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, volume 13, pages V13–109 –V13–112, oct. 2010.

[9] H. Shen, Z. Ding, and H. Chen. Reliable web services selection using a heuristic algorithm. In *Grid and Cooperative Computing (GCC), 2010 9th International Conference on*, pages 290 –295, nov. 2010.

[10] WEKA. Weka wiki. http://weka.wikispaces.com/Properties+file, 2012.

UbibusRoute: Usando Informações Contextuais de Redes Sociais para Sugestão de Rotas de Ônibus

Vanessa Gomes de Lima
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
vgl2@cin.ufpe.br

Ana Carolina Salgado
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
acs@cin.ufpe.br

RESUMO

Sistemas de Informação ao Usuário do transporte coletivo visam fornecer informações aos passageiros e apoiar suas decisões. A maioria dos sistemas com esse propósito utiliza informações estáticas ou auxiliadas por transmissores GPS instalados nos veículos. Os cidadãos podem contribuir com a melhoria dos serviços públicos e isto pode ser feito, por exemplo, com o uso das redes sociais. Este trabalho apresenta o UbibusRoute, um sistema móvel que considera informações contextuais dinâmicas do trânsito provenientes de redes sociais e apresenta informações sobre as rotas aos usuários, utilizando a API do Google Maps como fonte de geolocalização e a API do Twitter como provedora de informações dos cidadãos.

Palavras-chave

Sistemas Inteligentes de Transporte, Contexto, Aplicações Móveis, Redes Sociais.

1. INTRODUÇÃO

O cenário atual do trânsito nas grandes cidades brasileiras vem piorando a cada dia e os congestionamentos tornaram-se cada vez mais frequentes. Devido ao intenso tráfego, o horário em que os ônibus passam em seus respectivos pontos de parada pode se tornar imprevisível. Esse é um grande problema enfrentado pela população, pois de acordo com Cutolo [1], uma grande barreira para os passageiros é a ausência de informações relativas aos serviços e/ou sua baixa qualidade. Como opção de solução de alguns desses problemas surgiram os Sistemas Inteligentes de Transporte (SIT), que consistem na utilização da Tecnologia da Informação e Comunicação (TIC) para subsidiar a infraestrutura dos sistemas de transporte [2].

Como aplicações de SITs, existem os Sistemas Avançados de Transporte Público (SATP), que apoiam o aumento da eficiência e segurança dos sistemas de transporte público [3]. Dentro dos SATP, existem os Sistemas de Informações aos Usuários (SIU), que são as ferramentas de comunicação entre os gestores do transporte público e os usuários, fornecendo informações que satisfaçam necessidades específicas dos passageiros, tais como: horário de chegada dos ônibus nas paradas, tempo de espera e tempo de viagem.

Em paralelo, observa-se o aumento da popularidade dos dispositivos móveis com acesso à Internet, e o uso das redes sociais contribuindo para o surgimento de novos aplicativos que possam atender às diversas necessidades dos usuários. As redes sociais vêm, dessa forma, ajudando as pessoas a lidar com seus problemas diários, uma vez que facilitam a troca e o compartilhamento de informações dinâmicas, em tempo real, relacionados aos mais diversos aspectos do cotidiano, inclusive as situações que envolvem o trânsito. Segundo Levy [4], a área de *Crowdsourcing* (ou inteligência coletiva) incentiva a combinação do conhecimento individual provido por um grupo de pessoas para produzir novas informações ou ideias mais úteis.

Existem vários tipos de aplicativos baseados em localização que fornecem informações aos usuários sobre o trânsito (e.g. *Google Maps*[1] [5]). Esses aplicativos provêm, em sua maioria, informações sobre rotas e estimativa de tempo de chegada, mas não são totalmente direcionadas aos passageiros de ônibus e não levam em consideração acontecimentos dinâmicos como engarrafamentos, acidentes, alagamentos, entre outros. Estas informações dinâmicas são chamadas informações contextuais [6], que tornam a aplicação desenvolvida mais adaptativa ao usuário, satisfazendo suas preferências e necessidades.

O problema, nos casos descritos, é que os passageiros do transporte público não têm um serviço que os apoie na decisão sobre que ônibus e rota tomarem para chegar aos seus destinos, contornando situações de tráfego intenso ou acidentes. Em nosso projeto de pesquisa, denominado Ubibus [7], investigamos o uso de tecnologias relacionadas a dispositivos móveis, web e mídias sociais como apoio aos passageiros de transporte público em grandes e médios centros urbanos. Neste contexto, propomos o UbibusRoute [8], uma solução direcionada aos usuários de transporte coletivo por ônibus, que usa informações provenientes de redes sociais para recomendar rotas a esses usuários apoiando-os em suas tomadas de decisão. O sistema utiliza como fonte de informações contextuais dinâmicas a rede social *Twitter*[2] e a API de geolocalização *Google Maps*.

O restante desse artigo está organizado da seguinte forma: a Seção 2 descreve alguns trabalhos correlatos ao proposto; a Seção 3 apresenta o sistema UbibusRoute, sua arquitetura e questões técnicas de implementação; a Seção 4 detalha um cenário de uso experimental realizado com o intuito de avaliar o sistema proposto; a Seção 5 discute as contribuições e conclusões alcançadas com essa pesquisa e as perspectivas de trabalhos futuros.

2. TRABALHOS RELACIONADOS

Durante pesquisas encontramos alguns trabalhos que recomendam rotas aos usuários, como Mazhelis [5], mas não são voltados ao transporte público. Hoar [9] e Ferris [10] apresentam sistemas que exibem rotas de ônibus aos usuários, mas não processam dados dinâmicos de redes sociais. A utilização de tipo

[1] http://maps.google.com

[2] http://www.twitter.com

de dados contextuais é vista em um SIT denominado *Waze*[3], mas este não contempla o transporte público por ônibus. Existem ainda algumas propostas, como Alves [11], para desenvolvimento de um sistema de recomendação de rotas utilizando inteligência coletiva de redes sociais.

O presente trabalho apresenta um sistema de recomendação de rotas aos usuários de transporte público por ônibus, utilizando dados contextuais dinâmicos oriundos de redes sociais. Este sistema foi desenvolvido para uma plataforma móvel, usando *smartphones*, e será descrito na próxima seção.

3. UBIBUS ROUTE

Esta seção descreve a arquitetura e detalhes de implementação do UbibusRoute. O sistema que propõe a sugestão de rotas baseadas no conhecimento das informações dinâmicas.

4.1 Arquitetura

O UbibusRoute foi projetado em uma arquitetura cliente-servidor (Figura 1) onde o cliente contém um componente chamado Aplicativo Móvel, que pode ser nativo de qualquer sistema operacional móvel que se comunique com o servidor seguindo os padrões fornecidos pelo mesmo.

Existem duas bases de dados principais no sistema, a Base de Dados Contextuais de Trânsito, e a Base de Dados Estáticos. Os dados estáticos são paradas, linhas de ônibus e percursos. No módulo Cliente, o *Aplicativo Móvel* envia uma requisição de dados estáticos e de pedido de rota ao servidor, recebendo como resultado a rota e as instruções para percorrê-la. Já o módulo Servidor é dividido em três componentes principais: o *Identificador de Rotas,* responsável por identificar todas as rotas possíveis de acordo com a localização ou parada de ônibus selecionada; o *Indicador de Rotas*, responsável por selecionar a melhor rota de acordo com a preferência do usuário (tempo, preço ou distância); e o *Extrator de Informações Contextuais* que colhe informações de redes sociais e verifica como o trânsito está naquele momento.

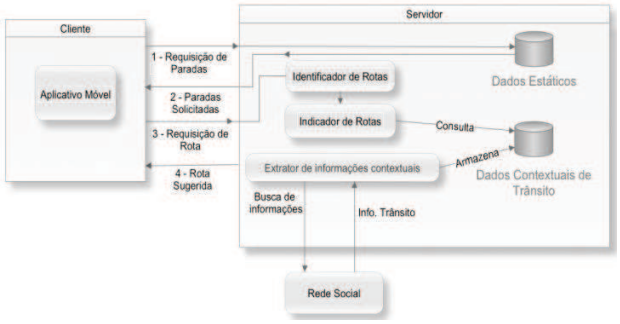


Figura 1 - Arquitetura Geral do UbibusRoute

4.1.1. Aplicativo Móvel

O Cliente, na arquitetura, contém um módulo chamado *Aplicativo Móvel*, que é a interface com o usuário. Na execução do aplicativo, é realizada inicialmente uma requisição automática de dados das paradas de ônibus, sendo fornecido ao usuário um mapa onde o mesmo pode visualizar todas as paradas de ônibus disponíveis e selecionar duas: uma parada de ônibus como origem e outra como destino. Além disso, o usuário deve informar se prefere que a rota sugerida seja baseada em alguns fatores como: menor preço, viagem com menor tempo ou menor distância percorrida. Os dados das paradas de ônibus selecionadas, bem como o tipo de viagem escolhido, serão enviados para o *Identificador de Rotas* no servidor.

A resposta obtida do servidor tem características visuais e textuais, onde consta uma rota traçada em um mapa, enfatizando o ponto de origem, o ponto de destino, os pontos de troca de ônibus (caso existam), o preço de cada um deles e uma mensagem textual. Tal visualização segue de acordo a opção selecionada pelo usuário inicialmente, contendo a distância total informada, o preço total a ser gasto, ou ainda o tempo total de viagem. Na resposta textual, o usuário recebe um texto explicando todas as etapas que o passageiro deve seguir para chegar ao seu destino.

4.1.2. Identificador de Rotas

O processo de identificar rotas, que acontece no componente *Identificador de Rotas*, é iniciado a partir de uma requisição por parte do *Aplicativo Móvel*. Desenvolveu-se um algoritmo de busca de rotas modificando o algoritmo guloso de Dijkstra[4] para que seja retornado o menor caminho, levando em consideração o parâmetro vindo do aplicativo móvel, ou seja, o *menor caminho* não é sempre pela distância; a depender do parâmetro, o peso das arestas do grafo que o algoritmo Dijkstra gerencia muda de acordo com a *preferência do usuário*.

A função do algoritmo desenvolvido é, portanto, procurar os possíveis caminhos que possam ser traçados entre a origem e o destino, oferecendo como saída essas rotas com parâmetros que indicam preço da viagem, distância a ser percorrida e tempo total. Este módulo classifica, então, cada possível rota com três pontuações numéricas: uma para a distância, uma para o tempo e uma para o preço a ser gasto. A saída desse módulo é enviada ao *Indicador de Rotas*.

4.1.3 Indicador de Rotas

O componente *Indicador de Rotas*, por sua vez, utiliza todas as rotas possíveis, recuperadas a partir do processo de identificação, e seleciona a melhor, baseada na preferência inicial definida pelo usuário (menor preço, menor distância ou menor tempo). Esta seleção é feita a partir da saída das rotas repassadas do módulo *Identificador de Rotas*. Como cada rota possui uma pontuação de tempo, custo de preço e distância, é escolhida a que contém o menor valor do parâmetro indicado pelo usuário. Este componente consulta, também, informações contextuais armazenadas pelo *Extrator de Informações Contextuais*, que estão guardadas na base de *Dados Contextuais*, provenientes de redes sociais. Sendo assim, o *Indicador de Rotas* é responsável por cruzar as preferências do usuário com as informações contextuais capturadas das redes sociais, atribuindo uma nova pontuação relativa ao tempo para as possíveis rotas.

4.1.4. Extrator de Informações Contextuais, Dados Contextuais e Redes Sociais

O objetivo do *Extrator de Informações Contextuais* é capturar informações dinâmicas de redes sociais. É possível obter mensagens recentes relacionadas aos trechos do trânsito por onde passam as linhas de ônibus cadastradas no UbibusRoute.

A rede social explorada no UbibusRoute foi o *Twitter*, devido à natureza curta de suas mensagens, com no máximo 140

[3] http://www.waze.com/

[4] http://www.cs.auckland.ac.nz/~jmor159/PLDS210/dijkstra.html

caracteres. O módulo das Redes Sociais está intrinsecamente ligado ao *Extrator de Informações Contextuais*. Os *tweets* são extraídos em intervalos de tempo periódicos. Na versão inicial deste sistema, a informação dos *tweets* foi interpretada com o uso de técnicas de expressões regulares. Por fim, esses *tweets* armazenados em uma *Base de Dados Contextuais de Trânsito,* localizada no servidor. Tais informações são utilizadas para conhecer a situação atual do trânsito.

As mensagens provenientes das redes sociais precisariam estar em um formato pré-estabelecido, a fim de identificar a informação. Essa limitação acabava perdendo outras mensagens que contenham informações importantes e relevantes sobre o trânsito. No entanto, tendo as mensagens nesse formato pré-definido, há um ganho de velocidade no processo de identificação textual das diversas situações de trânsito, melhorando o desempenho e garantido corretude na informação identificada.

No primeiro protótipo, para experimentação, foi desenvolvida uma gramática de reconhecimento e padronização das mensagens. As mensagens deveriam ser compostas por um número que identifica a hora, <*HORA*>, os minutos <*MIN*>, um fato agravante <*AGRAVANTE*>, que é uma palavra que intensifica uma situação de trânsito (a presença de um agravante na mensagem a ser identificada é facultativa). Uma situação de trânsito <*SITUAÇÃO*>, por sua vez, é uma palavra que indica congestionamento ou fluidez no trânsito, em diferentes níveis. Uma preposição <*PREPOSIÇÃO*> é apenas um conector entre a situação de trânsito e a localidade da ocorrência, composta por <LOCAL> e <TRECHO>. A gramática proposta é ilustrada na Figura 2.

<*HORA*> h <*MIN*> min Transito <*AGRAVANTE*> <*SITUAÇÃO*> <*PREPOSIÇÃO*> <*LOCAL*> <*TRECHO*>

Onde:

<*HORA*> → número de 00 a 23;

<*MIN*> → número de 00 a 59;

<*AGRAVANTE*> → muito | pouco | quase | bastante | meio

<*SITUAÇÃO*> → parado | lento | ruim | bom | livre | fácil | difícil | moderado | péssimo | complicado | devagar | rápido | congestionado | engarrafado | fluindo | normal | tranquilo | leve | pesado

<*PREPOSIÇÃO*> → em | na | no

<*LOCAL*> → nomes de ruas da base dados

<*TRECHO*> → trechos cadastrados na base de dados.

Figura 2 - Gramática para Extração de Mensagens de Trânsito de Redes Sociais.

As informações contextuais extraídas do *Twitter* são armazenadas na base de dados como um tipo composto [localização geográfica, dia, horário, *pontuação*]. A *pontuação*, que será detalhada na próxima seção, indica o grau de congestionamento de determinado local.

4.2. Aspectos de Implementação

O aplicativo móvel foi desenvolvido sobre a plataforma *Android,* em *Java* e usando alguns recursos da API do *Google Maps*. A comunicação com o Servidor é feita seguindo os princípios REST (*Representational State Transfer*), onde o módulo Cliente faz uma requisição HTTP, contendo *origem*, *destino* e *tipo de busca*, e o módulo Servidor responde com um objeto JSON (*JavaScript Object Notation*).

O servidor do UbibusRoute foi implementado com o framework *Django* e todos os seus módulos foram desenvolvidos em *Python*. A resposta do servidor ao aplicativo móvel é dada a partir da interação entre os componentes: *Identificador de Rotas e Indicador de Rota,* consultando os Dados Contextuais de Trânsito. Tal resposta possui vários parâmetros como: pontos geográficos da origem e destino, dados geográficos das paradas onde houver troca de ônibus; preço de cada troca de ônibus, caso exista; mensagem de resposta textual; e a mensagem final da requisição, que contém: a distância a ser percorrida, o custo de tempo total da viagem, ou o custo relativo ao preço.

A comunicação com o *Twitter* é realizada pelo *Extrator de Informações Contextuais* por meio da API REST[5] do *Twitter*. Isto permite acesso a dados essenciais como: *tweets* (mensagens postadas pelos usuários), prazos de atualização e informações dos usuários. A atualização dos dados contextuais de trânsito ocorre de acordo com a atual situação de trânsito informada pelos *tweets*.

Para manter os dados atualizados sobre o estado atual do trânsito, foi desenvolvido um *crawler* (componente que busca continuamente os *tweets* relativos ao trânsito), que se encontra dentro do *Extrator de Informações Contextuais*. Este *crawler* utiliza a API REST do *Twitter* e captura automaticamente as informações do perfil @Ubibus_PE de acordo com dois parâmetros: quantidade máxima de *tweets* por captura - definida como 20, para efeitos de desempenho, e a diferença entre os instantes de tempo atual e de criação do *tweet* - definido como 20 minutos na versão atual (podendo ser parametrizado em versões futuras). Com isso, é possível obter os 20 últimos *tweets* que foram publicados no perfil.

Também foi construído um *script* em *Python* que formata o conteúdo capturado do perfil do *Twitter*, extraindo o trecho e a situação atual do trânsito. Este *script* contém um algoritmo de interpretação de informações de trânsito que opera considerando a Gramática definida anteriormente (Seção 3.1.4). A situação do trânsito é calculada de acordo com algumas variáveis que classificam o adjetivo <*SITUAÇÃO*> combinado com o seu <*AGRAVANTE*>. De acordo com a *situação* e a presença ou não de um *agravante,* é calculada uma pontuação para aquele endereço. Essa pontuação é válida por 20 minutos, (tempo considerado suficiente para mudança na fluidez do tráfego). Para facilitar a extração das informações do trânsito, foi criado um conjunto de classificação das *situações* e dos *agravantes*. A situação pode estar classificada como [*bom*, *médio* e *ruim*] e os agravantes como [*alto*, *baixo*].

Toda rua que é representada na base de dados possui, estaticamente, um indicativo de velocidade da via. O tempo de trânsito (a pontuação) é calculado em função da mudança que pode ocorrer nessa velocidade. Uma vez que o trânsito está livre (sem interrupções ou congestionamentos), a velocidade não é alterada e, portanto, a pontuação fica zerada. Na primeira versão deste sistema, o retorno desse algoritmo consiste de uma pontuação que oscila de 0 a 3 onde: 0 (zero) indica que o trânsito está livre, como descrito anteriormente; 1 (um) indica que o trânsito está moderado, quando houve detecção dos classificadores de situação *médio*; quando da detecção do classificador *ruim,* a pontuação atribuída é 2 (dois) e indica que o trânsito está lento; e, por fim, 3 (três) indica que o trânsito está muito lento, quando da junção do classificador *ruim* com o agravante *alto*. Posteriormente, estes números são utilizados no algoritmo de busca de rotas para influenciar na decisão de qual rota sugerir e qual tempo informar ao usuário.

[5] http://dev.twitter.com/docs/api

5. CENÁRIO DE USO DO UBIBUS ROUTE

Foi desenvolvido um cenário de experimentação do sistema, que teve como objetivo verificar o comportamento das recomendações de acordo com as informações contextuais extraídas do *Twitter*. As bases de dados foram simuladas com a criação de um perfil no *Twitter* (@Ubibus_PE), e alguns endereços da cidade do Recife, simulando neles localizações de pontos de ônibus com latitude, longitude e identificadores. A partir daí, foram simuladas rotas de linhas de ônibus, preços e intervalo de tempo de saída dos ônibus dos terminais.

Ao usuário, foi disponibilizada uma aplicação móvel (como descrita anteriormente), onde ele poderia escolher origem e destino. À medida que determinado endereço fosse comentado, e sua pontuação depreciada, de acordo com as informações contextuais, a sugestão das rotas baseadas no tempo de viagem ia mudando. Na Figura 3, é possível constatar um resultado de consulta de rotas baseada em menor tempo (custo).

Figura 3 - Mapa e informações da rota indicada

À esquerda, a resposta textual dada ao usuário e à direita o mapa com origem e destino marcados e o trajeto traçado.

6. CONCLUSÃO

Este trabalho teve por objetivo desenvolver uma aplicação móvel para indicar e sugerir rotas aos usuários de transporte público, capaz de obter informações contextuais dinâmicas da rede social *Twitter*. Tal recomendação é baseada em pontos de origem, destino e tipo de busca, fornecidos pelos usuários. Para implementação do UbibusRoute [8] foram utilizadas ferramentas externas como a API do *Google Maps* e *Twitter*. O aplicativo mostrou que é possível recomendar rotas de ônibus aos usuários utilizando informações dinâmicas extraídas de redes sociais, apesar de demandar algumas melhorias que serão abordadas em pesquisas futuras.

Alguns trabalhos futuros serão investigados com o intuito de contribuir para a melhoria do sistema desenvolvido, entre eles podemos citar: a implementação de outros algoritmos para a busca por menor preço, a utilização de uma base de dados reais, o uso de informações provenientes de outras Redes Sociais, como *Facebook*, *Instagram* e *Foursquare*, além de mudanças na forma de busca e interpretação dessas informações. Estão também sendo desenvolvidas técnicas baseadas em Análise de Sentimentos, a fim de descobrir o *sentimento* em mensagens de usuários sobre o trânsito, para então extrair informação necessária sobre a pontuação das rotas, como funciona atualmente no UbibusRoute.

REFERÊNCIAS

[1] Cutolo, F. A. (2003) "Diretrizes para sistema de informação ao usuário". In: III Seminário Internacional PROMOTEO, Porto Alegre-RS.

[2] Gómez, A., Diaz, G. and Bousetta, K. (2009) "ITS Forecast: GIS Integration with Active Sensory System". In: Global Information Infrastructure Symposium, GlJS'09. Hammamet, Tunísia, pp. 1-6.

[3] Sussman, J. (2005), "Perspectives on Intelligent Transportation Systems". New York, USA: Springer.

[4] Levy, P. (2003), "A Inteligência Coletiva". São Paulo: Loyola.

[5] Mazhelis, O., Zliobaite, I. and Pechenizkiy, M. (2011) "Context-aware personal route recognition". In: The Fourteenth International Conference on Discovery Science (DS 2011), Espoo, Finland, pp. 365-379.

[6] Brézillon, P. (1999) "Context in Artificial Intelligence: IA Survey of the Literature", Computer & Artificial Intelligence, v. 18, pp. 321-340.

[7] Vieira, V. Salgado, A.C., Tedesco, P., Times, V. C., Ferraz, C., Huzita, E., Chaves, A. P., Steinmacher, I. The UbiBus Project: Using Context and Ubiquitous Computing to build Advanced Public Transportation Systems to Support Bus Passengers. In: Anais do VIII Simpósio Brasileiro de Sistemas de Informação, 2012, São Paulo, pp. 55-60.

[8] Lima, V., Magalhães, F., Tito, A., Santos, R., Ristar, A., Santos, L., Vieira, V., Salgado, A. C. (2012) "UbibusRoute : Um Sistema de Identificação e Sugestão de Rotas de Ônibus Baseado em Informações de Redes Sociais". In: VIII Simpósio Brasileiro de Sistemas de Informação (SBSI) – Cidades Inteligentes, São Paulo-SP, pp. 516-527.

[9] Hoar, R. (2010) "A Personalized Web Based Public Transit Information System with User Feedback". In: 13th International IEEE Conference on Intelligent Transportation Systems, Ilha da Madeira, Portugal.

[10] Ferris, B., Watkins, K. and Borning, A. (2010) "Location-Aware Tools for Improving Public Transit Usability", IEEE Pervasive Computing, v. 9 n. 1, pp. 13-19.

[11] Alves, L. P. S., Chaves, A. P e Steinmacher, I. F. (2011) "Um aplicativo baseado em inteligência coletiva para compartilhamento de rotas em redes sociais". In: VIII Simpósio Brasileiro de Sistemas Colaborativos (SBSC), Paraty-RJ.

Um Guia Eletrônico de Programação Baseado em Serviços Web e na Plataforma Android

Carlos Eduardo Ferreira Marins
Laboratory of Advanced Web Systems – UFMA
Av. dos Portugueses, Campus do Bacanga
São Luís/MA – 65085-580 - Brasil
edwardmarins@gmail.com

Mário Meireles Teixeira
Departamento de Informática – UFMA
Av. dos Portugueses, Campus do Bacanga
São Luís/MA – 65085-580 - Brasil
mario@deinf.ufma.br

ABSTRACT
The large volume of content provided by digital TV motivates the development of solutions that facilitate the choices of the viewers. This paper presents the architecture of an electronic programming guide based on web services using the REST architectural style. We also present the development of EPGdroid, an application on the Android platform, which consumes resources of the web service and provides a friendly interface to the end user.

Keywords
Web Services, SOA, REST, EPG, Android, Mobile Devices.

1. INTRODUÇÃO

Com a implantação da TV digital no Brasil tornaram-se muito bem-vindos esforços de pesquisa e produção de softwares que forneçam suporte ou ampliem as possibilidades dessa tecnologia.

Uma extensa variedade de programas e serviços vêm sendo disponibilizados por meio da TV Digital. A grande quantidade de conteúdos digitais disponíveis motiva a criação de aplicações que sejam capazes de coletar informações sobre os programas e serviços e apresentá-las de maneira organizada [5]. Nesse contexto, é essencial a existência de aplicações que apresentem para os usuários as diferentes opções de conteúdo oferecidas, de modo a auxiliar sua escolha. Essas aplicações são chamadas de Guias Eletrônicos de Programação [6].

Este artigo tem com objetivo o desenvolvimento de uma arquitetura interoperável, baseada em serviços web RESTful, que permita disponibilizar a programação atualizada das principais emissoras de TV, de modo que as mesmas possam ser facilmente consumidas por aplicações clientes desenvolvidas em diversas linguagens de programação e executando sobre plataformas heterogêneas.

2. SERVIÇOS WEB

Um Web Service, ou serviço web, é definido pelo W3C como sendo uma aplicação de software identificada por um URI, cujas interfaces e ligações são capazes de serem definidas, descritas, e descobertas como artefatos XML [8]. Um serviço web suporta interações diretas com outros agentes de software usando mensagens baseadas em XML trocadas via protocolos da Internet.

Dentre as principais abordagens para a utilização de serviços web destacam-se a Arquitetura Orientada a Serviços e o estilo arquitetural REST.

2.1 Arquitetura Orientada a Serviços

A Arquitetura Orientada a Serviços está baseada na relação entre um provedor, que oferece recursos e funcionalidades sob a forma de serviços web, e um consumidor de serviços, que se trata de um componente de software interessado em associar-se aos serviços. Com o propósito de facilitar essa associação, os serviços devem possuir uma especificação formal de sua interface e dos procedimentos necessários para que os consumidores de serviços possam se conectar a ele. Essa descrição é formalizada utilizando-se o padrão WSDL e o provedor de serviços pode, eventualmente, publicá-la em um registro de serviços, conforme ilustrado na Figura 1.

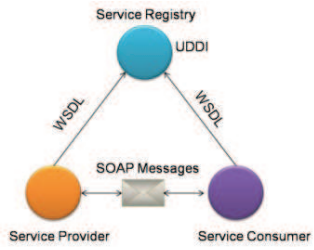


Figura 1: Estrutura da Arquitetura Orientada a Serviços

Um registro de serviços facilita a descoberta de serviços e a aquisição de informações de uso, provendo uma maneira uniforme de o consumidor descobrir quais serviços disponíveis estão aptos a satisfazer seus interesses. Este mecanismo uniforme é alcançado através do padrão UDDI.

Uma vez descoberto o serviço, o consumidor pode associar-se a ele para utilizar os recursos, dados ou funcionalidades fornecidas. Para isso, tanto provedor quanto consumidor de serviços trocam mensagens padronizadas através do protocolo SOAP.

A seqüência publicar/descobrir/associar e os elementos provedor, consumidor e registro de serviços constituem a essência da arquitetura orientada a serviços.

2.2 Estilo Arquitetural REST

Representational State Transfer (REST) é um estilo arquitetural híbrido para sistemas hipermídia distribuídos, derivado de vários

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WebMedia '12, Oct 15–18, 2012, São Paulo, SP, Brazil.
Copyright 2012 ACM 1-58113-000-0/00/0010…$10.00.

estilos arquiteturais de software em rede [3], que define um conjunto de princípios que podem ser aplicados na construção de sistemas com uma Arquitetura Orientada a Recursos (ROA). Sistemas construídos segundo os princípios REST são chamados aplicativos *RESTful*.

REST está baseado no conceito de recursos que são identificados e disponibilizados através de URIs. Essas URIs são acessíveis a partir de links que retornam as representações dos recursos. Quando o cliente acessa uma dessas representações ele passa a ocupar um novo estado em relação à aplicação como um todo. Cada representação pode conter links para outros recursos, o que permite ao cliente navegar pela aplicação a partir da transição entre estados, daí o termo *Representational State Transfer*.

Também vale destacar a utilização de uma interface uniforme, ou seja, em REST o conjunto de métodos para cada elemento do sistema é conhecido e padronizado, sendo que a cada execução de um método, sua semântica é visível.

3. PLATAFORMA ANDROID

Com a popularização massiva dos celulares e dispositivos móveis com cada vez maior poder de processamento, a computação demonstra uma tendência natural para aplicações baseadas em plataformas para ambientes móveis. Uma das mais recentes e bem sucedidas é a plataforma Android.

A plataforma Android foi lançada em 2008 pela Google, por meio do consórcio *Open Handset Alliance* e, segundo [1], é uma pilha de software para dispositivos móveis que inclui um sistema operacional, middleware e aplicações denominadas críticas. O Android SDK (*Software Development Kit*) fornece as ferramentas e APIs (*Application Programming Interfaces*) necessárias para começar a desenvolver aplicações na plataforma Android usando a linguagem de programação Java.

A plataforma Android é um ambiente para dispositivos móveis que inclui um sistema operacional baseado no kernel 2.6 do Linux, um ambiente rico para desenvolvedores, além de suporte às tecnologias presentes na maioria dos dispositivos móveis e funcionalidades convencionais de telefones celulares [7].

3.1 Arquitetura Android

A arquitetura da plataforma Android, representada na Figura 2, é organizada em camadas que possuem atribuições específicas e gerenciam seus próprios processos.

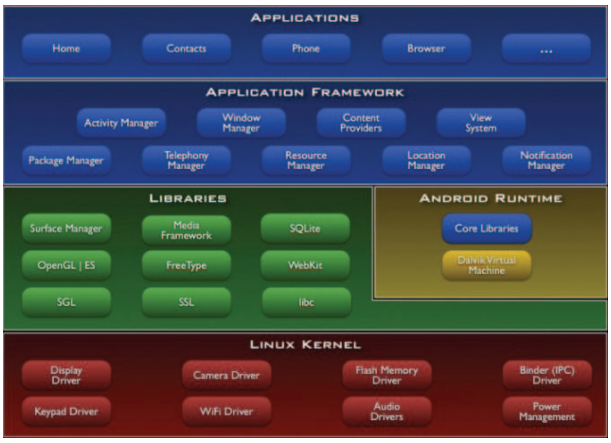


Figura 2: Arquitetura da plataforma Android.

A camada superior oferece um conjunto de aplicações de alto nível que são acessadas diretamente pelo usuário final, dentre as quais se destacam clientes de email, calendário, mapas, navegador, jogos e aplicações de terceiros. Logo abaixo da camada de aplicação, estão localizados os frameworks de aplicação, que incluem os programas que gerenciam as funções básicas do dispositivo como alocação de recursos, gerenciamento do telefone, oferecem informações sobre localização, notificações, alarmes etc.

Um pouco mais abaixo na pilha, Android inclui bibliotecas escritas em C/C++, bibliotecas de multimídia, visualização de camadas 2D e 3D, funções para navegadores web, funções de aceleradores de hardware, renderização 3D, funções para gráficos, fontes bitmap e vetorizadas e funções de acesso a banco de dados SQLite, dentre outras funcionalidades expostas aos desenvolvedores através dos frameworks de aplicação. No mesmo nível das bibliotecas está o ambiente de execução do Android (Android Runtime) que possui um conjunto de bibliotecas do núcleo Java e instâncias da máquina virtual Dalvik, uma máquina virtual própria, otimizada para dispositivos com poucos recursos computacionais e projetada especificamente para uso em ambientes embarcados.

Na base da pilha localiza-se o kernel do Linux, responsável pelo gerenciamento de memória, de processos e de E/S, pilha de rede, drivers de dispositivos, entre outros. É nesta camada que ocorre a abstração entre o hardware e o restante da pilha de software.

3.2 Principais Componentes

Existem quatro tipos principais de componentes Android: *activities*, *services*, *broadcast receivers* e *content providers*.

Atividades representam interfaces visuais da aplicação através das quais ocorre toda a interação com o usuário. Em geral, uma aplicação Android é composta por várias atividades, sendo que uma delas é carregada no inicio da aplicação, a qual é denominada atividade principal. A partir da atividade principal, outras atividades podem ser acionadas, cada uma com funções específicas, colaborando para a dinâmica do sistema. Apenas uma atividade pode estar ativa por vez, por isso, cada vez que uma nova atividade começa, a atividade anterior é interrompida e colocada em uma pilha (*back stack*).

A classe *Services* é usada para executar um serviço em segundo plano, geralmente vinculado a algum processo que deve ser executado por tempo indeterminado e possui um alto consumo de recursos, memória e unidade central de processamento [2].

Broadcast Receivers constituem uma implementação Android para um mecanismo produtor/consumidor, ou mais precisamente, um padrão do tipo observador. Através desse mecanismo uma aplicação pode registrar um receptor associado a um evento, de modo que no momento em que o evento ocorre o método *onReceive*() é disparado.

Por padrão, o Android executa cada aplicativo em sua própria *sandbox* para que todos os dados que pertençam a um aplicativo sejam totalmente isolados de outros aplicativos no sistema [4]. Os *content providers* (provedores de conteúdo) provém um nível de abstração para que qualquer dado guardado no dispositivo seja acessível por mais de uma aplicação [2].

4. ARQUITETURA PROPOSTA

Este trabalho descreve o desenvolvimento e a arquitetura de um serviço de guia eletrônico de programação, que adota uma abordagem de sistemas distribuídos baseada em serviços web que atendem às especificações do estilo arquitetural REST, e um cliente Android, responsável por fornecer uma interface amigável ao usuário final.

A partir da utilização do aplicativo os usuários são capazes de escolher um canal e visualizar sua programação diária, navegar na programação transmitida no momento corrente, podendo escolher um programa qualquer para visualizar seus detalhes e realizar uma pesquisa por programas de acordo com palavras-chave.

A arquitetura desenvolvida baseia-se nos componentes mostrados na Figura 3, os quais são detalhados a seguir.

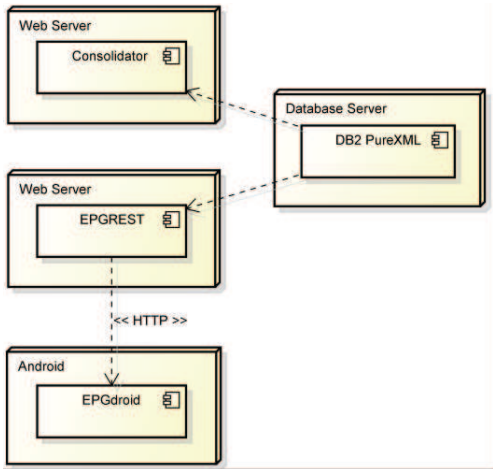


Figura 3: Arquitetura Proposta.

4.1 Componente Consolidator

Normalmente, as informações sobre programas e serviços são enviadas, por radiodifusão ou sob demanda, multiplexadas com outros tipos de dados como, por exemplo, áudio e vídeo [5]. No entanto, quando se deseja obter essas informações fora do contexto das televisões e decodificadores é necessária a utilização de fontes de dados alternativas, geralmente disponibilizadas na Web. Para este fim, em português, destacam-se como provedores de informações de programação de TV a tv.sapo.pt [1] e a Revista Eletrônica[2].

A Revista Eletrônica disponibiliza essas informações através do download de arquivos no formato XMLTV, que é um formato padronizado baseado em XML usado para obtenção, manipulação e compartilhamento de informações de programação de TV de vários canais.

O componente *Consolidator* tem por objetivo transformar estes arquivos em informação utilizável pelo serviço *RESTful*.

Para consolidar as informações e permitir o acesso às mesmas por outros componentes, o *Consolidator* efetua o download dos arquivos XMLTV, processando-os para os moldes específicos da aplicação e armazena estes arquivos em uma base de dados XML gerenciada pelo *SGBD DB2 PureXML*.

[1] http://tv.sapo.pt

[2] http://www.revistaeletronica.com.br

O *PureXML* é a nova tecnologia inserida no DB2 versão 9 permitindo que o banco de dados armazene documentos XML bem formados em colunas do tipo de dados XML, mantendo sua forma hierárquica. Deste modo não é mais necessário mapear documentos XML para o banco de dados ou armazená-los em objetos grandes.

4.2 Componente EPGREST

O componente EPGREST tem como função oferecer uma interface uniforme para acesso a informações de programação de TV, retornando seus resultados em um formato portátil, baseado em XML e facilmente interpretado por praticamente todas as linguagens de programação modernas.

EPGREST foi desenvolvido a partir dos princípios arquiteturais REST, utilizando serviços web para implementar funcionalidades bem definidas e independentes. Devido a estas características, os serviços podem ser consumidos por clientes em diferentes aplicações e processos de negócios, utilizando-se variadas plataformas e tecnologias.

Para a aplicação do guia eletrônico de programação foram definidos dois serviços básicos, *channel* e EPG, ambos possuindo apenas métodos GET.

O serviço *channel* possui somente um recurso, que disponibiliza uma lista com todos os canais que possuem suas programações registradas. O serviço *channel* é identificado pela URL */channel*.

O serviço EPG oferece um conjunto de subrecursos, representando documentos XMLTV, relacionados à programação de TV propriamente dita. O primeiro subrecurso recebe como parâmetro um código de canal e uma data, disponibilizando a programação inteira do canal na data especificada. O segundo subrecurso oferece um documento XMLTV contendo os programas, de todos os canais, que estão sendo transmitidos na data e hora atuais. Um terceiro subrecurso realiza uma busca na programação baseando-se em uma string passada como parâmetro. Outros subrecursos, embora não utilizados pelo cliente EPGdroid, estão disponíveis no serviço, entre os quais: a programação de um programa específico em um intervalo de datas, a programação de um dado canal em um intervalo de datas, todos os programas de uma determinada categoria em um intervalo de datas. As URLs para acesso aos recursos citados são, respectivamente:

- /EPG/{canal}/{data}
- /EPG/now
- /EPG/search/{palavra-chave}
- /EPG/program/{id_programa}/{data1}/{data2}
- /EPG/{canal}/{data1}/{data2}
- /EPG/category/{categoria}/{data1}/{data2}

4.3 Componente EPGdroid

Visto que o componente EPGREST foi desenvolvido como um serviço web *RESTful*, uma série de aplicações clientes, dos mais variados tipos, podem ser desenvolvidas a fim de consumir os recursos disponibilizados. Neste trabalho optou-se pela construção de um aplicativo sobre a plataforma Android, denominado EPGdroid.

O EPGdroid fornece um conjunto de interfaces para que o usuário possa ter acesso aos serviços do componente EPGREST. Estas interfaces são construídas usando os componentes básicos da plataforma Android descritos na Seção 3.2, entre outros, relacionados conforme diagrama da Figura 4.

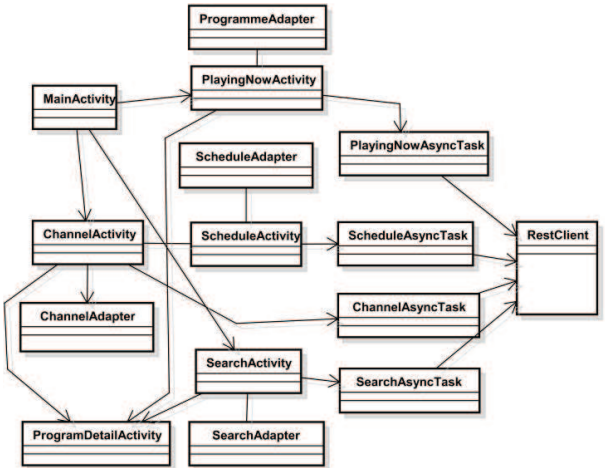


Figura 4: Diagrama de classes do EPGdroid.

A classe *MainActivity* é responsável pela navegação e acesso às funcionalidades providas pelo aplicativo. Ela utiliza o recurso *TabHost* da API Android para organizar o conteúdo da aplicação de uma forma intuitiva e prática, através do uso de abas. A partir de *MainActivity* três novas atividades podem ser acessadas: *ChannelActivity*, que exibe a lista dos canais disponíveis, PlayingNowActivity, que exibe a programação de cada canal no momento atual, e *SearchActivity*, que permite ao usuário efetuar uma busca na programação.

Outros elementos importantes do diagrama da Figura 4 são as classes que herdam de *AsyncTask*: *PlayingNowAsyncTask*, *ScheduleAsyncTask*, *ChannelAsyncTask* e *SearchAsyncTask*. Elas constituem um mecanismo apropriado para dar *feedback* ao usuário durante a execução de alguma operação que demande tempo.

Por fim, todo processamento relacionado ao serviço web foi encapsulado na classe *RestClient*. Esta classe é responsável por preparar e enviar as requisições HTTP e receber os resultados das solicitações.

Dentre as telas do EPGdroid, as que exibem os programas sendo transmitidos e a descrição de um programa são mostradas na Figura 5.

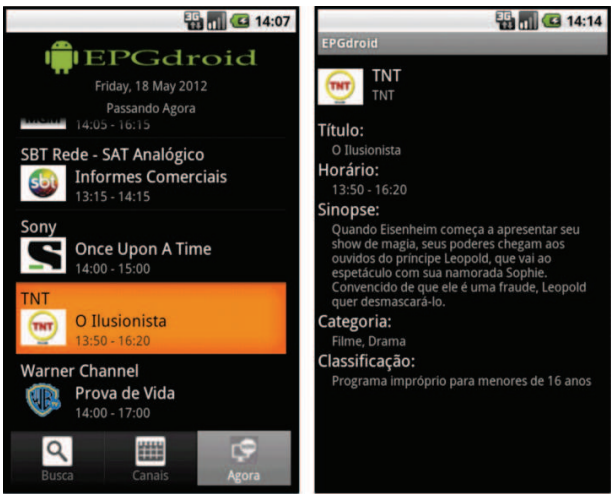


Figura 5: Telas do EPGdroid.

5. CONCLUSÃO

Este trabalho discutiu e implementou a arquitetura de um guia eletrônico de programação baseado em serviços web utilizando a abordagem REST, cuja interação com o usuário se dá por meio de um aplicativo desenvolvido sobre a plataforma Android.

Através deste trabalho pôde-se exemplificar como sistemas baseados em REST podem promover a dissociação entre as interfaces dos serviços e suas implementações, visto que os clientes precisam apenas descobrir essas interfaces e consumir os resultados disponibilizados sob o padrão XML, sem a necessidade de conhecer detalhes internos ao serviço.

Outro fator importante foi a utilização de um aplicativo Android interagindo com o serviço web *RESTful*, o que garante mobilidade e maior utilidade à aplicação, dada o aumento da preferência dos usuários por dispositivos móveis para a busca de informações relacionadas a consumo e entretenimento.

Como trabalhos futuros objetiva-se implementar um sistema de recomendação de programação baseado no perfil do usuário e realizar a integração com o IMDb (*Internet Movie Database*) de modo a aumentar o leque de informações sobre filmes e séries de TV, proporcionando maior interatividade ao usuário.

REFERÊNCIAS

[1] Android developers. User Interfaces. Disponível em: http://developer.android.com/guide/topics/ui/index.html. acessado em: 01 Maio 2012.

[2] Ferreira, G. D. Um middleware declarativo na plataforma AndroidTM para o Sistema Brasileiro de Televisão Digital (SBTVD). Dissertação, Universidade Federal do Espírito Santo, Vitória, ES, 2010.

[3] Fielding, R. T. Architectural Styles and The Design of Network-based Software Architectures. PhD thesis, University of California, Irvine, 2000.

[4] Gargenta, M. Learning Android. O'Reilly, Sebastopol, 2010.

[5] Maia, P. P. C., Leite, J., Batista, T. MyPersonal-EPG: Um EPG Personalizável e com Suporte à Recomendações. in WEBMEDIA (Belo Horizonte, 2010).

[6] Oliveira, F. N. B. de. Aplicação Adaptativa de Guia Eletrônico utilizando o Ginga-NCL. Dissertação, Pontifícia Universidade Católica Do Rio De Janeiro - Puc-Rio, Rio de Janeiro, RJ, 2010.

[7] Sousa, T. N. de. Desenvolvimento de uma Rede P2P para a Plataforma Android. Monografia, Universidade Federal do Maranhão, São Luis, Maranhão, 2011.

[8] W3C. Disponível em: http://www.W3C.org. Acessado em: 01 Maio 2012.

Buscando Fontes de Dados Relevantes para Aplicações Linked Data

Alberto Trindade Tavares
Centro de Informática - Universidade Federal de Pernambuco
Av. Jornalista Anibal Fernandes, s/n
Cidade Universitária, 50.740-560 - Recife – PE, Brasil
+55 81 2126.8430
att@cin.ufpe.br

Hélio Rodrigues de Oliveira
Centro de Informática - Universidade Federal de Pernambuco
Av. Jornalista Anibal Fernandes, s/n
Cidade Universitária, 50.740-560 - Recife – PE, Brasil
+55 81 2126.8430
hro@cin.ufpe.br

Bernadette Farias Lóscio
Centro de Informática - Universidade Federal de Pernambuco
Av. Jornalista Anibal Fernandes, s/n
Cidade Universitária, 50.740-560 - Recife – PE, Brasil
+55 81 2126.8430
bfl@cin.ufpe.br

ABSTRACT
The growing volume of Linked Data sources has motivated the interest in developing applications and tools focused on consuming linked data. One of the main challenges of developing such applications is the identification of relevant sources, i.e., sources that could contribute significantly to the results of user queries submitted to the application. In this paper, we discuss this problem and present an approach to detect sources that are potentially relevant to a specific application that uses linked data. One distinguishing issue of our approach is that the process of identifying new data sources employs the user requirements expressed in SPARQL queries posed to the application.

RESUMO
O crescimento no volume de fontes de dados *Linked Data* tem despertado um grande interesse no desenvolvimento de aplicações e ferramentas voltadas para o consumo de dados interligados. Diante disso, um dos principais desafios em relação ao desenvolvimento de aplicações que utilizam dados desta natureza é a identificação de fontes relevantes, ou seja, aquelas capazes de contribuir de maneira significativa com os resultados de consultas de usuários submetidas à aplicação. Neste artigo, discutimos este problema e apresentamos uma abordagem para detectar fontes de dados que sejam potencialmente relevantes para uma determinada aplicação que consome dados interligados. Uma característica importante da abordagem proposta é que o processo de identificação de novas fontes faz uso dos requisitos de usuários expressos nas consultas SPARQL da aplicação.

Categories and Subject Descriptors
H.4 [**Information System Applications**]: Miscellaneous;
H.2 [**Database Management**]: Miscellaneous

General Terms
Algorithms, Management, Experimentation.

Keywords
Semantic Web, Linked Data, Web Crawling.

1. INTRODUÇÃO
Diversas iniciativas, como as desenvolvidas pelo W3C (*World Wide Web Consortium*), buscam, por intermédio da criação de padrões, arquiteturas de metadados, serviços de inferências e ontologias, soluções que facilitem o compartilhamento e o processamento de dados disponíveis na Web. Dentre estas iniciativas, destaca-se a Web Semântica (*Semantic Web*), uma extensão da Web atual, onde o conteúdo publicado está associado a um significado que é compreensível tanto por um humano quanto por uma máquina. No contexto da Web Semântica, o termo *Linked Data* (Dados Interligados) é utilizado para descrever um conjunto de práticas para publicação de dados estruturados na Web, de forma a aumentar o valor e a utilidade desses dados.

Com a crescente adoção dos padrões de *Linked Data*, também tem aumentado o número de aplicações que fazem uso destes dados. Estas aplicações podem ser genéricas, como os navegadores e os motores de buscas, uma vez que oferecem acesso a dados de diversos domínios, ou podem ser aplicações específicas, uma vez que oferecem acesso a domínios específicos (como o de dados bibliográficos ou governamentais, por exemplo). Um dos principais desafios no desenvolvimento das aplicações de domínio específico é a identificação de fontes de dados relevantes, ou seja, fontes de dados que poderão contribuir com informações úteis do ponto de vista do usuário da aplicação [2, 5]. Considerando um número potencialmente grande de conjuntos de dados interligados atualmente disponíveis na Web, detectar manualmente as fontes relevantes para uma aplicação pode se tornar uma tarefa inviável.

Neste trabalho, abordamos o problema da identificação de fontes de dados relevantes para aplicações de domínio específico que consomem dados interligados, ou seja, dados publicados de acordo com os princípios de *Linked Data*. A abordagem proposta faz uso dos requisitos de usuários, extraídos a partir das consultas submetidas à aplicação, a fim de guiar a busca por novas fontes de dados. Especificamente, estamos interessados em encontrar novas fontes de dados RDF a partir das informações que podem ser extraídas dos padrões de triplas presentes em um conjunto de consultas SPARQL. A utilização desta abordagem permite a detecção de fontes de dados que não foram consideradas inicialmente, mas que, potencialmente, irão contribuir com os resultados das consultas fornecidos aos usuários.

O restante do artigo é organizado como se segue. Na Seção 2, são apresentados os principais conceitos relacionados às tecnologias da Web Semântica que compõem a base para o nosso trabalho. A Seção 3 descreve a abordagem proposta para busca de fontes RDF na Web. A Seção 4 dá detalhes de implementação da abordagem proposta e de experimentos realizados. A Seção 5 apresenta trabalhos relacionados. Por fim, a Seção 6 conclui o artigo, citando contribuições deste trabalho e indicando pesquisas futuras.

2. FUNDAMENTAÇÃO TEÓRICA

Nesta seção, serão apresentados alguns conceitos preliminares e terminologias que serão utilizadas ao longo deste artigo.

2.1 URI e RDF

URI[1] é uma sequência de caracteres que identifica ou denomina unicamente um recurso Web. O mecanismo básico para acessar recursos Web segundo os padrões de *Linked Data* se dá através de um processo chamado de derreferenciamento de URIs, que consiste no acesso via HTTP a uma URI, obtendo-se um conjunto de descrições RDF [1].

RDF[2] é um modelo de dados que permite descrever recursos na Web por meio de triplas, as quais podem ser organizadas como grafos direcionados. Os três componentes de uma tripla são: *Sujeito*, *Predicado* e *Objeto*. Como exemplo, é apresentada na Figura 1 uma tripla RDF, extraída da fonte de dados *DBpedia*[3], expressando que a UFPE se localiza na cidade do Recife. O nó mais à esquerda é o *Sujeito*, um recurso que representa a Universidade Federal de Pernambuco (UFPE), conectado ao *Objeto*, um recurso que representa a cidade Recife, através de uma aresta rotulada pelo *Predicado dbpprop:city*.

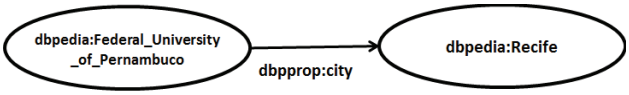


Figura 1. Exemplo de tripla RDF

2.2 SPARQL

SPARQL[4] é uma linguagem de consulta para dados RDF, permitindo a recuperação de informação contida em grafos. As principais partes de uma consulta SPARQL são [4]: O *padrão da consulta*, que é composto por um conjunto de padrões de triplas (*Triple Patterns)*, constituindo o denominado BGP (*Basic Graph Pattern)* da consulta; os *modificadores de solução*, que permitem reorganizar o resultado da consulta e a *saída*, que especifica o formato do resultado.

Como um exemplo, vamos considerar a consulta SPARQL apresentada na Figura 2, que extrai informações do *DBpedia* sobre pessoas que nasceram na cidade de Recife. Podemos identificar nesta consulta: i) o formato do resultado da consulta *SELECT ?nomePessoa*, ii) o BGP da consulta, contido na cláusula WHERE, que descreve os padrões com os quais as triplas resultantes devem estabelecer correspondência.

Q1. Retorne o nome de todas as pessoas que nasceram em Recife.

```
SELECT ?nomePessoa WHERE
{
        { ?pessoa dbp-owl:birthPlace ?cidade. }
        { ?pessoa foaf:name ?nomePessoa . }
        FILTER (?cidade = dbpedia:Recife)
}
```

Figura 2. Exemplo de consulta SPARQL

Fontes de dados interligados tipicamente fornecem um SPARQL *endpoint*[5], um serviço Web que permite ao usuário (humano ou máquina) submeter consultas SPARQL sobre os dados RDF disponibilizados na fonte.

2.3 Web Crawler Semântico

A extração de dados na Web pode ser realizada por meio de Web *crawlers*, agentes de software que acessam a Web de maneira automatizada, navegando entre os recursos por meio de links [7]. O processo realizado por este agente é chamado de *crawling*. Trabalhando sobre dados interligados, os Web *crawlers* semânticos diferem dos *crawlers* tradicionais em dois aspectos: o formato dos documentos em que navegam e o significado dos links entre as informações [6]. De maneira geral, os *crawlers* semânticos possuem a arquitetura apresentada na Figura 3 e descrita a seguir.

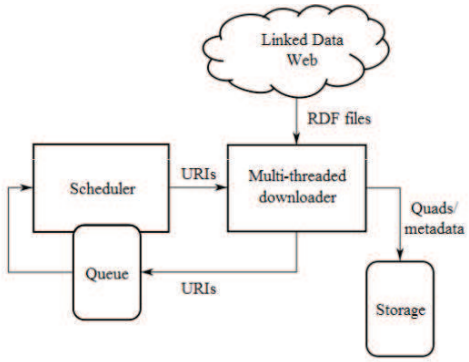


Figura 3. Arquitetura genérica de *crawlers* semânticos [7]

Um *crawler* semântico inicia sua busca de dados na Web a partir de um conjunto de recursos de origem, denominados s*eeds*, os quais são carregados em uma fila de URIs (*Queue*), onde são escalonadas uma a uma, pelo *Scheduler*, para a busca de conteúdo na Web de dados relacionados a cada recurso da fila. Cada URI escalonada é derreferenciada, obtendo-se um conjunto de triplas RDF, nas quais a URI aparece como sujeito ou objeto. A partir desta lista de triplas, são definidas novas URIs que serão inseridas na fila para próximas rodadas do *crawling*, opcionalmente pode-se restringir predicados para a obtenção de novas URIs [6].

3. A ABORDAGEM PROPOSTA

O número crescente de fontes de dados interligados disponível na Web faz surgir a necessidade de mecanismos que ajudem a filtrar tais fontes a fim de detectar aquelas que são mais relevantes de acordo com algum critério específico. Na abordagem proposta neste trabalho, aplicamos como filtro os requisitos de usuários que podem ser extraídos das consultas mais frequentemente submetidas a uma dada aplicação *Linked Data* de domínio específico.

A abordagem de busca de fontes proposta pode ser dividida em duas etapas:

(i) Extração de recursos mais relevantes de acordo com a frequência dos padrões de triplas das consultas da aplicação.

(ii) Realização de um *crawling* na Web para detectar fontes de dados interligados que descrevam os principais recursos extraídos na etapa anterior.

Estas etapas são descritas no Algoritmo *DatasetsSearch* (Algoritmo 1), que recebe como entrada um conjunto de consultas da aplicação e fornece como saída uma lista de fontes de dados candidatas que são possivelmente relevantes para esta aplicação,

[1]http://www.w3.org/TR/uri-clarification/
[2]http://www.w3.org/RDF/
[3]http://dbpedia.org/
[4]http://www.w3.org/TR/rdf-sparql-query/
[5]http://semanticweb.org/wiki/SPARQL_endpoint

especificamente, um conjunto de SPARQL *endpoints*. O Algoritmo 1 é detalhado nas subseções a seguir.

Algorithm DatasetsSearch
Input Q: A set of queries
k: Number of relevant resources to be considered during the crawling
Output *DE:* A set of SPARQL endpoints of fetched datasets
Begin
1. $RR \leftarrow \boldsymbol{ExtractRelevantResources}(Q)$
2. $Seeds \leftarrow SelectResources(RR, k)$
3. $Predicates \leftarrow \{sameAs,\ seeAlso,\ equivalentClass\}$
4. $FetchedTriples \leftarrow ExecuteCrawling(Seeds, Predicates)$
5. $ProvenanceTriples \leftarrow ExtractProvenance(FetchedTriples)$
6. $DE \leftarrow \emptyset$
7. **For each** $p \in ProvenanceTriples$ **do**
8. $DE \leftarrow DE \cup RetrieveSparqlEndpoint(p)$
9. **End for**
10. **Return** DE
End

Algoritmo 1. Algoritmo para Busca de Fontes de Dados

3.1 Extração de Recursos Relevantes

O primeiro passo do algoritmo consiste em usar a função *ExtractRelevantResources* para a identificação do conjunto de recursos relevantes, ou seja, recursos que irão guiar a busca por fontes de dados candidatas. De uma maneira geral, um recurso é identificado por uma URI e pode ser um sujeito ou objeto de um padrão de tripla. A função *ExtractRelevantResources* é apresentada no Algoritmo 2, onde podemos ver que este recebe como entrada o conjunto de consultas Q e retorna uma lista dos recursos mais frequentes de Q.

Especificamente, a extração de recursos consiste na recuperação do *BGP* para cada uma das consultas em Q e para cada padrão de tripla de um dado *BGP*, seus elementos (sujeito, predicado e objeto) são visitados. Ao longo desta visita, temos a construção de uma lista de recursos, presentes nos padrões de triplas das consultas, e de suas respectivas quantidades de ocorrência. No fim deste processo, os k recursos mais frequentes serão selecionados como sendo os mais relevantes.

3.2 Web Crawling para a Busca de Fontes

Uma vez que os recursos mais relevantes são identificados, o próximo passo consiste em realizar um processo de *crawling* na Web para buscar o conjunto de fontes candidatas. O processo de *crawling* considera como *seeds* os k primeiros recursos da lista *RR* (*RelevantResources)*, que é composta por URIs que representam recursos relevantes identificados a partir do conjunto de consultas Q. Um conjunto de predicados (*rdfs:seeAlso*, *owl:sameAs* e *owl:equivalentClass*) é utilizado durante o *crawling* para permitir a obtenção de novos recursos da Web que são similares aos recursos s*eeds*.

Ao final do *crawling*, temos um conjunto de triplas recuperadas da Web que descrevem tais recursos, as quais são armazenadas em um repositório RDF. O próximo passo do processo de busca é a construção da lista de fontes candidatas relevantes. Para esta tarefa, é extraída a proveniência das triplas coletadas pelo Web *crawler*. A informação sobre a proveniência de uma tripla é identificada por uma URI, indicando a localização do arquivo de origem da respectiva tripla. Para cada URI de proveniência, é utilizada a função *RetrieveSparqlEndpoint*, a qual é usada para a recuperação da URI do SPARQL *endpoint* das fontes de dados de procedência das triplas.

Algorithm ExtractRelevantResources

Input Q: A set of queries
Output *RR:* A sorted list by frequency of query resources
Begin
1. $FrequencyList \leftarrow \emptyset$
2. **For each** $q \in Q$ **do**
3. $BGP \leftarrow ExtractBGP(q)$
4. **For each** $triplePattern \in BGP$ **do**
5. $Resources \leftarrow VisitTriplePattern(triplePattern)$
6. $FrequencyList \leftarrow FrequencyList \cup Resources$
7. **End for**
8. **End for**
9. $RR \leftarrow DecreasingElementsList(FrequencyList)$
10. **Return** RR
End

Algoritmo 2. Algoritmo para Extração de Recursos Relevantes

A URI de cada *endpoint* coletado é inserida em um conjunto que é retornado como resultado final. Este conjunto de SPARQL *enpoints* fornece à aplicação em questão o acesso a novas fontes de dados da Web. Essas fontes são potenciais candidatas a integrarem o conjunto de fontes de dados que podem ser acessados a partir da aplicação, uma vez que, possivelmente, contribuem com a melhoria dos resultados das consultas submetidas à aplicação.

4. IMPLEMENTAÇÃO E EXPERIMENTOS

A abordagem para busca de fontes de dados interligados da Web proposta neste trabalho foi implementada como módulo de uma ferramenta, denominada *RDFilter*[6], que está sendo desenvolvida como parte de uma dissertação de Mestrado. Esta ferramenta tem o objetivo de auxiliar desenvolvedores a encontrar fontes de dados relevantes durante o processo de construção de aplicações de dados interligados sobre um domínio específico [5].

O *RDFilter* envolve duas tarefas principais: i) encontrar fontes de dados que possam responder à consultas da aplicação e ii) escolher as fontes de dados mais relevantes dentre as encontradas. Para a primeira tarefa, onde temos uma seleção de potenciais fontes candidatas, é utilizada a abordagem apresentada neste artigo. Enquanto na tarefa seguinte, as fontes de dados detectadas são classificadas de acordo com o *feedback* do usuário sobre os resultados das consultas, por meio de medidas de *precision* e *recall*, como descrito em [5].

Para o desenvolvimento do módulo para busca de fontes, foram consideradas diversas ferramentas e serviços já existentes. Dentre eles, destacamos a API *Jena*[7], que permite trabalhar com a linguagem de programação Java e com tecnologias semânticas como RDF e SPARQL. Como Web *crawler* semântico, adotamos o *LDSpider*[8]. Como repositório RDF, para armazenamento de dados extraídos pelo *crawler*, foi usado o *Jena TDB*[9], e por fim, para o acesso a esses dados por meio de consultas SPARQL através de um *endpoint* local, foi utilizado o servidor RDF *Joseki*[10].

Para a avaliação da abordagem proposta, foram realizados experimentos sobre o domínio de dados bibliográficos. O objetivo dos testes executados foi o de analisar as fontes de dados encontradas, a fim de identificar se tais fontes fornecem

[6]http://code.google.com/p/rdfilter/
[7]http://jena.apache.org/
[8]http://code.google.com/p/ldspider/
[9]http://jena.apache.org/documentation/tdb/
[10]http://www.joseki.org/

informações semelhantes e/ou complementares à fonte de dados considerada como fonte base para a execução das consultas. Na Tabela 1, são apresentados dados de um dos experimentos realizados.

Tabela 1. Dados de Experimento sobre Publicações

Fonte de dados base	*DBLP RKBExplorer*[11]
# consultas	15
# recursos relevantes selecionados	5
Fontes de dados retornadas	*ACM*[12]
	CiteSeer[13]
	DBLP L3S[14]

Neste experimento, um conjunto de consultas SPARQL foi construído considerando a fonte de dados *DBLP RKBExplorer* como fonte base. Este conjunto é constituído por 15 consultas que extraem informações sobre autores, artigos, conferências, revistas científicas, entre outros. As consultas foram fornecidas como entrada para o módulo de seleção de fontes, que selecionou os 5 recursos mais frequentes e, a partir destes, realizou uma busca na Web. Como resultado final foram obtidas URIs dos SPARQL *endpoints* de três novas fontes: *ACM*, *CiteSeer* e *DBLP L3S*. A análise destas fontes mostrou que, de fato, elas armazenam dados em comum com os da fonte base, o que indica que estas fontes podem contribuir com a melhoria dos resultados das consultas da aplicação.

5. TRABALHOS RELACIONADOS

Nesta seção são apresentadas brevemente algumas pesquisas relacionadas ao nosso trabalho. Em [3] uma abordagem é descrita para identificar fontes relevantes para interligação de dados em novas fontes publicadas. Esta abordagem envolve dois principais passos: (i) busca por recursos relevantes sobre índices semânticos, utilizando palavras-chave associadas às novas fontes e (ii) filtro de fontes irrelevantes utilizando técnicas de *matching* de ontologias. Este trabalho se difere do presente, entre outros aspectos, por sua abordagem focar na detecção de fontes relevantes para interligação, ao invés da execução de consultas.

Outro trabalho relacionado é apresentado em [8], o qual propõe uma abordagem para guiar a adição de novas fontes em um sistema de integração de dados baseado em busca por palavras-chave. Este processo permite a construção de um grafo a partir das fontes e seus relacionamentos, sobre o qual é realizada uma busca que retorna os resultados mais relevantes para o usuário.

6. CONCLUSÃO

Considerando o crescente volume de dados que estão sendo publicados na Web seguindo os padrões de *Linked Data*, se torna cada vez mais difícil detectar fontes que sejam relevantes para uma dada aplicação. Portanto, ter uma solução que ajude a identificar boas fontes de dados a serem usadas como entrada para um sistema se torna crucial.

Neste artigo, apresentamos uma abordagem para buscar entre as diversas fontes de dados interligados da Web, fontes que possivelmente vão contribuir com os resultados de consultas de determinada aplicação, utilizando os requisitos dos usuários expressos nas próprias consultas como meio para se realizar este busca.

Como trabalhos futuros, gostaríamos de destacar algumas direções:

(i) Experimentos adicionais para avaliação mais precisa da abordagem: planejamento de mais experimentos tanto sob o domínio de Publicações, assim como outros domínios.

(ii) Aperfeiçoamento do processo de busca de fontes: avaliação de uso de índices semânticos [3] para detecção de fontes de dados candidatas. Também será investigado o uso de descrições VoiD[15] para a melhoria da identificação de fontes de dados relevantes.

(iii) Estudo de caso em integração de dados governamentais: integração de um módulo de busca em um sistema de recomendação de fontes de dados interligados no domínio de dados abertos do governo brasileiro.

7. AGRADECIMENTOS

A FACEPE por amparar esta pesquisa e a todos que contribuíram para o desenvolvimento e avaliação da abordagem proposta neste trabalho.

8. REFERÊNCIAS

[1] Bizer, C., Heath, T., Berners-Lee, T. 2009. Linked data - the story so far. In *Proceedings of the International Journal on Semantic Web and Information Systems*, 5(3), 1-22.

[2] Das Sarma, A., Dong, X. L., Halevy, A. 2011. Data Integration with dependent sources. In *Proceedings of the 14th International Conference on Extending Database Technology*, EDBT/ICDT 2011, New York, NY, USA.

[3] Nikolov, A., d'Aquin, M. 2011. Identifying Relevant Sources for Data Linking using a Semantic Web Index. In *Proceedings of the Linked Data on the Web*, LDOW 2011, Hyderabad, India.

[4] Pérez, J., Arenas, M., Gutierrez, C. 2009. Semantics and complexity of SPARQL. In *Proceedings of the ACM Transactions on Database Systems*, TODS 2009, Nova York, NY, USA, 34(3).

[5] Oliveira, H. R., Tavares, A. T., Lóscio, B. F. 2012. Feedback-based Data Set Recommendation for Building Linked Data Applications. In *Proceedings of the International Conference on Semantic Systems*, I-SEMANTICS 2012, Graz, Austria.

[6] Tavares, A. T., Oliveira, H. R., Lóscio, B. F. 2012. RDFMat – Um serviço para criação de repositórios de dados RDF a partir de *crawling* na Web de dados. In *I Escola Regional de Informática de Pernambuco*, 2012, Recife, Brasil.

[7] Castillo, Carlos. 2005. Effective Web Crawling. In *ACM SIGIR Forum, Vol.39*, Nova York, NY, USA.

[8] Talukdar, P. P., Ives, Z. G., Pereira, F. 2010. Automatically incorporating new sources in keyword search-based data integration, In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD 2010, Indianapolis, Indiana, USA, 2010, 387-398.

[11] http://dblp.rkbexplorer.com
[12] http://acm.rkbexplorer.com
[13] http://citeseer.rkbexplorer.com
[14] http://dblp.l3s.de/d2r
[15] http://semanticweb.org/wiki/VoID

OLA – Objeto Lúdico de Aprendizagem para pessoas com deficiência visual. Um estudo de caso: formas geométricas

Andreia C. G. Machion Queliane S. Oliveira Suelen S. Bastos Tamiris Mucci

FATEC Carapicuíba

Avenida Francisco Pignatari, 650 – Vila Gustavo Correa, Carapicuíba – SP. CEP: 06310 390

55 11 997312602 andreia.machion@fatec.sp.gov.br

55 11 41841570 queli.so@hotmail.com

55 11 967122209 suelenbastos@yahoo.com.br

55 11 36914743 tamucci3@hotmail.com

ABSTRACT

This article introduces a specific type of learning object, named OLA – Playful Learning Object (Objeto Lúdico de Aprendizagem), the main goal of such a kind of object is to support a lighthearted teaching-learning process. Also it is meant to integrate regular students with disabled ones, for instance with vision impairment. In order to show the complexity to develop OLAs, a case study is presented. The referred subject is common geometric shapes for children. Its creation was motivated, largely, by digital, social and educational inclusion incentive to people with disabilities, also by the increasing use of technology on education. Each OLA must be designed so each learner can develop his/her own knowledge, by using the technology to support the activities. It's important to reinforce the main premise for OLA development: to integrate different students in learning process to promote digital and social inclusion.

Categories and Subject Descriptors

K.3.1 [**Computer Uses in Education**]: Collaborative learning, computer-assisted instruction.

General Terms

Design, Experimentation, Human Factors

Keywords

Learning object, disabled students, interactivity, inclusion, technology in education.

1. INTRODUÇÃO

Há muito tempo ouve-se falar de inclusão social, integração entre alunos com deficiência e regulares. Entendem-se como regulares, aqueles alunos sem deficiência. Diversas iniciativas são realizadas periodicamente com o intuito de promover o cumprimento dos direitos já previstos em lei, tais como acesso à educação e à saúde entre outros.

Em uma pesquisa realizada pelo INEP, no ano de 2009 aproximadamente 639.781 alunos se matricularam na educação especial, e no ano passado esse número passou para 702.603. Desse total, aproximadamente 69% dos alunos estavam matriculados em salas comuns de ensino regular, correspondendo a 484.332, os demais alunos cursavam ensino exclusivo em escolas especializadas ou em classes especiais [1]. Esses índices mostram a crescente procura de alunos com deficiência por classes regulares, evidenciando a necessidade de políticas de inclusão social, o que está diretamente ligado à educação.

A partir desse cenário, surgem questões importantes que devem ser respondidas: Existem materiais acessíveis capazes de suprir as necessidades desse público? Por exemplo, materiais didáticos ricos em cores e gráficos são muito úteis para alunos videntes, mas e deficientes visuais? Como tirariam proveito desses?

Sendo assim, é preciso desenvolver materiais e atividades que supram as necessidades gerais de todo indivíduo sem distinção ou exclusão de qualquer uma das partes. Uma ferramenta que pode vir a ajudar são os Objetos de Aprendizagem (OAs), por serem bastante flexíveis e permitirem o uso de diversas mídias.

Um OA pode ser um texto, uma imagem, um mapa, uma animação, um vídeo, ou até mesmo um *software* completo, um OA é um recurso utilizado como suporte ao ensino, seja ele digital ou não. Em virtude de sua flexibilidade os objetos de aprendizagem estão ganhando cada vez mais espaço como recursos pedagógicos, pois além de auxiliarem o ensino, promovem a interação do aluno com o assunto a ser estudado, tornando o estudante um agente ativo no processo de aprendizagem.

Entretanto, para que um OA possa exercer seu propósito, é necessário que o professor tenha um objetivo muito claro do seu uso, pois como mencionado: "Não há fórmula sobre como escolher o melhor objeto e fazer uma aula a partir dele, já que o que fará dele o melhor recurso é justamente a forma como será utilizado" [2].

Outro ponto a ser considerado é o uso do lúdico na aprendizagem, o que pode favorecer o ensino, pois ao introduzir atividades lúdicas em sala de aula, o estudante pode sentir-se motivado para a construção do seu próprio conhecimento de forma ativa.

2. OBJETOS DE APRENDIZAGEM

Um objeto de aprendizagem é: "qualquer recurso de aprendizagem digital que possa ser utilizado em vários contextos instrucionais para dar suporte á aprendizagem" e que as principais características de um OA que justificam seu desenvolvimento são [3]:

- Acessibilidade: disponibilidade para qualquer um em qualquer tempo;
- Modularidade: permite a sua utilização em contextos diferentes;

- Reusabilidade: característica que é consequência da anterior;

- Durabilidade: como ele pode ser usado em contextos diferentes ele vai ser útil por muito tempo;

- Interoperabilidade: necessidade de que os OAs possam ser utilizados em qualquer plataforma tecnológica.

Para que um OA possa ser utilizado de forma abrangente, em ambientes computacionais e aproveitando-se todas as características citadas anteriormente, eles são descritos em metadados, que são como um catálogo relacionado a cada OA.

3. A DEFICIÊNCIA VISUAL

Nos dias atuais, fala-se muito sobre necessidades especiais, tanto no que se refere à acessibilidade e inclusão social das pessoas com deficiência, quanto à sua educação.

Por outro lado, é notório que os conteúdos escolares a cada dia vêm ganhando mais cores, gráficos, imagens e símbolos privilegiando os indivíduos de boa capacidade visual.

Então como prover recursos pedagógicos que possuam elementos diferenciados de acordo com as limitações de pessoas com deficiência visual?

Para tratar de um questionamento desse tipo, é necessário entender primeiro os conceitos que definem uma pessoa com deficiência.

Segundo o 3° artigo do decreto nº 3.298 de 20 de dezembro de 1999 tem-se a seguinte definição:

> Pessoa portadora de deficiência é aquela que apresenta de forma permanente perdas ou anormalidades de sua estrutura ou função psicológica, fisiológica ou anatômica, que gerem incapacidade para o desempenho de atividade, dentro do padrão considerado normal para o ser humano [4].

Podemos encontrar ainda pensamentos como os de Alberto David de Araújo que em sua tese de Doutorado "A proteção Constitucional das Pessoas Portadoras de Deficiência" propõe a seguinte definição:

> O que define a pessoa portadora de deficiência não é a falta de um membro nem a visão ou audição reduzidas. O que caracteriza a pessoa portadora de deficiência é a dificuldade de se relacionar, de se integrar na sociedade. O grau de dificuldade para a integração social definirá quem é ou não portador de deficiência [5].

É possível perceber nessas afirmativas duas visões para a questão deficiência. Porém, ambas remetem à conclusão de que a pessoa com deficiência é aquela que necessita de cuidados e adaptações especiais, uma vez que o que a caracteriza é a perda ou anormalidade da estrutura ou função psicológica, fisiológica ou anatômica.

Em características gerais a pessoa com deficiência visual é aquela que possui perda total ou parcial da capacidade visual, seja por questões hereditárias ou congênitas. Essa diminuição da resposta visual classifica este tipo de pessoa em dois grupos: os cegos e pessoas com visão subnormal.

Hoje em dia, as adaptações para a pessoa com deficiência visual têm ganhado cada vez mais força. Já é possível encontrar desde adaptações de *software*s, games ou até metodologias de ensino com características especiais que se adéquam às limitações individuais de cada pessoa com deficiência.

4. OBJETOS DE APRENDIZAGEM PARA PESSOAS COM DEFICIÊNCIA VISUAL

Já é possível encontrar exemplos de OAs para pessoas com deficiência visual, para os quais alguns estudos determinam que além das premissas gerais de desenvolvimento de OA, há requisitos específicos para abordagem sobre este tipo de público [6]:

- A formação de conceitos depende do íntimo contato da criança com as coisas do mundo; materiais concretos são importantes na formação de conceitos;

- Tal como qualquer criança, a criança com deficiência visual necessita de motivação para a aprendizagem;

- O manuseio de diferentes materiais possibilita o treinamento da percepção tátil, facilitando a discriminação de detalhes e suscitando a realização de movimentos delicados com os dedos.

Segundo FERRONATO [7] para os deficientes com perda total de visão (cegos) estudar determinadas áreas de conhecimento é razoavelmente fácil por haver no mercado materiais já adaptáveis, porém quando se trata do ensino da matemática, em especial os tópicos de álgebra e geometria (que necessitam de recursos concretos) a ideia de ensinar pode ser considerada inconcebível frente à ausência (carência) de visão que essas pessoas possuem.

Alguns materiais no mercado utilizados para o ensino matemático para deficientes visuais são:

- Ábaco e soroban – auxiliam no aprendizado das quatro operações matemáticas;

- Multiplano – auxilia no desenvolvimento de formas geométricas.

Diante da situação exposta, percebe-se a oportunidade para o desenvolvimento de ferramentas tecnológicas que auxiliem o aprendizado para este público. Os *software*s leitores de tela já auxiliam na interação do aluno com o computador, porém isso não basta. Para motivar o aprendizado é necessário o desenvolvimento de recursos pedagógicos, identificando suas necessidades e limitações de maneira que eles se sintam envolvidos. O grande desafio é a integração social destes indivíduos em sala de aula com recursos pedagógicos apropriados, ou seja, promover o aprendizado de forma homogênea para alunos regulares e deficientes visuais.

5. ATIVIDADES LÚDICAS

A educação pode ser caracterizada pela troca de aprendizado e ensino em busca de geração de conhecimento. O processo de ensino aprendizagem geralmente leva o estudante a ser tratado como agente passivo. No entanto, quando se procura colocar a educação em prática de forma eficiente, é recomendado levar em conta o estudante como agente ativo, isto é, um ser capaz de construir seu próprio conhecimento.

Para tanto, pode-se fazer uso efetivo de atividades práticas, incluindo lúdicas. Segundo SÁ [8],

> [...] viver ludicamente significa uma forma de intervenção no mundo, indica que não apenas estamos inseridos no mundo mas, sobretudo, que somos ele. Logo, conhecimento, prática e

> reflexão são as nossas ferramentas para exercermos um protagonismo lúdico ativo.

A atividade lúdica pode ser representada por um jogo ou uma brincadeira. A prática do jogo ou brincadeira pode remeter, em um primeiro momento, a qualquer atividade sem grandes pretensões ou obrigação de chegar a um resultado, uma espécie de entretenimento. No entanto, ao trazer essas atividades para a sala de aula é possível extrair delas muito mais do que apenas uma maneira de diversão.

> [...] antes de qualquer distração, as brincadeiras são experiências de vida para a criança [...] A brincadeira é o que existe de mais sério e universal na linguagem infantil, pois quando as crianças brincam juntas, as barreiras culturais, sociais e de linguagem tendem a desaparecer [8].

Sendo assim, o lúdico pode ser o parceiro do professor, e a partir dessa ideia, ele pode prover para seus alunos condições para que expressem seu mundo ou conhecimento de formas diferentes, por exemplo, assumindo sua corporeidade [9] e o uso da tecnologia tem servido de apoio para a inserção do lúdico em sala de aula, por exemplo, por meio de jogos de computador.

6. OBJETOS LÚDICOS DE APRENDIZAGEM

Conforme visto anteriormente, a integração da tecnologia com o lúdico pode promover um aprendizado efetivo. Por sua vez, se esses dois recursos puderem ser utilizados também para promover a integração entre alunos em sala de aula, sejam eles regulares ou com deficiência, melhor será o resultado obtido.

Definimos assim, o OLA – Objeto Lúdico de Aprendizagem – um tipo específico de OA que agrega tecnologias e recursos lúdicos para integrar alunos regulares e crianças com deficiência.

7. UM ESTUDO DE CASO: FORMAS GEOMÉTRICAS

Para este estudo, foi criado um conjunto de OLAs que abrangem o ensino de formas geométricas básicas. Esses objetos integram crianças regulares e crianças com deficiência visual. São eles:

- OLA – Ensinando formas geométricas – Linhas (*software*)
- OLA – Ensinando formas geométricas – Triângulo equilátero (*software*)
- OLA – Ensinando formas geométricas – Quadrado (*software*)
- OLA – Ensinando formas geométricas – Circunferência (*software*)
- OLA – Brincando com as formas geométricas – Formas concretas

Os OLAs *software*/digitais foram desenvolvidos com o programa de computador Adobe® Flash®, já as formas concretas foram desenvolvidas com papel emborrachado, também conhecido como EVA (Etil Vinil Acetato).

Além disso, para que os OAs sejam facilmente acessados por qualquer um, foi criado um website (Figura 1) que utiliza recursos comuns de acessibilidade para pessoas com deficiência visual. Também, foram acrescentadas instruções tanto para o professor quanto para o aluno que irão utilizá-lo como ferramenta dentro do processo ensino-aprendizagem.

Figura 1. Página Inicial do Website OLA.

Como o objetivo principal desses OLAs é a integração, tanto o *software* quanto as formas concretas utilizam recursos não só de sons e navegabilidade diferenciados, mas também cores bem realçadas para estimular alunos regulares e com deficiência visual parcial. O lúdico também foi explorado. Eles podem ser usados no ensino de algumas formas geométricas básicas para crianças.

7.1 Os Sons

A definição do som para cada forma geométrica foi feita considerando as percepções que um som pode provocar ao ser ouvido. O primeiro passo foi utilizar as definições sobre formas geométricas (quadrado, triângulo e circunferência) que determinam que tais formas são constituídas por linhas [10]. Portanto, o primeiro desafio foi pensar em uma maneira que uma pessoa com deficiência visual pudesse acompanhar como seria desenhada cada linha (direção e sentido). Definiu-se, então que cada linha seria representada por um som diferente que intuitivamente favorecesse suas características.

A segunda questão a ser respondida foi: Como uma pessoa com deficiência visual pode identificar onde e como está sendo desenhada cada linha.

Foi definido então que toda forma seria desenhada sempre a partir do canto inferior esquerdo seguindo, portanto, nas direções ascendente e da esquerda para direita e depois começaria a ser fechada nas direções descendente e da direita para a esquerda.

Além dos sons que representam as linhas, foi necessário pensar em sons de fundo que despertassem o interesse da pessoa com deficiência visual e a mantivesse motivada durante a apresentação das formas geométricas. Para isso, foram utilizados sons dinâmicos e divertidos para a interação com o usuário e melodias adequadas como som de fundo.

7.2 Definições de cores e tamanhos

Outro recurso do *software* é o contraste de cores, para que as crianças regulares e também as crianças com deficiência visual parcial possam identificar cada uma das formas também visualmente e aprenderem brincando juntas. Por exemplo, o triângulo (Figura 2) é formado pelas linhas:

- diagonal ascendente: cor amarela;
- diagonal descendente: roxa; e
- horizontal: laranja.

Figura 2. Representação do Triângulo.

É importante ressaltar que as cores são colocadas sobre fundo preto, para reforçar o contraste.

7.3 Implementação dos OLAs

Primeiramente, foram definidos os roteiros para os OLAs *software*, seguindo a seguinte estrutura:

- Apresentação: OLA e seu tema são introduzidos;
- Contextualização: nesta etapa, mostram-se objetos comuns ao cotidiano do aluno. Por exemplo, a circunferência que pode ser encontrada na bola ou num anel;
- Conceitualização: momento em que são apresentadas as cores e sons das linhas isoladamente ou das linhas que compõem as formas geométricas;
- Validação do aprendizado: composta por perguntas e respostas;
- Despedida.

O OLA Brincando com as Formas Geométricas, composto por elementos concretos, foi feito em papel emborrachado. Ele é composto por três diferentes contextos, são eles: a casa, o carro e o trem.

Nesses contextos, há formas geométricas, no caso do carro, as janelas e as rodas. E algumas destas formas geométricas são feitas para encaixar no contexto e são compostas por linhas que possuem as mesmas cores definidas e utilizadas nos outros OLAs.

7.4 Testes

O conjunto de OLAs desenvolvido foi submetido a um conjunto de testes iniciais. Eles foram validados por profissionais da área da educação, do ensino da matemática e também por uma estudante do ensino superior com deficiência visual (embora os objetos destinem-se a crianças, esse teste serviu para que fossem feitos ajustes na sua usabilidade).

Os testes com as especialistas apontaram a relevância de materiais desse tipo e a sua contribuição num ambiente de aprendizagem. Elas já vislumbraram uma série de atividades em que eles seriam úteis.

A aluna com deficiência visual conseguiu utilizar os OLAs sem necessidade de qualquer ajuda (o que era esperado pela sua experiência). Além disso, ela identificou todos os elementos pelo som. Vale ressaltar que os testes foram efetuados utilizando-se a ferramenta NVDA, um *software* leitor de tela.

8. CONSIDERAÇÕES FINAIS

A inclusão social é um grande desafio da atualidade. Integrar estudantes com deficiência e regulares pode ser um caminho e a tecnologia pode contribuir fortemente para esse fim. Além disso, o uso do lúdico pode favorecer situações interessantes de aprendizagem. O uso de OAs vem crescendo, o que mostra que esta é uma ferramenta bastante adequada dentro do processo ensino-aprendizagem. Sendo assim, a construção de OAs específicos tem papel fundamental. Daí os OLAs, que agregam as tecnologias e o lúdico para que seja possível atender tais necessidades. Os primeiros testes mostraram a sua viabilidade. O próximo passo deve ser a validação com classes características.

9. REFERENCIAS

[1] Brasil. Cresce a inclusão em salas de aula. 2011. Disponível em: <http://www.brasil.gov.br/noticias/arquivos/2011/04/18/cresce-inclusao-de-deficientes-em-sala-comum>. Acesso em: 23 de Dez de 2011.

[2] Webeduc. Compreendendo OAs. [2010?]. Disponível em: <http://webeduc.mec.gov.br/linuxeducacional/curso_le/modulo4_4_1.html>. Acesso em: 12 jan. 2012.

[3] Willey, David A. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. 2000. Disponível em: <http://penta3.ufrgs.br/objetosaprendizagem/>. Acesso em: 16 nov. 2011.

[4] Brasil. Decreto n. 3.298 (20/12/1999). Regulamenta a Lei 7.853, dispõe sobre a Política Nacional para a Integração da Pessoa Portadora de Deficiência, consolida as normas de proteção e dá outras providências. Lex: Coletânea de Legislação e Jurisprudência, Brasília. 1999.

[5] Araújo, Luís Alberto David. A proteção Constitucional das Pessoas Portadoras de Deficiência. São Paulo: PUCSP, 1992. Tese (Doutorado), São Paulo, 1992.

[6] Dias (2010) apud Cerqueira, Jonir Bechara; Ferreira, Elise de Melo Borba. Recursos didáticos da educação especial. Benjamin Constant. Rio de Janeiro. n. 5, 1996.

[7] Ferronato, Rubens. A construção de instrumentos de inclusão no ensino da matemática. Dissertação de mestrado em Engenharia de Produção. Universidade Federal de Santa Catarina. 2002.

[8] Sá, N. M. C. Brincadeira de adulto. Revista Vida Simples, Revista Vida Simples, p. 48 – 49, 2008.

[9] Araújo, S. C. Brincar É Preciso: Considerações Sobre A Atividade Lúdica Da Criança Deficiente Visual. Revista de Terapia Ocupacional da Baiana, Salvador, V. 2, p.17-22, 2005.

[10] Jakubo, J. J. et al. Matemática na medida certa. 5a série: Ensino fundamental. 6..ed. São Paulo: Scipione, 2001.

Multimodal Interaction System for Disabled People

Use Case: Playback of multimedia content

Sebastián Sastoque H.
Universidad Militar Nueva Granada
Bogotá, Colombia

Soraya Colina
Clinical Phonoaudiologist and Researcher
Bogotá, Colombia

Marcela Iregui
Universidad Militar Nueva Granada
Bogotá, Colombia

ABSTRACT

In this paper, a multimodal interaction system for people with disabilities is presented. The system combines four different interaction techniques in a single graphical user interface for playback multimedia content. The combination of different interaction modalities aims to expand the number of potential users, by allowing the selection of appropriate interactions that fulfil their specific needs. In this work, we validate, for a study case, the idea that different methods of interaction allow a higher accessibility, this way, individuals experiencing permanent or temporary disabilities could benefit from technology applications.

Keywords

Human Computer Interaction, Interaction Techniques, Multimedia, Multimodal Interaction, Augmentative and Alternative Communication (AAC), Accessibility.

1. INTRODUCTION

People with disabilities face numerous daily challenges. Unfortunately the use of computers for these individuals is one of them. Although computers are becoming more advanced, people with disabilities encounter several problems that limit their access to technology. For a long time getting computers to be accessible for people with certain disabilities, was achieved. In many cases the approach was catering to the specific needs of a particular user. In that case, Human Computer Interaction practices focusing on people with disabilities, were made initially from handmade adaptations of surprising quality and usefulness. The disadvantage was being unable to perform effectively with others disabilities.

The mouse and keyboard, traditional devices for computer interaction, are useless for people with disabilities. For instance, Parkinson patients can hardly use a mouse because of the required precision to achieve any task [4]. Likewise, graphical user interfaces are usually very complicated for people with cognitive limitations, such as patients with aphasia who present problems interpreting complex instructions or processing the information in parallel. Interfaces with too many menus and buttons causes people with disabilities to refuse the use of computers because they get confused[8].

As Dumas et al [5] defined, multimodal systems interpret information from combining user input modes. They offer alternatives for human machine interaction that involve diverse and underserved users groups, achieving universal access. Therefore, different research projects, concerning multimodal interaction for people with disabilities, have been developed. The authors of *MailSaw* and *NaviSaw*[7] present a system for users with visual disabilities to browse the Internet, using voice synthesis, speech recognition and mouse commands to interact. In the project *BlindAid*[9], users interact with a haptic device in a 3D interface, through various unknown spaces, in order to carry out therapeutic activities for blind people rehabilitation. There are also several applications of unimodal interaction, such as mouse control from head movement[6], eye gaze[11] and hand gestures[12]. Some applications use virtual keyboards, speech recognition systems, communication boards and screen readers[1].

The applications referred above are based on the traditional model of HCI practices for people with disabilities, which focus on satisfying the needs of a specific group of users, limiting its use to a restricted group of users. For this reason, in this paper a multimodal interaction system is proposed, which allows to manipulate an application interface in different ways. It could be configured according to each user's preferences. The four methods of interaction implemented are: detection of head movements, image detection and recognition, trackball manipulation and command selection with an infrared pen (IRPen) in a GUI projection. The main goal of this work is to show, with a study case, that multimodal interaction improves applications usability, making them broadly accessible. The use case selected for validation consists of an application for playback multimedia content using the four methods of interaction mentioned above.

This paper is organized as follows: next section deals with the description of the use case. The third section focuses on the explanation of the four interaction techniques implemented, and their methods. The fourth section describes the proposed system with each of its components and their relationship. Finally, the last sections correspond to discussion of the results and conclusions of the work.

2. USE CASE

Multimedia technologies provide fuller entertainment experiences that generally are not accessible to disabled people. That is why, the selected use case consists on a playback multimedia content application, considering that many current players have usability limitations. This work is focused on two issues: The first one, concerning the design and implementation of accessible interaction techniques for people with sensory, motor or cognitive disabilities. The second one, concerning the software application design.

3. INTERACTION TECHNIQUES

The interaction methods implemented in this work, i.e. detection of head movements, image detection and recognition, trackball manipulation and command selection with an IR pen in a GUI projection, were designed based on the requirements of users with reduced motor skills and difficulties to follow complex instructions.

3.1 Head Movements Detection (HMD)

For users with reduced motor skills (i.e. quadriplegics), detection of head movements provides the possibility to control a software application. For the implemented use case, the instructions are identified by tracking the direction of the head movements, as shown in figure 1. That is, horizontal movement of the user's head is used to navigate between functions and vertical movements to perform a selected action.

The method implemented for motion detection is based on the *Frame Difference*[10] algorithm. Once the frame differences are performed, the movement areas are detected and the centers of these two bounding boxes are calculated to find the angle between these two points. According to the angle value the movement is associated with a direction between up, down, left or right. This process is repeated for the next 15 frames (1 second). The final decision is determined by calculating the more frequent direction.

3.2 Image Detection and Recognition (IDR)

The use of symbols or pictograms is broadly used in Augmentative and Alternative Communication (AAC) systems, specially for people with communication disorders (i.e. aphasia, autism) [2]. That is to say, images can be easily associated with an idea and then with an instruction.

Fifteen (15) standard symbols were stamped in plastic cards, each one related with an action on the playback multimedia system. Once a symbol card is placed in front of the camera, the system recognizes the symbol printed and performs an action according to the associated instruction, as shown in figure 2.

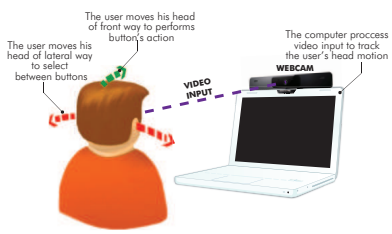


Figure 1: Head Movements Detection Interaction

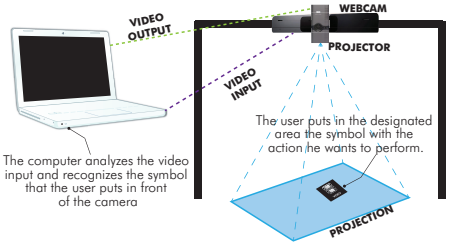


Figure 2: Image Detection and Recognition Interaction

When the video detects a symbol card the system proceeds to the recognition. The method used for image classification is based on the well known emphSpeeded Up Robust Features (SURF)[3] algorithm. In that method, feature extraction is performed by doing a scale-space representation of the image, followed by detection and description of the interest points. Secondly, the classification is performed by features matching using the nearest neighbor criteria.

3.3 Command selection with an IRPen (CSIR)

For some users it is difficult to use a mouse in order to interact with a graphical user interface, due to the precision required for that task (i.e Parkinson). The method proposed to overcome that limitation is the use of an infrared pen to select the action from a projection of the GUI in a work surface. This method requires a particular arrangement of devices consisting of a projector, an IRPen and a Wiimote disposed as shown in the figure 3. Once the IR pen touches the surface, it emits a signal which is detected by the Wiimote and then transmitted to the application via bluetooth. The IR pen coordinates are used to select the action from the GUI.

3.4 Trackball Manipulation (TM)

Some users, particularly those with visual limitations and motion reduced upper limbs endure problems with traditional interaction systems [4]. Interaction based on movement fits these kind of users (see figure 4).

This interaction is based on gesture characterization from the movements performed by the user in a trackball device. These gestures are grasped from the momentum and the mo-

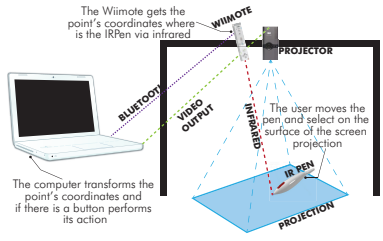


Figure 3: Command selection with an IRPen Interaction

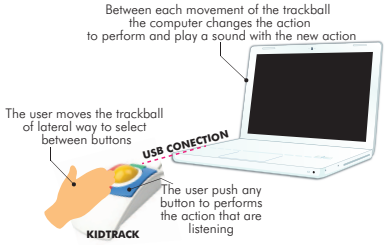


Figure 4: Trackball Manipulation Interaction

tion direction. The Euclidean distance between the starting point and the ending point is calculated to determine the displacement and the angle formed by these two points to establish the direction.

4. SYSTEM DESCRIPTION

The playback of multimedia content system, shown in figure 5, has a step by step choice where the final result is the reproduction of media. The system was designed to enhance the usability for people with limitations.

The system uses the same GUI for the different interaction modalities. The user first selects a content type and the *Interaction Analysis Module* interprets the user's input. Subsequently, the *Content Selection Module* returns the list of media for the selected type. To select the option, the Interaction Analysis Module interprets the user's action and finally, the *Content Playback Module* decodes the file and displays the content. The considered actions for this use case are playing, pausing and stopping the current content; looking for the next or previous content; and selection of music, text, images or video.

Summing up, the system is composed of three main modules, which are:

1. **Interaction Analysis Module:** The IAM applies algorithms and methods to grasp the user preferences from the different interaction possibilities and translates them to system instructions.
2. **Content Selection Module:** The CSM searches the indexed content locations for a specific content type.
3. **Content Playback Module:** The CPM opens and controls the media file. Mainly, it decodes the file and displays the content.
4. **Multimedia Content Indexes:**The structure of indexes of media files is organized in an XML file into four categories, one for each type of content (audio, video, text or image). Each indexed file in a category has the *ID*, *Title*, *Author*, *Duration* and *Location* attributes.

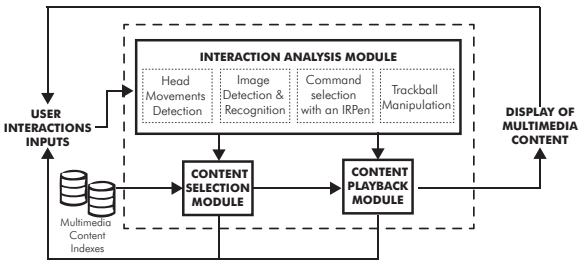


Figure 5: General description of the system

Figure 6: 55 years old woman with Broca's aphasia using the system

5. RESULTS

A case study of a 55 years old woman with Broca's aphasia due to stroke in the Sylvian fissure of the left middle cerebral was considered. This person has deficits in language production and comprehension, she has gross motor skills, and she had previous contact with technology therefore knew how to use a computer. Before the evaluation, an explanation on the using of the system was given to the patient (see figure 6).

The following activities were performed with an allowed maximum of five (5) attempts: Play a song, play a video, find an image of your home, play the second video from the list, display a text, navigate in a text, show the duration of a song, show a song artist and show the title of a song.

Each activity was performed by using isolated interactions, and using multimodal interaction (see figure 7). For each trial, the time undertaken for each activity was measured, from the moment when the system shows the options for choose until the person achieve the task. When the user failed to do the task, the time is measured again from the beginning in a new attempt. The results reported in table 1 is the time of the first attempt that the user achieve the task.

A comparison was made with certain players using the mouse for the interaction, for the same tasks and test conditions mentioned above, obtaining the results shown in table 2.

Figure 7: Use of the proposed interaction

Table 1: Comparison between Interaction techniques results (seconds to achieve a task)

Task	HMD	IDR	CSIR	TM	Multimodal
Play a song	23s	21s	20s	28s	20s
Play a video	42s	38s	39s	40s	37s
Find an image of your home	78s	67s	72s	69s	65s
Play the second video from the list	39s	37s	35s	40s	35s
Display a text	31s	28s	38s	32s	30s
Navigate in a text	21s	18s	23s	20s	15s

Table 2: Comparison between players (seconds to achieve a task)

Task	iTunes	Windows Media Player	VLC	Our System
Play a song	50s	57s	63s	20s
Play a video	48s	60s	53s	37s
Find an image of your home	Player doesn't have this function			65s
Play the second video from the list	45s	60s	Don't achieve	35s
Display a text	Player doesn't have this function			30s
Navigate in a text	Player doesn't have this function			15s
Show the duration of a song	10s	12s	8s	4s
Show a song artist	15s	22s	20s	3s
Show the title of a song	20s	17s	25s	6s

6. DISCUSSION

Analyzing the obtained results in the study case, it is straightforward to affirm that the proposed system is usable for people with disabilities related with aphasia. This is due to several facts:

- The way in which the options are sequentially presented, since the user must not process a high number of information to decide what content to play, although this can become rather uncomfortable and unnecessary for a user who doesn't have a cognitive disability.
- Each interaction evaluated in an isolated manner improves usability due to the implementation of accessible interfaces adapted to specific user disabilities..
- The possibility of having multimodal interfaces allows fast interactions, because the user has different alternatives to attain the control of the application. Unlike isolated interactions that provide easier interactions for some options but not for the whole set.
- The proposed system is adequate and efficient for users with disabilities compared to other systems of the same kind. This is confirmed by the shorter amount of times used to successfully utilize applications as reported by the proposed system. This, being a result of accessible interfaces and simpler graphical interfaces. Moreover the implemented application presents a use case richer in options because it is adapted to different media as text, music, video and images.

7. CONCLUSIONS

This work has demonstrated that an application, with user driven interfaces for disabled and multimodality, enhances the accessibility and as well as the user's experience. In doing so, people experiencing a permanent or temporary situation of disability could benefit from the use of entertainment technology.

The development validates for a study case the fact that multimodal techniques permit the appropriation of technology and could be considered to be a applied in software applications.

In order to have more facts allowing the generalization of the conclusions, further work must be accomplished to develop an extended quantity of use cases that enable the evaluation of user experience variables in different scenarios. Additionally, this work must be validated by people with different disabilities and the population must be extended for each condition.

8. ACKNOWLEDGEMENTS

This work has been funded by the Universidad Militar Nueva Granada in the framework of their PIC projects program.

We thanks Helioth Sánchez and Eduard Sierra for useful discussions and the comments provided during the development of this project.

9. REFERENCES

[1] J. Abascal and R. Moriyón. Tendencias en interacción persona computador, 2002.

[2] C. Basil, E. Soro-Camats, and C. Rosell. *Sistemas de signos y ayudas técnicas para la comunicación aumentativa y la Escritura*. Masson, 2004.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008.

[4] A. M. Cook, J. Polgar, and S. Hussey. *Cook and Hussey's assistive technologies: Principles and practice*. Mosby Elsevier, St. Louis, 2008.

[5] B. Dumas, D. Lalanne, and S. Oviatt. Multimodal interfaces: A survey of principles, models and frameworks. In *Human Machine Interaction*. Springer Berlin / Heidelberg, 2009.

[6] J. Gips, M. Betke, and P. Fleming. The camera mouse. In R. Press, editor, *Proceedings of RESNA 2000*, pages 98–100, 2000.

[7] J. González, F. Sánchez, A. Lozano, M. Macias, and F. Sanchez Herrera. Navisaw y mailsaw herramientas multimodales para el acceso a internet para usuarios con discapacidad visual. Technical report, Universidad de Extremadura, 2007.

[8] M. Iza. Tecnología computacional en afasia. *Revista de Psciología General y Aplicada*, 56(1):101–123, 2003.

[9] O. Lahav, D. Schloerb, S. Kumar, and M. Srinivasan. Blindaid: A learning environment for enabling people who are blind to explore and navigate through unknown real spaces. In *Virtual Rehabilitation*, 2008.

[10] P. Rivas Perea and M. Chacón Murguía. Evaluación de métodos de detección de movimiento. In *ITCH - ELECTRO 2005*, 2005.

[11] L. E. Sibert and R. J. Jacob. Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–288. ACM, 2000.

[12] X. Zabulis, H. Baltzakis, and A. Argyros. Vision-based hand gesture recognition for human-computer interaction. In *The Universal Access Handbook*. LEA, 2009.

Parte IV

Minicursos do Webmedia 2012

Prefácio

Este livro é uma coletânea de minicursos com resultados de atividades de pesquisa científica ou tecnológica. O conteúdo de cada minicursos foi fornecido pelos seus autores e apresentado durante o Simpósio Brasileiro de Bancos de Dados (SBBD 2012), o Simpósio Brasileiro de Sistemas Colaborativos (SBSC 2012) e o Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia 2012), de 15 a 18 de Outubro de 2012 em São Paulo/SP. Os minicursos, eventos de curta duração (4 horas), visam apresentar tópicos atuais de pesquisa e/ou tecnologia de interesse da comunidade de Sistemas Multimídia e Web. Pretende-se oferecer oportunidades de aprendizagem e de utilização dos novos conhecimentos nas áreas de atuação ou pesquisa dos participantes dos minicursos. Os minicursos selecionados foram submetidos à chamada pública, divulgada no site dos eventos, em emails e na lista eletrônica da SBC (Sociedade Brasileira de Computação). Foram recebidas diversas propostas de minicursos, as quais foram avaliadas pelo comitê científico de minicurso do WebMedia: Alessandra Alaniz Macedo (USP), Carlos Alberto Kamienski (UFABC), João Paulo Góis (UFABC), Itana Stiubiener (UFABC), Marcelo Zuffo (EPUSP), Romero Tori (EPUSP), Roseli Lopes (EPUSP), Tatiana Tavares (UFPB) e Thais Batista (UFRN). Quatro propostas foram selecionadas. Cada proposta de minicurso aceita é apresentada em um capítulo deste livro.

O Capítulo 1 do minicurso *Introdução ao Desenvolvimento Colaborativo de Regras SWRL com o SWRL Editor* apresenta uma introdução ao desenvolvimento de conjunto de regras em SWRL (Semantic Web Rule Language). A linguagem SWRL (um padrão do W3C) permite a combinação de regras e termos de ontologias (definidos em OWL - Web Ontology Language) para aumentar a expressividade de ambos. O capítulo apresenta uma breve introdução sobre web semântica, na qual é abordada a linguagem OWL, uma linguagem padrão do W3C para representar e compartilhar ontologias na Web. Em seguida, é feita uma introdução prática da linguagem SWRL com a construção de uma pequena ontologia com um conjunto de regras. Para a criação desse conjunto de regras é usado o SWRL Editor, um novo editor de regras SWRL integrado ao Web-Protégé (versão Web do editor de ontologias Protégé).

O Capítulo 2 do minicurso *Software as a Service: Desenvolvendo Aplicações Multi-Tenancy com Alto Grau de Reúso* apresenta software como serviço (SaaS) que representa um novo paradigma e um modelo de negócios onde as empresas não precisam comprar e manter sua própria infraestrutura de TI. Ao invés disso, elas adquirem um serviço de software de terceiros, obtendo assim consideráveis benefícios, principalmente no que tange a redução de custos de manutenção dessa infraestrutura. O objetivo desse minicurso é apresentar os principais conceitos relacionados à arquitetura multi-tenancy, uma das abordagem para implementação de Software como Serviço. Durante esse trabalho, são apresentadas as principais tecnologias associadas ao assunto e apresentar um exemplo prático da implementação de um aplicativo multi-tenancy utilizando o framework Grails e componentes reutilizáveis.

O Capítulo 3 do minicurso *Desafios em Cloud Computing: Armazenamento, Banco de Dados e Big Data* apresenta que com o aumento da velocidade de conectividade e evolução dos sistemas Web, começa a surgir os sistemas de Internet, que mais comumente são chamados de Computação nas Nuvens. Este termo designa uma plataforma de suporte a sistemas de software que provê aos seus usuários: gerenciamento, uso sob demanda, adequação às necessidades, racionalização do uso dos recursos e automação dos processos relacionados à criação de infra-estruturas. Neste contexto, surgem os sistemas de armazenamento de dados em nuvem, escalabilidade de banco de dados e busca e recuperação que hoje chamamos de BIGDATA. Este minicurso aborda estes temos exemplificando como implementar e utilizar tais plataformas.

O Capítulo 4 apresenta o texto do minicurso *Análise de Informações Contextuais através de Técnicas de Aprendizagem de Máquinas* que apresenta os conceitos introdutórios de técnicas de aprendizado de máquina como rede bayesiana, árvore de decisão e redes neurais e discuti como essas técnicas podem ser aplicadas na análise de informações contextuais. Neste minicurso são utilizadas as APIs da ferramenta Weka, desenvolvida na Universidade de Waikato na Nova Zelândia, que contempla, um conjunto de técnicas de aprendizagem de máquina implementado. Também são apresentados exemplos de uso de contexto na personalização e recomendação de conteúdo em ambientes Web, móveis e de TV Digital.

São Paulo, Outubro, 2012
Alessandra Alaniz Macedo (USP)
Itana Stiubiener (UFABC)

Capítulo

1

Introdução ao desenvolvimento colaborativo de regras SWRL com o SWRL Editor

João Paulo Orlando, Adriano Rívolli e Dilvan de Abreu Moreira
Dept. of Computer Science, ICMC Universidade de São Paulo – Campus de São Carlos, Caixa Postal 668, 13560-970, São Carlos-SP, Brazil
{orlando, rivolli, dilvan}@icmc.usp.br

Abstract

This chapter presents an introduction to the development of rule sets in SWRL (Semantic Web Rule Language). The SWRL language (a W3C standard) allows the combination of rules and ontology terms (defined in OWL Web Ontology Language) to increase the expressiveness of both. The chapter presents a brief introduction to the semantic web, in which we introduce the OWL language, a W3C standard for representing and sharing ontologies over the Web. After that, the chapter shows a hands-on introduction to SWRL where we build a small ontology with a rule set. This rule set will be created and tested using the SWRL Editor, a new SWRL rule editor integrated with Web-Protégé (the Web version of the Protégé ontology editor).

Resumo

Este capítulo apresenta uma introdução ao desenvolvimento de conjuntos de regras em SWRL (Semantic Web Rule Language). A linguagem SWRL (um padrão do W3C) permite a combinação de regras e termos de ontologias (definidos em OWL Web Ontology Language) para aumentar a expressividade de ambos. O capítulo apresenta uma breve introdução sobre web semântica, na qual será abordada a linguagem OWL, uma linguagem padrão do W3C para representar e compartilhar ontologias na Web. Em seguida, é feita uma introdução prática da linguagem SWRL com a construção de uma pequena ontologia com um conjunto de regras. Para a criação desse conjunto de regras, será usado o SWRL Editor, um novo editor de regras SWRL integrado ao Web-Protégé (versão Web do editor de ontologias Protégé).

1.1. Introdução

A Web Semântica é uma maneira de explorar a associação de significados explícitos aos conteúdos de documentos presentes na Web, para que esses possam ser processados diretamente ou indiretamente por máquinas [Berners-Lee e Fischetti 2008]. Para possibilitar esse processamento, os computadores necessitam ter acesso a coleções estruturadas de informações (dados e metadados) e a conjuntos de regras de inferência sobre esses conteúdos (que ajudem no processo de dedução automática) para que seja possível o raciocínio automatizado sobre os mesmos [Berners-Lee et all 2001].

A Web Semântica renovou e aumentou o interesse em sistemas baseados em regras e seu desenvolvimento [Zacharias 2008]. A SWRL é a linguagem padrão para regras da Web Semântica, muitas áreas de estudo na computação estão usando SWRL, entre elas podemos citar: Serviços sensíveis ao contexto [Wusheng et all 2011], gestão de energia em ambientes domésticos [Rossello-Busquet et all 2011], sistemas de *e-learning* [Vesin et all 2011], gerenciamento de SLA (*Service Level Agreement*) para serviços de IPTV (*Internet Protocol Television*) [Seo et all 2011], cálculos de redes de co-autoria em redes sociais [Ahmedi et all 2011], sensores em ambientes inteligentes [Sadoun et all 2011], etc. Na área da informática biomédica, é possível citar outras aplicações de regras SWRL: Atribuição de notas a tumores [Levy et all 2009] e para classificar fenótipos de Autismo [Hassanpour et all 2011]. Como pode ser visto muitas áreas tem usado regras SWRL para facilitar a modelagem de problemas.

1.2. Ontologia e a Web Semântica

1.2.1. Ontologia

O W3C [Heflin 2004] afirma que as ontologias são vistas como a tecnologia de consolidação para a construção da Web Semântica. O termo é emprestado da Filosofia, em que uma ontologia é um relato sistemático da existência [Gruber 1993].

[Studer et all 1998] define ontologia como uma especificação **formal** e **explícita** de uma **conceituação compartilhada**. **Conceituação** se refere a um modelo abstrato de algum fenômeno no mundo, identificando os conceitos relevantes daquele fenômeno. **Explícito** significa que os conceitos utilizados e as restrições sobre seu uso são explicitamente definidos. **Formal** refere-se ao fato de que a ontologia deve ser legível pelas máquinas. **Compartilhado** refere-se à noção de que uma ontologia captura o conhecimento consensual, isto é, não é privado de algum indivíduo, mas aceito por um grupo.

Discussões e estudos aprofundados sobre a definição e conceituação do termo ontologia podem ser encontrados em [Guarino and Giaretta 1995]; [Chandrasekaran et all 1999]; [Smith et all 2001]; [Almeida and Bax 2003] e [Corazzon 2010]. Mesmo sem um consenso sobre sua definição, ontologias compartilham características comuns e impulsionam o desenvolvimento de diversos trabalhos referentes a metodologias, ferramentas, linguagens e aplicações.

Uma ontologia é especificada por meio de componentes básicos que são as **classes**, **relações**, **axiomas** e **instâncias**, além de ser expressa por meio de uma linguagem de construção [Almeida and Bax 2003].

As **classes**, o foco da maioria das ontologias, são utilizadas para descrever os conceitos de um domínio, possibilitando a organização das classes em um sistema lógico e hierárquico contendo subclasses que representam conceitos mais específicos [Noy and McGuinness 2001].

As **relações** representam o tipo de interação entre os conceitos de um domínio e as propriedades presentes nas classes e indivíduos. Os termos *slot*, *role*, propriedade e até mesmo atributo podem ser empregados como sinônimo para as relações [Noy and McGuinness 2001] [Horridge et all 2004]. As relações podem ter características próprias como serem transitivas, simétricas, ou terem uma cardinalidade definida.

Os **axiomas** são utilizados para modelar regras assumidas como verdadeiras no domínio em questão, de modo que seja possível associar o relacionamento entre os indivíduos, além de fornecer características descritivas e lógicas para os conceitos. Para [Uschold and Grüninger 1996] os axiomas são especificados para definir a semântica e significado dos termos (classes e propriedades) e sugere que a fase de definição dos axiomas (especificação da ontologia) é a mais difícil na construção de ontologias.

Por fim, os **indivíduos**, ou instâncias das classes, são utilizados para representar elementos específicos, ou seja, os próprios dados, que juntamente com a definição de uma ontologia constituem a base de conhecimento [Noy and McGuinness 2001]. Os indivíduos representam objetos do domínio de interesse [Horridge et all 2004].

Os componentes básicos de uma ontologia são definidos por meio de uma linguagem de representação. [Horridge et all 2004] aponta que diferentes linguagens para ontologias proporcionam diferentes facilidades e recursos. Alguns exemplos de linguagens que se prestam à construção de ontologias são apresentados a seguir, na Tabela 1.1. Uma listagem mais completa de linguagens de representação de ontologias pode ser encontrada em [Su and Ilebrekke 2002] e [Almeida and Bax 2003].

Tabela 1.1 – Linguagens para construção de ontologias

Linguagens	Breve descrição
DAML	*DARP Agent Markup Language* (DAML) é precursora do DAM+OIL que originou a OWL. Sua sintaxe permite que o computador possa fazer as mesmas inferências simples que os seres humanos podem fazer [Ouellet and Ogbuji 2002].
OBO	*Open Biological Ontologies* (OBO) é uma linguagem de ontologia que tem sido frequentemente utilizada para a modelagem de ontologias nas áreas biomédicas. Faz uso de uma sintaxe textual simples que foi projetada para ser compacta, legível por humanos e fácil de ser processada [Golbreic et all 2007].
OIL	Ontology Interchange Language (OIL) é base para as linguagens da Web Semântica é precursora do DAM+OIL que originou o OWL. É uma linguagem baseada em frames juntamente com as características de lógica descritiva (*Description Logic* – DL) [Fensel et all 2001].
OWL	*Web Ontology Language* (OWL) projetada para atender a necessidade de uma linguagem de representação de ontologias para a Web. Utiliza RDF para a sintaxe e lógica descritiva para a definição de regras de inferência. É o padrão adotado no W3C como linguagem da Web Semântica [McGuinness and Harmelen 2004].
RDF	*Resource Description Framework* (RDF) é uma linguagem para definição de informação na Web. Descreve os dados (significado dos termos e conceitos) em um formato que máquinas podem processar. Utiliza o *eXtensible Markup Language* (XML) e *Uniform Resource Identifier* (URI) e conjuntos de triplas (sujeito, verbo e objeto) formam as sentenças elementares [Berners-Lee et all 2001].

Por fim, as ontologias são utilizadas para promover a interoperabilidade entre sistemas, ao representarem os dados compartilhados por diversas aplicações [Uschold

and Grüninger 2004]. Ontologias são amplamente utilizadas para fins diferentes e em diferentes comunidades.

1.2.2. Web Semântica

A Web Semântica, uma extensão da Web atual, é uma representação capaz de associar significados explícitos aos conteúdos dos documentos disponíveis na Internet, sendo que sua principal meta é possibilitar que programas processem e interpretem automaticamente esses documentos [Berners-Lee et all 2001]. Para Berners-Lee, a Web Semântica deve possibilitar que computadores sejam capazes de acessar dados estruturados e de definir regras de inferências, transformando grandes volumes de dados em informação.

Seu emprego é fortemente recomendado pelo W3C, que busca desenvolver padrões, arquiteturas de metadados e linguagens para ontologias que juntos possibilitem a integração e entendimento dos dados por computadores, agregando aos mesmos significados. Sua exploração é motivada pelo potencial que tem em transformar a Internet, vista hoje como um repositório de dados, em um repositório explícito de conhecimento, disponível tanto para pessoas como para máquinas. O papel do W3C no contexto da Web atual é o desenvolvimento de padrões, recomendações e orientações com o objetivo de levar a Web ao seu potencial máximo. Além dos avanços relacionados com as aplicações da Web, o W3C tem mobilizado grandes esforços e iniciativas para o desenvolvimento de uma Web para todos, em todos os dispositivos, baseada no conhecimento, com confiança e confiabilidade [Diniz and Cecconi 2008].

A adoção das tecnologias da Web Semântica e de ontologias na representação de dados, embora não tenha sido amplamente difundida e adotada até o momento [Shadbolt et all 2006], conta com o apoio de inúmeras empresas na divulgação e desenvolvimento de soluções que fazem o uso da Web Semântica [Feigenbaum et all 2007].

A tarefa de associar significados aos dados é possível pelo uso de tecnologias como RDF e OWL. O RDF utiliza XML e URI para proporcionar uma representação minimalista do conhecimento na Web e tem como característica principal ser simples. Por outro lado, a OWL é uma tecnologia complexa e voltada para a representação de objetos que requerem grande poder de expressividade. OWL usa RDF e possibilita a criação de ontologias para representação de conhecimento [Shadbolt et all 2006].

1.2.2.1. *RDF*

Resource Description Framework (RDF) é um esquema para a definição de informações na Web, provendo tecnologia para expressar o significado de termos e conceitos de modo que os computadores possam facilmente processá-los [Berners-Lee et all 2001]. Sua sintaxe pode ser definida com o uso de marcações XML e URI, empregadas para especificar entidades, conceitos, propriedades e relacionamentos. [Lassila and Swick 2004] define que RDF é a tecnologia base para o processamento de metadados, provendo assim interoperabilidade entre aplicações que trocam e processam informações na Web. Para os autores, um dos objetivos do RDF é possibilitar a criação de semântica para dados baseados em XML. Contudo, o principal objetivo do desenvolvimento do RDF é permitir a descrição de recursos de qualquer domínio específico.

Definida pelo W3C, o RDF organiza o conhecimento por meio da idéia de redes semânticas, e permite a representação de conceitos, taxonomia de conceitos e relações binárias [Almeida and Bax 2003]. Em um relato histórico, [Smith and Welty 2004] afirma que o RDF foi a primeira linguagem especificada pelo W3C para representar informações semânticas na Web.

O modelo de dados em RDF é uma forma neutra de representar as expressões utilizadas para atribuir significado aos dados. O modelo básico consiste em três tipos de objetos: recursos, propriedades e afirmações[1] que são chamadas respectivamente de sujeito, verbo e objeto, formando a tripla RDF [Berners-Lee et all 2001]. A codificação do conhecimento se concretiza em conjuntos de triplas, representadas graficamente na Figura 1.1.

Figura 1.1 – Tripla RDF [Klyne et all 2004].

O sujeito compreende qualquer coisa que pode ser expressa e descrita em RDF. Alguns exemplos de recursos são: um documento HTML; uma parte de uma página Web; um elemento XML específico; um conjunto de páginas; e objetos que não podem ser acessados diretamente na Web, como um livro impresso ou uma pessoa. Os recursos são sempre nomeados por URI, pois a expressividade deste recurso possibilita que qualquer entidade possa ter seu identificador [Lassila and Swick 2004].

O verbo consiste nas características, atributos ou relacionamentos utilizados para descrever um recurso. Cada propriedade tem um significado específico, define seus valores permitidos, os tipos de recursos que podem descrever, e sua relação com outras propriedades. O valor de uma propriedade atribuída a um recurso específico é o objeto. O valor da propriedade pode ser outro recurso, especificado por um URI, ou um valor literal (simples string ou outro tipo primitivo definido em XML). Tendo como base a tripla, a linguagem RDF possui uma sintaxe abstrata que pode ser expressa em XML, Notation 3 (N3) ou graficamente, de modo que a semântica formal é definida com base nesta sintaxe abstrata. Uma completa descrição desta sintaxe é apresentada em [Klyne et all 2004].

Uma poderosa extensão do RDF é o *RDF Schema*, usado para descrição de vocabulário utilizado para descrever os relacionamentos internos de propriedades e valores, sendo considerado um mecanismo de extensão semântica para o RDF [Brickley 2004]. O *RDF Schema* prove mecanismos para descrever grupos de recursos relacionados, além do relacionamento entre estes recursos. É possível utilizar o conceito de classes, indivíduos, propriedades e subpropriedades semelhantemente ao recurso de hierarquia de classes.

Para [Feigenbaum et all 2007], o RDF é o mais fundamental bloco de construção para a Web Semântica, pois, além de poder ser utilizado para criar dados semânticos, é também utilizado como base para as linguagens de ontologia da Web Semântica. Entretanto para [Staab et all 2001], os dados em RDF são fracamente interligados, de modo que a Web Semântica necessita de técnicas mais sofisticadas.

[1] Tradução livre do o termo *statement*

1.*2.2.2. OWL*

A linguagem de ontologia *Web Ontology Language* (OWL) é destinada para os que desejam grande expressividade na descrição dos objetos e seus relacionamentos [Shadbolt et all 2006]. O *RDF Schema* é uma linguagem ontológica considerada leve [Staab et all 2001], pois não contém a complexidade e riqueza de outras linguagens como OWL. Todavia, OWL usa as ligações RDF e trabalha uma camada acima do mesmo [Feigenbaum et all 2007].

O W3C publicou em 2004 a linguagem de marcação semântica OWL como recomendação para representar e compartilhar ontologias na Web [Smith and Welty 2004]. Contudo, OWL foi concebida de uma revisão da linguagem DAML+OIL incorporando as lições aprendidas durante o desenvolvimento e aplicação desta [McGuinness and Harmelen 2004].

A OWL 1 provê três sublinguagens com expressividade crescente: *OWL Lite*, *OWL DL* e *OWL Full*, sendo a primeira a mais simples e a última a mais complexa. A simplicidade e complexidade referidas aqui dizem respeito ao vocabulário, recursos e capacidades de expressividade de cada sublinguagem.

OWL Lite suporta necessidades primárias, como classificação de hierarquia e construções simples, entretanto possui restrições de cardinalidade e tem um número menor de propriedades. O uso desta sublinguagem é recomendado para migração de thesauri e outras taxonomias. *OWL DL* e *OWL Full* compartilham a máxima expressividade incluindo todas as construções da linguagem OWL. A diferença entre estas sublinguagens consiste que *OWL DL* garante que a ontologia seja sempre computável e decidível enquanto que, com *OWL Full*, não se pode ter esta garantia. Isso ocorre pelo fato de *OWL DL* respeitar as restrições de definição presentes na lógica *Description Logic*, daí o sufixo DL.

[Barder and Nutt 2003] define *Description Logic* como um formalismo matemático que representa o conhecimento de um domínio de aplicação (o "mundo") primeiramente definindo os conceitos relevantes desse domínio (sua terminologia) e depois usando esses conceitos para especificar propriedades de objetos e indivíduos que ocorrem nesse domínio (a descrição do mundo). Esse formalismo é que garante que ontologias em *OWL DL* vão ser computáveis e decidíveis. A finalidade de uma linguagem de representação de conhecimento que usa *Description Logic* vai além de armazenar e definir conceitos e afirmações. Suas definições podem ser validadas (checagem de consistência) e podem conter conhecimento implícito, obtido através de inferências lógicas.

A ferramenta capaz de gerar as inferências lógicas e checar a consistência em OWL é chamada de *reasoner*. Um dos principais testes que essa ferramenta pode realizar é verificar se uma classe é uma subclasse de outra classe, de modo que isso ajuda a manter uma hierarquia correta e identificar classes inconsistentes [Horridge et all 2004]. Em OWL, uma inconsistência acontece se uma classe não pode ter instâncias.

Para [Smith and Welty 2004], uma das vantagens do uso da linguagem OWL para a construção de ontologias é que existem ferramentas de classificação. Estes tipos de ferramenta proporcionam um suporte genérico, aplicável a todos os domínios de aplicação. Uma lista de classificadores é encontrada em [Grau et all 2008].

OWL trabalha com suposições de mundo aberto, considerando que as descrições de recursos podem estar presentes em mais de um único arquivo ou escopo. Isso implica na impossibilidade de assumir que algo não existe até que isso seja explicitamente definido, ou ainda, algo não pode ser definido como verdadeiro, pelo fato de não poder ser assumido como falso, pois o conhecimento pode existir sem ter sido adicionado à base de conhecimento [Horridge et all 2004].

Proporcionando uma visão geral da OWL, [McGuinness and Harmelen 2004] apresenta a sintaxe expressa em RDF/XML. Com isso, alguns termos possuem o prefixo rdf: ou rdfs: compreendendo os mesmos termos presentes em RDF ou *RDF Schema* respectivamente.

Recentemente o W3C agregou novos recursos à linguagem OWL dando origem a uma nova recomendação, chamada OWL 2, que é uma extensão e revisão da anterior [W3C OWL Working Group 2009].

A nova linguagem é consequência de um amadurecimento e evolução da Web Semântica, e com isso, novos recurso e possibilidades foram incorporados em OWL 2. O uso de outras linguagens além do RDF/XML para definição da sintaxe, novos recursos semânticos como redes de propriedades, definição de intervalo de dados, tipos de dados mais ricos e novos tipos de propriedades são exemplos dos novos recursos incorporados.

A justificativa encontrada por [Grau et all 2008] para justificar a necessidade da evolução da OWL, é o seu sucesso. Pois, devido ao sucesso da "OWL 1" foram identificados problemas quanto ao design da linguagem, sendo alguns de natureza crítica. Problemas como as limitações de expressividade juntamente com o fato da linguagem OWL ter sua sintaxe influenciada pelo paradigma de frames, são alguns dos problemas identificados pelo autor que justificam a evolução da linguagem.

1.3. Regras SWRL

Infelizmente a expressividade de OWL nem sempre é suficiente para modelar todos os tipos de problemas. Especialmente OWL não tem suporte a cláusulas no formato de Horn (representa uma sentença de implicação). Para suprir essa deficiência, a SWRL (*Semantic Web Rule Language*) foi criada.

A SWRL é baseada em uma combinação de *OWL DL* e *OWL Lite* [O'Connor et all 2005]. Essa combinação amplia o conjunto de axiomas da linguagem OWL, mais especificamente para poder incluir cláusulas no formato de Horn [O'Connor et all 2005]. Essas regras podem ser usadas para inferir novos conhecimentos a partir de bases de conhecimento em OWL. SWRL permite que os usuários escrevam regras para raciocínio sobre os indivíduos OWL (instâncias) que podem inferir novos conhecimentos sobre esses indivíduos.

Regras em SWRL são compostas de duas partes: o antecedente (*body*) e o consequente (*head*). Cada regra é uma implicação entre o antecedente e o consequente, que pode ser entendida como: quando as condições do antecedente são verdadeiras, então as condições do consequente também são verdadeiras. Ambas as partes consistem em uma conjunção de zero ou mais átomos, não permitindo disjunções ou negação.

Os átomos, por sua vez, são formados por um predicado e um ou mais argumentos (o número e o tipo destes argumentos são determinados pelo tipo do átomo,

que por sua vez, é definido pelo predicado utilizado). Embora a especificação W3C defina sete tipos de átomos, neste capítulo foi adotada a convenção usada em [Hassanpour et all 2009], que trabalha com seis tipos de átomos, pois, SameAs e DifferenteFrom embora possuam semânticas diferentes, sintaticamente são idênticos. Na Tabela 1.2 são apresentados os tipos de átomos e suas descrições.

Tabela 1.2 – Tipos de átomos SWRL [Hassanpour et all 2009]

Tipo de átomo	Descrição
Class	O predicado corresponde a uma classe definida na ontologia e recebe um indivíduo[2] como argumento.
Object property	O predicado corresponde a uma propriedade definida na ontologia e recebe dois indivíduos como argumentos e os relaciona entre si.
Datavalued property	O predicado corresponde a uma propriedade definida na ontologia e recebe dois argumentos: um indivíduo e um valor[3] relacionando-os.
Data range	O predicado corresponde a um tipo de dado definido na ontologia e recebe um valor como argumento.
Same/different	O predicado corresponde ao termo *"same as"* ou *"different from"* e recebe dois indivíduos como argumentos definindo que estes são iguais ou diferentes respectivamente.
Built-in	O predicado corresponde a um conjunto de funções pré-definidas (comparações, funções matemáticas, lógicas, ...) ou definidas pelo usuário que recebem um ou mais argumentos e retornam verdadeiro quando estes satisfazem o predicado.

Os tipos *Class*, *Object property* e *Datavalued property* correspondem a elementos (conceitos) definidos na ontologia. Data range correspondem aos tipos de dados utilizados na ontologia. O tipo *Same*/*different* é utilizado para explicitar a igualdade ou diferença entre dois indivíduos, exigida pelo fato de OWL trabalhar com o paradigma de mundo aberto. Por fim, os Built-ins são tipos que encapsulam as mais diversas funções, podendo inclusive ser estendidos pelos usuários. Cada tipo de átomo define o número e o tipo de argumentos suportados, conforme ilustrado na Tabela 1.3.

Tabela 1.3 – Números e tipos de argumentos suportados pelos tipos de átomos

Tipo de átomo	Número de argumentos	Tipo de argument
Class	1	i-object
Object property	2	i-object, i-object
Datavalued property	2	i-object, d-object
Same/Different	2	i-object, i-object

[2] Na Tabela "Tipos de átomos SWRL" o termo indivíduo é empregado para representar: 1) um indivíduo específico definido na ontologia; ou, 2) variáveis para indivíduos, que podem ser quaisquer indivíduos definidos na ontologia. Ou seja, corresponde ao tipo I-Object.

[3] O termo valor neste caso é uma livre tradução de *data value* e representa um dado de um tipo primitivo definido na ontologia (Ex: inteiro, real, *string*). Na Tabela "Tipos de átomos SWRL", o termo valor é empregado para representar valores constantes ou variáveis para valores. Ou seja, corresponde ao tipo D-Object.

Tipo de átomo	Número de argumentos	Tipo de argument
Built-in	2 ou mais	d-object ou i-object
Data range	1	d-object

As variáveis são tratadas como quantificadores universais e possuem o escopo limitado à regra a qual pertencem. Apenas variáveis que ocorrem no antecedente podem ocorrer no consequente.

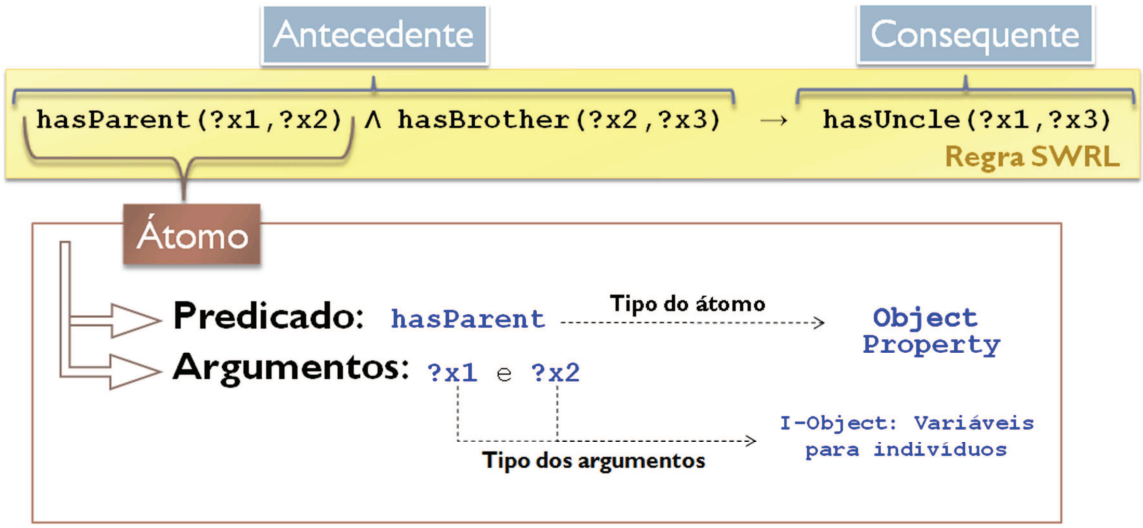


Figura 1.2 – Regra SWRL especificada no formato humano e suas partes.

Embora as regras SWRL possam ser representadas em mais de um formato, o formato de leitura humano é adotado neste capítulo. Neste formato, a seta (→) é usada para separar antecedente e consequente, o acento circunflexo (^) representa a conjunção entre os átomos e o sinal de interrogação (?) distingue as variáveis dos nomes de indivíduos. Na Figura 1.2 é apresentada uma regra SWRL, representada no formato humano, ressaltando suas partes e características.

Semanticamente, o exemplo anterior ilustra que **se** o pai/mãe de um indivíduo possui um irmão, **então** este irmão é tio do indivíduo em questão. A regra possui três átomos (todos do tipo *Object property*) sendo dois no antecedente e um no consequente (cujos predicados são: hasParent, hasBrother e hasUncle respectivamente) e faz uso de três variáveis para indivíduos: ?x1, ?x2 e ?x3. Outro exemplo, suma seria uma regra SWRL que expresse a seguinte afirmação: "uma pessoa qualquer que tenha qualquer carro será considerada um motorista", ficaria assim:

pessoa(?x) ^ temCarro(?x, true) -> motorista(?x)

Ao executar-se a regra acima, o seu efeito seria classificar todos os indivíduos da classe pessoa que possuíssem carro como também pertencentes à classe motorista. As regras SWRL também permitem trabalhar com um indivíduo específico de uma ontologia. Por exemplo, é possível escrever uma regra diretamente para um indivíduo João, pertencente à classe pessoa, e fazer uma classificação direta deste indivíduo. Um exemplo pode ser visto na regra abaixo:

pessoa(João) ^ temCarro(João, true) -> motorista(João)

A regra acima apenas funciona para o indivíduo conhecido como João. Um dos recursos mais poderosos do SWRL é a capacidade de usuários usarem *built-ins* definidos [SWRLLanguage 2012]. *Built-in* é um predicado que pode ter um ou mais argumentos e realiza uma operação com os argumentos, avaliando e retornando verdadeiro ou falso. Por exemplo, uma regra que classifica um indivíduo da classe pessoa que seja maior de 18 anos como da classe adulto pode ser vista abaixo:

pessoa(?x) ^ temIdade(?x, ?idade) ^ swrlb:greaterThan(?idade, 18) -> adulto(?x)

SWRL permite o uso de novas bibliotecas de *built-ins* e os usuários podem definir suas próprias bibliotecas, o que torna SWRL uma linguagem muito rica em recursos de representação. Por exemplo, poderia ser criada uma biblioteca que tivesse operações de conversão de valores e busca de termos em taxonomias.

Um exemplo mais complexo do uso de SWRL é mostrado por [Hassanpour et all 2009], neste exemplo o objetivo é estabelecer relações entre duas ou mais entidade de uma ontologia. O exemplo cria uma regra que estabelece regras de condução no estado da Califórnia para menores de 18 anos:

Person(?p) ^ has_Driver_License(?p,?d) ^ issued_in_State_of(?d,?s) ^ swrlb:notEqual(?s,"CA") ^ has_Age(?p,?g) ^ swrlb:lessThan(?g,18) ^ number_of_Visiting_Days_in_CA(?p,?x) ^ swrlb:lessThan(?x,10) ^ Car(?c) ^ has_Weight_in_lbs(?c,?w) ^ swrlb:lessThan(?w,26000)

→ can_Drive(?p,?c)

Traduzindo essa regra para linguagem humana teríamos: “Qualquer indivíduo com idade inferior a 18 anos, mas que possua carta de motorista, sendo de fora da Califórnia, visitando o estado por menos de 10 dias e dirigindo um veiculo com menos de 26000 libras, pode dirigir normalmente”.

Cabe ressaltar que as regras são armazenadas na própria ontologia e o uso deste tipo de anotação permite a inferência de novos conhecimentos sobre indivíduos, uma vez que as regras correspondem a afirmações condicionais que, ao serem satisfeitas, agregam novas informações à base de conhecimento.

1.4. SWRL Editor

Nesta seção é descrita uma nova ferramenta, intitulada SWRL Editor, que foi desenvolvida nos trabalhos [Orlando 2012] e [Silva 2012]. O SWRL Editor foi desenvolvido como um plug-in para o Web-Protégé. O Web-Protégé é um editor open-source de ontologias, baseado no Protégé que funciona na Web [Tudorache et all 2008]. Ele foi desenvolvido usando o GWT (*Google Web Toolkit*). O principal motivo de se ter uma versão Web do Protégé é facilitar a colaboração entre desenvolvedores, já que não é necessário instalar programas locais. Nessa ferramenta, as ontologias estão disponíveis de forma centralizada e compartilhada. Como as ontologias estão se tornando cada vez

maiores, é difícil que uma pessoa, ou um grupo pequeno de pessoas, consiga manter grandes ontologias de forma eficaz [Tudorache et all 2008]. Daí a necessidade do desenvolvimento colaborativo de ontologias e regras.

O SWRL Editor foi integrado ao Web-Protégé e está dividido em Servidor e Cliente. O Servidor centraliza as operações com as regras, por exemplo, ontologias e regras são carregadas de seus arquivos apenas uma vez e ficam disponíveis para todos os clientes através de chamadas RPC. Além disso, a centralização das operações é um ponto chave para que a ferramenta possa funcionar colaborativamente, já que o servidor atua como nó central de distribuição das alterações feitas por cada usuário.

Já no Cliente estão as interfaces gráficas (as quais que são priorizadas nessa seção) são divididas em duas partes: Visualização (Seção 1.4.1), Filtros (Seção 1.4.2), Opções (Seção 1.4.3) e Composição (Seção 1.4.4). Para acessar o cliente do SWRL Editor é necessário escolher uma das ontologias. Na Figura 1.3 é mostrado o Web-Protégé após se ter acessado a Ontologia da Família. Nessa figura é possível ver a guia SWRL Editor (A), onde se encontra a nova ferramenta.

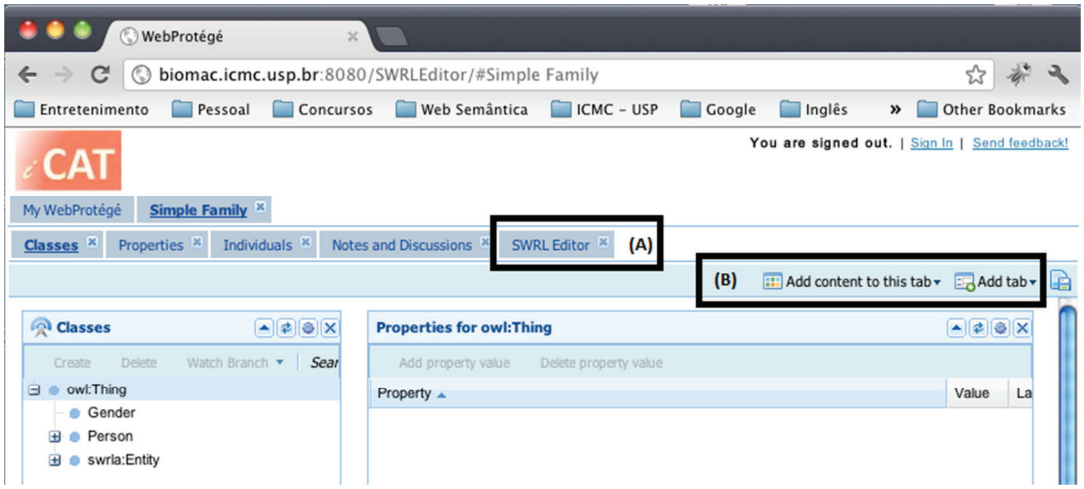


Figura 1.3 – Tela após o acesso a ontologia da família: (A) Acesso ao plug-in; (B) Acesso a todos os plug-ins não carregados automaticamente.

O Web-Protégé, após acessar a ontologia, carrega automaticamente os principais plug-ins (guias). Como não existe no Web-Protégé um plug-in para manipulação de SWRL, foi definido que o SWRL Editor é carregado automaticamente. Porém, caso seja fechado o plug-in, é possível acessá-lo novamente usando os botões Figura 1.3 (B).

1.4.1. Visualização

A página inicial do SWRL Editor é a *Visualizations* (Figura 1.4). Essa página é dividida em:

- Barra de Ferramentas (A) – Da esquerda para a direita: *Options* é o acesso as configurações do SWRL Editor; *Info* é o acesso as informações gerais do conjunto de regras; *New Rule* acessa a tela de composição para inserção de uma nova regra; *Run Rules* executa as regras no servidor. O botão Edit em *Rule Filter* acessa a tela para modificar o filtro das regras exibidas.
- Formas de Visualização dos Conjuntos de Regras (B) – Da esquerda para a direita: A guia *List* mostra as regras em uma visualização hierárquica (Seção

1.4.1.2). A guia *Text* mostra as regras usando Parafraseamento (Seção 1.4.1.3). A guia SWRL mostra as regras usando SWRL com *Highlight* (1.4.1.1). A guia *Autism* mostra uma representação visual especial só para um conjunto especifico de regras para classificação de fenótipos de autismo [Silva 2012] e que não será abordada nesse capítulo. A guia *Groups* dispõe das técnicas para agrupamento (Seção 1.4.1.4). A guia *Decision Tree* contém as técnicas para transformar os conjuntos de regras para um formato de árvore (Seção 1.4.1.5).

- Botões para cada regra (C) – Para cada regra, nas guias *List*, *Text*, *SWRL* e *Autism*, existem botões para manipulação. Da esquerda para a direita são eles: O botão *Edit Rule* que coloca a regra em modo de edição. O botão *Duplicate and Edit* que faz uma cópia da regra e coloca a essa copia em modo de edição. O botão *Delete* excluí uma regra do conjunto. O botão *Similar Rules* mostra uma lista de regras similares usando um dos algoritmos de agrupamento.
- Barra de Status (D) – Serve como status do sistema, sendo utilizada para mostrar o numero de regras filtradas/número total de regras do conjunto.

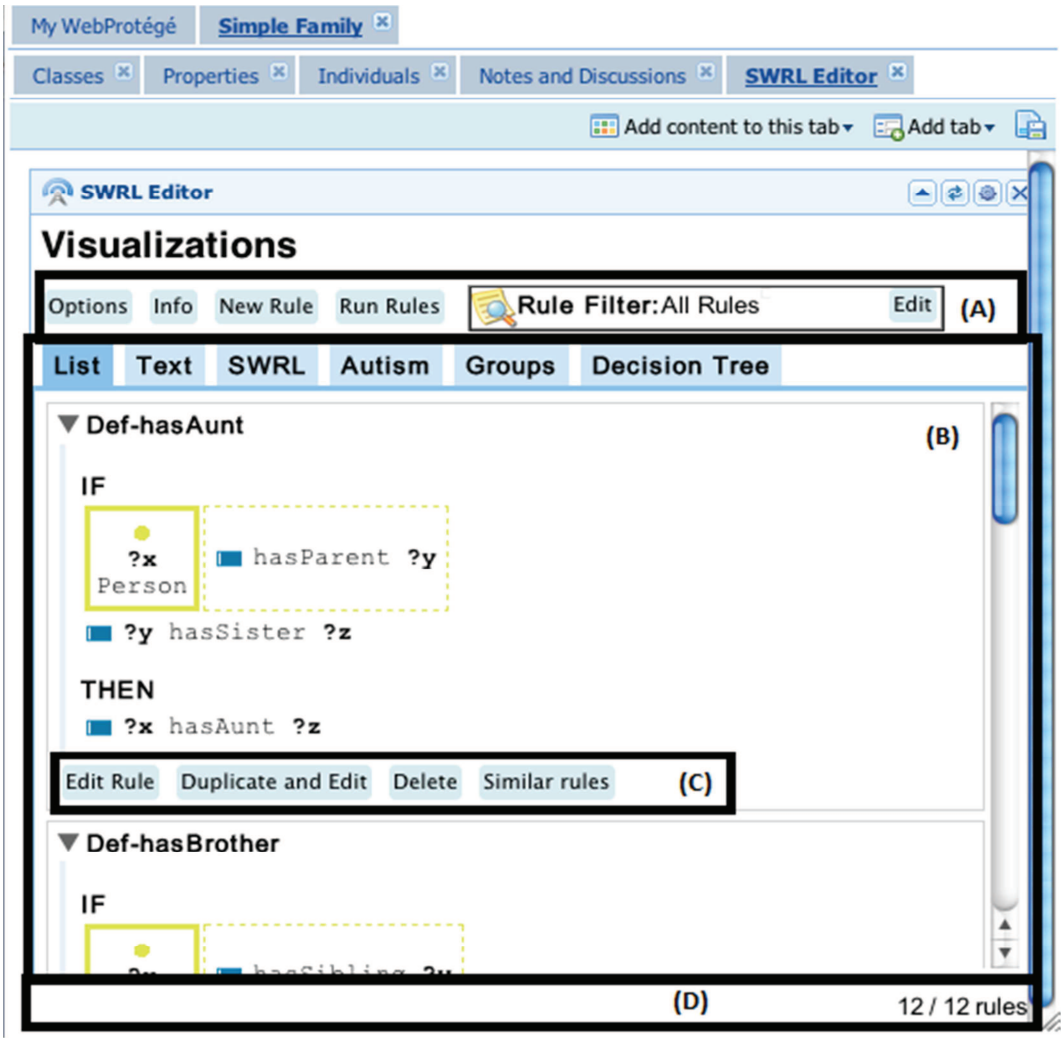


Figura 1.4 – SWRL Editor – Visualização: (A) Barra de Ferramentas; (B) Formas de Visualização dos Conjuntos de Regras; (C) Botões para cada regra; (D) Barra de Status do plug-in.

1.4.1.1. *SWRL com Highlight*

A representação visual *SWRL* exibe a regra sem ocultar os detalhes da sintaxe, contudo, agrega cores aos átomos e argumentos semelhantemente a recursos encontrados em alguns IDEs. Ao invés de utilizar a quebra automática no final da linha, cada átomo é apresentado em uma linha e o símbolo de implicação →, que divide o antecedente do consequente, é disposto em uma linha separada, o que torna a regra mais legível. Na Figura 1.5, a regra **def_hasAunt** é apresentada no formato original (A) e na representação com *highlight* (B).

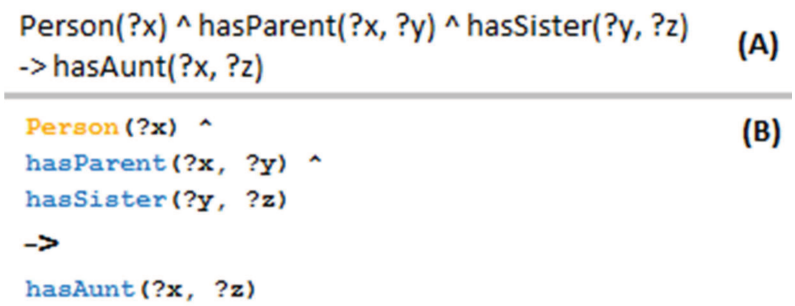


Figura 1.5 – Comparativo entre (A) uma regra apresentada no formato original e (B) a representação visual com *Highlight*.

Na Tabela 1.4 são apresentadas as cores utilizadas na geração dos *highlights* e seus significados. A representação visual destaca os tipos dos átomos, variáveis e valores literais sem sobrecarregar a apresentação da regra. Algumas das cores utilizadas foram extraídas dos símbolos utilizados pelo editor de ontologias Protégé, enquanto outras foram extraídas do editor Eclipse. O uso de uma fonte de tamanho fixo (monoespaçada) foi adotado seguindo o padrão encontrado nos editores de texto usados na programação, o que torna a leitura da regra mais organizada.

Tabela 1.4 – Cores utilizadas na representação *SWRL Highlight*

Cor	Exemplo	Descrição
Laranja	Person_Female	Representa predicados do tipo *Class*[4]
Azul	has_natural_father	Representa predicados do tipo *Object property*
Verde	has_current_age	Representa predicados do tipo *Datavalued property*
Vermelho	sqwrl:avg	Representa predicados do tipo *Built-in*
Cinza	differentFrom	Representa predicados do tipo *Same/different*
Preto em negrito	**?a**	Representa as variáveis
Roxo	32; "teste"	Representa os valores literais (Strings e números)

[4] Embora o Protégé utilize amarelo no símbolo que representa as classes, foi adotado para esta visualização o tom mais alaranjado pelo fato da cor amarela apresentar pouco contraste com fundos claros.

1.4.1.2. *Visualização Hierárquica*

A representação denominada visualização hierárquica busca abstrair a sintaxe da regra apresentando os átomos agrupados hierarquicamente a partir dos átomos do tipo *Class*. Assim, ao mesmo tempo em que torna a regra mais clara para apresentação, enriquece com cores e símbolos o seu significado. Os símbolos empregados na ferramenta Protégé foram reutilizados, contudo, para representar os demais elementos (*Same/Different* e *Built-ins*) foram empregados retângulos com as cores utilizadas na representação com *highlight* (Seção 1.4.1.1). Na Tabela 1.5 são apresentados exemplos dos símbolos utilizados na visualização.

Tabela 1.5 – Cores utilizadas na representação *SWRL Highlight*

Tipo	Símbolos e Cores	Exemplo
Class	Circulo amarelo	?b Person_Female
Object property	Retângulo azul	has_natural_father(?a,?f)
Datavalued property	Retângulo verde	has_current_age(?a,?b)
Built-in	Retângulo vermelho	?c > 18
Same/different	Retângulo cinza	?m same ?n ?m different ?n

O resultado dessa visualização pode ser visto na Figura 1.6, onde é apresentada a mesma regra (**def_hasAunt**) da representação com *highlight*. Embora esta representação oculte alguns detalhes da sintaxe SWRL e exiba os átomos organizadamente, seu uso se torna apropriado a desenvolvedores não técnicos em computação, apenas quando são utilizados na ontologia nomes apropriados nos labels (rdfs: Label) dos termos.

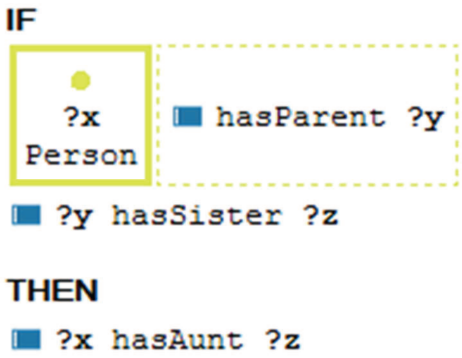


Figura 1.6 – Representação Hierárquica.

1.4.1.3. *Parafraseamento*

Essa representação fornece uma explicação em forma de texto da regra. É uma re-implementação de uma técnica do Axiomé [Hassanpour 2010] para gerar paráfrases em inglês. Apenas foi implementada uma mudança para essa técnica: a utilização de negrito em termos de ligação (IF, IS, AN, ...) entre os átomos de uma regra. A mudança é simples, porém torna mais fácil a leitura (Figura 1.7). Além disso, o SWRL Editor

oferece a possibilidade de entrar em modo de edição a partir da representação em parafraseamento. Não edita o parafraseamento, mas acessa a edição desta regra.

```
IF
        "y" IS A Person
        AND "y" HAS Child "x"
        AND "y" HAS Child "z"

THEN
        "x" HAS Sibling "z"
```

Figura 1.7 – Parafraseamento.

1.*4.1.4. Agrupamento*

O agrupamento tem o objetivo de facilitar a visualização de grandes conjuntos de regras. Essa técnica facilita a localização de regras pelo usuário dentro de um conjunto de regras. O agrupamento é feito por meio de algoritmos que implementam interfaces Java e são carregados automaticamente no SWRL Editor. Atualmente, o SWRL Editor possui 3 algoritmos de agrupamento:

- *K-means* – Similaridade de átomos: Implementa o algoritmo de *K-means* considerando a similaridade dos átomos de cada regra [Silva 2012];
- *K-means* – Similaridade de predicados: Implementa o algoritmo de *K-means* considerando a similaridade dos predicados de cada átomo [Silva 2012];
- Estrutura da regra: Separa as regras em grupos por estrutura sintática das regras [Orlando 2012];

Além disso, o SWRL Editor possui sistemas de carregamento automático de algoritmos para o agrupamento. Esse sistema carrega os algoritmos de forma automática usando a classe Java: ServiceLoader. Para o ServiceLoader carregar os algoritmos é necessário que esses sejam implementados usando as interfaces Java que o SWRL Editor disponibiliza para agrupamento ou para árvore de decisão. Mais detalhes de como desenvolver novos algoritmos para o SWRL Editor estão disponíveis em [Orlando 2012].

O sistema de carregamento automático de algoritmos tornou-se uma ótima estratégia para usuários mais avançados que tenham interesse em desenvolver seus próprios algoritmos. Ele facilita essa operação já que para inserir um novo algoritmo é apenas necessário gerar um arquivo JAR com as implementações e o colocá-lo numa pasta do servidor, sem ser necessário alterar o código do SWRL Editor.

1.*4.1.5. Árvore de Decisão*

Da mesma forma que no agrupamento, árvores de decisão usam o sistema de carregamento automático de algoritmos e podem ser adicionados novos algoritmos. Atualmente, o SWRL Editor conta com 3 implementações de árvore de decisão:

- Ocorrência de átomos (Figura 1.8) – Essa técnica apenas cria uma árvore com base em um ranking de ocorrência de átomos, ou seja, átomos que mais ocorrem tendem a estar mais perto da raiz.

- Dependência de variáveis (Figura 1.9) – Essa técnica cria as árvores levando em conta a dependência das variáveis.
- Paráfrases (Figura 1.10) – Divide o parafraseamento (Seção 1.4.1.3) em pequenas paráfrases e com isso cria uma árvore mantendo a ordem original da paráfrase.

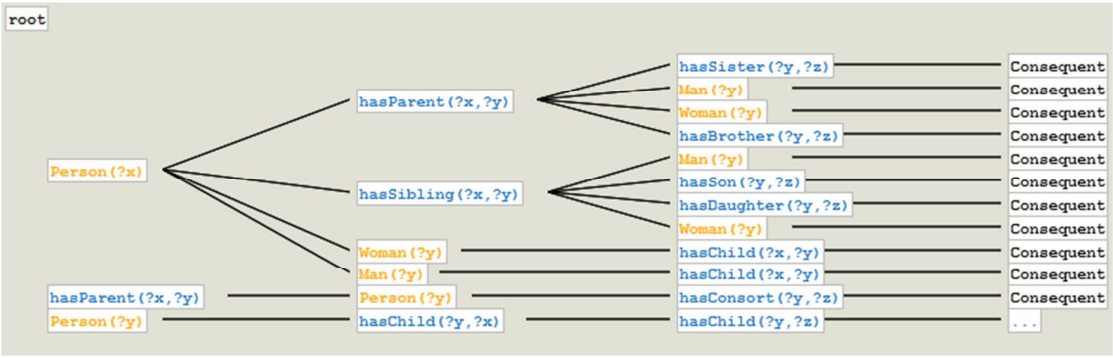


Figura 1.8 – Árvore de decisão: Ocorrência de átomos.

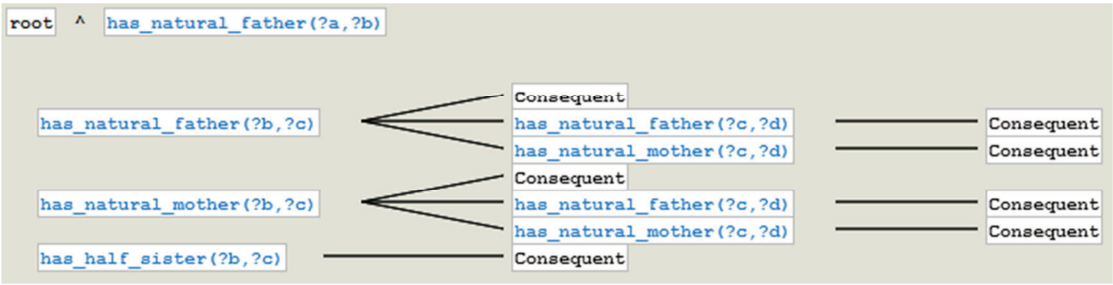


Figura 1.9 – Árvore de decisão: Dependência de variáveis.

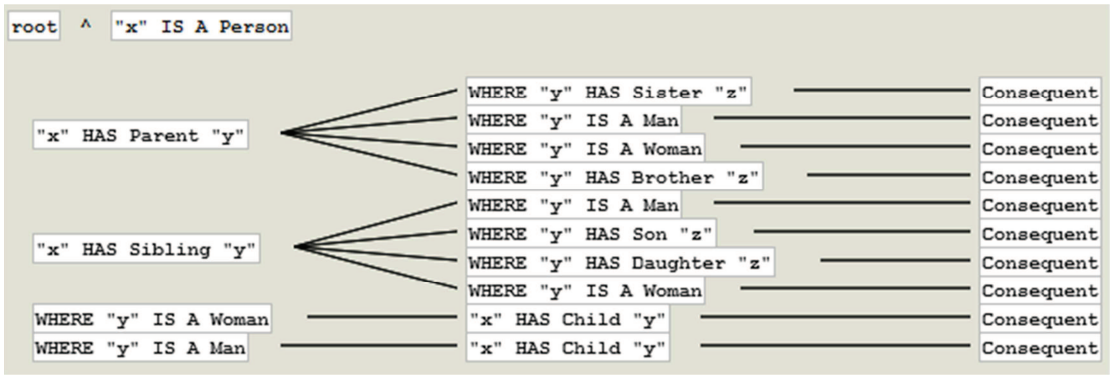


Figura 1.10 – Árvore de decisão: Paráfrases.

A funcionalidade das árvores de decisão é uma das principais contribuições do SWRL Editor, pois essa técnica nunca foi usada para regras SWRL e mostra-se como uma forma fácil de navegar em conjuntos de regras. Através dela é possível encontrar facilmente regras com antecedentes iguais ou semelhantes. Na Figura 1.11 é possível identificar que ocorrem antecedentes com mais de um consequente. De cima para baixo, na primeira ocorrência de um antecedente com dois consequentes, ao passar o mouse sobre os dois consequentes é possível perceber que eles são diferentes. Sendo assim, pode ser que não seja um erro, mas uma opção de projeto. Já na segunda ocorrência, quando são olhados os consequentes é possível identificar que eles são iguais (um erro).

Como os algoritmos de árvore de decisão juntam átomos semelhantes é possível identificar os tipos de repetição que merecem atenção.

Figura 1.11 – Árvore de decisão: Identificação de regras com antecedentes iguais.

Muitas regras se diferenciam às vezes por um ou dois átomos. Então como ocorrem muitas repetições nos átomos das regras, a árvore de decisão pode ser usada para fazer o reaproveitamento de átomos. A Figura 1.12 mostra a tela da árvore de decisão, como pode ser visto na imagem, ao se clicar sobre um nodo da árvore são mostradas opções em um menu popup. Quando o nodo for um átomo do antecedente, o menu irá mostrar a opção para reaproveitar todos os átomos até a raiz, em uma nova regra. Já quando o nodo for um consequente, o menu mostrará a opção de editar essa regra.

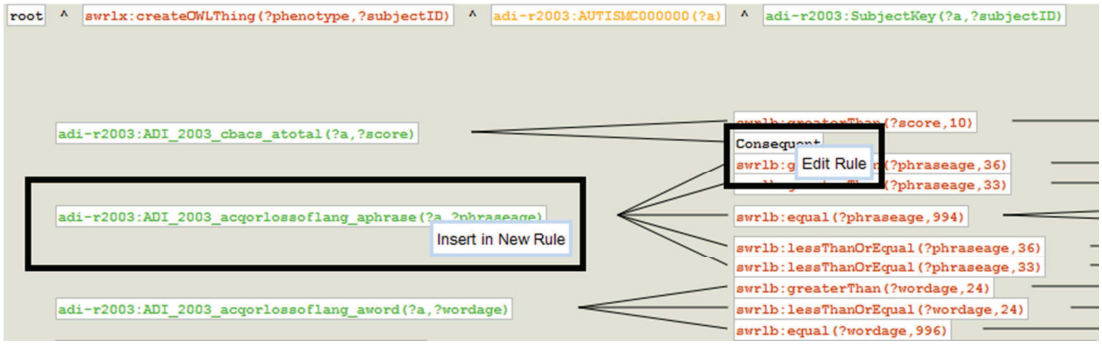


Figura 1.12 – Árvore de decisão: Botão direito sobre os nós.

1.4.2. Filtros

Como conjuntos de regras tendem a crescer, torna-se cada vez mais difícil localizar regras, então o SWRL Editor possui uma interface para filtrar regras SWRL. Nessa interface, o usuário pode usar os operadores lógicos AND, OR e NOT para criar uma expressão lógica para filtrar regras. O filtro é montado usando a interface do usuário na Figura 1.13, em que há um campo para cada operador lógico. Nesses campos, os valores de filtro são separados por espaços, expressões com espaços devem ser delimitadas por "" (exemplo "casa e carro").

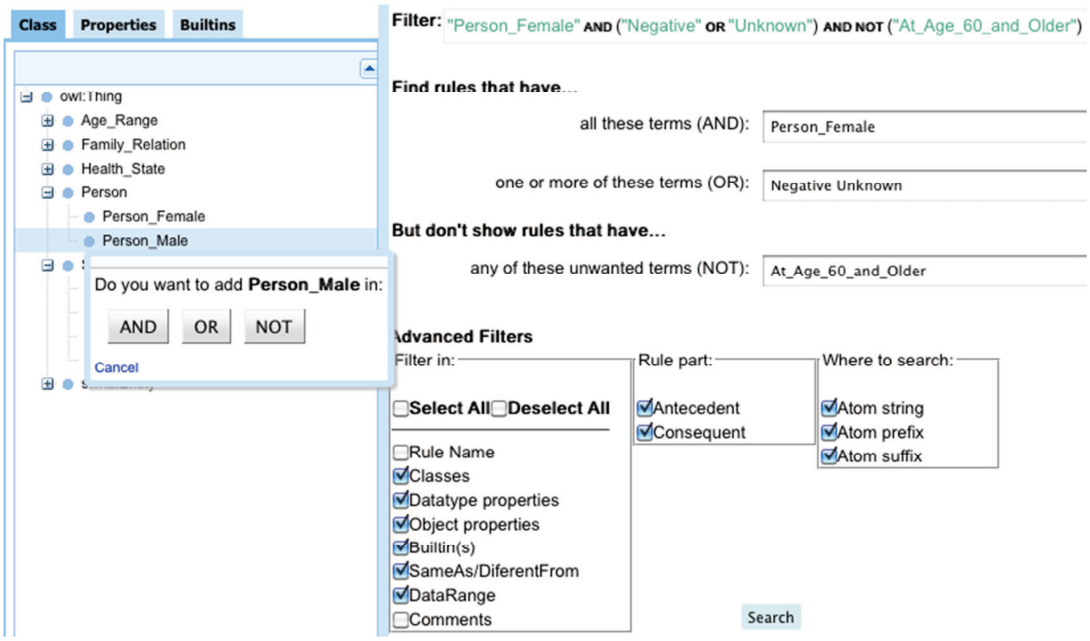


Figura 1.13 – Filtro das regras.

O filtro utiliza a expressão lógica montada (Figura 1.13, parte superior) para procurar automaticamente no rdf:ID e no rdfs:Label dos predicados de cada átomo. Além disso, a interface fornece opções avançadas para filtrar, sendo possível escolher partes de uma regra e tipos de átomos a serem filtrados. É possível escolher filtrar no Nome da regras, em *Class*, *Datatype properties*, *Object properties*, *Buitins*, *SameAs/DiferentFrom*, *DataRange*, *Comments* (Comentários dos predicados de cada átomo). Também é possível filtrar somente no antecedente ou no consequente. As opções para a filtragem ainda permitem os usuários realizem buscas nos prefixos, sufixos ou qualquer parte da string que representa cada átomo.

Na Figura 1.13, na lateral esquerda, encontram-se três guias (*Class*, *Properties* e *Bultins*) com os termos da ontologia. Essas guias são usadas para facilitar a inserção de termos da ontologia em uma busca. Como pode ser visto nessa figura, assim que o usuário clica na classe Person_Male a interface oferece a opção para escolher a qual operador lógico quer inserir.

Quando o usuário termina de montar o filtro, basta clicar no botão *Search* (localizado no lado direito parte inferior) que o filtro será executado e o seu resultado será mostrado na tela de visualização do SWRL Editor. Na Figura 1.14 é mostrada a tela de visualização com o filtro de regras modificado. Apenas as regras que obedecerem as condições desse filtro serão mostradas na interface e poderão ser usadas nas outras ferramentas.

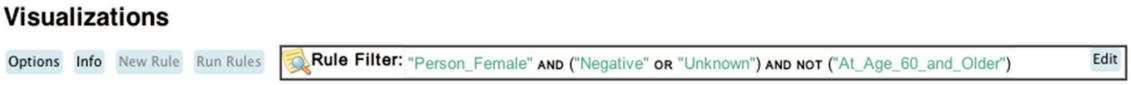


Figura 1.14 – Resultado do filtro de regras.

1.4.3. Opções

As configurações do SWRL Editor servem para tornar a ferramenta mais amigável e facilitar os processos de visualização e edição de regras. As configurações do SWRL

Editor foram divididas em três partes (Figura 1.15): *General*, *Composition* e *Visualization*. Só serão detalhadas as configurações gerais, as demais configurações são apenas detalhes de interface, já as configurações gerais servem para toda a ferramenta.

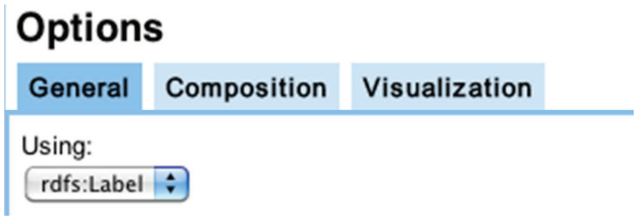


Figura 1.15 – Opções Gerais do SWRL Editor.

Na Figura 1.15 é apresentada a tela das configurações gerais, nela somente existe uma configuração para usar rdf:ID ou rdfs:Label. O rdf:ID é um identificador único para cada termo da ontologia, já o rdfs:Label é uma descrição para o termo da ontologia. Usando a ontologia do autismo [Hassanpour 2011], na Tabela 1.6 é mostrado alguns rdf:ID de termos da ontologia com o seu respectivo rdfs:Label. Fica evidente que os rdfs:Labels podem tornar mais fácil o entendimento e a edição das regras.

Tabela 1.6 – rdf:ID versus rdfs:Label

rdf:ID	rdfs:Label
adi-r2003:AUTISMC000000	*Autism Diagnostic Interview-Revised*
ados1:AUTISMC000004	*Autism Diagnostic Observation Schedule*
ados4:AUTISMC000001	*Autism Diagnostic Observation Schedule*
vabs_survey:AUTISMC000003	*Vineland Adaptive Behavior Scales*
Autism-core:AUTISMC100000	*Phenotype Record*
Autism-core:AUTISMC100002	*Quantitative value*
NIF-Invertigation:birnlex_2387	*Citation Record*
Autism-core:AUTISMC1000069	*Present*
Autism-core:AUTISMC1000070	*Repetitive stereotypical behavior*

Essa é uma importante técnica, pois permite ao usuário escolher a opção de visualizar e editar as regras usando rdfs:Labels. Porém, isso pode acarretar problemas na edição, pois um mesmo rdfs:Label pode estar em mais de um termo da ontologia. Um exemplo está na Tabela 1.6 em que os termos ados1:AUTISMC000004 e ados4:AUTISMC000001 compartilham um mesmo rdfs:Label. A ferramenta trata esse tipo de redundância, pois mesmo com a opção para usar rdfs:Label selecionada, o usuário poderá usar um rdf:ID durante a edição.

1.4.4. Composição

O processo de criação de regras é ativado no SWRL Editor quando o usuário seleciona a opção “New Rule” na visualização ou opta por editar uma regra específica. Na Figura 1.16 é apresentada a tela da ferramenta SWRL Editor no modo composição. Na lateral esquerda (A) dessa figura são exibidos os termos disponíveis na ontologia que podem ser selecionados pelos usuários (da mesma forma que nas Opções Seção 1.4.2). Quando

um deles é selecionado, uma janela é aberta com as opções de adicioná-lo como um átomo no antecedente ou consequente.

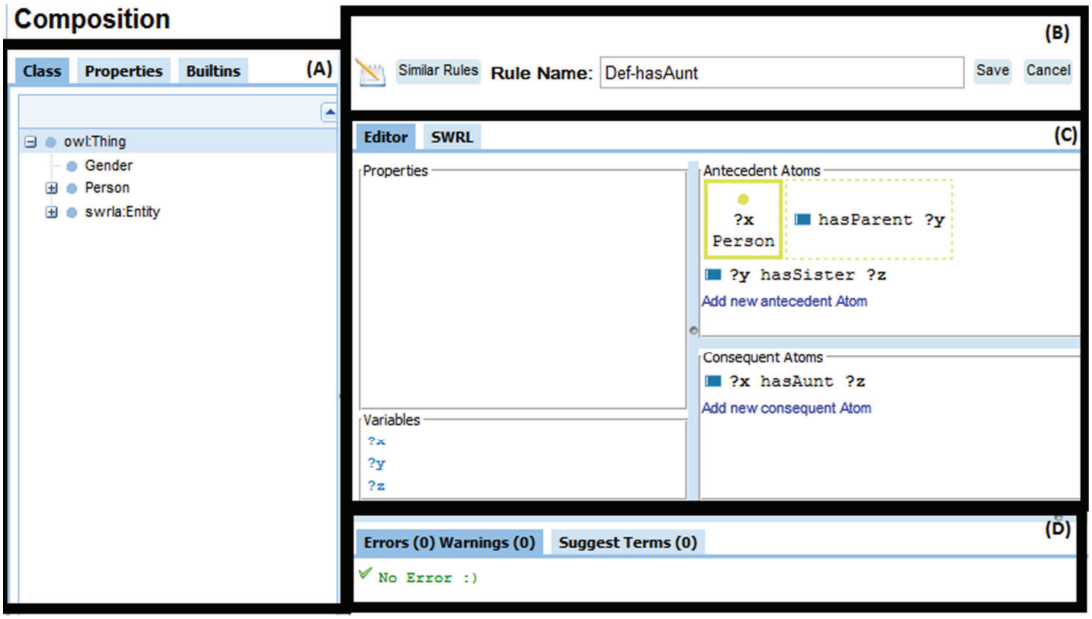


Figura 1.16 – SWRL Editor – Composição: (A) Termos disponíveis na ontologia; (B) Nome da regra e opção de salvar; (C) Editores da regra; (D) Avisos e sugestões de regras.

Na Figura 1.16 (B), é possível definir ou alterar o nome da regra. O botão salvar (*Save*) fica desativado enquanto são identificados erros na regra em desenvolvimento. Além disso, quando o desenvolvedor opta por voltar ao modo de visualização sem salvar (*Cancel*) a regra, a ferramenta exibe uma mensagem de confirmação. Ainda nessa figura em (C) é possível ver os dois modos de edição dos átomos de uma regra:

- *Editor* (ativo na Figura 1.16) – Formulário que utiliza a representação hierárquica para apresentar e organizar os átomos da regra. Divide o ambiente de desenvolvimento nas seguintes janelas de trabalho:
    - Caixa de propriedades (*Properties*), contendo as propriedades do átomo que o desenvolvedor selecionar;
    - Lista de variáveis (*Variables*) utilizadas na regra;
    - Átomos do antecedente (*Antecedent Atoms*), contendo os átomos presentes na parte antecedente da regra. Os átomos são apresentados conforme a representação hierárquica e podem ser alterados na caixa de propriedades;
    - Átomos do consequente (*Consequent Atoms*), contendo os átomos presentes na parte consequente da regra. Os átomos são apresentados conforme a representação hierárquica e podem ser alterados na caixa de propriedades;
- SWRL – Editor SWRL com *Highlight*, no qual o desenvolvedor escreve a regra utilizando a sintaxe SWRL. Os átomos e argumentos são apresentados conforme o padrão de cores adotado na visualização do SWRL com *Highlight*;

Na parte inferior da Figura 1.16 (D), são apresentadas as seguintes abas:

- Problemas (*Errors* e *Warnings*) – contem as possíveis notificações de erros e avisos para o desenvolvedor da regra;
- Termos sugeridos (*Suggest Terms*) – conterá uma lista com os termos sugeridos;

Um ponto importante, já citado, é o da ferramenta permitir que o usuário utilize os rdfs:Labels (mais fácies que os identificadores únicos rdf:ID). Porém para que não ocorram redundâncias são apresentados erros como o da Figura **1**.17. Nesta Figura é mostrada a seguinte mensagem de erro: "The predicate label **Autism Diagnostic Observation Schedule** represents more than one ID: **ados4:AUTISMC000001**, **ados1:AUTISMC000004**". O usuário poderá então usar um dos dois rdf:IDs para representar o elemento e assim evitar o erro.

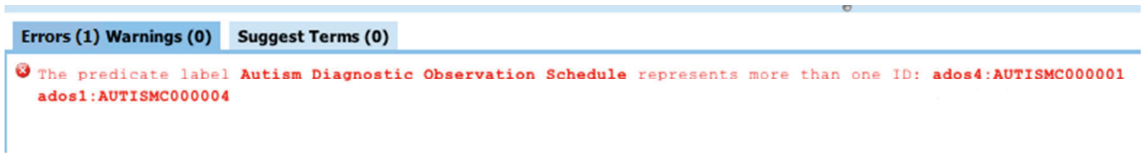


Figura 1.17 – Mensagem de erro de redundância de rdfs:Labels.

1.5. Construção de uma ontologia com regras SWRL

Essa seção descreve a criação de uma ontologia de relações familiares. A idéia de utilizar esse exemplo reside no fato de que esse domínio é de fácil entendimento por todos e pode gerar bons exemplos. Vamos dividir essa demonstração em duas etapas:

- Ontologia – Definir os termos da ontologia e suas relações:
  - Classes (Conceitos) O objetivo criar classes e subclasses, modificando a hierarquia de classes da ontologia;
  - Propriedades (Relações) As propriedades representam relacionamentos entre dois indivíduos;
- SWRL:
  - SWRL Regras de inferências;
  - Indivíduos da ontologia representam objetos no domínio de interesse (ou indivíduos de uma classe);
  - Execução – Demonstração dos resultados gerados após a execução de um conjunto de regras;

1.5.1. Construção da ontologia

Toda ontologia contém uma classe chamada owl:Thing. Conforme mencionado, as classes OWL são interpretadas como conjuntos de indivíduos (ou conjunto de objetos). A classe owl:Thing é a classe que representa o conjunto que contém todas as classes e os indivíduos, uma vez que todas as classes são subclasses de owl:Thing. Para exemplificar será apresentada nessa seção a criação de uma ontologia que representem relações parentesco entre indivíduos.

Na Figura 1.18 são definas as subclasses de owl:Thing que farão parte da ontologia de exemplo. Já nas Figuras 1.19 e 1.20 são definidas as outras classes que são subclasses de Man e Woman respectivamente.

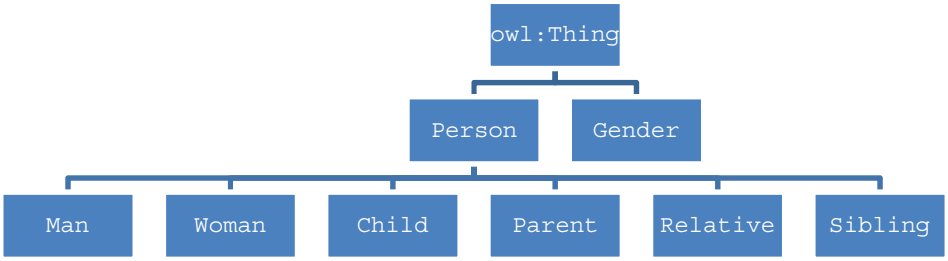


Figura 1.18 – Definição da ontologia: owl:Thing.

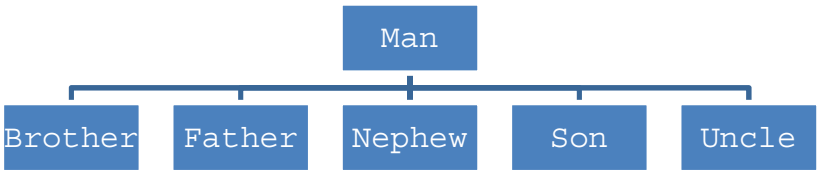


Figura 1.19 – Definição da ontologia: Man.

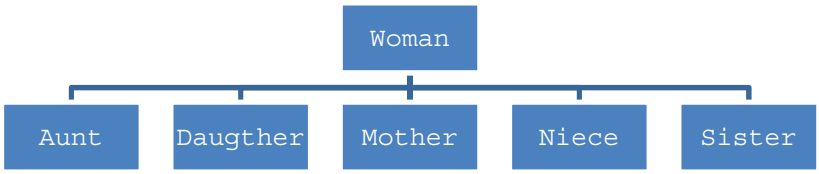


Figura 1.20 – Definição da ontologia: Woman.

Como apresentado anteriormente, esses são identificadores únicos (rdf:ID) da ontologia, ou seja, só podem existir uma vez. Porém muitas vezes, quando não existe um consenso sobre o nome ideal de um termo, é adotado um código para ser o rdf:ID e são usados rdfs:Labels para nome. Outra boa prática é usar o termo em inglês (ex: Person) e colocar rdfs:Labels para outras linguagens (seguindo o ex: Pessoa).

Com todos os termos definidos para a ontologia vamos demonstrar a criação de uma classe no Web-Protégé. Para pode editar é necessário estar logado no Web-Protégé. O Web-Protégé disponibiliza uma guia chamada "Classes" que edita as classes da ontologia. Essa guia possui por default 3 parte:

- *Classes* (Figura 1.21A): Usada exibir a hierarquia de classes;
- *Properties* (Figura 1.21B): Exibe as propriedades de uma classe;
- *Asserted Conditions* (Figura 1.21 C): Contém as restrições de uma classe;

Na Figura 1.21, para criar a classe Person é necessário selecionar a classe owl:Thing e clicar em *Create* (Figura 1.21A), irá aparecer um popup para informar o nome da nova classe. Após informar basta clicar em OK e a classe já fica disponível na hierarquia. Ao selecionar a nova classe, ela não possuirá nenhuma propriedade e possuirá uma restrição (a relação com owl:Thing). Ainda na Figura 1.21(B) foram criadas 3 propriedades de anotação para essa nova classe:

- Dois rdfs:Labels: Não tem limite de *labels*, no exemplo foi disponibilizada a tradução do termo em português e espanhol;

- Um rdfs:Comment: os comentário são interessantes para ter uma explicação do termo da ontologia;

Já nas restrições Figura 1.21(C) existem somente dois tipos que podem ser definidos:

- *Necessary*: Essa restrição pode ser lida assim: "se alguma coisa é um membro da classe então é necessário que preencha essas condições". Na Figura 1.21 (C) é informado que para pertencer a Person o individuo deve estar em owl:Thing.
- *Necessary & Sufficient*: Essa restrição agrega mais esse fato: "se alguma coisa preenche essas condições então deve ser um membro dessa classe". Ex: Qualquer individuo que pertença a Man ou Womem é uma Person.

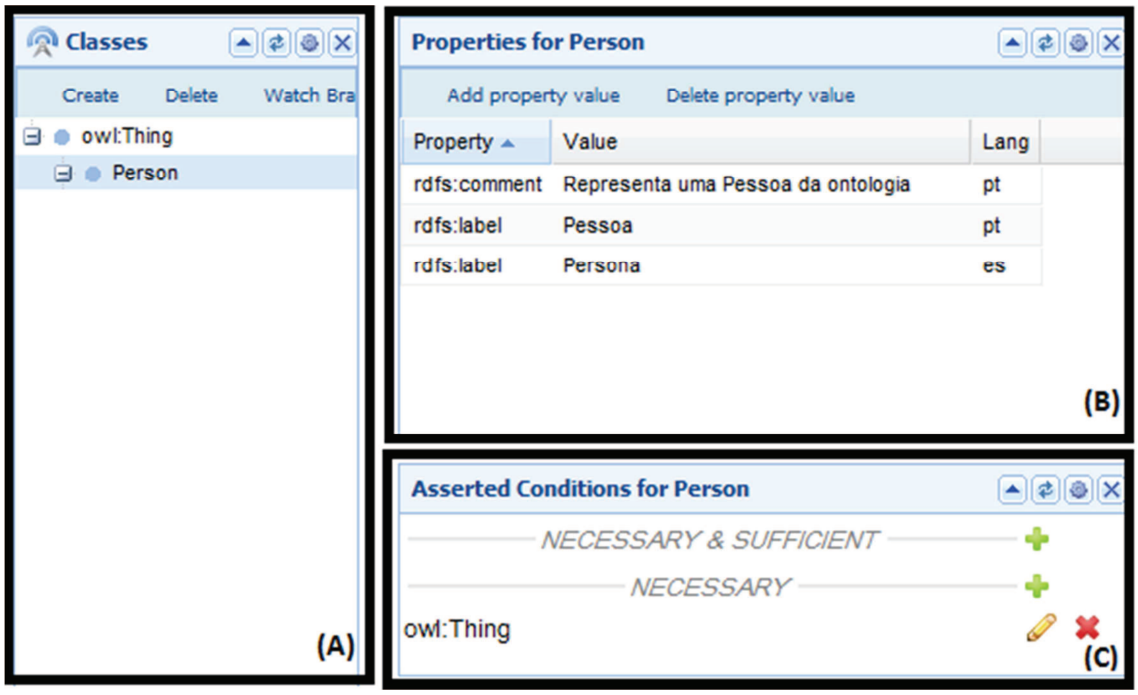


Figura 1.21 – Web-Protégé Interface para edição de classes: (A) Hierarquia de Classes; (B) Propriedades de uma classe; (C) Restrições de uma classe.

Se analisarem as Figuras 1.18, 1.19 e 1.20 pode-se perceber que, por exemplo, a classe Son criada para Man, também pertence a classe Child, porém como são identificadores únicos não podem ser inseridos duas vezes na hierarquia de classes. Então a solução é inserir uma restrição em *Necessary & Sufficient* para esses termos. No caso de Son ele já possui essa restrição para Man, então apenas basta inserir para Child também. Na tabela abaixo são apresentados os termos que necessitam de restrições:

Tabela 1.7 – Lista de restrições para estabelecer a hierarquia de classes

Classe	Tipo de Restrição	Restrição
Brother	*Necessary & Sufficient*	Sibling
Father	*Necessary & Sufficient*	Parent
Nephew	*Necessary*	Relative
Son	*Necessary & Sufficient*	Child

Classe	Tipo de Restrição	Restrição
Uncle	*Necessary*	Relative
Aunt	*Necessary*	Relative
Daugther	*Necessary & Sufficient*	Child
Mother	*Necessary & Sufficient*	Parent
Niece	*Necessary*	Relative
Sister	*Necessary & Sufficient*	Sibling

Com isso é possível montar toda a hierarquia de classes chegando ao que é apresentado na Figura 1.22.

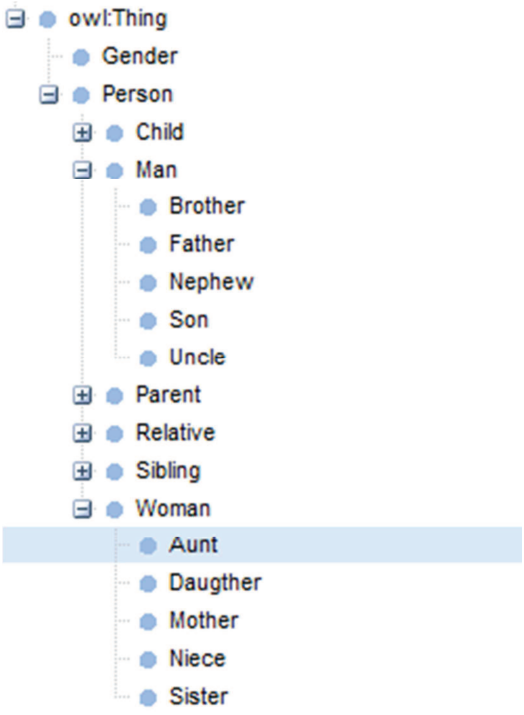


Figura 1.22 – Hierarquia Inicial de Classes.

O ideal agora é preencher todos os rdfs:Label da ontologia e disponibilizar, por exemplo, os termos da ontologia em português, porém não será apresentado neste capítulo, por não ser obrigatório. Antes de preencher as demais restrições é necessário criar as *Properties*. O Web-Protégé disponibiliza uma guia chamada "Properties" (Figura 1.23). Por default as *Properties* precisam de duas propriedades de anotação preenchidas:

- rdfs:domain: Na Figura 1.23 está selecionada a propriedade hasAunt, o domínio dessa propriedade é qualquer pessoa (Person), ou seja, qualquer pessoa pode ter uma tia;
- rdfs:range: O range já é quem pode ser classificada para aquela classe. Ex Figura 1.23: só pode ser uma tia quem for da classe Woman.

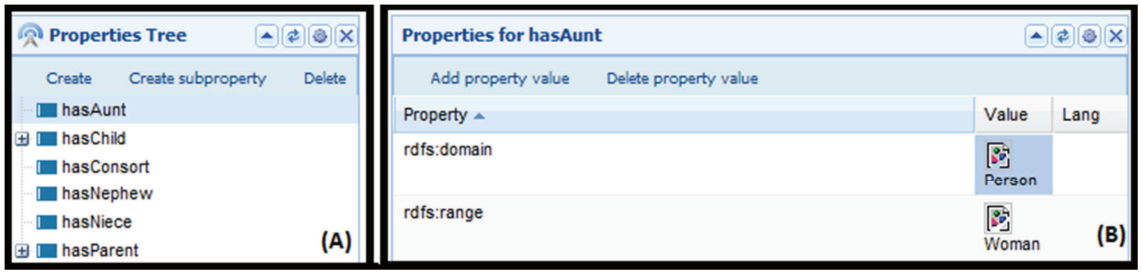


Figura 1.23 – Web-Protégé Interface para edição de propriedades: (A) Hierarquia de propriedades; (B) Propriedades de uma propriedade.

Na tabela abaixo seguem todas as propriedades da ontologia com os seus respectivos rdfs:domain e rdfs:range. Para inserir uma propriedade na Ontologia basta clicar em *Create* em (Figura 1.23 – A) e irá aparecer um popup para informar o nome da nova propriedade. Após informar basta clicar em OK e a propriedade já fica disponível. Caso queira inserir uma subpropriedade basta selecionar a propriedade pai e clicar em *Create subproperty*.

Tabela 1.8 – Lista de propriedades da ontologia

Propriedade pai	Propriedade a criar	rdfs:domain	rdfs:range
-	hasAunt	Person	Woman
-	hasChild	Person	Person
hasChild	hasDaughter	Person	Woman
hasChild	hasSon	Person	Man
-	hasConsort	Person	Person
-	hasNephew	Person	Man
-	hasNiece	Person	Woman
-	hasParent	Person	Person
hasParent	hasFather	Person	Man
hasParent	hasMother	Person	Woman
-	hasSex	Person	Gender
-	hasSibling	Person	Person
hasSibling	hasBrother	Person	Man
hasSibling	hasSister	Person	Woman
-	hasUncle	Person	Man

Ainda antes de inserir as demais restrições nas classes é necessário instanciar dois indivíduos (Female e Male) para classe Gender na guia "*Individuals*" do Web-Protégé (Figura 1.24). Para inserir um indivíduo na Ontologia é necessário primeiramente selecionar a classe referente (Figura 1.24 A). O próximo passo é clicar em *Create* em (Figura 1.24 – B) e irá aparecer um popup para informar o nome do novo indivíduo. Após informar basta clicar em OK e o indivíduo já fica disponível.

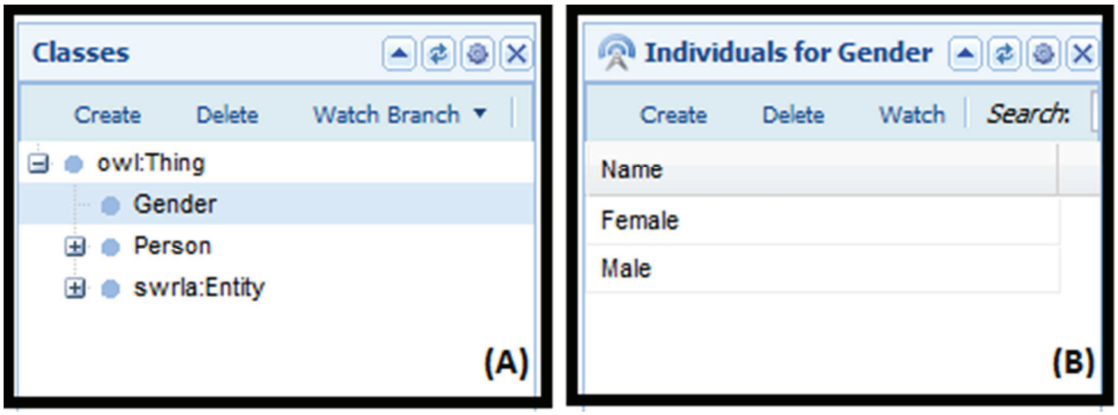


Figura 1.24 – Web-Protégé Interface para edição de indivíduos: (A) Hierarquia de classes; (B) Indivíduos de uma classe.

Após isso é possível inserir as restrições que envolvem propriedades ou indivíduos com em (Figura 1.21 C). Essas restrições podem ser vistas na Tabela abaixo.

Tabela 1.9 – Lista das demais restrições

Classe	Tipo de Restrição	Restrição
Gender	*Necessary & Sufficient*	{Female Male}
Person	*Necessary & Sufficient*	Man or Woman
Child	*Necessary & Sufficient*	Hasparent min 1
Man	*Necessary & Sufficient*	hasSex has Male
Nephew	*Necessary & Sufficient*	(hasUncle min 1) or (hasAunt min 1)
Uncle	*Necessary & Sufficient*	(hasNephew min 1) or (hasNiece min 1)
Parent	*Necessary & Sufficient*	hasChild min 1
Relative	*Necessary & Sufficient*	Child or Parent or Aunt or Nephew or Niece or Uncle or Sibling
Aunt	*Necessary & Sufficient*	(hasNephew min 1) or (hasNiece min 1)
Niece	*Necessary & Sufficient*	(hasUncle min 1) or (hasAunt min 1)
Sibling	*Necessary & Sufficient*	hasSibling min 1
Woman	*Necessary & Sufficient*	hasSex has Female

1.5.2. Construção e execução das regras SWRL

Nesta seção são apresentadas e explicadas 12 regras SWRL, as regras estão disponíveis nas Tabelas 1.10 e 1.11 abaixo:

Tabela 1.10 – Lista das demais restrições

Nº	Nome	Regra
1	Def-hasAunt	Person(?x) ∧ hasParent(?x, ?y) ∧ hasSister(?y, ?z) → hasAunt(?x, ?z)
2	Def-hasBrother	Person(?x) ∧ hasSibling(?x, ?y) ∧ Man(?y) → hasBrother(?x, ?y)

Nº	Nome	Regra
3	Def-hasDaughter	Person(?x) ∧ hasChild(?x, ?y) ∧ Woman(?y) → hasDaughter(?x, ?y)
4	Def-hasFather	Person(?x) ∧ hasParent(?x, ?y) ∧ Man(?y) → hasFather(?x, ?y)
5	Def-hasMother	Person(?x) ∧ hasParent(?x, ?y) ∧ Woman(?y) → hasMother(?x, ?y)
6	Def-hasNephew	Person(?x) ∧ hasSibling(?x, ?y) ∧ hasSon(?y, ?z) → hasNephew(?x, ?z)
7	Def-hasNiece	Person(?x) ∧ hasSibling(?x, ?y) ∧ hasDaughter(?y, ?z) → hasNiece(?x, ?z)
8	Def-hasParent	Person(?y) ∧ hasConsort(?y, ?z) ∧ hasParent(?x, ?y) → hasParent(?x, ?z)
9	Def-hasSibling	Person(?y) ∧ hasChild(?y, ?x) ∧ hasChild(?y, ?z) ∧ differentFrom(?x, ?z) → hasSibling(?x, ?z)
10	Def-hasSister	Person(?x) ∧ hasSibling(?x, ?y) ∧ Woman(?y) → hasSister(?x, ?y)
11	Def-hasSon	Person(?x) ∧ hasChild(?x, ?y) ∧ Man(?y) → hasSon(?x, ?y)
12	Def-hasUncle	Person(?x) ∧ hasParent(?x, ?y) ∧ hasBrother(?y, ?z) → hasUncle(?x, ?z)

Tabela 1.11 – Explicação das Regras em forma de texto

Nº	Explicação da Regra
1	Uma regra em que uma Pessoa X que tem um Pai/Mãe Y e que Y tem uma Irmã Z pode-se concluir que X tem uma Tia Z.
2	Uma regra em que uma Pessoa X que tem um Irmão/Irmã Y e Y seja Homem pode-se concluir que X tem um irmão Y.
3	Uma regra em que uma Pessoa X que tem um Filho/Filha Y e Y seja Mulher pode-se concluir que X tem uma filha Y.
4	Uma regra em que uma Pessoa X que tem um Pai/Mãe Y e Y seja homem pode-se concluir que X tem um Pai Y.
5	Uma regra em que uma Pessoa X que tem um Pai/Mãe Y e Y seja mulher pode-se concluir que X tem uma Mãe Y.
6	Uma regra em que uma Pessoa X que tem um Irmão/Irmã Y e que Y tem um filho Z pode-se concluir que X tem um sobrinho Z.
7	Uma regra em que uma Pessoa X que tem um Irmão/Irmã Y e que Y tem uma filha Z pode-se concluir que X tem uma sobrinha Z.
8	Uma regra em que uma Pessoa Y que tem um Cônjuge Z e que Y é Pai/Mãe de X pode-se concluir que Z também é Pai/Mãe X.
9	Uma regra em que uma Pessoa Y que tem um Filho/Filha X e que tem outro Filho/Filha Z pode-se concluir X tem um Irmão/Irmã Z.
10	Uma regra em que uma Pessoa X que tem um Irmão/Irmã Y e Y seja Mulher pode-se concluir que X tem uma irmã Y.
11	Uma regra em que uma Pessoa X que tem um Filho/Filha Y e Y seja Homem pode-se concluir que X tem um filho Y.
12	Uma regra em que uma Pessoa X que tem um Pai/Mãe Y e que Y tem um Irmão Z pode-se concluir que X tem um Tio Z.

Com as regras definidas, vamos criar e executar a inferência de uma regra. Para criar uma nova regra no SWRL Editor basta clicar no botão *New Rule* da tela de visualização (Figura 1.4). Após clicar, a ferramenta acessa a tela de composição (Figura 1.25). Para adicionar átomos a uma regra é possível a partir da navegação nas classes, propriedades e built-ins. Quando achar o termo, basta clicar sobre ele que o SWRL

Editor exibirá um popup que serve para selecionar se o predicado será inserido no antecedente ou no consequente (Figura 1.25 A). Assim que for selecionada a opção do popup, o novo predicado é carregado para edição (Figura 1.25 B).

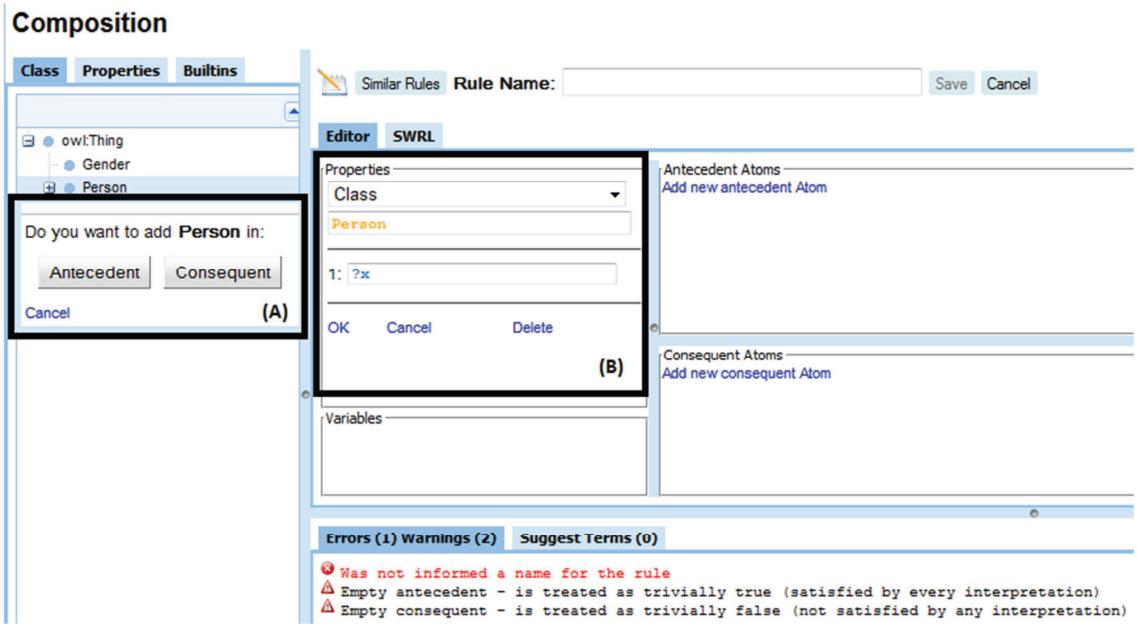


Figura 1.25 – SWRL Editor Tela de composição para nova regra: (A) Selecionar entre antecedente e consequente; (B) Átomo sendo editado.

Após informar o nome e todos os átomos da regra 1 se obtém a Figura 1.26. Na sequência é só necessário salvar a regra em "Save" e assim foi criada a primeira regra SWRL.

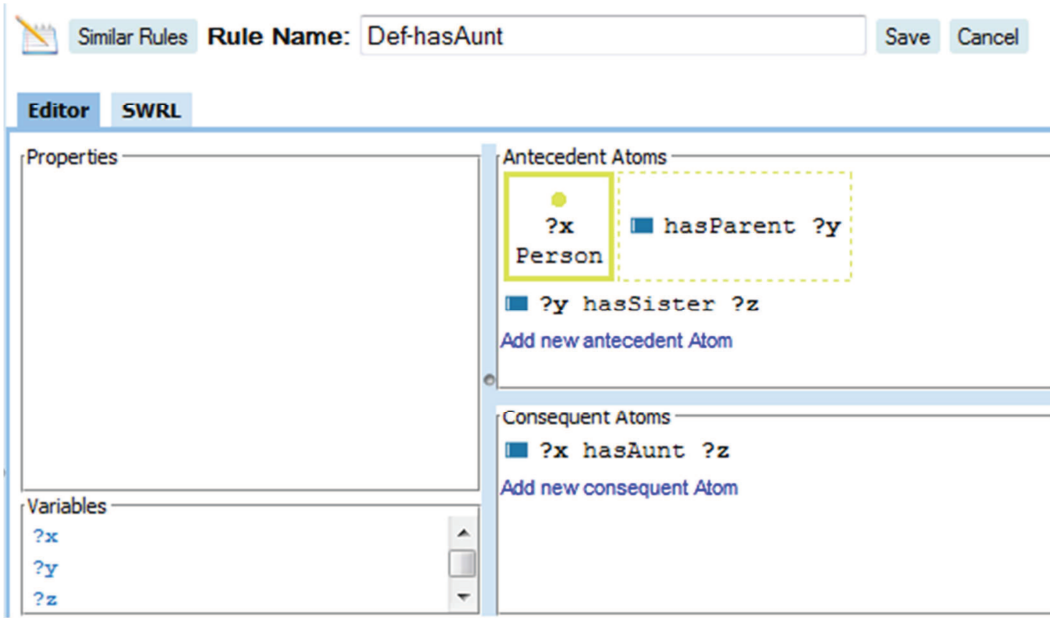


Figura 1.26 – SWRL Editor Tela de composição com uma regra preenchida.

A Regra 1 criada tem função de criar a relação de uma Pessoa ter uma Tia. Para testar regra 1 é necessário criar os seguintes indivíduos na ontologia:

- Criar o indivíduo “Filho” na classe Man.
    - Atribuir “Pai” a propriedade hasFather de “Filho”;
- Criar o indivíduo “Pai” na classe Man.
    - Atribuir “Filho” a propriedade hasSon de “Pai”;
    - Atribuir “Tia” a propriedade hasSister de “Pai”;
- Criar o indivíduo “Tia” na classe Woman.
    - Atribuir “Pai” a propriedade hasBrother de “Tia”;

Agora ao executar as regras (*Run Rules* na Figura 1.4 B) será preenchida com “Tia” a propriedade hasAunt no “Filho”

1.6. Conclusão

A Web Semântica é uma maneira de explorar a associação de significados explícitos aos conteúdos de documentos presentes na Web, para que esses possam ser processados diretamente ou indiretamente por máquinas [Berners-Lee and Fischetti 2008]. Para possibilitar esse processamento, os computadores necessitam ter acesso a coleções estruturadas de informações (dados e metadados) e a conjuntos de regras de inferência sobre esses conteúdos (que ajudem no processo de dedução automática) para que seja possível o raciocínio automatizado sobre os mesmos [Berners-Lee et all 2001).

A linguagem recomendada pelo W3C para representação e compartilhamento de ontologias na Web Semântica é o OWL. Essa linguagem foi projetada para aplicações que necessitam processar o conteúdo da informação em vez de apenas apresentar informações em texto [McGuinness and Harmelen 2004]. Infelizmente a expressividade de OWL nem sempre é suficiente para modelar todos os tipos de problemas. Especialmente OWL não tem suporte a cláusulas no formato de Horn (se a → b). Para suprir essa deficiência, a SWRL (Semantic Web Rule Language) foi criada. A linguagem SWRL é muito útil para desenvolvedores de ontologias, pois faz inferências de novos conhecimentos sobre indivíduos da ontologia (OWL).

Este capítulo apresentou conceitos ligados a Web Semântica, especialmente os ligados a regras SWRL e o SWRL Editor, um editor de regras SWRL encontrado no Web-Protégé. Por último, foi mostrado um guia passo a passo para a criação de uma ontologia, com regras, que descreve relações de parentescos entre indivíduos.

1.7. Agradecimentos

Dilvan A. Moreira agradece o apoio recebido da Fapesp para apresentação do trabalho.

1.8 Referências

Ahmedi, L., Abazi-Bexheti, L. and Kadriu, A. (2011). “A Uniform Semantic Web Framework for Co-authorship Networks”. In: IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC), p. 958-965. doi: 10.1109/DASC.2011.159.

Almeida, M. B. and Bax, M. P. (2003). Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção. Ci. Inf., Brasília, v. 32, n. 3. doi: 10.1590/S0100-19652003000300001.

Barder, F. and Nutt, W. (2003). Basic Description Logics. In: BARDER, F. et al. The Description Logic Handbook. Cambridge University Press. Capítulo 2, p. 43-90.

Berners-Lee, T. and Fischetti, M. (2008). Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web. HarperSanFrancisco. ISBN: 978-0062515872.

Berners-Lee, T., Hendler, J. and Lassila, O. (2001). The Semantic Web. Scientific American, p. 34–43. Disponível em: http://www.scientificamerican.com/article.cfm?id=the-semantic-web, acesso em Jan. 2012.

Brickley, D., Guha, R. V. and Mcbride, B. (Editors). (2004). RDF Vocabulary Description Language 1.0: RDF Schema. W3C. Disponível em: http://www.w3.org/TR/rdf-schema/, acesso em Jan. 2010.

Chandrasekaran, B., Josephson, J. R. and Benjamins, V. R. (1999). What Are Ontologies, and Why Do We Need Them?. IEEE Intelligent Systems, Piscataway, NJ, USA, v. 14, n. 1, p. 20-26. doi: 10.1109/5254.747902.

Corazzon, R. (2010). Theory and History of Ontology. A Resource Guide for Philosophers. Disponível em: http://www.formalontology.it/, acesso em Jan. 2010.

Diniz, V. and Cecconi, C. (2008). Padrões Web: Passado, presente e futuro, V Conferência Latino Americana de Software Livre. Disponível em: http://www.w3c.br/palestras/internet-web-jun-jul-2008/internetWeb_Out08.html, acesso em Dez. 2009.

Feigenbaum, L., Herman, I., Hongsermeier, R, Neumann, E. and Stephens, S. (2007). The Semantic Web in action, Scientific American, p. 64-71. Disponível em: http://www.ncbi.nlm.nih.gov/pubmed/18237102, acesso em Jan. 2010.

Fensel, D., Van Harmelen, F., Horrocks, I., Mcguinness, D. L. and Patel-Schneider, P. F. (2001). OIL: An Ontology Infrastructure for the Semantic Web. IEEE Intelligent Systems, v. 16, n. 2, p. 38-45. doi: 10.1109/5254.920598.

Golbreich, C., Horridge, M., Horrocks, I., Motik, B. and Shearer, R. (2007). OBO and OWL: leveraging semantic web technologies for the life sciences. In: Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference. Busan, Korea, p. 169-182.

Grau, B. C., Horrocks, I., Motik, B., Parsia, B., Patel-Scjmeoder, P. and Sattler, U. (2008). OWL 2: The next step for OWL. Journal of Web Semantics: Science, Services and Agents on the World Wide Web, Amsterdam, Netherlands, v. 6, n. 4, p. 309–322. doi: 10.1016/j.websem.2008.05.001.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowl. Acquis. London, UK, v. 5, n. 2, p. 199-220. doi: 10.1006/knac.1993.1008.

Guarino, N. and Giaretta, P. (1995). Ontologies and Knowledge Bases: Towards a Terminological Clarification. Journal Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing. v. 1, n. 9, p. 25-32.

Hassanpour, S., O'connor, M. J. and Das, A. K. (2009). Exploration of SWRL Rule Bases through Visualization, Paraphrasing, and Categorization of Rules. In: Proceedings of the 2009 International Symposium on Rule Interchange and Applications (RuleML 2009), Las Vegas, Nevada, p. 246–261. doi: 10.1007/978-3-642-04985-9_23.

Hassanpour, S., O'connor, M. J. and Das, A. K. (2010). Visualizing Logical Dependencies in SWRL Rule Bases. The International RuleML Symposium on Rule Interchange and Applications, Washington, DC, p. 259-272.

Hassanpour, S., O'Connor, M. J. and Das, A. K. (2011). "Evaluation of Semantic-Based Information Retrieval Methods in the Autism Phenotype Domain". In: AMIA Annual Symposium.

Heflin, J. (2004). W3C Recommendation: OWL Web Ontology Language Use Cases and Requirements. Disponível em: www.w3.org/TR/2004/REC-webont-req-20040210. acesso Fev. 2012.

Horridge, M., Knublauch, H., Rector, A., Stevens, R. and Wroe, C. (2004). A practical guide to building OWL ontologies using the protégé-OWL plugin and CO-ODE tools edition 1.0. University Of Manchester. Disponível em: http://www.co-ode.org/resources/, acesso em Out. 2009.

Horrocks, I, Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B. and Dean, M. (2004). SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C. Disponível em: http://www.w3.org/Submission/SWRL/, acesso em Jan. 2010.

Klyne, G., Carrol, J. J. and Mcbride, B. (Editors). (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C. Disponível em: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/, acesso em Set. 2009.

Lassila, O. and Swick, R. (Editors). (2004). Resource Description Framework (RDF) model and syntax specification. W3C. Disponível em: http://www.w3.org/TR/REC-rdf-syntax/, acesso em Out. 2009.

Levy, M., O'Connor, M. J. and Rubin, D. L. (2009). "Semantic Reasoning with Image Annotations for Tumor Assessment". In: AMIA Annual Symposium.

McGuinness, D. L. and Harmelen, F. (Editors). (2004). OWL Web Ontology Language Overview. W3C. Disponível em: http://www.w3.org/TR/2004/REC-owl-features-20040210/, acesso em Jun. 2009.

Noy, N. F. and McGuinness, D. L. (2001). Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05.

O'connor, M., Knublauch, H., Tu, S., Grosof, B., Dean, M., Grosso, W. and Musen, M. (2005). Supporting Rule System Interoperability on the Semantic Web with SWRL Fourth International Semantic Web Conference (ISWC2005), Galway, Ireland. Disponível em: http://bmir.stanford.edu/file_asset/index.php/1157/BMIR-2005-1080.pdf, acesso em Maio de 2012.

Orlando, J. P. (2012). Usando aplicações ricas para internet na criação de um ambiente para visualização e edição de regras SWRL. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos.

Recuperado em 2012-08-26, de http://www.teses.usp.br/teses/disponiveis/55/55134/tde-25072012-101810/

Ouellet, R. and Ogbuji, U. (2002). Introduction to DAML: Part I. Disponível em: http://www.xml.com/pub/a/2002/01/30/daml1.html, acesso em Jan. 2010.

Rossello-Busquet, A., Brewka, L. J., Soler, J. and Dittmann, L. (2011). "OWL Ontologies and SWRL Rules Applied to Energy Management". In: International Conference on Computer Modelling and Simulation (UKSim), p. 446-450. doi: 10.1109/UKSIM.2011.91.

Sadoun, D., Dubois, C., Ghamri-Doudane, Y. and Grau, B. (2011). "An Ontology for the Conceptualization of an Intelligent Environment and Its Operation". In: Mexican International Conference on Artificial Intelligence (MICAI), p. 16-22. doi: 10.1109/MICAI.2011.32.

Seo, S., Kwon, A., Kang, J. and Hong, J. W. (2011). "OSLAM: Towards ontology-based SLA management for IPTV". In: IEEE International Symposium on Integrated Network Management (IM), p. 1228-1234. doi: 10.1109/INM.2011.5990570.

Shadbolt, N., Hall, W. and Berners-Lee, T. (2006). The Semantic Web Revisited. IEEE Intelligent Systems, v. 21, n. 3, p. 96-101. doi: 10.1109/MIS.2006.62.

Silva, A. R. (2012). Aprimorando a visualização e composição de regras SWRL na Web. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos. Recuperado em 2012-08-26, de http://www.teses.usp.br/teses/disponiveis/55/55134/tde-27022012-142801/

Smith, B. and Welty, C. (2001). Ontology: Towards a New Synthesis. In: FOIS'01: Proceedings of the international conference on Formal Ontology in Information Systems, Ogunquit, Maine, USA, p. 3-9. doi: 10.1145/505168.505201.

Smith, M. K., Welty, C. and McGuinness, D. L. (Editors). (2004). OWL Web Ontology Language Guide. W3C. Disponível em: http://www.w3.org/TR/owl-guide/, acesso em Nov. 2008.

Staab, S., Maedche, A. and Handschuh, S. (2001). An annotation framework for the semantic web. In: Proceedings of the First Workshop on Multimedia Annotation, Tokyo, Japan, p. 30-31. doi: 10.1.1.25.910.

Studer, R., Benjamins, R. and Fensel, D. (1998). Knowledge Engineering: Principles and Methods. IEEE Transactions on Data and Knowledge Engineering, v. 25(1-2), p. 161-197. doi: 10.1.1.41.1007.

Su, X. and Ilebrekke, L. (2002). A Comparative Study of Ontology Languages and Tools. In: CAiSE'02:Proceeding of the 14th Conference on Advanced Information Systems Engineering, Toronto, Canada, v. 2348, p. 761-765. doi: 10.1007/3-540-47961-9_62.

SWRLLanguage. (2012). SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Disponível em: http://protege.cim3.net/cgi-bin/wiki.pl?SWRLLanguageFAQ, acesso em Maio 2012.

Tudorache, T., Vendetti, J. and Noy, N. F. (2008). Web-Protégé: A Lightweight OWL Ontology Editor for the Web. In: OWLED 2008: OWL: Experiences and Directions, Karlsruhe, Germany. doi: 10.1.1.142.8568.

Uschold, M. and Grüninger, M. (1996). Ontologies: principles, methods and applications. The Knowledge Engineering Review, v. 11, n. 2, p. 93-155. doi: 10.1017/S0269888900007797.

Uschold, M. and Grüninger, M. (2004). Ontologies and semantics for seamless connectivity. SIGMOD Rec, NY, USA v. 3, n. 4, p. 58-64. doi: 10.1145/1041410.1041420.

Vesin, B., Ivanovic, M., Klasnja-Milicevic, A. and Budimac, Z. (2011). "Rule-based reasoning for altering pattern navigation in Programming Tutoring System". In: International Conference on System Theory, Control, and Computing (ICSTCC), p. 1-6.

W3C OWL Working Group. (2009). OWL 2 Web Ontology Language Document Overview. W3C. Disponível em: http://www.w3.org/TR/owl2-overview/, acesso em Dez. 2009.

Wusheng, W., Weiping, L., Zhonghai, W., Weijie, C. and Tong, M. (2011). "An Ontology-Based Context Model for Building Context-Aware Services". In: International Conference on Intelligent Systems, Modelling and Simulation (ISMS), p. 296-299. doi: 10.1109/ISMS.2011.52.

Zacharias, V. (2008). Development and verification of rule based systems – a survey of developers. In: Rule Representation, Interchange and Reasoning on the Web: International Symposium, Orlando, Florida, USA, v. 5321, p. 6-16. doi:10.1007/978-3-540-88808-6_4.

Capítulo

2

Software as a Service: Desenvolvendo Aplicações Multi-tenancy com Alto Grau de Reuso

Josino Rodrigues Neto, Vinicius Cardoso Garcia, Andrêza Leite de Alencar, Júlio César Damasceno, Rodrigo Elia Assad e Fernando Trinta

Abstract

Software as a Service (SaaS) represents a new paradigm and business model with witch companies don't need buy and maintain their own IT infrastructure. Instead, they acquire a software as service of third party, obtaining significant benefits, especially regarding the reduction of infrastructure maintenance costs. The aim of this work is to present the main concepts related to multi-tenancy architecture, and an approach to implementation of Software as a Service. During this work the key technologies will be presented associated with the subject and a practical example of implementing a multi-tenancy application using Grails Framework and reusable components.

Resumo

Software como serviço (SaaS) representa um novo paradigma e um modelo de negócios onde as empresas não precisam comprar e manter sua própria infraestrutura de TI. Ao invés disso, elas adquirem um serviço de software de terceiros, obtendo assim consideráveis benefícios, principalmente no que tange a redução de custos de manutenção dessa infraestrutura. O objetivo desse minicurso é apresentar os principais conceitos relacionados à arquitetura multi-tenancy, uma das abordagem para implementação de Software como Serviço. Durante esse trabalho serão apresentadas as principais tecnologias associadas ao assunto e um exemplo prático da implementação de um aplicativo multi-tenancy utilizando o framework Grails e componentes reutilizáveis.

2.1. Introdução

Em 1969, Leonard Keinrock, um dos cientistas chefe da ARPANET, precursora da internet, disse: "A partir de agora, redes de computadores estão ainda na sua infância, mas a medida que crescem e tornam-se sofisticadas, nós iremos provavelmente ver a disseminação do 'computador utilitário' que, como telefones e energia elétrica, irão servir casas e escritórios por todo país" [30]. Essa visão de computação utilitária baseada no modelo de fornecimento de serviços antecipa a transformação massiva de toda a indústria de computação nos últimos anos. Serviços de computação estarão prontamente disponíveis sob demanda, como os outros serviços utilitários disponíveis atualmente como água, energia e telefone. Similarmente, os usuários (consumidores) precisam pagar os provedores somente quando eles acessam os serviços de computação. Além disso, consumidores não precisam mais realizar altos investimentos ou enfrentar a dificuldade de construir e manter complexas infra-estruturas de TI para instalar um aplicativo em suas dependências.

Os profissionais de desenvolvimento de software estão enfrentando inúmeros novos desafios com o objetivo de criar serviços que atendam a milhões de consumidores ao invés de prover um software que seja executado em computadores pessoais. E ao longo dos anos, o surgimento de avanços tecnológicos como processadores multicore e ambientes de computação em rede como Cluster Computing [62], Grid Computing [25], computação P2P [57] e o mais recente Cloud Computing [42], tornam esse objetivo cada vez mais factível.

Os serviços de computação citados anteriormente precisam ser altamente confiáveis, escaláveis e dinâmicos para suportar acesso ubíquo e fornecer a possibilidade de composição de outros serviços [15]. Além disso, consumidores devem ter a capacidade de determinar o nível de serviço requerido através de parâmetros de QoS (Quality of Service) e SLA (Service Level Agreement) [7].

Cloud Computing foi o último desses paradigmas a emergir e promete entregar serviços confiáveis através da nova geração de datacenters que são construídos utilizando tecnologias de virtualização de processamento e armazenamento. Consumidores estarão habilitados a acessar aplicações e dados da "cloud" em qualquer lugar do mundo e sob demanda, como é o caso do Gmail[5], Google Docs[6], Office Web Apps[7], Dropbox[8], dentre outros.

[5] http://www.gmail.com
[6] http://docs.google.com
[7] http://office.microsoft.com/web-apps
[8] http://dropbox.com

2.2. Conceitos Fundamentais

2.2.1. Cloud Computing

Atualmente não existe uma definição oficial do que seja *Cloud Computing*. Nesse trabalho usaremos a definição proposta pelo National Institute of Standards and Technology(NIST), que define Cloud Computing como um modelo que permite, de forma conveniente, o acesso a um pool de recursos computacionais compartilhados (rede, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente provisionados e liberados com o mínimo de esforço de gerenciamento ou interação do provedor de serviço.

Ainda segundo o NIST, *Cloud Computing* é composta por cinco características essenciais [42]:

- **Alocação de recursos sob demanda:** O usuário pode adquirir unilateralmente recursos computacionais, como tempo de processamento do servidor ou armazenamento, através da rede na medida em que necessite e sem precisar de interação humana com os provedores de cada serviço;

- **Amplo acesso a rede:** recursos estão disponíveis através da rede e podem ser acessados por meio de mecanismos que funcionem em plataformas heterogêneas (por exemplo, telefones celulares, laptops e PDA);

- ***Pooling* de recursos:** os recursos do provedor de computação são agrupados para atender vários consumidores através de um modelo *multi-tenancy*, com diferentes recursos físicos e virtuais atribuídos dinamicamente e novamente de acordo com a demanda do consumidor. Há um senso de independência local em que o cliente geralmente não tem nenhum controle ou conhecimento sobre a localização exata dos recursos disponibilizados, mas pode ser capaz de especificar o local em um nível maior de abstração (por exemplo, país, estado ou data center). Exemplos de recursos incluem o armazenamento, processamento, memória, largura de banda de rede e máquinas virtuais;

- **Elasticidade rápida:** recursos podem ser adquiridos de forma rápida e elástica, em alguns casos automaticamente, caso haja a necessidade de escalar com o aumento da demanda, e podem também ser liberados, na retração dessa demanda. Para os usuários, os recursos disponíveis para uso parecem ser ilimitados e podem ser adquiridos em qualquer quantidade e a qualquer momento; e

- **Serviço medido:** sistemas em nuvem automaticamente controlam e otimizam a utilização dos recursos, alavancando a capacidade de medição em algum nível de abstração adequado para o tipo de serviço (por exemplo, armazenamento, processamento, largura de banda, e contas de usuários ativos). Uso de recursos

pode ser monitorado, controlado e relatado a existência de transparência para o fornecedor e o consumidor do serviço utilizado.

Ainda segundo o NIST [42], Cloud Computing é dividido em três modelos de serviço (Figura 2.1):

• **Software como Serviço (SaaS):** A capacidade fornecida ao consumidor é a de usar as aplicações do fornecedor em uma infraestrutura da nuvem. As aplicações são acessíveis de vários dispositivos cliente através de uma interface thin client, como por exemplo um browser web. O consumidor não administra ou controla a infraestrutura básica, incluindo rede, servidores, sistemas operacionais, armazenamento, ou mesmo capacidades individuais da aplicação. Mesmo assim ainda é possível definir algumas configurações específicas para o usuário na aplicação;

• **Plataforma como Serviço (PaaS):** A capacidade fornecida ao consumidor é a de realizar deploy de uma aplicação em uma infraestrutura pré-definida ou adquirir aplicações criadas usando linguagens de programação e as ferramentas suportadas pelo provedor de PaaS. O consumidor não administra ou controla a infraestrutura básica como rede, servidores, sistemas operacionais, ou armazenamento, mas tem controle sobre os aplicativos utilizados e, eventualmente, hospedagem de aplicativos e configurações de ambiente; e

• **Infraestrutura como Serviço (IaaS) :** A capacidade prevista para o consumidor é a de processamento, armazenamento, redes e outros recursos computacionais fundamentais para que o consumidor seja capaz de implantar e executar programas arbitrários. Esses recursos podem incluir sistemas operacionais e aplicativos. O consumidor não administra ou controla a infraestrutura de nuvem subjacente, mas tem controle sobre os sistemas operacionais, armazenamento, aplicativos implantados, e, eventualmente, o controle limitado de componentes de rede (por exemplo, firewall).

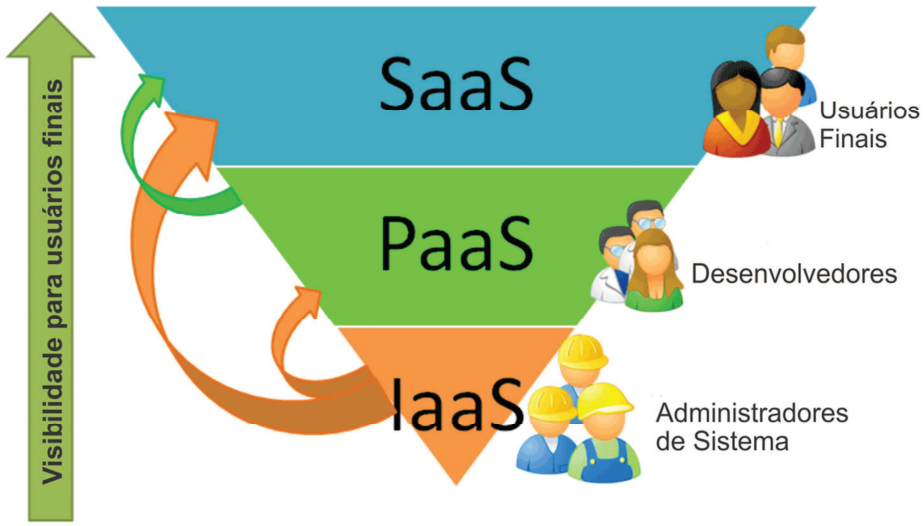


Figura 2.1. Modelos de serviço de *Cloud Computing*

Nesse trabalho teremos como foco principal os tópicos relacionados a SaaS, embora sejam mencionados alguns conceitos associados à PaaS e IaaS. O motivo disso é que multitenancy é um conceito existente dentro do contexto de Software como serviço.

2.2.2. Software como Serviço (SaaS)

Nos últimos anos muitas empresas têm saído do modelo de entrega de software empacotado para o modelo de fornecimento de software na web [23]. Essas aplicações entregues através da web vão desde emails a calendários, sistemas colaborativos, publicações online, processadores de texto simples, aplicações para negócios e aplicações para uso pessoal.

Salesforce.com [35], por exemplo, criou um aplicativo para CRM (*Customer Relationship Management*) e o configurou não como um software empacotado, mas como software rodando sobre servidores acessível através do browser. Quando fez isso, criou a própria plataforma in-house para entregar o software como serviço a seus consumidores.

Logo em seguida Salesforce.com criou o *AppExchange*, uma plataforma de integração aberta para outras empresas de software construírem produtos utilizando algumas funcionalidades do CRM do *salesforce*. Pouco tempo depois *Salesforce* estendeu o conceito de plataforma aberta como force.com, um ambiente de desenvolvimento e deployment usando a infraestrutura de SaaS do *Salesforce*. Posteriormente, algumas outras empresas ingressaram nesse mercado, a Amazon com o EC2 [2] e Google com o Google AppEngine[9], abrindo suas infraestruturas de cloud para hospedarem aplicações de terceiros, além de produzirem serviços online.

Amazon, em especial, vem se tornando bastante atraente porque possui uma rica infraestrutura para suportar operações de varejo online e tem oferecido vários serviços para usuários de cloud como: *data storage*, processamento, fila de mensagens, *billing* e etc. O artigo *"Amazon S3's Amazing Growth"* [24] menciona o crescimento vertiginoso no uso do serviço S3 da Amazon [2] nos últimos anos, é possível ver o gráfico de crescimento desse serviço na Figura 2.2.

Cloud Computing e SaaS também parecem ser eficientes tanto para usuários quanto para fornecedores de software. Vários consumidores podem usar a mesma instalação do software e consequentemente isso melhora as taxas de utilização de hardware e rede. Por exemplo, Amazon[10] e Google[11] têm enormes *datacenters* que eles não utilizam completamente o tempo todo. Eles podem executar seus próprios produtos enquanto hospedam aplicações de outras empresas. Empresas como Amazon, Google e

[9] https://appengine.google.com
[10] http://aws.amazon.com
[11] http://google.com

Salesforce geralmente garantem a qualidade de serviço para todos os seus consumidores de cloud através de SLAs (*Service Level Agreements*) detalhadas.

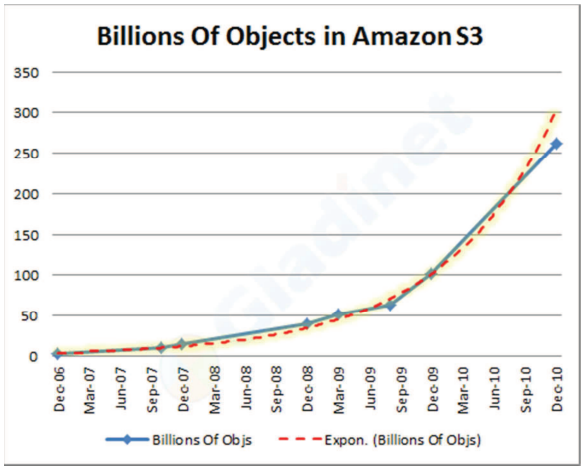


Figura 2.2. Amazon S3's Growth (Fonte: [24])

Os serviços providos por esses grandes players do mercado permitem que empresas possam desenvolver software no modelo SaaS e hospedar em algum de seus datacenters pagando apenas pelo que usarem. Isso torna-se um atrativo principalmente para Startups e empresas com poucos recursos financeiros.

Segundo Harris et al.[21] existem três tipos diferentes de aplicações SaaS:

- **Single-instance:** as aplicações proveem serviços para um único cliente e executam em um servidor exclusivo. Nesse caso cada servidor web possui uma única instância da aplicação. Essa pode considerada a abordagem com maior desperdício de recursos, dado que um servidor pode, por exemplo, executar com apenas 10% de sua capacidade;

- **Multi-instance:** essas aplicações executam em ambientes onde o servidor web é compartilhado. Nessa abordagem uma cópia da aplicação é inicialmente configurada para cada cliente, e então cada cópia é implantada como um contexto no mesmo servidor web; e

- **Multi-tenancy:** provê uma única aplicação compartilhada por vários clientes. Nessa abordagem várias aplicações "virtuais" são criadas na mesma instância.

Implementar o conceito de SaaS nem sempre é tão simples como parece. Chong[12] propõe 4 níveis de maturidade para aplicações que utilizam o modelo de SaaS:

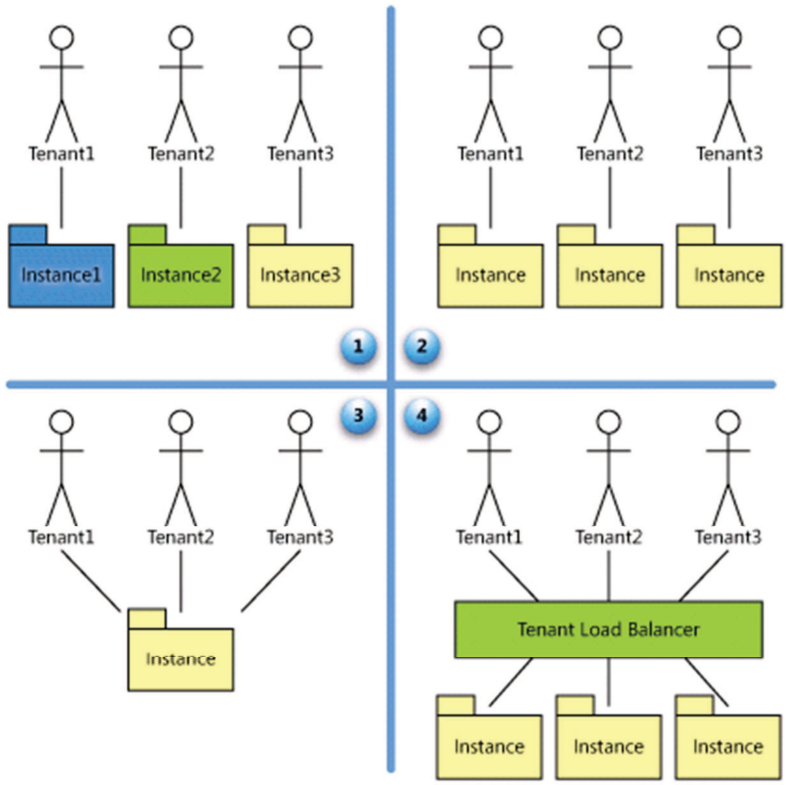


Figura 2.3. Níveis de Maturidade SaaS (Fonte: [12])

- **Nível 1 Ad-Hoc/Personalizado:** O primeiro nível de maturidade é semelhante ao modelo de entrega de software do provedor de serviços de aplicativos (ASP Application Service Provider) tradicional, que data da década de 1990. Nesse nível, cada cliente tem a sua própria versão personalizada do aplicativo hospedado e executando nos servidores do provedor. Pensando em arquitetura, software nesse nível de maturidade é muito semelhante aos softwares corporativos vendidos tradicionalmente, em que diferentes usuários de uma organização conectam a uma instância quaisquer outras instâncias ou processos que o host esteja executando para os seus outros clientes;

- **Nível 2 Configurável:** No segundo nível de maturidade, o fornecedor hospeda uma instância separada do aplicativo para cada tenant. Enquanto no primeiro nível cada instância é personalizada individualmente para o tenant, neste nível todas as instâncias utilizam a mesma implementação de código e o fornecedor atende as necessidades dos clientes fornecendo opções de configuração detalhadas que permitem ao cliente alterar a aparência e o comportamento do aplicativo para os seus usuários. Apesar de serem idênticas a nível do código, cada instância permanece totalmente isolada de todas as demais;

- **Nível 3 Configurável e eficiente para vários tenants:** No terceiro nível de maturidade, o fornecedor executa uma única instância que serve a todos os clientes. Metadados configuráveis são usados para fornecer uma experiência de usuário e um conjunto de recursos exclusivos para cada instância. Políticas de

autorização e de segurança garantem que os dados de cada cliente sejam mantidos separados dos dados de outros clientes e que, da perspectiva do usuário final, não exista qualquer indicação de que a instância do aplicativo esteja sendo compartilhada entre vários tenants; e

- **Nível 4 Escalonável, configurável e eficiente para vários tenants:** No quarto e último nível de maturidade, o fornecedor hospeda vários clientes em um ambiente com balanceamento de carga. Os dados de cada cliente são mantidos separados e com metadados configuráveis fornecendo uma experiência do usuário e um conjunto de recursos exclusivos para cada cliente. Um sistema de SaaS é escalonável para um número de clientes arbitrariamente grande, uma vez que a quantidade de servidores e instâncias no lado do fornecedor pode ser aumentada ou diminuída conforme necessário para corresponder à demanda sem a necessidade de remodelar a arquitetura aplicativo, além disso as alterações ou correções podem ser transmitidas para milhares de *tenants* tão facilmente quanto para um único *tenant*.

Normalmente se esperaria que o quarto nível fosse a meta definitiva para qualquer aplicativo de SaaS, mas não é sempre assim. É necessário verificar as necessidades operacionais, arquiteturais e de negócio relacionadas à aplicação. Uma abordagem *singletenant* faz sentido financeiramente? O seu aplicativo pode ser feito para executar em uma única instância lógica? Você pode garantir que a qualidade de serviço desejada por cada cliente seja atendida? Essas são questões que devem ser respondidas quando se pretende adotar o modelo SaaS.

2.2.3. Multi-tenancy

Multi-tenancy é uma abordagem organizacional para aplicações SaaS. Bezemer e Zaidman [8] definem multi-tenancy como aplicações que permitem a otimização no uso dos recursos de hardware, através do compartilhamento de instâncias da aplicação e da instância do banco de dados, enquanto permite configurar a aplicação para atender às necessidades do cliente como se estivesse executando em um ambiente dedicado. Tenant é uma entidade organizacional que aluga uma aplicação multi-tenancy. Normalmente, um tenant agrupa um número de usuários que são os stakeholders da organização.

A definição anterior foca no que nós consideramos aspectos importantes em aplicações multi-tenancy:

- Possibilidade de compartilhamento de recursos de hardware, permitindo a redução de custos [58];

- Alto grau de configurabilidade, permitindo que cada consumidor customize sua própria interface e seu workflow na aplicação [47, 26]; e

- Uma abordagem arquitetural na qual os tenants fazem uso de uma única aplicação e banco de dados [34].

É possível que algumas pessoas confundam multi-tenancy com o conceito de multi-usuário. Multi-tenancy não é multi-usuário. Em uma aplicação multi-usuário nós assumimos que os usuários estão usando a mesma aplicação com opções de acesso limitadas e que essa instância da aplicação é utilizada por apenas um único cliente. Nesse caso podemos ter vários usuários com o perfil "gerente", com o perfil "vendedor", ou com qualquer perfil de usuário necessário para o funcionamento da aplicação. Em uma aplicação multi-tenancy nós assumimos que existe uma instância da aplicação que atende a vários clientes (tenants) e possui um alto grau de configuração. Dependendo da definição dessas configurações, dois tenants podem possuir aparência e workflows diferentes. Um argumento adicional para essa distinção é que o SLA para cada tenant pode ser diferente [39].

Uma abordagem mais profunda sobre o tema multi-tenancy será apresentado na seção seguinte.

2.3. Propostas de Arquitetura Multi-tenancy

Um ponto importante durante o desenvolvimento de software em qualquer segmento é conhecer implementações semelhantes já realizadas por outros desenvolvedores. Partindo desse princípio essa seção apresenta propostas de arquitetura de software que auxiliem no desenvolvimento de aplicações multi-tenancy. Foram encontradas propostas de arquitetura para aplicações multi-tenancy [31, 9, 8, 49, 56, 63, 28, 11, 32] e para plataformas de suporte a multi-tenancy [60, 48, 5].

Dentre essas propostas algumas já foram aplicadas à indústria, como é o caso do Force.com [60] e EXACT [9]. Force.com é uma plataforma de desenvolvimento de software que utiliza arquitetura dirigida à metadados para construção de aplicações multitenancy. Nessa arquitetura tudo que é exposto para os desenvolvedores e usuários da aplicação é internamente representado como metadado. Formulários, relatórios, workflows, priviégios de acesso, customizações específicas do tenant e lógica de negócio, tudo é armazenado como metadados. Outro exemplo de arquitetura dirigida à metadados pode ser encontrada em [48]. Bezemer et al. [9] apresentam uma arquitetura utilizada no desenvolvimento de aplicações na empresa EXACT, uma empresa de desenvolvimento de software especializada em ERP, CRM e aplicações financeiras. Em sua arquitetura os autores apresentam 3 componentes básicos para a implementação de aplicações multitenancy: componente de autenticação, customização e banco de dados. Além disso os autores apresentam um estudo de caso e uma lista de lições aprendidas.

Outros autores apresentam Arquiteturas Orientadas a Serviço (SOA ServiceOriented Architecture) que implementam requisitos de multi-tenancy como é o caso de Jing and Zhang [28],Azeez et al. [5] e Pervez et al. [49]. Jing and Zhang [28] apresentam OSaaS, uma arquitetura que utiliza tecnologias SOA para desenvolvimento de aplicações SaaS. Azeez et al. [5] apresentam uma arquitetura para implementar

multi-tenancy a nível de SOA, que permite a usuários executar seus serviços e outros artefatos SOA em um framework multi-tenancy SOA. Já Pervez et al. [49] apresentam uma arquitetura para SaaS que foca nos aspectos de segurança e balanceamento de carga em ambiente de cloud computing.

Além das propostas de arquitetura citadas anteriormente, foi encontrado também uma proposta de estilo arquitetural, o SPOSAD (*Shared, Polymorphic, Scalable Application and Data*), que é descrito em [31] e [32]. Um estilo arquitetural define os tipos de elementos que podem aparecer em uma arquitetura e as regras que regem a sua interconexão. O SPOSAD descreve componentes, conectores e elementos de dados de uma arquitetura multi-tenancy, bem como restrições impostas nesses elementos. Os autores também apresentam os benefícios do uso dessa arquitetura e informações que podem auxiliar em decisões de projeto.

Tsai et al. [56] apresentam uma arquitetura de duas camadas que foca em escalabilidade, que trabalha a nível de serviço e aplicação para economizar o uso de recursos, e a idéia chave é aumentar os recursos somente onde houver gargalos. Várias técnicas de duplicação de recursos computacionais são propostas, incluindo duplicação preguiçosa e duplicação pro-ativa para alcançar a melhor performance do sistema. Além da arquitetura esse trabalho apresenta um algoritmo para alocação de recursos para cluster em ambientes de cloud. Já Yuanyuan et al. [63] apresentam uma arquitetura multi-tenancy para sistemas de suporte a negócios (BSS Business Support System) e discutem brevemente uma abordagem para alcançar configurabilidade, seguranca e escalabilidade na arquitetura. Focando em escalabilidade de dados, os autores propõem um particionamento horizontal baseado em grupos de entidades pela análise das relações entre as entidade de negócio do sistema.

Calero et al. [10] apresentam a arquitetura de um sistema de autorização multitenancy apropriado para um serviço de middleware para PaaS. Cada empresa pode provê vários serviços de cloud que podem colaborar com outros serviços tanto da mesma organização quanto de organizações diferentes. Esse sistema de autorização suporta acordos de colaboração entre empresas (também conhecidos como federações).

2.4. Componentes Básicos de uma Aplicação Multi-tenancy

Multi-tenancy afeta quase todas as camadas de uma aplicação típica e possui um grande potencial para ser implementado como interesse transversal [43]. Para reduzir a complexidade do código, a implementação de requisitos multi-tenancy deve ser separada da lógica de negócio o máximo possível. Caso contrário, a manutenção pode se tornar um pesadelo, porque:

- Implementar código de requisitos da arquitetura multi-tenancy juntamente com a lógica de negócio dos tenants, exige que todos os desenvolvedores sejam reeducados para entenderem os conceitos de multi-tenancy; e

- Misturar multi-tenancy com código de lógica de negócio dos tenants leva ao aumento da complexidade da implementação, pois é mais difícil manter o controle de onde o código multi-tenancy é introduzido.

Estes problemas podem ser superados integrando cuidadosamente multi-tenancy na arquitetura. No restante desta seção, descrevemos os componentes da arquitetura abordada por Bezemer et al. [9] para implementação de multi-tenancy como um interesse transversal. A Figura 2.4 apresenta a descrição dessa aplicação e as subseções seguintes descrevem cada componente da aplicação.

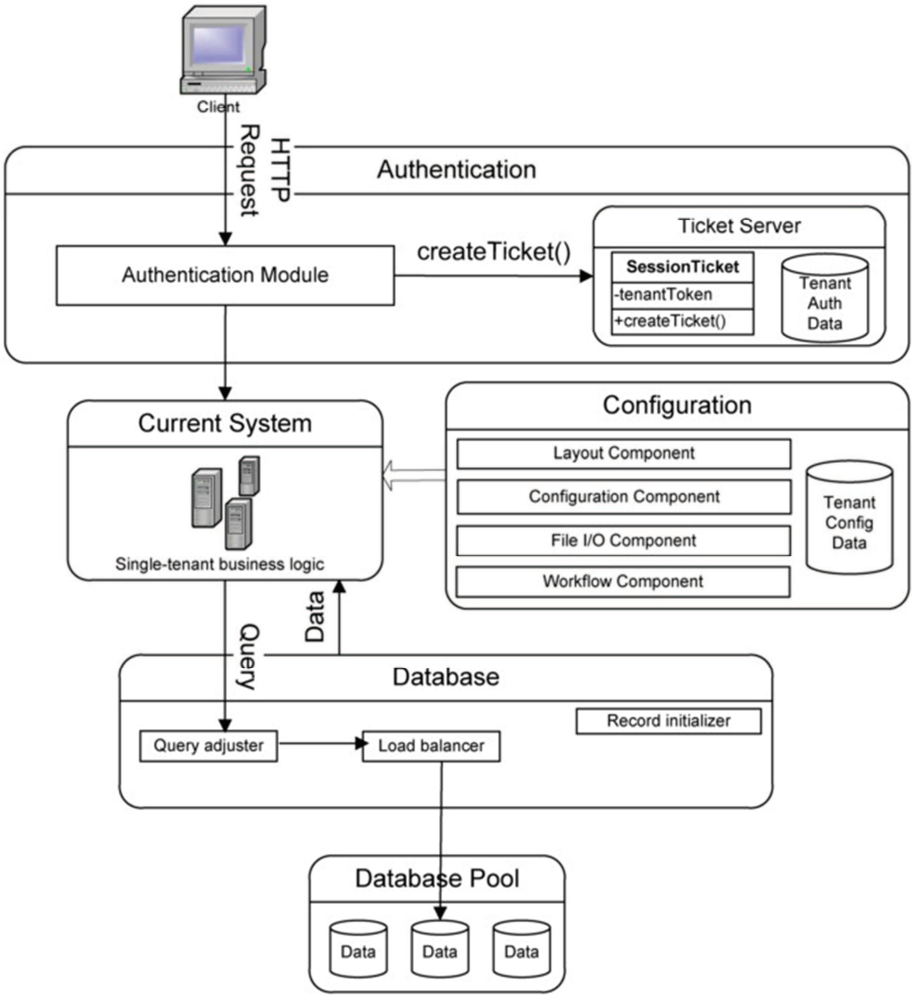


Figura 2.4 Arquitetura de Referência adotada (Fonte: [9])

2.4.1. Autenticação

Devido a uma aplicação multi-tenancy ter apenas uma instância da aplicação e do banco de dados, todos os tenants usam o mesmo ambiente físico. A fim de oferecer a customização do ambiente e ter certeza de que os tenants podem acessar somente os seus próprios dados, tenants devem ser autenticados. Enquanto autenticação de usuário é, possivelmente, já presente na aplicação de destino, um mecanismo separado de autenticação de tenants específicos pode ser necessário, por duas razões: (1) geralmente

é muito mais fácil introduzir um mecanismo de autenticação adicional do que mudar um já existente, e (2) autenticação de tenants permite que um único usuário faça parte de mais do que uma organização lógica, o que estende a idéia de autenticação de usuários com "grupos".

2.4.2. Configuração

Em uma aplicação multi-tenancy a customização deve ser possível através de configuração. A fim de permitir que o usuário tenha uma experiência como se ele estivesse trabalhando em um ambiente dedicado, é necessário permitir pelo menos os seguintes tipos de configuração:

- **Estilo de Layout (Layout Style):** O componente de configuração de estilo de layout permite o uso de temas e estilos específicos;
- **Configuração Geral (General Configuration):** O componente de configuração geral permite a especificação de configurações específicas, como configurações de chave de criptografia e detalhes do perfil pessoal;
- **Entrada e saída de arquivo (File I/O):** O componente de configuração de I/O de arquivo permite a especificação de caminhos de arquivos, que podem ser usados para, por exemplo, geração de relatório; e
- **Fluxo de trabalho (Workflow):** O componente de configuração de fluxo de trabalho permite a configuração de fluxos específicos. Por exemplo, configuração de fluxos é necessária em uma aplicação de planejamento de recursos empresariais ERP, em que os passos para se realizar uma mesma tarefa pode variar significativamente entre diferentes companhias.

2.4.3. Banco de dados (Database)

Em uma aplicação multi-tenancy há uma grande exigência pelo isolamento dos dados. Pelo fato de os tenants usarem a mesma instância de um banco de dados é necessário ter certeza de que eles podem acessar somente seus próprios dados. Atualmente sistemas de gerenciamento de dados (Data Base Management Systems DBMS) de prateleira não são capazes de lidar com multi-tenancy de forma nativa, isso deve ser feito em uma camada entre a camada lógica de negócios e o pool de banco de dados da aplicação. As principais tarefas dessa camada são as seguintes:

- **Criação de novos tenants no banco de dados:** Se a aplicação armazena ou recupera dados que podem ser de tenants específicos, é tarefa da camada de banco de dados criar os registros do banco de dados correspondente quando um novo tenant se inscreveu para a aplicação;

- **Adaptação de consulta:** A fim de prover um isolamento de dados adequado, a camada de banco de dados deve ter certeza que consultas são ajustadas de forma que cada tenant possa acessar somente seus próprios registros; e

- **Balanceamento de carga:** Para melhorar o desempenho de uma aplicação multi-tenancy é necessário um balanceamento de carga eficiente para o pool de banco de dados. Note que qualquer acordo feito no SLA de um tenant e quaisquer restrições impostas pela legislação do país onde o tenant está localizado deve ser satisfeita. Por outro lado, nossa expectativa é a de que é possível criar algoritmos de balanceamento de carga mais eficientes usando as informações coletadas sobre as características de funcionamento dos tenants.

2.5. Implementando um Protótipo de Aplicação Multi-tenancy

2.5.1. Tecnologias

Após a escolha de uma arquitetura de referência para ser adotada por nossa aplicação, tem-se a necessidade de escolher as tecnologias que serão adotadas para a implementação. Durante a escolha dessas tecnologias avaliou-se a possibilidade do uso de JSF (Java Server Faces)[12], Struts 2[13], Spring Framework[14] e Grails[15]. Para a implementação do protótipo descrito nesse trabalho optou-se por utilizar o Grails Framework, pois apresenta um conjunto de recursos que podem dar produtividade para a equipe de desenvolvimento como geração de CRUD (Create, Read, Update e Delete); arquitetura baseada em plugins, que permite a criação e reuso de componentes; e total integração com frameworks e APIs Java. É possível desenvolver aplicações multi-tenancy em outras linguagens de programação como PHP, C#, Python, Ruby, etc. O que pode variar de uma linguagem para outra é o esforço necessário para desenvolver esse tipo de aplicação e algumas variações de performance.

Grails é um framework web open-source que utiliza a linguagem Groovy[16], e outros frameworks consagrados como Hibernate[17], Spring Framework e Sitemesh[18]. Uma descrição visual da arquitetura do Grails é apresentada na Figura 2.5. Grails foi projetado para desenvolver aplicações CRUD de forma simples e ágil, utilizando o modelo de "escrever código por convenção" introduzido pelo Ruby on Rails. O Grails propõe trazer a produtividade do Ruby on Rails para a plataforma Java, porém ele possui uma grande vantagem, já que é baseado na linguagem Groovy. Groovy (padronizado pela JSR-241) é uma linguagem dinâmica e ágil para a plataforma Java, que possui muitas características de linguagens de script como Ruby, Python e Smalltalk; e, além disso, aplicações Groovy podem utilizar classes Java facilmente.

[12] http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html
[13] http://struts.apache.org/2.3.1.2/index.html
[14] http://www.springsource.org/spring-framework
[15] http://grails.org
[16] http://groovy.codehaus.org
[17] http://hibernate.org
[18] http://sitemesh.org

Linguagens de script estão ganhando cada vez mais popularidade, devido a quantidade reduzida de código fonte necessário para implementar determinadas funcionalidades, se comparado com uma implementação em Java.

O protótipo multi-tenancy descrito nesse trabalho, em particular, se beneficiará da capacidade de definir métodos e propriedades em tempo de execução, disponibilizado pela linguagem Groovy. Essa característica da linguagem Groovy é chamada de metaprogramação. Em uma linguagem estática como Java, o acesso a uma propriedade ou invocação de um método é resolvido em tempo de compilação. Em comparação, Groovy não resolve o acessos à propriedades ou invocação de métodos até que a aplicação seja executada [50]. Uma aplicação que utiliza essa linguagem pode dinamicamente definir métodos ou propriedades em tempo de execução, isso vai de encontro à necessidade de customização das aplicações, dado que, com a utilização desse recurso, pode-se adicionar atributos e customizar comportamentos de um tenant específico futuramente.

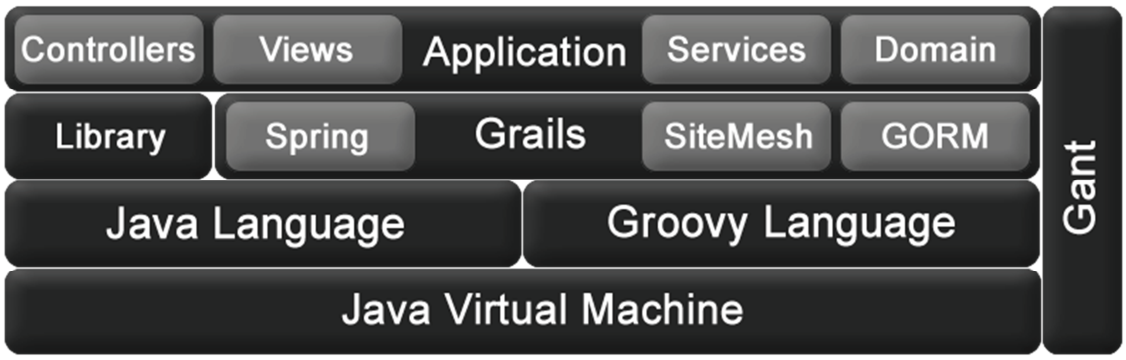


Figura 2.5. Arquitetura do Grails Framework (Fonte: [29])

Outro ponto que influenciou bastante na escolha de Grails foi sua arquitetura baseada em plugins. Grails não se propõe a ter todas as respostas e soluções para o desenvolvimento de aplicações web. Ao invés disso, ele provê uma arquitetura baseada em plugins e um repositório mantido pela comunidade de desenvolvedores onde é possível encontrar plugins que implementam as mais diversas funcionalidades como segurança, teste, busca, geração de relatórios, REST, web services, etc. Essa abordagem proporciona o reuso e permite que funcionalidades de difícil implementação possam ser adicionadas na aplicação de forma bastante simples.

Além das tecnologias relacionadas à programação, foi necessário escolher a tecnologia utilizada para armazenamento de dados. Como já existiam dados legados cadastrados em um banco de dados relacional, decidiu-se adotar o SGBD (Sistema de Gerenciamento de Banco de Dados) PostgreSQL. A escolha desse SGBD deu-se pelo fato do mesmo ser open-source, bastante consolidado no mercado e possuir integração com várias ferramentas de relatórios, o que poderia facilitar futuras extrações de dados para os clientes.

2.5.2. Prototipagem

Para exemplificar uma aplicação real que implementa o conceito de multi-tenancy será descrita a implementação de uma aplicação desenvolvida utilizando Grails Framework. A Arquitetura dessa aplicação é apresentada na Figura 2.6. Em nossa proposta os módulos associado à lógica de negócio da aplicação são implementados o mais independente possível dos requisitos multi-tenancy. Juntamente com o Framework Grails foram adotados alguns plugins existentes em seu repositório. A implementação do protótipo tem o objetivo de apresentar como esses conceitos podem ser aplicados na prática.

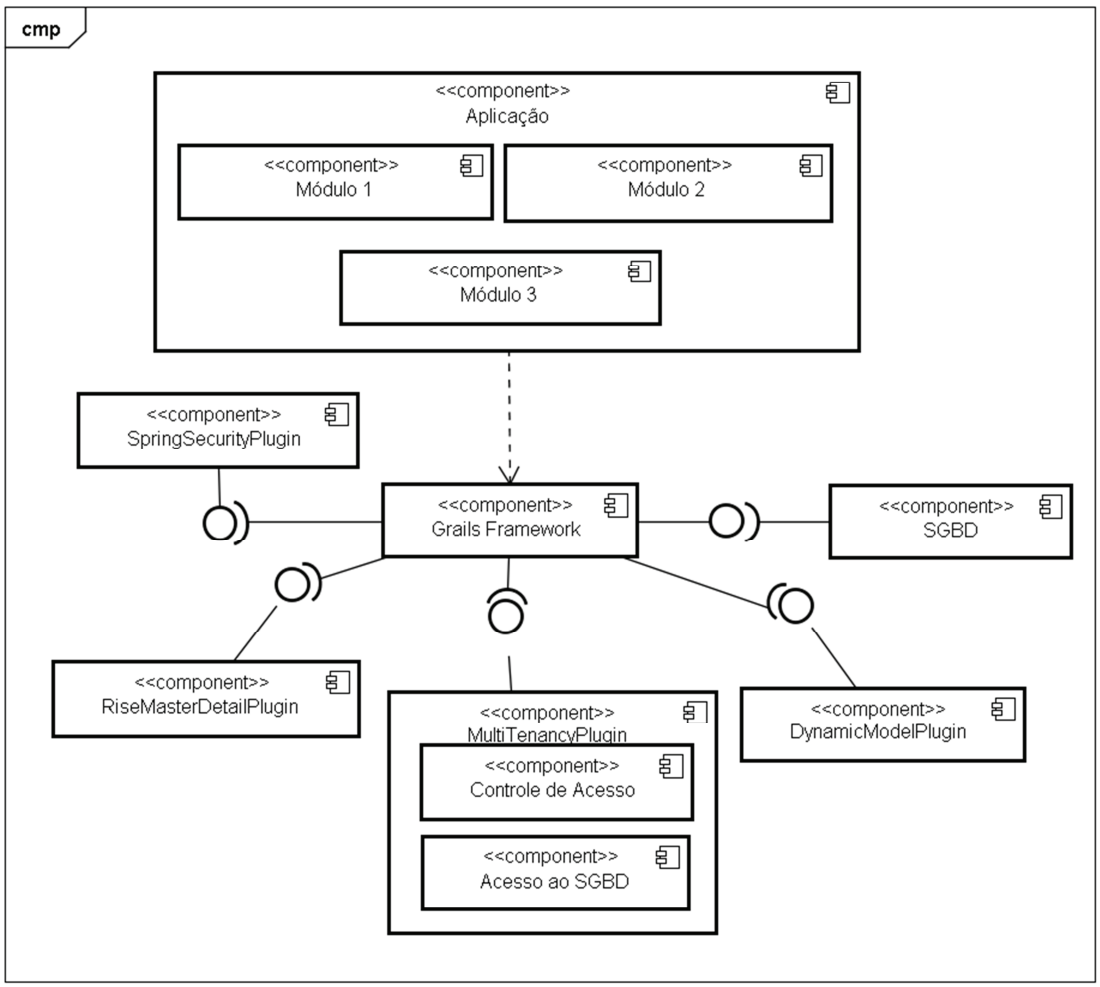


Figura 2.6. Arquitetura da Aplicação

Como mencionado anteriormente, existe uma grande necessidade de separar o código de lógica de negócio do código relacionado aos requisitos multi-tenancy. Com o objetivo de implementar os requisitos multi-tenancy de forma reutilizável, a solução foi implementar esses requisitos como um plugin do Grails, dessa forma aplicações futuras poderiam se beneficiar do código implementado. Uma vantagem do uso dessa abordagem é que dentro de um plugin grails é possível adicionar não só ter classes

implementadas em Groovy ou Java, mas também páginas da camada de visão e outros tipos de arquivo. Antes de implementar os requisitos multi-tenancy como plugin, foi realizado uma pesquisa no repositório de plugins do Grails para verificar se já existia algo semelhante ao que se pretendia implementar.

Durante a pesquisa foi encontrado o plugin *Multi-tenancy Core*[19] que implementa as funcionalidades de dois dos componentes mencionados em nossa arquitetura de referência[9]: o componente de autenticação e o componente de acesso ao banco de dados. Esse plugin implementa a funcionalidade de associar um usuário da aplicação a um tenant específico e além disso realiza de forma dinâmica o filtro dos dados durante uma consulta ao banco de dados. Dessa forma um usuário da aplicação tem acesso apenas aos dados vinculados ao seu tenant. Para utilizar esse plugin é necessário apenas instalá-lo na aplicação e adicionar uma anotação *@MultiTenant* nas classes de domínio da aplicação. Essa anotação indica que todos os objetos dessa classe deverão ser gerenciados pelo plugin e sempre que um registro for salvo no banco deverá ser associado ao tenant do usuário que estiver logado no momento. Um exemplo simplificado de uma classe Groovy com essa anotação é apresentada na Figura 2.7.

```
package br.com.rise.alexandria.fur

import grails.plugin.multitenant.core.groovy.compiler.MultiTenant;

/**
 * The Product entity.
 *
 * @author Josino Rodrigues
 */
@MultiTenant
class Product {

    String name
    String description

    static hasMany = [modules : Module]


    static constraints = {
        project(blank:false)
        name(size:1..100, blank:false, unique:['tenantId','project'])
        description()
    }

    //MÉTODOS ...

}
```

Figura 2.7. Exemplo de classe escrita em linguagem Groovy que utiliza o plugin Multi-Tenancy Core

A autenticação e controle de acesso pode ser realizado através da integração desse plugin com o plugin Spring Security[20], um framework de controle de acesso bastante utilizado em aplicação Java. A Figura 2.8 apresenta a tela de autenticação gerada pelo próprio plugin do Spring Security. De forma integrada com o plugin Multi-tenancy Core, ele associa cada usuário logado ao tenant ao qual ele pertence.

[19] http://multi-tenancy.github.com/grails-multi-tenancy-core
[20] http://static.springsource.org/spring-security/site/index.html

O terceiro componente mencionado em nossa arquitetura de referência é o componente de configuração. O objetivo desse componente é gerenciar as configurações relacionadas a cada tenant e prover funcionalidades de customização da aplicação para estes.

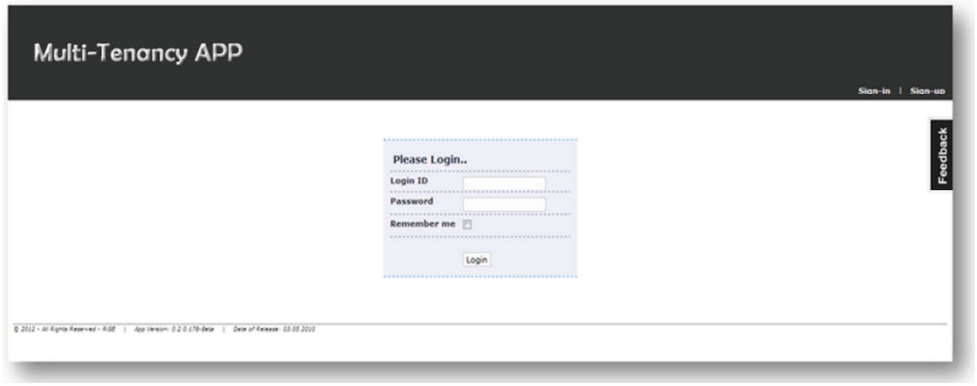


Figura 2.8. Tela de autenticação gerada pelo Spring Security Plugin

Essa customização pode ir desde alterações em interface com o usuário até alteração nas classes de negócio, tudo isso realizado de forma dinâmica. Durante a pesquisa foi encontrado o plugin *Dynamic Domain Class*[21] que proporciona a criação de classes de domínio de forma dinâmica, esse plugin ajudou a validar a idéia de usar Groovy e Grails para customizar as classes de domínio em tempo de execução. A tela apresentada na Figura 2.9 apresenta o formulário utilizado na aplicação para criação de classes dinâmicas. Para cada classe dinâmica criada o Grails Framework cria automaticamente o CRUD para essa entidade.

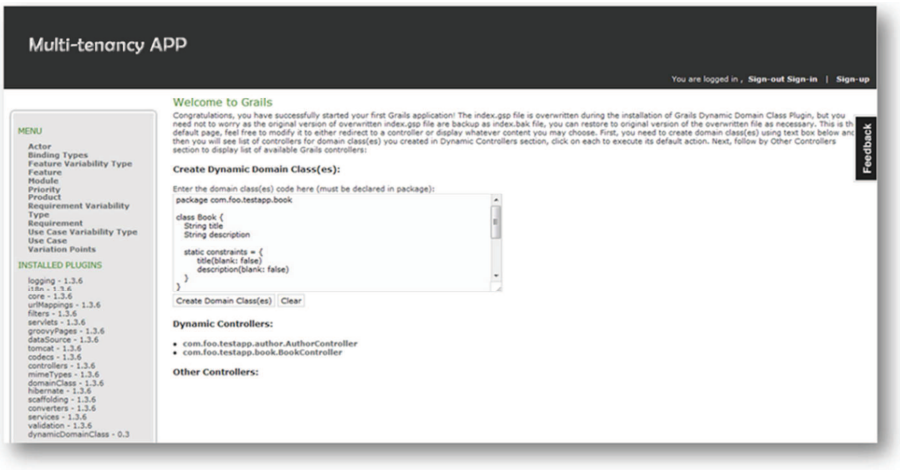


Figura 2.9. Tela com Formulário para criação de classes dinâmicas

Durante a implementação de um projeto real, detectamos que o padrão de geração de telas do Grails não satisfazia aos anseios dos usuários quanto à forma de interação com o aplicativo. Apartir dessa necessidade foi desenvolvido um plugin para implementar o

[21] http://code.google.com/p/grails-dynamic-domain-class-plugin/

padrão de tela Mestre/Detalhe [45], não implementado nativamente pelo gerador de telas padrão do Grails. Durante a implementação desse plugin identificou-se que poderia ser viável implementar módulos completos da aplicação como plugins Grails, desde classes de negócio à camada de visão.

Após a implementação do protótipo foi realizado a avaliação do protótipo para verificar alguns atributos de qualidade atendidos pela proposta. A Tabela 2.1 apresenta para cada atributo de qualidade uma breve descrição de como ele foi atendido.

Tabela 2.1. Resultado da aplicação dos critérios de exclusão (Fonte: Elaboração própria)

ATRIBUTO	AVALIAÇÃO
Disponibilidade	Como foi adotado um SGBD *PostgreSQL*, é possivel agendar tarefas automáticas no servidor para executar *backups* periódicos.
Integridade Conceitual	Tanto as entidades de negócio quanto as classes utilitárias da aplicação foram modularizadas em *plugins* de forma que os interesses da aplicação foram bem definidos e separados.
Flexibilidade	A flexibilidade pode ser alcançada através do uso das característica de linguagem dinâmica existente na linguagem *Groovy*. Dessa forma é possivel criar Entidade de domínio e alterar entidades já existentes em tempo de execução, caso seja necessário.
Interoperabilidade	O uso de *plugins Grails* permite que os dados da aplicação possam ser expostos através de REST ou *Web Services* para os usuários.
Manutenibilidade	A arquitetura de *plugins* do *Grails* facilita a manutenção dos código já que cada plugin pode ser mantido separadamente, reduzindo as dependências entre os componentes da aplicação.
Reusabilidade	As partes da aplicação implementadas como *plugins* podem ser reutilizadas em aplicações futuras facilmente, sem a necessidade de qualquer adaptação no *plugin*.
Segurança	A autenticação e controle de acesso puderam ser garantidos pelo uso do *Framework Spring Security*, que pôde ser facilmente integrado com o *plugin* de *multi-tenancy* no atendimento do requisito de isolamento de acesso entre dados dos *tenants*.
Testabilidade	O *Grails Framework* provê uma infraestrutura para auxiliar na implementação de testes unitários que utiliza *JUnit*, um *framework* de teste para aplicações *Java*. Durante a criação de uma classe o *Grails Framework* já cria uma classe de testes correspondente para otimizar o tempo de desenvolvimento.
Usabilidade	*Grails* trabalha com templates para geração de telas. Embora já possua um template padrão, o desenvolvedor pode implementar um template próprio que atenda à necessidades específicas de interação com o usuário. O template padrão do *Grails Framework* já implementa alguns padrões de usabilidade bem estabelecidos no mercado.

Esse protótipo foi baseado em uma experiência adquirida durante o desenvolvimento de uma aplicação real. A aplicação desenvolvida entrou em produção e atendeu satisfatoriamente aos requisitos do cliente. Durante o desenvolvimento do software em questão foi utilizado Grails porque era, na época, a opção open source mais madura no tocante a multi-tenancy. Atualmente já existem outros frameworks que

dão suporte ao desenvolvimento de aplicações multi-tenancy como Atena Framework[22], Hibernate 4[23], Rails[24], etc.

2.6. Vantagens e desvantagens

Durante uma vasta pesquisa bibliográfica foram identificados 12 trabalhos que citam vantagens ou desvantagens da adoção de multi-tenancy. Apartir da leitura dos mesmos identificamos as seguintes vantagens:

- O provedor de serviço pode usar a mesma versão da aplicação (com o único código base) para prover serviços para várias organizações [55];
- Consumidores obtêm acesso à capacidade de processamento elástico que pode ser aumentada ou diminuída de acordo com a demanda sem a necessidade de realizar manutenção em servidores [55];
- Dois benefícios de uma abordagem baseada em uma plataforma multi-tenancy são colaboração e integração. Pelo fato de todas as aplicações executarem em um único espaço, torna-se mais fácil permitir a um usuário de qualquer aplicação acesso à uma coleção de dados de outra aplicação semelhante. Essa capacidade simplifica o esforço necessário para integrar aplicações relacionadas e os dados que elas gerenciam [60];
- Atualização do software de uma só vez para todos os tenants [44];
- Economia nos custos com infra-estrutura de hardware [53, 33, 4];
- Economia nos custos de gerenciamento de infra-estrutura [53, 33, 4];
- Aumento da margem de lucro para o provedor de serviço através da redução dos custos de entrega e diminuição dos custos de assinatura do serviço para os clientes [36];
- Possibilidade de reusar regras de negócio com o mínimo de adaptação [9];
- Organização dos usuários em vários níveis de acordo com suas necessidades e melhor gerenciamento de recursos [27];
- O usuário pode customizar sob-demanda os serviços providos pelo fornecedor do software [67]; e
- Redução dos custos de venda e manutenção do software por porte do provedor do software [67].

As desvantagens da adoção de multi-tenancy foram pouco mencionadas nos trabalhos encontrados. Em geral os trabalhos mencionavam mais vantagens do que desvantagens. A seguir são listados alguns pontos considerados desvantagens por alguns autores:

- É difícil calcular os recursos requeridos por cada novo tenant e ao mesmo tempo garantir que as restrições de todos os outros tenants da mesma instância sejam atendidas[33];

[22] http://athenasource.org
[23] http://docs.jboss.org/hibernate/orm/4.1/devguide/en-US/html/ch16.html
[24] http://rubyonrails.org

- Fatores limitantes e gargalos nos recursos computacionais exigidos pelas várias instâncias devem ser identificados [33], e isso não é trivial nesse tipo de ambiente;
- Dificuldade de comparar e otimizar a redução de custos das diferentes formas de distribuição dos tenants entre os servidores, pelo fato de existirem várias variáveis envolvidas [33];
- Preocupação das empresas com o custo inicial de reestruturas suas aplicações legadas para multi-tenancy, [8]; e
- Preocupação dos mantenedores de software com a possibilidade de multi-tenancy introduzir problemas adicionais de manutenção decorrentes do fato desses novos sistemas serem altamente customizáveis [8].

2.7. Desafios da área

2.7.1. Alocação de Recursos

Para que se possa desfrutar dos benefícios que uma aplicação multi-tenancy traz, um conjunto de desafios devem ser solucionados [33]:

1. Calcular os recursos computacionais necessários para o funcionamento de cada novo tenant, e ao mesmo tempo atender às restrições de todos os tenants em instância da aplicação compartilhada;
2. Identificar fatores limitantes ou gargalos nos recursos computacionais requeridos para as várias instâncias, cada uma com vários tenants contendo diferentes restrições que devem ser atendidas;
3. Durante a distribuição dos tenants é necessário indicar a melhor localização para que nenhuma restrição de SLA seja violada;
4. A distribuição dos tenants e instâncias em um ambiente de computação distribuído deve ser automatizada; e
5. A economia alcançada entre as diferentes formas de distribuições de tenants e instâncias deve ser comparada e otimizada, de forma que seja encontrada a melhor distribuição possível.

O cálculo do número máximo de usuários e tenants em uma instância compartilhada em um servidor, sem que haja violação de restrições definidas no contrato de SLA de cada tenant é um desafio não trivial. Kwok e Mohindra [33] propõem uma abordagem para o cálculo de recursos requeridos para o bom funcionamento de vários tenants em uma instância de aplicação compartilhada. Nesse trabalho também é descrito um método para otimizar a distribuição de tenants e instâncias de uma aplicação em um conjunto de servidores sem violar qualquer requisito de SLA dos tenants. Por fim os autores apresentam uma ferramenta que tem o objetivo de auxiliar o deployment de aplicações multi-tenancy usando o número mínimo de servidores, fornecendo portanto o máximo de economia em um ambiente de computação distribuído.

Fehling et al. [18] também realizam um estudo sobre as oportunidades de otimização do uso de recursos, mas nesse caso o cenário considerado é um ambiente de computação heterogêneo onde os recursos utilizados são oriundos de clouds públicas e privadas. Nesse trabalhos os autores utilizam um algoritmo Smarter Simulated Annealing para auxiliar na busca de uma distribuição otimizada dos recursos computacionais.

Outra questão associada à alocação de recursos é a priorização das requisições recebidas por uma instância de uma aplicação multi-tenancy. Em um cenário multi-tenancy é comum que cada tenant necessite priorizar as requisições de seus consumidores de tal maneira que um consumidor com prioridade alta poderá ser atendido mais rapidamente que outro, e que a instância da aplicação também priorize os tenants, de forma que as requisições de um tenant tenham maior prioridade de atendimento que as de outro. Diante desse cenário, Tsai et al.e [55] propõem um modelo para priorizar requisições de vários tenants enquanto preserva as prioridades locais das requisições de um tenant específico. Os autores propõem um algoritmo chamado Crystalline Mapping que mapeia prioridades internas de um tenant específico para prioridades globais.

2.7.2. Banco de Dados

Uma das preocupações quando pretende-se implementar uma aplicação multi-tenancy é planejar como os dados da aplicação serão armazenados. Atualmente, boa parte das aplicações existentes no mercado utilizam bancos de dados relacionais. Existem várias estratégias para implementar um esquema de banco de dados relacional de forma que ele permita o armazenamento de dados de vários tenants. Aulbach et al. [3] apresenta várias dessas técnicas, que são mostradas na Figura 2.10 e brevemente descritas a seguir:

- ***Basic Layout -*** a técnica mais básica para implementar multi-tenancy é adicionar uma coluna TENANTID em cada tabela para armazenar um valor que identifica a qual tenant um registro pertence. Essa abordagem provê uma boa consolidação mas não provê uma boa extensibilidade, já que não possui nenhum mecanismo para customização de armazenamento de dados para um tenant específico;
- ***Private Table -*** é a forma mais básica para suportar extensibilidade, dado que cada tenant possui sua próprias tabelas privadas. Nessa abordagem, é necessário uma camada de transformação de queries para ajustar o nome das tabelas nas queries e substituir pelo nome da tabela privada;
- ***Extension Table -*** essa técnica é a combinação das duas técnicas anteriores, as tabelas de extensão bem como as tabelas base devem possuir uma coluna para armazenar os dados de identificação do tenant. Outra coluna também precisa ser adicionada para armazenar a tabela lógica para que os dados possam ser reconstruídos. Essa abordagem é utilizada para mapear esquemas orientados a objeto com herança nos modelos relacionais atualmente;

- ***Universal Table -*** estruturas genéricas permitem a criação de um número arbitrário de tabelas com formas arbitrárias. Universal Table é uma estrutura genérica com uma coluna Tenant, uma coluna Table e um grande número de colunas de dados genéricas. A coluna de dados tem um tipo flexível, como VARCHAR, no qual outro tipo pode ser convertido;
- ***Pivot Table -*** nessa técnica as entidades de domínio são representadas por tabelas lógicas, cujas informações são montadas dinâmicamente. Uma Pivot Table possui as seguintes colunas: Tenant (identifica o tenant), Table (identifica a tabela à qual o dado está associado), Row (identificar a linha à qual o dado está associado) Col (identifica o campo que a linha representa) e uma coluna para armazenar o dado em si, que normalmente é de um tipo flexível como VARCHAR. Essa abordagem proporciona uma alta flexibilidade mas possui muitas colunas de metadados que podem impactar negativamente na performance da aplicação, durante a manipulação dos dados;
- ***Chunk Table -*** essa técnica é uma extensão da técnica anterior e trabalha com o conceito de Chunk Table. Uma Chunk Table é semelhante a uma Pivot Table exceto pelo fato de possuir um conjunto de colunas de dados de vários tipos, e a coluna Col é substituida pela coluna Chunk. Em comparação com a técnica Pivot Table, essa técnica armazena uma quantidade menor de metadados, o que diminui o tempo de reconstrução da tabela lógica; e
- ***Chunk Folding -*** as tabelas lógicas são verticalmente particionadas em chunks, os quais permitem junção quando necessário.

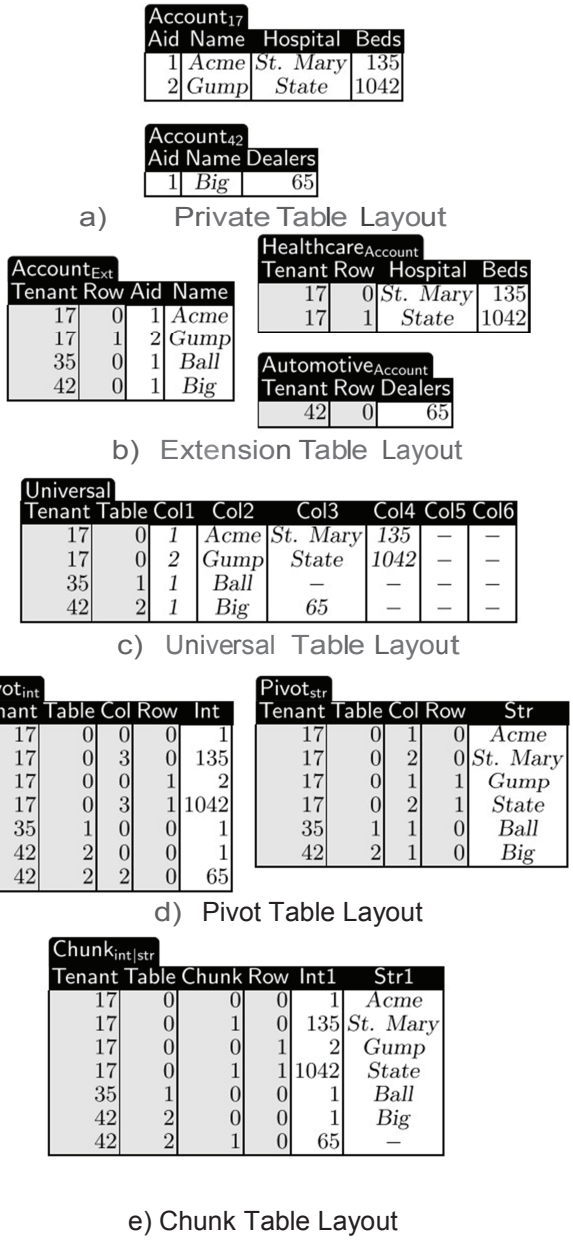
Essas são as técnicas básicas para a implementação de modelos de dados para aplicações multi-tenancy, outras abordagens foram criadas a partir delas [19, 61, 59]. Aulbach et al. [4] realizam um estudo experimental para comparar cinco técnicas de implementação de aplicações multi-tenancy a nível de banco de dados. Os autores concluíram que ainda não existe um sistema de gerenciamento de banco de dados (SGBD) ideal para esse tipo de aplicação e que um SGBD para SaaS deveria ser baseado na técnica Private Table.

Schiller et al. [52] apresentam em seu trabalho as primeiras funcionalidades para que um SGBD relacional possa suportar múltiplos tenants nativamente. Em sua proposta tenants são introduzidos como objetos de primeira classe e é proposto o conceito de "contexto" para isolar um tenant de outro. Além disso, o conceito de herança permite compartilhar o esquema da aplicação entre os tenants, ao mesmo tempo em que permite que o esquema seja extendido com estruturas de dados adicionais. Ao final do trabalho o autor realiza uma avaliação da implementação de sua proposta através de experimentos.

2.7.3. Customização

Um ponto importante em uma aplicação multi-tenancy é a customização. Em aplicações web customizáveis o código torna-se mais complexo, problemas de performance impactam todos os tenants da aplicação e planejar segurança e robustez tornam-se pontos importantes. Segundo Jansen et al. [26], existem três áreas de pesquisa que são

diretamente relacionadas a Técnicas de Realização de Customização (Customization Realization Techniques CRT) em aplicações multi-tenancy: variabilidade em linhas de produtos de software, personalização do usuário final em aplicações web e arquiteturas multi-tenancy. Pesquisadores que pretendem atacar esse problema devem direcionar seus estudos nessas três áreas.



$Account_{17}$

Aid	Name	Hospital	Beds
1	*Acme*	*St. Mary*	135
2	*Gump*	*State*	1042

$Account_{42}$

Aid	Name	Dealers
1	*Big*	65

a) Private Table Layout

$Account_{Ext}$

Tenant	Row	Aid	Name
17	0	1	*Acme*
17	1	2	*Gump*
35	0	1	*Ball*
42	0	1	*Big*

$Healthcare_{Account}$

Tenant	Row	Hospital	Beds
17	0	*St. Mary*	135
17	1	*State*	1042

$Automotive_{Account}$

Tenant	Row	Dealers
42	0	65

b) Extension Table Layout

Universal

Tenant	Table	Col1	Col2	Col3	Col4	Col5	Col6
17	0	*1*	*Acme*	*St. Mary*	*135*	–	–
17	0	*2*	*Gump*	*State*	*1042*	–	–
35	1	*1*	*Ball*	–	–	–	–
42	2	*1*	*Big*	*65*	–	–	–

c) Universal Table Layout

$Pivot_{int}$

Tenant	Table	Col	Row	Int
17	0	0	0	1
17	0	3	0	135
17	0	0	1	2
17	0	3	1	1042
35	1	0	0	1
42	2	0	0	1
42	2	2	0	65

$Pivot_{str}$

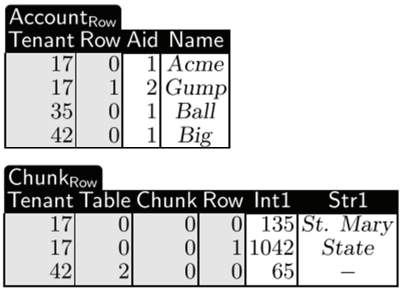
Tenant	Table	Col	Row	Str
17	0	1	0	*Acme*
17	0	2	0	*St. Mary*
17	0	1	1	*Gump*
17	0	2	1	*State*
35	1	1	0	*Ball*
42	2	1	0	*Big*

d) Pivot Table Layout

$Chunk_{int|str}$

Tenant	Table	Chunk	Row	Int1	Str1
17	0	0	0	1	*Acme*
17	0	1	0	135	*St. Mary*
17	0	0	1	2	*Gump*
17	0	1	1	1042	*State*
35	1	0	0	1	*Ball*
42	2	0	0	1	*Big*
42	2	1	0	65	–

e) Chunk Table Layout



$Account_{Row}$

Tenant	Row	Aid	Name
17	0	1	*Acme*
17	1	2	*Gump*
35	0	1	*Ball*
42	0	1	*Big*

$Chunk_{Row}$

Tenant	Table	Chunk	Row	Int1	Str1
17	0	0	0	135	*St. Mary*
17	0	0	1	1042	*State*
42	2	0	0	65	–

f) Chunk Folding

Figura 2.10. Tecnicas de mapeamento de esquema (Fonte: [3])

Outro ponto importante é que os tenants de uma aplicação multi-tenancy não somente têm diferentes requisitos com respeito a propriedades funcionais, mas também podem exigir diferentes propriedades de qualidade de serviço como privacidade e performance. Alguns tenants exigem que a aplicação possua alta disponibilidade e estão dispostos a pagar um alto preço pelo uso desse serviço. Outros tenants não estão interessados em alta disponibilidade mas preocupam-se mais com um baixo preço. Em casos como esse é necessário que exista na aplicação algum mecanismo para ajustar a aplicação às reais necessidades do usuário, no tocante aos casos citados anteriormente.

2.7.4. Escalabilidade

Esse é um dos problemas mais críticos para serem resolvidos em um cenário real. Dado um número fixo de servidores, é necessário otimizar a distribuição dos tenants de forma a maximizar o número total de tenants possíveis sem violar seus requisitos de SLA e ainda estar preparado para o crescimento do volume de dados e de requisições.

Yuanyuan et al. [63] apresentam uma arquitetura focada na escalabilidade de dados. Eles propõem uma entidade com base em grupos de particionamento horizontal, através da análise das relações entre as entidades de uma aplicação multi-tenancy. Nessa abordagem, entidades de negócio altamente coesas formam um grupo de entidades e o particionamento ocorre com base nesse grupo. Dessa forma cada operação acessa um único grupo de instâncias, podendo obter os dados necessários através do acesso a um único banco de dados e evitando a necessidade de transações distribuídas.

Koziolek [31, 32] apresenta um estilo arquitetural focado em escalabilidade de dados e ilustra como os conceitos apresentados nesses trabalhos podem ajudar a fazer as Plataformas como Serviço (PaaS) atuais, como Force.com, Windows Azure, e Google App Engine, escaláveis e customizáveis. Já Zhang et al. [66] focam no problema de localização de tenants, propõem um algoritmo heurístico chamado Tenant Dispatch Heuristic (TDH) e realizam um conjunto de simulações e comparações onde é avaliado como o uso desse algoritmo impacta na escalabilidade e economia de recursos.

2.7.5. Migração

Migrar aplicações web legadas que trabalham com um único tenant (Isolated Tenancy Hosted Applications ITHA) para multi-tenancy (multi-tenancy Enabled Application MTEA) não é uma tarefa trivial, devido a quantidade de ferramentas e técnicas de

migração apropriadas. De acordo com Zhang et al. [65] os métodos existentes para migração são muito abstratos ou muito específicos quando tentamos aplicá-los como guias para migrar uma aplicação que trabalha com um tenant isolado para uma aplicação multi-tenancy. Nesse mesmo trabalho os autores propõem um método sistemático que provê diretrizes para migrar aplicações ITHA para MTEA, levando em conta custo, risco e efetividade do método de migração.

Dado a característica de demanda sazonal existente em vários tipos de aplicação, uma funcionalidade essencial para um ambiente multi-tenancy é a funcionalidade que permite a migração de tenants entre hosts, uma técnica chamada live migration [13, 40] é apresentada por alguns autores como uma possível solução.

Elmore et al. [17] apresentam Zephyr, uma técnica para live migration que tem o objetivo de minimizar a interrupção de serviço do tenant migrado. Através da introdução de uma sincronização dupla que permite a ambos os nós, o nó origem dos dados e o nó destino, executarem simultaneamente transações para o tenant. A migração inicia-se com a transferência dos metadados do tenant para o nó destino, que pode realizar novas transações enquanto o nó origem completa as transações que foram iniciadas quando a migração se iniciou. De acordo com os autores, live migration é uma importante característica para habilitar elasticidade em bancos de dados multi-tenancy para plataformas de cloud.

2.7.6. Monitoramento

Para obter uma visão geral do funcionamento de seus serviços, provedores precisam de informações sobre todas as camada de seu sistema, incluindo a aplicação, máquinas virtuais e uso de rede. Por razões de simplicidade, todas essas informações devem idealmente ser entregues através de uma única interface de monitoramento. Hasselmeyer e D'Heureuse [22] apresentam alguns requisitos básicos de uma infraestrutura de monitoramento para ambientes multi-tenancy:

- ***Multi-tenancy:*** a infraestrutura de monitoramento precisa naturalmente ser capaz de lidar com informações de monitoramento provinientes de diferentes clientes (tenants);
- ***Escalabilidade:*** uma solução de monitoramento precisa ser escalável em vários aspectos. Ela precisa escalar para um grande número de agentes de monitoramento, de notificação de eventos, de tenants, de recursos (virtuais e físicos, servidores, elementos de rede e aplicações), e de tipos de informação de monitoramento;
- ***Dinamismo:*** sistemas de monitoramento multi-tenancy precisam suportar o dinamismo que é inerente a um ambiente multi-tenancy. Esse dinamismo decorre da rápida e frequente adição e remoção de tenants no datacenter. Além disso, a alocação de recursos para os tenants também está em constante mudança e o conjunto de recursos que está sendo monitorado também pode mudar;

- ***Simplicidade:*** o sistema de monitoramento deveria ser simples em dois aspectos: primeiro, a interface para monitoramento do sistema precisa ser fácil de entender e usar. Segundo, o sistema precisar ser fácil de instalar e manter, tanto para quem gerencia o datacenter quanto para os consumidores. Uma API simples é desejável, uma vez que facilita o trabalho de desenvolvedores que criam soluções de controle e monitoramento; e
- ***Abrangência:*** esse requisito aborda a necessidade de uma infraestrutura única de monitoramento que deve ser utilizável para todos os tipos de informação de monitoramento, não importa o que está relacionado ao recurso ou qual o significado que ele transmite. Essa abrangência aplica-se a múltiplos aspectos: tipos de dados, fontes de notificação e tenants.

Já Cheng et al. [11] propõem um mecanismo de controle que monitora a qualidade de serviço por tenant, detecta situações anormais e dinamicamente ajusta o uso de recursos baseado nos parâmetros de SLA definidos para o tenant. Nesse trabalho os autores apresentam sua proposta e realizam um estudo experimental para avaliar os resultados.

2.7.7. Performance

Esse assunto está diretamente ligado à monitoramento de aplicações multi-tenancy, isso porque aplicações são monitoradas para que se possa verificar se a mesma está executando em um nível de performance compatível com a SLA definida para o cliente. Desde 2008 temos um bom número de trabalhos que estudam esse problema, dentre eles estão propostas de arquitetura, frameworks, métodos, técnicas, ferramentas e experimentos, que utilizam as mais variadas abordagens. O primeiro trabalho encontrado relacionado à performance foi o de Wang et al. [58], onde os autores apresentam os principais padrões de implementação de aplicações multi-tenancy que tratam dos aspectos de isolamento e segurança, e avaliam a performance desse padrões através de uma série de experimentos e simulações.

Para permitir que um provedor de serviço ofereça diferentes níveis de performance baseado no nível de SLA definido para um tenant específico, Lin et al.[38] propõe uma solução que utiliza um regulador de performance baseado no controle de feedback. O regulador possui uma estrutura hierárquica, na qual um controlador de alto nível gerencia a taxa de admissão de requisições para prevenir sobrecarga e um controlador de baixo nível que gerencia os recursos alocados pelas requisições admitidas para alcançar um nível específico de diferenciação de serviço entre os tenants que compartilham os mesmos recursos. Um protótipo dessa abordagem é implementado utilizando Tomcat e MySQL, e ao final são realizados alguns experimentos para validar a proposta.

Outro ponto importante é o isolamento de performance para prevenir potenciais anomalias de comportamento de um tenant, que possam afetar a performance de outros tenants que compartilham os mesmos recursos. Li et al. [36] propõem o SPIN (Service Performance Isolation Infrastructure) que visa permitir um abrangente

compartilhamento de recursos em ambientes multi-tenancy. Uma vez que alguns tenants agressivos podem interferir na performance de outros, SPIN fornece um relatório de anomalias, identifica os tenants agressivos, e fornece um mecanismo de moderação para eliminar o impacto negativo em outros tenants. Durante sua pesquisa os autores implementaram um protótipo do SPIN e realizaram alguns experimentos para demonstrar sua eficiência.

Um desafio interessante na área de gerenciamento de performance é entender e predizer performance de sistemas. Durante a realização desse mapeamento foram encontradas duas abordagens relacionadas a esse assunto. Schaffner et al. [51] apresentam um estudo sobre a automação de tarefas operacionais em clusters multi-tenancy de bancos de dados em memória orientados a coluna. Foi desenvolvido um modelo para predizer se a alocação de um tenant em um servidor no cluster, levaria a uma violação no tempo de resposta esperado. Já Ahmad e Bowman [1] realizaram um estudo experimental para entender e predizer a performance de sistemas, nesse trabalho os autores sugerem uma abordagem de máquina de aprendizado que usa dados monitorados para entender a performance do sistema.

Guo et al. [20] apresentam um framework que possui um componente específico para isolamento de performance. Segundo os autores os objetivos principais do isolamento de performance incluem dois aspectos. Primeiro, evitar que o comportamento (potencialmente ruim) de um tenant afete o comportamento de outro, de maneira imprevista. Segundo, evitar a injustiça entre tenants em termos de performance de uso. Para alcançar esse objetivo os autores sugerem um padrão de isolamento de performance híbrido, baseado em várias técnicas apresentadas em seu trabalho.

2.7.8. Segurança

Confiança é um dos maiores desafios que influenciam a ampla aceitação de SaaS. Na ausência de confiança em SaaS, segurança e privacidade de dados figuram entre os principais e mais importantes assuntos relacionados a arquitetura multi-tenancy. Foram encontrados durante esta pesquisa alguns trabalhos publicados nos últimos 2 anos que estão relacionados a esse assunto.

Um assunto bastante relevante é a avaliação de credibilidade de tenants, que indica quais tenants têm um comportamento que possa prejudicar o funcionamento dos demais. Zhiqiang [46] apresenta um algoritmo de avaliação de credibilidade de tenants baseado na experiência. Essa abordagem visa realizar a detecção e gerenciamento de tenants maliciosos a um baixo custo. De acordo com o histórico de acesso dos tenants, ele pode calcular a credibilidade de tenants, atribuir privilégios de acesso e determinar estratégias de detecção e monitoramento.

De acordo com Li et al. [37], separação de responsabilidade de segurança entre provedores SaaS e consumidores precisa ser suportada em um ambiente de cloud. Arquitetura multi-tenancy baseada em modelo de controle de acesso (MTACM multi-

tenancy based access control model) foi projetada para inserir o princípio de separação de responsabilidade de segurança em cloud; essa arquitetura define dois níveis de granularidade no mecanismo de controle de acesso. O primeiro nível está relacionado ao provedor de serviço, que compartilha sua infraestrutura entre vários clientes. O segundo nível é o nível da aplicação onde uma mesma aplicação hospeda informações de vários tenants.

Zhang et al. [64], Calero et al. [10] e Tsai et al. [54] apresentam três abordagens diferentes para implementação de mecanismos de segurança e controle de acesso para aplicações multi-tenancy. Zhang et al. [64] apresentam uma abordagem de controle de acesso baseado em restrições de privacidade customizáveis. Essa abordagem combina criptografia de dados e separação de informação e define três tipos de restrições de privacidade baseado na funcionalidade de customização em aplicações SaaS. Calero et al. [10] descrevem um sistema de autorização para um serviço de middleware em uma PaaS, que suporta controle de acesso baseado em hierarquia de funções, hierarquia de objetos baseada em caminho e federações. Os autores apresentam também uma arquitetura de sistema de autorização para implementação do modelo.

De acordo com Tsai et al. [54] as abordagens atuais para controle de acesso em cloud não escalam bem para requisitos multi-tenancy porque eles são baseados principalmente em identificadores (IDs) de usuário individual em diferentes níveis de granularidade. No entanto, o número de usuários pode ser enorme e causar um overread significativo no gerenciamento de segurança. RBAC (Role-Based Access Control) tornase atrativo nesse caso pelo fato do número de papéis de usuário em uma aplicação ser significativamento menor, e usuários podem ser classificados de acordo com seus papéis. Os autores propõem um RBAC usando uma ontologia de papéis para arquitetura multitenancy em clouds.

2.7.9. Integração com outros sistemas

De acordo com o cenário apresentado até aqui, é possível perceber que seria muito difícil e caro criar uma aplicação multi-tenancy que fosse tão configurável a ponto de permitir que todos os requisitos do usuário pudessem ser atendidos através de configurações. Mesmo nesse cenário é possível permitir que os próprios usuários implementem parte de suas soluções e as integrem à plataforma multi-tenancy utilizada. Para solucionar esse problema, alguns autores propuseram abordagens para implementar arquitetura multi-tenancy com SOA.

Azeez et al. [5] apresentam uma arquitetura para implementar multi-tenancy no nível de SOA, que habilita usuários a executar seus serviços e outros artefatos SOA em um framework multi-tenancy SOA. Em seu trabalho os autores discutem arquitetura, decisões de projeto, e problemas encontrados, juntamente com potenciais soluções. Diferentes aspectos de arquitetura de sistemas no que se refere a multi-tenancy para SOA também são considerados como: deployment de serviços, envio de mensagens, segurança, execução de serviços e finalmente acesso a dados.

2.8. Considerações Finais

Grandes empresas de TI (Tecnologia da Informação), como HP e IBM têm investido bastante nessa área, isso é comprovado com o fato de vários artigos encontrados durante esse estudo serem escritos por membros dessas empresas. Além disso, algumas empresas também vendem aplicativos que podem ser configurados para executar como SaaS em nuvens privadas; alguns sistemas da SAP por exemplo, podem ser usados como um SaaS oferecido dentro das empresas.

A primeira dificuldade encontrada durante as pesquisas foi identificar quais os requisitos de uma aplicação multi-tenancy que são essenciais para sua implementação. Durante a tentativa de identificá-los, foi possível perceber que duas características chave de multi-tenancy são o auto grau de automação nas atividades de manutenção da aplicação e uma interface amigável para que o usuário possa realizar suas customizações, reduzindo ao máximo a necessidade de intervenções por parte de desenvolvedores e administradores de sistema.

Durante a elaboração desse trabalho foi possível notar também a existência de muitos trabalhos associados à multi-tenancy e armazenamento de dados. Propostas de métodos, técnicas e até uma proposta de SGBD multi-tenancy foi encontrada. Embora esse seja o campo de multi-tenancy onde mais se encontrou publicações, ainda é um campo onde existem muitos pontos de melhoria, principalmente pelo fato do armazenamento de dados influenciar diretamente no tempo de resposta de todas as operações que manipulam dados neste tipo de ambiente.

Embora vários experimentos já tenham sido realizados com as abordagens existentes na literatura para armazenamento de dados em aplicações multi-tenancy, não foi encontrado um consenso sobre qual abordagem é a melhor. Migração de dados, modelagem dos dados, tempo de resposta, armazenamento distribuído, integração com ferramentas de BI (Business Inteligence), todos esses fatores podem influenciar na escolha de uma abordagem para armazenamento de dados.

Atualmente, novas formas de armazenamento de dados estão surgindo como: HBase[25], Cassandra[26], Hipertable[27], MongoDB[28], CouchDB[29], etc. Esses sistemas armazenam dados de uma forma diferente do usado nos modelos relacionais e compõem um movimento chamado NoSQL. Uma lacuna pouco explorada foi a utilização dessas novas tecnologias no desenvolvimento de aplicações multi-tenancy, dado que bancos de dados NoSQL surgiram como uma solução para a questão da escalabilidade no armazenamento e processamento de grandes volumes de dados[14].

[25] http://hbase.apache.org
[26] http://cassandra.apache.org
[27] http://hypertable.com
[28] http://www.mongodb.org
[29] http://couchdb.apache.org

A necessidade de uma nova tecnologia de banco de dados surgiu como consequência da ineficiência dos bancos de dados relacionais em lidar com a tarefa de manipulação de grandes volumes de dados. Vale ressaltar que o modelo relacional foi proposto na década de 70, quando as aplicações de banco de dados caracterizavam-se por lidar com dados estruturados, ou seja, dados que possuem uma estrutura fixa e bem definida, e esse não é o caso de aplicações multi-tenancy que podem ser customizadas [41].

O crescente surgimento de APIs [16] web mostra que a utilização de APIs para permitir que usuários possam criar extensões de aplicações faz muito sentido no contexto de aplicações web e SaaS. Um exemplo de sucesso é o Tweetdeck, uma aplicação cliente do twitter que foi desenvolvida utilizando a API disponibilizada pelo twitter aos desenvolvedores e teve grande aceitação pelos usuários do twitter.com. O valor real da aquisição não foi declarado, mas especula-se algo em torno de 40 milhões de dólares [6]. Uma API web bem definida pode permitir que usuários de uma aplicação multi-tenancy possam, eles mesmos, criarem extensões para a aplicação e até fazer disso um negócio.

Outro exemplo de sucesso é o marketplace desenvolvido pela Salesforce, o AppExchange, que possui mais de 1 milhão de aplicações cadastradas, desde extensões da aplicação de CRM do Salesforce até aplicações para outros propósitos. Exemplos como esse mostram quão vantajoso é trazer desenvolvedores independentes para o mercado de SaaS.

A realidade é que para entender o campo de customização de aplicações multitenancy é necessário entender o contexto no qual essas aplicações estão surgindo. Dado que não é possível atender com uma única aplicação a necessidade de todos os clientes, como já foi mencionado anteriormente, enfrenta-se em aplicações multi-tenancy o desafio de permitir de alguma forma que o cliente adicione novas regras e customize a aplicação ou faça extensões que auxiliem na solução de seus problemas específicos. Durante esse trabalho foram encontrados vários estudos relacionados ao tema, muitos deles utilizando metadados, orientação a aspectos, SOA, etc.

Customização de aplicações já é um tema bastante explorado pelos pesquisadores da área de linha de produtos de software. Basicamente podemos considerar que a customização de aplicações multi-tenancy é equivalente a variabilidade de linha de produtos de software em tempo de execução. Uma iniciativa de customização de aplicações multi-tenancy tomando como base os conhecimentos de linha de produtos de software é apresentado por Jansen et al. [26].

Dado que SaaS tem como objetivo prover software para uma grande massa de consumidores, o atendimento aos requisitos de alocação de recursos, monitoramento, escalabilidade e performance são requisitos não funcionais que podem até ser ignorados em uma aplicação comum, mas que são de vital importância em aplicações multi-tenancy, principalmente se a aplicação estiver hospedada em um ambiente de cloud onde a tarifação (pagamento) é realizada por consumo. Nesses ambientes uma aplicação

mal projetada pode levar a custos operacionais tão altos que podem inviabilizar o funcionamento da aplicação, dado o valor que deverá ser pago para manter a infraestrutura de servidores funcionando.

Por outro lado, em uma contexto onde a alocação de recursos é feita de forma inteligente, o monitoramento indica quais recursos estão ociosos e quais os gargalos da aplicação. Nesse caso, se a elasticidade (capacidade de aumentar e diminuir os recursos de hardware) é realizada de forma automática e inteligente, os custos operacionais tendem a ser diminuidos drasticamente se a aplicação consumir somente o que é estritamente necessário para atender aos requisitos de SLA definidos aos clientes.

Nos últimos anos muitas empresas têm saído do modelo de entrega de software empacotado para o modelo de fornecimento de software na web. Essas aplicações entregues através da web vão desde emails a calendários, sistemas colaborativos, publicações online, processadores de texto simples, aplicações para negócios e aplicações para uso pessoal. Com o aumento da popularidade de marketing baseado em web, e-commerce e muitos outros serviços baseados em web, mesmo os pequenos negócios têm sido obrigados oferecer seus produtos e serviços na internet para serem competitivos no mercado. A característica de compatilhamento de recursos e diminuição de custos trazidos por multitenancy, têm feito dessa abordagem uma excelente alternativa para prover software de qualidade e com um baixo custo para pequenas e médias empresas.

Por fim, consideramos que a implementação de multi-tenancy não só é viável do ponto de vista de negócios como do ponto de vista técnico, embora ainda existam problemas e pontos de melhoria que devem ser tratados.

2.9. Referências

[1] M. Ahmad and I. T. Bowman. Predicting system performance for multi-tenant database workloads. In Proceedings of the Fourth International Workshop on Testing Database Systems, DBTest ’11, pages 6:1–6:6, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0655-3. doi: http://doi.acm.org/10.1145/1988842.1988848. URL http://doi.acm.org/10.1145/1988842.1988848.

[2] Amazon. What is AWS? Amazon Company, 2011. URL http://aws.amazon.com/pt/what-is-aws. Disponível em: <http://aws.amazon.com/pt/what-is-aws>. Acesso em: 10 dez. 2011.

[3] S. Aulbach, T. Grust, D. Jacobs, A. Kemper, and J. Rittinger. Multi-tenant databases for software as a service: schema-mapping techniques. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, SIGMOD ’08, pages 1195–1206, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-102-6.doi: http://doi.acm.org/10.1145/1376616.1376736. URL http://doi.acm.org/10.1145/1376616.1376736.

[4] S. Aulbach, D. Jacobs, A. Kemper, and M. Seibold. A comparison of flexible schemas for software as a service. In Proceedings of the 35th SIGMOD international conference on Management of data, SIGMOD '09, pages 881–888, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-551-2. doi: http://doi.acm.org/10.1145/1559845.1559941. URL http://doi.acm.org/10.1145/1559845.1559941.

[5] A. Azeez, S. Perera, D. Gamage, R. Linton, P. Siriwardana, D. Leelaratne, S. Weerawarana, and P. Fremantle. Multi-tenant soa middleware for cloud computing. In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pages 458 – 465, july 2010. doi: 10.1109/CLOUD.2010.50.

[6] BBC News. Twitter acquires Tweetdeck app for undisclosed fee. BBC News, 2011. URL http://www.bbc.co.uk/news/technology-13518382. Disponível em: <http://www.bbc.co.uk/news/technology-13518382>. Acesso em: 10 dez. 2011.

[7] D. Berberova and B. Bontchev. Design of service level agreements for software services. In Proceedings of the International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing, CompSysTech'09, pages 26:1–26:6, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-986-2. doi: http://doi.acm.org/10.1145/1731740.1731769. URL http://doi.acm.org/10.1145/1731740.1731769.

[8] C.-P. Bezemer and A. Zaidman. Multi-tenant saas applications: maintenance dream or nightmare? In Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE), IWPSE-EVOL '10, pages 88–92, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0128-2. doi: http://doi.acm.org/10.1145/1862372.1862393. URL http://doi.acm.org/10.1145/1862372.1862393.

[9] C.-P. Bezemer, A. Zaidman, B. Platzbeecker, T. Hurkmans, and A. 't Hart. Enabling multi-tenancy: An industrial experience report. In Software Maintenance (ICSM), 2010 IEEE International Conference on, pages 1 –8, sept. 2010. doi: 10.1109/ICSM.2010.5609735.

[10] J. Calero, N. Edwards, J. Kirschnick, L. Wilcock, and M. Wray. Toward a multitenancy authorization system for cloud services. Security Privacy, IEEE, 8(6):48–55, nov.-dec. 2010. ISSN 1540-7993. doi: 10.1109/MSP.2010.194.

[11] X. Cheng, Y. Shi, and Q. Li. A multi-tenant oriented performance monitoring, detecting and scheduling architecture based on sla. In Pervasive Computing (JCPC), 2009 Joint Conferences on, pages 599 –604, dec. 2009. doi: 10.1109/JCPC.2009.5420114.

[12] F. Chong and G. Carraro. Architecture strategies for catching the long tail. 2006. URL http://msdn2.microsoft.com/en-us/library/aa479069.aspx. Disponível em: <http://msdn2.microsoft.com/en-us/library/aa479069.aspx>. Acesso em: 10 dez. 2011.

[13] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. In Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation Volume 2, NSDI'05, pages 273–286, Berkeley, CA, USA, 2005. USENIX Association. URL http://dl.acm.org/citation.cfm?id=1251203.1251223.

[14] D. Diana, Mauricio, Gerosa, and M. Aurélio. Nosql na web 2.0: Um estudo comparativo de bancos não-relacionais para armazenamento de dados na web 2.0. Communications, 2010.

[15] S. Dustdar and W. Schreiner. A survey on web services composition. Int. J. Web Grid Serv., 1:1–30, August 2005. ISSN 1741-1106. doi: 10.1504/IJWGS.2005.007545. URL http://dl.acm.org/citation.cfm?id=1358537.1358538.

[16] A. DuVander. Api growth doubles in 2010, social and mobile are trends. 2011. URL http://blog.programmableweb.com/2011/01/03/api-growth-doubles-in-2010-social-and-mobile-are-trends/.

[17] A. J. Elmore, S. Das, D. Agrawal, and A. El Abbadi. Zephyr: live migration in shared nothing databases for elastic cloud platforms. In Proceedings of the 2011 international conference on Management of data, SIGMOD '11, pages 301–312, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0661-4. doi: http://doi.acm.org/10.1145/1989323.1989356.

[18] C. Fehling, F. Leymann, and R. Mietzner. A framework for optimized distribution of tenants in cloud applications. In Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing, CLOUD '10, pages 252–259, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4130-3. URL http://dx.doi.org/10.1109/CLOUD.2010.33.

[19] F. Foping, I. Dokas, J. Feehan, and S. Imran. A new hybrid schema-sharing technique for multitenant applications. In Digital Information Management, 2009. ICDIM 2009. Fourth International Conference on, pages 1 –6, nov. 2009. doi:10.1109/ICDIM.2009.5356775.

[20] C. J. Guo, W. Sun, Y. Huang, Z. H. Wang, and B. Gao. A framework for native multitenancy application development and management. In E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, 2007. CEC/EEE 2007. The 9th IEEE International Conference on, pages 551 –558, july 2007. doi: 10.1109/CEC-EEE.2007.4.

[21] I. S. Harris and Z. Ahmed. An open multi-tenant architecture to leverage smes. European Journal of Scientific Research, 65(4):601–610, 2011.

[22] P. Hasselmeyer and N. d'Heureuse. Towards holistic multi-tenant monitoring for virtual data centers. In Network Operations and Management Symposium Workshops (NOMS Wksps), 2010 IEEE/IFIP, pages 350 –356, april 2010. doi: 10.1109/NOMSW.2010.5486528.

[23] J. Huang. Gartner says worldwide software as a service revenue is forecast to grow 21 percent in 2011. 2011. URL http://www.gartner.com/it/page.jsp?id=1739214. Disponível em: <http://www.gartner.com/it/page.jsp?id=1739214>. Acesso em: 10 dez. 2011.

[24] J. Huang. Amazon s3's amazing growth. Cloud Computing Journal, 2011. URL http://cloudcomputing.sys-con.com/node/1699373. Disponível em: <http://cloudcomputing.sys-con.com/node/1699373>. Acesso em: 10 dez. 2011.

[25] B. Jacob. Introduction to Grid Computing. IBM redbooks. IBM, International Technical Support Organization, 2005. ISBN 9780738494005. URL http://books.google.com.br/books?id=4GSUtgAACAAJ.

[26] S. Jansen, G.-J. Houben, and S. Brinkkemper. Customization realization in multitenant web applications: case studies from the library sector. In Proceedings of the 10th international conference on Web engineering, ICWE'10, pages 445–459, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-13910-8, 978-3-642-13910-9. URL http://dl.acm.org/citation.cfm?id=1884110.1884147.

[27] A. Jasti, P. Shah, R. Nagaraj, and R. Pendse. Security in multi-tenancy cloud. In Security Technology (ICCST), 2010 IEEE International Carnahan Conference on, pages 35 –41, oct. 2010. doi: 10.1109/CCST.2010.5678682.

[28] J. Jing and J. Zhang. Research on open saas software architecture based on soa. In Computational Intelligence and Design (ISCID), 2010 International Symposium on, volume 2, pages 144 –147, oct. 2010. doi: 10.1109/ISCID.2010.125.

[29] C. M. Judd, J. F. Nusairat, J. Shingler, M. Moodie, J. Ottinger, J. Pepper, F. Pohlmann, B. Renow-clarke, D. Shakeshaft, M. Wade, and et al. Beginning groovy and grails from novice to professional. Specialist, page 440, 2008. URL http://books.google.com/books?hl=en&lr=&id=9AxFrcvawvAC&oi=fnd&pg=PR15&dq=Beginning+Groovy+and+Grails:+From+Novice+to+Professional&ots=AWQ-JVaL8&sig= BzgKgJMF9jI7E27n9UwfxSRT4OY.

[30] L. Kleinrock. An internet vision: the invisible global infrastructure. Ad Hoc Networks, 1(1):3–11, 2003. URL http://linkinghub.elsevier.com/retrieve/pii/S157087050300012X.

[31] H. Koziolek. Towards an architectural style for multi-tenant software applications. Proc Software Engineering 2010 SE10 Fachtagung des GIFachbereichs Softwaretechnik, To appear:81–92, 2010.nURL http://www.koziolek.de/docs/ Koziolek2010-SE.pdf.

[32] H. Koziolek. The sposad architectural style for multi-tenant software applications. In Software Architecture (WICSA), 2011 9th Working IEEE/IFIP Conference on, pages 320 –327, june 2011. doi: 10.1109/WICSA.2011.50.

[33] T. Kwok and A. Mohindra. Resource calculations with constraints, and placement of tenants and instances for multi-tenant saas applications. In Proceedings of the 6th International Conference on Service-Oriented Computing, ICSOC '08, pages 633–648, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-89647-0. URL http://dx.doi.org/10.1007/978-3-540-89652-4_57.

[34] T. Kwok, T. Nguyen, and L. Lam. A software as a service with multi-tenancy support for an electronic contract management application. In Services Computing, 2008. SCC '08. IEEE International Conference on, volume 2, pages 179 –186, july 2008. doi: 10.1109/SCC.2008.138.

[35] S. Lehnen. Understanding the basic building blocks of sales force crm. 2011. URL http://www.salesforce.com/customer-resources/learning-center/details/best-practices/understanding-the-basic-building.jsp. Disponível em: <http://www.salesforce.com/customer-resources/learning-center/details/best-practices/understanding-the-basic-building.jsp>. Acesso em: 28 dez. 2011.

[36] X. H. Li, T. C. Liu, Y. Li, and Y. Chen. Spin: Service performance isolation infrastructure in multi-tenancy environment. In Proceedings of the 6th International Conference on Service-Oriented Computing, ICSOC '08, pages 649–663, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-89647-0. doi: http://dx.doi.org/10.1007/978-3-540-89652-4_58. URL http://dx.doi.org/10.1007/978-3-540-89652-4_58.

[37] X.-Y. Li, Y. Shi, Y. Guo, and W. Ma. Multi-tenancy based access control in cloud. In Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on, pages 1 –4, dec. 2010. doi: 10.1109/CISE.2010.5677061.

[38] H. Lin, K. Sun, S. Zhao, and Y. Han. Feedback-control-based performance regulation for multi-tenant applications. In Parallel and Distributed Systems (ICPADS), 2009 15th International Conference on, pages 134 –141, dec. 2009. doi: 10.1109/ICPADS.2009.22.

[39] H. Lin, K. Sun, S. Zhao, and Y. Han. Feedback-control-based performance regulation for multi-tenant applications. In Parallel and Distributed Systems (ICPADS), 2009 15th International Conference on, pages 134 –141, dec. 2009. doi: 10.1109/ICPADS.2009.22.

[40] H. Liu, H. Jin, X. Liao, L. Hu, and C. Yu. Live migration of virtual machine based on full system trace and replay. In Proceedings of the 18th ACM international symposium on High performance distributed computing, HPDC '09, pages 101–110, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-587-1. URL http://doi.acm.org/10.1145/1551609.1551630.

[41] B. F. Lóscio, H. R. d. Oliveira, and J. C. d. S. Pontes. Nosql no desenvolvimento de aplicações web colaborativas. VIII Simpósio Brasileiro de Sistemas Colaborativos, 2011.

[42] P. Mell and T. Grance. The nist definition of cloud computing. National Institute of Standards and Technology, 53(6):50, 2009. URL http://www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf.

[43] A. Mesbah and A. van Deursen. Crosscutting concerns in j2ee applications. In Web Site Evolution, 2005. (WSE 2005). Seventh IEEE International Symposium on, pages 14 – 21, sept. 2005. doi: 10.1109/WSE.2005.4.

[44] R. Mietzner, T. Unger, R. Titze, and F. Leymann. Combining different multitenancy patterns in service-oriented applications. In Proceedings of the 2009 IEEE International Enterprise Distributed Object Computing Conference (edoc 2009), EDOC '09, pages 131–140, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-0-7695-3785-6. doi: http://dx.doi.org/10.1109/EDOC.2009.13. URL http://dx.doi.org/10.1109/EDOC.2009.13.

[45] T. Neil. 12 standard screen patterns, 2010. URL http://designingwebinterfaces.com/designing-web-interfaces-12-screen-patterns. Disponível em:<http://designingwebinterfaces.com/designing-web-interfaces-12-screen-patterns>. Acesso em: 10 dez. 2011.

[46] Z. Nie. Credibility evaluation of saas tenants. In Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on, volume 4, pages V4–488 –V4–491, aug. 2010. doi: 10.1109/ICACTE.2010.5579322.

[47] Nitu. Configurability in saas (software as a service) applications. In Proceedings of the 2nd India software engineering conference, ISEC '09, pages 19–26, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-426-3. doi: http://doi.acm.org/10.1145/1506216.1506221. URL http://doi.acm.org/10.1145/1506216.1506221.

[48] B.-T. Oh, H.-S. Won, and S.-J. Hur. Multi-tenant supporting online application service system based on metadata model. In Advanced Communication Technology (ICACT), 2011 13th International Conference on, pages 1173 –1176, feb. 2011.

[49] Z. Pervez, S. Lee, and Y.-K. Lee. Multi-tenant, secure, load disseminated saas architecture. In Advanced Communication Technology (ICACT), 2010 The 12th International Conference on, volume 1, pages 214 –219, feb. 2010.

[50] C. Richardson. Orm in dynamic languages. Commun. ACM, 52(4):48–55, Apr.2009. ISSN 0001-0782. doi: 10.1145/1498765.1498783. URL http://doi.acm.org/10.1145/1498765.1498783.

[51] J. Schaffner, B. Eckart, D. Jacobs, C. Schwarz, H. Plattner, and A. Zeier. Predicting in-memory database performance for automating cluster management tasks. In Data Engineering (ICDE), 2011 IEEE 27th International Conference on, pages 1264–1275, april 2011. doi: 10.1109/ICDE.2011.5767936.

[52] O. Schiller, B. Schiller, A. Brodt, and B. Mitschang. Native support of multitenancy in rdbms for software as a service. In Proceedings of the 14th International Conference on Extending Database Technology, EDBT/ICDT '11, pages 117–128, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0528-0. doi: http://doi.acm.org/10.1145/1951365.1951382. URL: http://doi.acm.org/10.1145/1951365.1951382.

[53] L. Shwartz, Y. Diao, and G. Grabarnik. Multi-tenant solution for it service management: A quantitative study of benefits. In Integrated Network Management, 2009. IM '09. IFIP/IEEE International Symposium on, pages 721 –731, june 2009. doi: 10.1109/INM.2009.5188879.

[54] W.-T. Tsai and Q. Shao. Role-based access-control using reference ontology in clouds. In Autonomous Decentralized Systems (ISADS), 2011 10th International Symposium on, pages 121 –128, march 2011. doi: 10.1109/ISADS.2011.21.

[55] W.-T. Tsai, Q. Shao, and J. Elston. Prioritizing service requests on cloud with multitenancy. In e-Business Engineering (ICEBE), 2010 IEEE 7th International Conference on, pages 117 –124, nov. 2010. doi: 10.1109/ICEBE.2010.38.

[56] W.-T. Tsai, X. Sun, Q. Shao, and G. Qi. Two-tier multi-tenancy scaling and load balancing. In e-Business Engineering (ICEBE), 2010 IEEE 7th International Conference on, pages 484 –489, nov. 2010. doi: 10.1109/ICEBE.2010.103.

[57] Q. H. Vu, M. Lupu, and B. C. Ooi. Peer-to-peer computing. Media, 2010. URL http://www.springerlink.com/index/10.1007/978-3-642-03514-2.

[58] Z. H. Wang, C. J. Guo, B. Gao, W. Sun, Z. Zhang, and W. H. An. A study and performance evaluation of the multi-tenant data tier design patterns for service oriented computing. In Proceedings of the 2008 IEEE International Conference on e-Business Engineering, pages 94–101, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3395-7. doi: 10.1109/ICEBE.2008.60. URL http://dl.acm.org/citation.cfm?id=1471605.1472222.

[59] C. Weiliang, Z. Shidong, and K. Lanju. A multiple sparse tables approach for multitenant data storage in saas. In Industrial and Information Systems (IIS), 2010 2nd International Conference on, volume 1, pages 413 –416, july 2010. doi: 10.1109/INDUSIS.2010.5565824.

[60] C. D. Weissman and S. Bobrowski. The design of the force.com multitenant internet application development platform. In Proceedings of the 35th SIGMOD international conference on Management of data, SIGMOD '09, pages 889–896, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-551-2. URL http://doi.acm.org/10.1145/1559845.1559942.

[61] W. Xue, L. Qingzhong, and K. Lanju. Multiple sparse tables based on pivot table for multi-tenant data storage in saas. In Information and Automation (ICIA), 2011 IEEE International Conference on, pages 634 –637, june 2011. doi: 10.1109/ICINFA.2011.5949071.

[62] C. S. Yeo, R. Buyya, H. Pourreza, R. Eskicioglu, P. Graham, F. Sommers, and G. Computing. Cluster computing: High-performance, high-availability, and highthroughput processing on a network of computers. Communication, pages 1–24, 2006. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.66.1453.

[63] D. Yuanyuan, N. Hong, W. Bingfei, and L. Lei. Scaling the data in multi-tenant business support system. In Knowledge Engineering and Software Engineering, 2009. KESE '09. Pacific-Asia Conference on, pages 43 –46, dec. 2009. doi: 10.1109/KESE.2009.20.

[64] K. Zhang, Y. Shi, Q. Li, and J. Bian. Data privacy preserving mechanism based on tenant customization for saas. In Multimedia Information Networking and Security, 2009. MINES '09. International Conference on, volume 1, pages 599 –603, nov. 2009. doi: 10.1109/MINES.2009.256.

[65] X. Zhang, B. Shen, X. Tang, and W. Chen. From isolated tenancy hosted application to multi-tenancy: Toward a systematic migration method for web application. In Software Engineering and Service Sciences (ICSESS), 2010 IEEE International Conference on, pages 209 –212, july 2010. doi: 10.1109/ICSESS.2010.5552407.

[66] Y. Zhang, Z. Wang, B. Gao, C. Guo, W. Sun, and X. Li. An effective heuristic for on-line tenant placement problem in saas. In Web Services (ICWS), 2010 IEEE International Conference on, pages 425 –432, july 2010. doi: 10.1109/ICWS.2010. 65.

[67] J. Zheng, X. Li, X. Zhao, X. Zhang, and S. Hu. The research and application of a saas-based teaching resources management platform. In Computer Science and Education (ICCSE), 2010 5th International Conference on, pages 765 –770, aug. 2010. doi: 10.1109/ICCSE.2010.5593500.

Capítulo

3

Desafios em Cloud computing: Armazenamento, Banco de Dados e BIG Data

Rodrigo Elia Assad[1], Marco André Santos Machado[1,2], Paulo Fernando Almeida Soares[1,2], Anderson Fonseca e Silva[1] , Thiago Jamir e Silva[1,2], Vinicius Cardoso Garcia[2] Fernando Mota Trinta[3], Silvio Romeiro Lemos Meira[2]

[1]USTO.RE {assad@usto.re, marco@usto.re, paulo@usto.re, anderson@usto.re, tjamir@usto.re}
[2]Universidade Federal de Pernambuco { vcg@cin.ufpe.br, srlm@cin.ufpe.br }
[3]Universidade Federal do Ceará { fernando.trinta@lia.ufc.br }

Abstract

With increasing connectivity speed and Web systems evolution, emerges the Internet systems, which are more commonly called cloud computing. It´s designates a support platform that provides: management, on-demand use, fitness requirements, rational use of resources and automation of processes related to creation of infrastructure. On this context emerge systems of data storage in the cloud, database scalability and data search and retrieval now called as BIGDATA. This short course addresses these exemplifying how we implement and use such platforms.

Resumo

Com o aumento da velocidade de conectividade e evolução dos sistemas WEB, começa a surgir os sistemas de Internet, que mais comumente são chamados de Computação nas Nuvens. Este termo designa uma plataforma de suporte a sistemas de software que provê aos seus usuários: gerenciamento, uso sob demanda, adequação às necessidades, racionalização do uso dos recursos e automação dos processos relacionados a criação de infra-estruturas. Neste contexto surge os sistemas de armazenamento de dados em nuvem, escalabilidade de banco de dados e busca e recuperação que hoje chamamos de BIGDATA. Este minicurso aborda estes temos exemplificando como implementar e utilizar tais plataformas.

3.1 Introdução

Em 1990, Tim Berners-Lee deu início à implementação das linguagens, dos protocolos e dos artefatos de software necessários à construção da *World Wide Web*, ou simplesmente web, como conhecemos hoje [W3C, 2000]. Duas décadas depois, a web ganhou proporções mundiais, atingindo mais de um bilhão e meio de pessoas em todos os continentes[30]. Ao longo desta história, a mesma passou por constantes evoluções, permitindo uma variedade de aplicações não antes previstas por Tim Berners-Lee, as quais compõem o momento atual da web. A **Erro! Fonte de referência não encontrada.** apresenta a distribuição dos usuários de internet por continente em 2009.

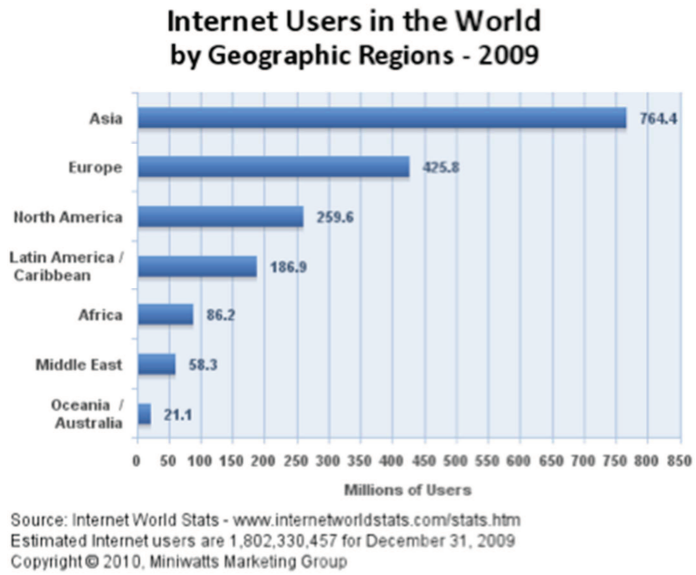


Figura 3.1 - Distribuição dos usuários de internet por continente em 2009

Adicionalmente, de acordo com um estudo do *The Economist*[31] no início de 2010 estimava-se que a quantidade de informação gerada no planeta, no ano de 2010, seria duas vezes maior do que o armazenamento disponível para guardá-la, caracterizando assim um "*dilúvio informacional*". Isto faz com que haja um grande aumento na demanda de volume de recursos para o apoio as tarefas de elicitação, compartilhamento, manipulação e exploração de grandes conjuntos de dados, bem como para o desenvolvimento e execução de serviços de apoio à Pesquisa.

A *McKinsey Quartely*[32] [Bughin et al., 2010], jornal de negócios da *McKinsey & Company*, divulgou artigo em que relaciona 10 tendências de tecnologia que devem estar no radar dos estrategistas das empresas para geração de novos modelos de negócios. O artigo faz um alerta aos altos executivos das empresas sugerindo que eles devem pensar estrategicamente sobre como preparar as organizações para o novo ambiente desafiador que se aproxima com o crescimento de tecnologias como computação em nuvem, internet das coisas, colaboração em escala entre outras.

[30] URL: http://bit.ly/MM7XnF, Acessado em 05/07/2012
[31] URL: http://econ.st/MM7YrC, Acessado em 05/07/2012
[32] URL: http://bit.ly/9stJjH, Acessado em 05/07/2012

Ainda segundo o artigo da *McKinsey*, a tecnologia continua a evoluir rapidamente. *Facebook*[33], em pouco mais de dois curtos anos, quintuplicou de tamanho a uma rede que atinge mais de 500 milhões de usuários. Mais de 4 bilhões de pessoas ao redor do mundo já utilizam telefones celulares, e para 450 milhões de pessoas a Web é uma experiência completamente móvel. A maneira como utilizamos tecnologia, como Computação em Nuvem (*Cloud Computing*) e virtualização, redistribuem os custos de tecnologia, criam novas formas para os indivíduos consumirem produtos e serviços e torna possível que empresários e empresas pensem em novos modelos de negócios que muitos consideram impossíveis como, por exemplo, marketing contextual.

A *McKinsey* também alerta que as empresas não devem se limitar a compreender as 10 tendências abaixo descritas. Elas precisam também pensar estrategicamente sobre como adaptar a gestão e estruturas organizacionais para atender as novas demandas proporcionadas por essas 10 tendências, pois muitas dessas tendências exigirão a necessidade de atravessar fronteiras organizacionais tradicionais fazendo com que líderes criem conexões entre as equipes em diferentes e espalhados cantos da organização, buscando maior interdisciplinaridade de forma que toda a empresa compreenda a necessidade de explorar plenamente essas tendências. Isso implica em transformar as organizações em pequenos laboratórios capazes de testar mais rapidamente e aprender mais rapidamente em pequena escala e depois expandir o sucesso rapidamente.

As 10 tendências citadas pela *McKinsey* [Bughin et al., 2010] são:

1. **Distribuição estimula a Co-Criação** (*Distributed cocreation moves into the mainstream*): Inspirados pelo pioneirismo da *Wikipedia* e uma comunidade de desenvolvedores de software de código aberto, a capacidade de organizar as comunidades da Web para desenvolver , comercializar produtos e serviços de apoio passou das margens da prática empresarial para o *mainstream*, onde nos dias de hoje diversas empresas tem pautado seu negócio (ou o relacionamento com seus clientes, no mínimo) que tem como objetivo redes abertas de compartilhamento de conhecimento;
2. **Transformar a organização em uma rede** (*Making the network the organization*): Muitas empresas estão indo além fronteiras estaduais, nacionais ou até continentais, construindo redes flexíveis que se estendem além das fronteiras internas e externas da própria organização. Barreiras culturais, burocráticas ou até processuais em torno de determinados silos organizacionais podem impedir que os gestores acessem os melhores talentos em toda a organização para resolver problemas críticos com soluções criativas. Utilizando tecnologias da Web, por exemplo, para expandir o acesso a especialistas de todo o mundo, as empresas podem montar comunidades de inovação entre as unidades de negócios em silos, acelerando a prestação de serviços e melhorando simultaneamente a qualidade destes serviços e do relacionamento com todos os seus *stakeholders*;
3. **Colaboração em escala** (*Collaboration at scale*): Com o advento da Internet e da WEB 2.0, o número de pessoas que realizam trabalho, muitas vezes colaborativamente, de conhecimento (voltado para inovação e criação de riqueza em cada setor da economia) tem crescido muito mais rapidamente do que os

[33] URL: http://www.facebook.com, Acessado em 05/07/2012

trabalhadores de produção de bens. Como resultado, é crescente o interesse em tecnologias de colaboração que prometem melhorar o rendimento e a eficácia desses trabalhadores;

4. **A crescente "Internet das Coisas"** (*The growing 'Internet of Things'*): Dispositivos, eletrodomésticos, embalagens, *containers*, entre outros objetos (ou genericamente "*coisas*"), embutidos de sensores, atuadores e capacidades de comunicações, em breve serão capazes de absorver e transmitir informações em grande escala e, em alguns casos, adaptar-se e reagir às mudanças no ambiente automaticamente. Estes ativos "inteligentes" podem tornar os processos mais eficientes, fornecendo novos recursos e novos modelos de negócio das empresas;
5. **Experimentação e Grandes Volumes de Dados** (*Experimentation and big data*): O volume de dados vem crescendo a taxas nunca antes vistas, e tecnologias para a captura e análise de informações estão amplamente disponíveis a preços cada vez mais baixos. Muitas empresas estão levando o uso desses dados para novos níveis, utilizando a Tecnologia da Informação e Comunicação (TIC) para suportar a experimentação constante de negócios que orientem as decisões e para testar novos produtos, modelos de negócios e inovações na experiência do cliente. Esta tendência tem o potencial de conduzir a uma transformação radical na pesquisa, inovação e na forma como as empresas conduzem seus negócios. Isso exigirá que TIC e Negócios não sejam mais compreendidos como áreas estanques e separadas, e sim como um único problema, ou seja, TIC não é mais parte do negócio, na verdade, é necessário um modelo de negócio que utilize a TIC como instrumento do negócio;
6. **Cabeamento para um mundo sustentável** (*Wiring for a sustainable world*): Mesmo com os marcos regulatórios continuando a evoluir, gestão ambiental e sustentabilidade são temas prioritários da agenda empresarial. Além do mais, a sustentabilidade está se tornando rapidamente uma importante métrica de desempenho corporativo, que os *stakeholders*, incluindo até mesmo os mercados financeiros começaram a dar mais atenção. TIC desempenha um duplo papel neste debate: ela é uma fonte significativa de emissões para o ambiente e um fator essencial de muitas estratégias para mitigar os danos ambientais. A pesquisa da *McKinsey* mostra, no entanto , que o uso da TIC em áreas como as redes de energia inteligentes, edifícios eficientes e melhor planejamento de logística poderia eliminar cinco vezes as emissões de carbono que a indústria produz;
7. **Imagine TUDO como Serviço** (*Imagining anything as a service*): Cada vez mais podemos afirmar que TIC é serviço e não produto. Consumidores desejam pagar apenas pelo uso e, dessa forma, evitar grandes gastos e futuras dificuldades de compra e manutenção de um produto. Na indústria de TIC, o crescimento de "*cloud computing*" [Armbrust et al., 2009; Galán & Sampaio, 2009; Leavitt, 2009; Smith et al., 2009; Sun, 2009] exemplifica esta mudança, além de software como serviço (SaaS) [Turner et al., 2003], que permite às organizações acesso a serviços como, por exemplo, a gestão de relacionamento com clientes. A aceitação pelos consumidores dos serviços de *cloud* baseada na Web para tudo, desde e-mail até armazenamento de vídeos e fotos, tende naturalmente a tornar-se universal, e as companhias estão seguindo essa tendência;

8. **A era do modelo de negócios multilaterais** (*The age of the multisided business model*): O maior exemplo desse modelo de negócios é o Google, que vende publicidade oferecendo serviços de TIC que permitem a busca e organizam a informação na Web. Outro exemplo, citado no artigo da *McKinsey*, é a empresa *Spiceworks*[34] oferece gerenciamento de aplicativos de TIC para 950 mil usuários, sem qualquer custo, enquanto vende espaço publicitário para empresas de B2B que querem acesso a profissionais de TIC. Quanto mais pessoas migrarem para atividades on-line, o valor dos modelos de negócio multilaterais são ampliados por estes efeitos proporcionados pela rede, sendo que quanto maior o número de usuários, utilizando os serviços gratuitamente, mais valioso o serviço torna-se para todos os clientes;
9. **Inovando na base da pirâmide** (*Innovating from the bottom of the pyramid*): A adoção de tecnologia é um fenômeno global, e a intensidade de seu uso é particularmente impressionante nos mercados emergentes. A pesquisa da *McKinsey* mostrou que os modelos de negócios disruptivos surgem quando a tecnologia se combina com condições extremas de mercado, tais como a demanda dos clientes por preços muito baixos, pouca infraestrutura, fornecedores de difícil acesso e curvas de baixo custo para o talento;
10. **Expandir serviços públicos** (*Producing public good on the grid*): É preciso entender que o custo de transação das empresas, como o trânsito caótico para movimentação das pessoas, cresce cada vez mais tornando-se inviável do ponto de vista macroeconômico e isso tende a se agravar, uma vez que cerca de metade da população mundial vive em áreas urbanas, e essa parte é projetada para atingir 70 por cento em 2050. Políticas públicas criativas, que incorporem as novas tecnologias, poderiam ajudar a aliviar as tensões econômicas e sociais da densidade populacional. Londres, Singapura e Estocolmo têm utilizado recursos inteligentes para gerenciar o congestionamento do tráfego em seus núcleos urbanos, e muitas cidades em todo o mundo estão implementando essas tecnologias para melhorar a confiabilidade e a previsibilidade dos sistemas de transporte de massa. Da mesma forma, redes de água inteligentes (*networked water smart grids)* serão fundamentais para responder à necessidade de água limpa [Baumgarten & Chui, 2009]. A plena exploração do potencial da tecnologia na esfera pública significa re-imaginar a forma como os serviços públicos são criados , entregues e gerenciados;

Torna-se imperativo que as instituições considerem adotar tais tendências de tecnologia para sua indústria de Tecnologia de Informação e Comunicação, tanto para atender de forma efetiva sua demanda interna, quanto para tornar-se uma potência exportadora de sistemas de software que, com as características citadas, teriam qualidade e rapidez superior aos seus concorrentes.

Como uma plataforma chave de fornecimento de serviço na área de computação de serviços [Zhang et al., 2007], Computação em Nuvem oferece ambientes que permitem o compartilhamento de recursos em termos de infraestruturas escaláveis, middleware e plataformas de desenvolvimento de aplicações, algumas com alto valor agregado ao negócio. Os modelos de operação podem incluir modelos de utilidade de

[34] URL: http://www.spiceworks.com/, acessado em 28/08/2010

pagamento pelo uso (*pay-as-go*), serviços de infraestrutura grátis com plataformas de serviço com valor agregado, serviços de infraestruturas baseados em taxas composto por serviços de aplicações com valor agregado ou de serviços gratuitos para os fornecedores, mas compartilhando as receitas geradas pelos consumidores.

Tipicamente, existem quarto tipos de recursos que podem ser provisionados e consumidos através da Internet [Zhang et al., 2008]. Estes recursos podem ser compartilhados entre os usuários por meio da economia de escala. O provisionamento é uma forma de compartilhar recursos com solicitantes através da rede. Um dos objetivos principais da computação em nuvem é aproveitar a Internet ou uma Intranet para provisão de recursos para os usuários.

O primeiro tipo de recursos são os recursos de infraestrutura, que incluem o poder de computação, armazenamento e fornecimento de máquinas. Por exemplo, o Amazon EC2 oferece interface de serviço web para facilmente solicitar e configurar capacidades online [Amazon EC2, 2012]. O serviço *Xdrive Box* oferece armazenamento online para os usuários [Xdrive, 2012]. O *Microsoft SkyDrive* oferece serviço de armazenamento gratuito, com um modelo integrado *offline* e *online* que mantém a privacidade relacionada com arquivos em discos rígidos, e permite às pessoas acessar esses arquivos remotamente [Microsoft, 2012]. Na área de compartilhamento de poder computacional, a computação em Grid tomou como seu foco principal a utilização de clusterização e tecnologias de computação paralela para compartilhar o poder computacional com os outros usuários (consumidores do serviço), com base no agendamento de tarefas quando os computadores estão ociosos.

O segundo tipo de recursos em computação em nuvem são os recursos de software, incluindo middlewares e recursos de desenvolvimento. O middleware consiste em sistemas operacionais centrados na nuvem, servidores de aplicação, banco de dados, entre outros. Os recursos de desenvolvimento compreendem plataformas de projeto, ferramentas de desenvolvimento, teste e implantação além de projetos de referência baseados em outros projetos de código aberto [Zhang and Zhou, 2009].

O terceiro tipo de recursos em computação em nuvem são os recursos de aplicação. As empresas de TIC estão migrando gradualmente aplicações e dados para Internet (nuvem). As aplicações de software são entregues através do modelo de Software como Serviço (*SaaS – Software as a Service*) ou *mashups*[35] de aplicações de valor agregado. Por exemplo, o Google tem utilizado computação em nuvem para oferecer aplicações Web para comunicação e colaboração [Google, 2012]. O pacote de escritório do Google, o Google Docs, moveu gradativamente para a Web as aplicações de produtividade anteriormente existentes apenas no modelo de licenciamento por máquina. Desta forma, fica claro que desenvolvendo uma arquitetura aberta de computação em nuvem reutilizável e customizável possibilitando um efetivo ambiente de desenvolvimento de aplicações como serviço é a chave para o compartilhamento (ou distribuição) de aplicações por meio da Internet.

O quarto tipo de recursos na computação em nuvem são os processos de negócio. Algumas aplicações podem ser compartilhadas (publicadas) como serviços públicos (*utilities*), ou seja, por meio de sub-processos com baixo acoplamento ou tarefas dentro de processos de negócio de clientes. O compartilhamento de processos de

[35] URL: http://pt.wikipedia.org/wiki/Mashup, Acessado em 09/07/2012

negócios é a terceirização de aplicativos orientados a negócios que dá suporte ao reuso, composição e provisionamento de aplicações.

Neste contexto, está claro o porque de grandes empresas estarem investindo recursos de esforço neste novo modelo computacional. Principalmente as fornecedoras de serviço como Google, Amazon e Microsoft que possuem hoje suas plataformas de serviços que permitem o aluguel de sua infraestrutura de máquinas para que terceiros possam hospedar aplicações, repassando questões como administração de máquinas, controle de escalabilidade, dentre outras, para as empresas contratadas.

Atualmente, boa parte da economia mundial já é baseada em sistemas de TIC, e a demanda por software vai aumentar ainda mais nas próximas décadas [Osterweil, 2007]. Estas mudanças se darão devido a fatores como o uso de software em larga escala por meio de infraestrutura de serviços, novos tipos de aplicações de integração de negócios e novas plataformas, além de combinação de serviços oferecidos via Web e dispositivos móveis.

3.1.1 Sistemas de Informação e os modelos de computação em nuvem

Sistemas de informação compreendem pessoas, componentes físicos (hardware) e lógicos (software) que, ligados por redes de comunicação, permitem a transformação de dados brutos em informação e conhecimento. A quantidade e diversidade dos dados requerem cuidados adicionais sobre seu armazenamento, como por exemplo, segurança, confiabilidade, entre outros, além de suporte a procedimentos comuns como: backups, salvamento de arquivos e sua recuperação.

Um sistema ao utilizar o novo paradigma de computação em nuvem, o faz por meio de três modelos básicos:

a) Software como Serviço (*SaaS* - do inglês, *Software as a Service*): Neste modelo, o provedor da infraestrutura de computação em nuvem cuida de todos os aspectos relacionados à disponibilidade, manutenção e adição de novas funcionalidades para possíveis clientes. Exemplos são *Google Docs*, *OfficeLive (+)* e o *SalesForce* que fornecem versões online de editores de texto, planilhas de cálculo e demais programas de escritório;
b) Plataforma como Serviço (*PaaS* - do inglês, *Platform as a Service*): Nesta outra abordagem, o provedor oferta uma plataforma que será utilizada pela aplicação do usuários final. Exemplos: *Amazon S3*, *AllMyData*, *Windows Azure* que fornecem sistemas de armazenamento;
c) Infraestrutura como Serviço (*IaaS*, do inglês, *Infrastructure as a Service*): O provedor oferece um ambiente virtualizado e gerenciável pelo usuário. Exemplos: *Amazon EC2*, *CloudBR* que fornecem sistemas virtualizados para processamento sobre demanda.

Em qualquer um dos modelos apresentados, um propósito básico é a racionalização da utilização dos recursos computacionais por uma organização ou um conjunto de organizações. A ideia é que clientes em um modelo de computação em nuvem possam melhor administrar o dimensionamento de seus recursos de infraestrutura, software, aplicação e/ou de negócio. Neste modelo, aplicações tornam-se escaláveis de forma automática, onde os encargos relacionados ao seu correto funcionamento são cobrados de acordo com a demanda das próprias aplicações.

No caso especifico de uma plataforma oferecida como serviço no paradigma de computação em nuvem, deve ser entendida com uma infraestrutura que permita a seus usuários a utilização de serviços por meio de uma interface pública, estabelecida através de uma interface de programação - *API* (do inglês, *Application Programming Interface*). Ao utilizar esta *API*, cada participante pode utilizar e oferecer recursos para os demais integrantes da nuvem, permitindo uma melhor utilização dos recursos compartilhados, otimização de processamento, armazenamento, transferência, dentre outros fatores.

Os dados de uma plataforma em modelo de computação em nuvem podem ser distribuídos entre os vários participantes da nuvem, aumentando a sua disponibilidade, assim como otimizando o uso de recursos subutilizados. Porém, há problemas a serem enfrentados, tais como escalabilidade horizontal segurança, disponibilidade.

Em geral, a infraestrutura de uma plataforma de computação em nuvem é construída sob uma tradicional forma da topologia cliente/servidor, onde o lado servidor da nuvem é composto por um alto número de máquinas que criam a abstração de um único provedor de serviços em cluster. Apesar de ser um modelo consolidado, existem problemas associados ao custo da manutenção desta infraestrutura, bem como do consumo de banda, energia, dentre outros fatores. Como alternativa para este modelo, ao longo dos anos tem se visto o crescimento de soluções baseados em topologia ponto a ponto - P2P (do inglês, *Peer to Peer*), onde não há a figura de um host ou conjunto de hosts que assume o papel de servidor ou coordenador. Os exemplos mais conhecidos de aplicações P2P são as redes de compartilhamento de arquivos, como *gnutella* ou *e-donkey*.

Aplicações P2P lidam melhor com questões de escalabilidade e tolerância a falhas do que aquelas no modelo cliente/servidor, que também contam com um custo adicional devido à complexidade necessária para o gerenciamento das buscas e roteamento dos dados entre seus participantes.

Unir os conceitos de redes P2P e computação em nuvem cria um cenário muito interessante de pesquisa para novas aplicações. Busca-se neste casamento trazer o melhor dos dois mundos. De computação em nuvem, temos **(i)** a alta disponibilidade de informações, **(ii)** a elasticidade das aplicações, ou seja, a alocação ou liberação de recursos de forma automática, **(iii)** a garantia de serviços e **(iv)** o acesso ubíquo através da rede. De redes P2P, tem-se a descentralização e a possibilidade de serviços mais tolerantes a falhas.

3.2 Fundamentação teórica

Este projeto tem por objetivo explorar os conceitos apresentados anteriormente, principalmente os ligados a abordagens P2P, para compartilhar recursos de armazenamento que estejam ociosos e disponíveis pelos nós (*peers*) que compõem uma rede para formar uma solução de backup, sincronização e compartilhamento de dados por meio de uma plataforma de computação em nuvem. Esta solução se baseia ainda no paradigma orientado a serviço e tem por princípio os principais benefícios e práticas de reutilização de software. Doravante chamaremos de USTO.RE a plataforma confiável em computação em nuvem para a realização de backups, sincronização e compartilhamento de dados que seja eficiente, segura e totalmente distribuída.

3.2.1 Reutilização e Arquitetura Orientada a Serviços

A ideia de reutilização de software não é nova. Em 1968, durante a conferência da NATO de engenharia de software, Mcllroy [McIlroy, 1968], em seu revolucionário artigo, intitulado: "*Mass Produced Software Components*", apresentou a tese que dizia: "*a indústria de software é fracamente fundada e um aspecto dessa fraqueza é a ausência de uma indústria de componentes de software*". O autor propunha investigar técnicas de produção em massa de software, conforme algumas ideias da construção industrial.

Mcllroy idealizava, com a indústria de componentes, encontrar catálogos de rotinas padronizadas, classificadas por precisão, robustez, performance; aplicar rotinas existentes no catálogo para uma variedade de máquinas; e, ter confidência na qualidade das rotinas.

Durante cerca de quatro décadas, diversas pesquisas têm sido apresentadas e discutidas em conferências, como: *International Conference on Software Engineering* (ICSE), *International Conference on Software Reuse* (ICSR) e, em periódicos, na área de engenharia de software, tentando seguir as direções apontadas por Mcllroy.

Dentre as principais razões para a distância entre as ideias de Mcllroy e o atual estado da arte, está o atraso da indústria de software, comparado a indústria de hardware em termos de princípios de manufatura e catálogo de padrões; a mudança cultural dos desenvolvedores, que, normalmente, utilizam o verbo desenvolver, ao contrário do verbo reutilizar; o fato dos atuais sistemas de repositórios raramente considerarem processos de Engenharia de Domínio ou Linha de Produtos para o desenvolvimento dos artefatos, além de processos de garantia de qualidade antes de publicar os artefatos para reutilização.

Há muito vem se formando uma consciência na comunidade de engenharia de software de que, a fim de se obter produtos com alta qualidade e que sejam economicamente viáveis, torna-se extremamente necessário um conjunto sistemático de processos, técnicas e ferramentas. Nesse conjunto, a reutilização está entre as técnicas mais relevantes. Ao longo dos últimos anos, muitas técnicas têm sido propostas para favorecer o reuso. Dentre estas técnicas, destacam-se: a engenharia de domínio, frameworks, padrões, o Desenvolvimento Baseado em Componentes (DBC) e, atualmente, Linhas de Produto de Software que tem atraído o maior foco de pesquisa e desenvolvimento [Clements & Northrop, 2001].

Segundo Papazoglou et al. [Papazoglou et al., 2007], o paradigma de desenvolvimento orientado a serviços utiliza serviços para o suportar o desenvolvimento efetivo, com baixo custo e alta interoperabilidade de aplicações distribuídas. Serviços são entidades autônomas e independentes de plataforma que podem ser descritas, desenvolvidas, publicadas e descobertas. Os serviços realizam funções que podem variar de simples requisições, como serviços meteorológicos, até executar processos de negócios sofisticados, que requerem relacionamentos ponto a ponto entre camadas de consumidores e serviços [Erl, 2008]. Quando tratamos de serviços, o processo natural de criação dos serviços passa por uma fase inicial conhecida como modelagem de serviços.

Segundo Bell [Bell, 2008], a disciplina de modelagem é um campo de conhecimento que oferece boas práticas, padrões e procedimentos para facilitar as

atividades de desenvolvimento orientado a serviços durante o ciclo de vida de um serviço. Em geral, as atividades de modelagem orientada a serviços são realizadas antes da construção e implantação de serviços. As disciplinas de modelagem orientada a serviços identificam os processos chave nos quais o negócio e as pessoas envolvidas com o desenvolvimento devem estar engajados para produzir os artefatos de projeto e implementação.

Anos atrás, com o objetivo de analisar o mercado de engenharia de software baseada em componentes, o *Software Engineering Institute* (SEI) analisou a indústria de software, com ênfase em componentes. O estudo [Bass et al., 2000], conduzido durante o período de setembro de 1999 até fevereiro de 2000, examinou componentes de software sob a perspectiva técnica e de negócios.

A partir dessa pesquisa, um distinto conjunto de inibidores para adoção de componentes de software foi identificado. Com base nestes dados e de entrevistas realizadas, o SEI identificou os seguintes inibidores para a adoção de componentes, apresentados aqui em ordem decrescente de importância:

1. Carência de componentes disponíveis;
2. Carência de padrões estáveis para a tecnologia de componentes;
3. Carência de componentes certificados; e
4. Carência de um método efetivo para construção de sistemas baseados em componentes.

Com o crescimento do paradigma de desenvolvimento de software orientados a serviços, podemos perfeitamente fazer uma associação destes inibidores para o reuso de componentes, com a reutilização de serviços, uma vez que os próprios serviços podem ser vistos como componentes de software (ativos) reutilizáveis.

Vale ressaltar também, que em uma recente pesquisa envolvendo o estado da arte e direções de pesquisa na área de desenvolvimento orientado a serviços, Michael Papazoglou, principal investigador da área, em conjunto com outros pesquisadores, constataram que um dos grandes desafios está na definição de processos voltados para o desenvolvimento de software orientado a serviço [Papazoglou et al., 2007]. Assim, analisando as visões das duas comunidades, tanto de reuso de software e linhas de produto, quanto de desenvolvimento orientado a serviços, pode-se identificar a relevância do tema e a oportunidade para o projeto.

Quando tratamos do desenvolvimento de software, principalmente em relação à reutilização (seja por meio do DBC ou de linha de produtos de software) um ponto chave é a arquitetura de software. No caso do paradigma orientado a serviços, novos desafios são identificados por meio das arquiteturas de software orientadas a serviços.

Uma arquitetura orientada a serviço estabelece um modelo arquitetural que visa aumentar a eficiência, agilidade e produtividade das empresas com a utilização de serviços representando os seus processos de negócio. Como uma tecnologia de arquitetura de software, uma implementação de uma arquitetura orientada a serviço pode consistir da combinação de tecnologias, produtos e *APIs* [Erl, 2008]. Web Services [Stal, 2002], uma opção tecnológica para se implementar SOA, tem sido utilizado na indústria nos últimos anos com o objetivo de construir aplicações que reutilizam serviços disponíveis na rede [Stal, 2002]. Esta tecnologia permite disponibilizar funcionalidades através de serviços, utilizando padrões conhecidos da Web, tais como, SOAP, XML e HTTP, provendo uma maior interoperabilidade entre as aplicações.

Desta maneira, o desenvolvimento de aplicações distribuídas através de serviços se torna mais rápido, aumentando a qualidade e diminuindo o custo e tempo de entrega do produto.

3.2.2 Arquiteturas Peer-to-Peer (P2P)

Os grandes responsáveis pelo impulso e popularização dos sistemas distribuídos P2P, foram o Napster [Napster, 2012] e Gnutella [Gnutella, 2012]. Além de protagonizar inúmeras questões judiciais quanto a direitos autorais e pirataria, foi através destas ferramentas que se evidenciou o potencial da tecnologia no que diz respeito a compartilhamento de recursos sem desprender maiores investimentos em hardware. Desde a época dos precursores a tecnologia sofreu diversas mudanças.

Pesquisas mostram que os sistemas tendem a sofrer um processo de descentralização contínuo, onde o primeiro grande avanço se deu com o modelo cliente-servidor. Este consiste em uma máquina central no qual disponibiliza algum serviço que é consumido por máquinas com um hardware com bem menos recursos. A segunda forma de descentralização mostra o surgimento das aplicações P2P, onde podem ser desenvolvidas sobre sua forma completamente descentralizada, denominada de pura, ou um modelo híbrido, onde se desenvolvem estruturas de controle centralizadas e a utilização de recursos descentralizados. Cada um dos modelos de descentralização possui suas vantagens e desvantagens conforme se pode acompanhar na sessão seguinte.

3.2.3 Arquiteturas Puras

As redes denominadas puras possuem como principal característica a não existência de nenhum tipo de controle central. Todo o funcionamento se dá com uso de um algoritmo descentralizado onde é possível localizar peers e/ou serviços [Schollmeier & Rudiger, 2002].

Esta localização é feita fazendo uso da técnica de enchente (flooding) [Schollmeier & Rudiger, 2002], onde a mensagem é enviada a todos os computadores diretamente ligados ao emissor e cada máquina que recebe a mensagem faz o mesmo. Para evitar que a rede entre em colapso (loop), um tempo de vida é atribuído a mensagem para que caso esta não chegue a seu destino seja descartada. Os pontos negativos aqui são:

- Algumas máquinas podem não receber a mensagem, negando assim um serviço que estaria disponível em tese;
- Há um grande volume de mensagens enviadas até que a mesma encontre a máquina de destino.

A técnica mais utilizada na implementação de arquiteturas puras que gerou um avanço significativo na área é o *Distributed Hash Tables* (DHT) [Balakrishnan et al., 2003]. Utilizada nas seguintes ferramentas: *pStore* [Oliveira, 2007], *Pastiche* [Mattos, 2005] e *Oceanstore* [Kubiatowicz et al., 2000], *PeerStore* [Landers et al., 2004] e *BitTorrent* [BitTorrent, 2012]. As DHTs pertencem à classe de sistemas distribuídos descentralizados e oferecem recursos de localização similar às *hash tables* (chave, valor). Um par de chave e valor é armazenado na DHT e qualquer participante da rede pode acessar o valor desejado apenas informando a chave associada. As primeiras

quatro especificações de *DHTs* (*Pastry* [Rowstron & Druschel, 2001], *Chord* [Stoica et al., 2001], *CAN* [Ratnasamy et al., 2001] e *Tapestry* [Zhao et al., 2004]) surgiram quase simultaneamente no ano 2001, depois disso, sua utilização se popularizou em aplicações destinadas ao compartilhamento de arquivos na internet. Um estudo mais aprofundado sobre as diferentes implementações de DHT pode ser encontrado em [Balakrishnan et al., 2003]. As *DHTs* possuem como principais características:

- Descentralização: os próprios nós criam e mantêm o sistema, sem a necessidade de um servidor;
- Escalabilidade: o sistema suporta a participação de um crescente número de nós simultaneamente;
- Tolerância a erros: o sistema deve ser confiável, mesmo com nós entrando e saindo continuamente da rede.

Para alcançar os objetivos supracitados, as redes *DHTs* utilizam a técnica de que um nó na rede deve estar em contato direto com apenas uma fração de todos os nós participantes. Isso reduz o custo de manutenção quando um nó entra ou sai do sistema.

Para armazenar um arquivo numa *DHT*, primeiro se calcula uma chave (geralmente o código *hash SHA-1* [FIPS, 1995] do seu nome ou do seu conteúdo), em seguida esse arquivo é enviado para a rede até ser encontrado o conjunto de nós responsáveis por seu armazenado. Para recuperá-lo, uma mensagem é enviada informando a chave do arquivo desejado, essa mensagem é encaminhada até um nó que possua o conteúdo desejado e então o mesmo é enviado como resposta. A seguir descreveremos uma das implementações de *DHT* mais utilizadas, o *Chord* [Stoica et al., 2001].

3.2.4 Arquitetura Chord

A implementação de *DHT* utilizando *Chord* se destaca pela sua simplicidade em oferecer uma única operação em que dada uma determinada chave, ela será mapeada para um nó na rede. Segundo Stoica [Stoica et al., 2001], sua arquitetura foi projetada para se adaptar facilmente a entrada e saída de novos *peers* na rede. Embora seja uma implementação que possui apenas uma tarefa (associar chaves a nós da rede), o *Chord* possibilita ao desenvolvimento de aplicações p2p uma série de benefícios:

- Balanceamento de carga, as chaves são distribuídas igualmente entre os nós da rede;
- Descentralização, nenhum nó é considerado mais importante que outro;
- Escalabilidade, o uso do *Chord* é viável mesmo com uma grande quantidade de nós;
- Disponibilidade, ajuste de sistema automático a entrada e saída de novos nós, fazendo que um nó sempre esteja visível na rede; e,
- Flexibilidade, não é necessário seguir nenhuma regra para o nome das chaves.

Para atribuir chaves aos nós, o *Chord* usa uma variação de *consistent hashing* [Karger et al., 1997] que cuida do balanceamento de carga, uma vez que cada nó tende a receber naturalmente um mesmo número de chaves. Em trabalhos anteriores ao *Chord* usando *consistent hashing* se assumia que cada nó possuísse conhecimento de todos os

outros, diferentemente, no *Chord* cada nó precisa ter conhecimento de apenas uma fração dos outros nós na rede.

Supondo uma rede de n nós, um *peer* mantém informações sobre *O (log n)* nós, e para encontrar um determinado nó na rede basta que ele possua apenas uma referência válida.

3.2.5 Arquiteturas Híbridas

Em alguns sistemas P2P é necessária a identificação dos peers conectados na rede. Para tal, sistemas como o *OurBackup* [Oliveira, 2007], que fazem uso de redes sociais para backup, é utilizado em sua arquitetura um servidor responsável pela autenticação dos usuários, manutenção da rede e dos *metadados* onde as cópias estão armazenadas. Pode-se ressaltar que a utilização de servidores não é obrigatória para a localização dos *peers* e dos *metadados*, podendo essa ser feita utilizando as DHT mencionadas anteriormente.

Nesses sistemas, o papel do servidor está em oferecer uma interface aos *peers* da rede com diversas operações tais como: autenticação do usuário; manipulação dos dados armazenados por outros *peers*, adicionar, remover, excluir, atualizar; manipulação dos usuários cadastrados no sistema; localização dos *peers* e relacionamento entre eles; dentre outras tarefas que venham a atender os requisitos do sistema. Em todos os casos geralmente é verificado se o cliente possui permissão para executar tais operações.

Essa centralização de informações pode trazer prejuízos de escalonamento no sentido de que sempre vai existir uma exigência maior do servidor à medida que se aumenta o número de requisições, usuários, *metadados* ou quaisquer outras informações delegadas ao serviço. Porém, sistemas como o *eDonkey* [Aidouni et al., 2009] se mostraram bastante eficientes quanto ao gerenciamento centralizado de informações dessa natureza. Se necessário esse escalonamento também pode ser resolvido com a utilização de clusters ou grids no lado do servidor, fazendo-se mais importante a disponibilização da interface de comunicação com a máquina cliente.

3.2.6 Detalhamento de ferramentas existentes

Em um estudo conduzido pela equipe deste projeto de pesquisa, foi constatado que existem diversas ferramentas implementadas e funcionais, ou seja, que fazem backup, sincronização, compartilhamento e recuperação em um ambiente P2P. Entretanto, foi constatado que existem apenas mecanismos que aumentam a disponibilidade. Em nenhuma das ferramentas investigadas foi identificado um o mecanismo que quantifique e garanta, em termos percentuais, a disponibilidade da recuperação. As alternativas que possuíam essa características só atendiam a esta necessidade por meio da adição de um componente extra (i.e. ferramentas, suítes ou ambientes).

Vignatti [Vignatti et al., 2009] faz um levantamento das soluções existentes e propõe um mecanismo de armazenamento digital a longo prazo baseado na escolha de repositórios confiáveis. Em essência a contribuição do trabalho é a preservação do dado, ou seja, que o mesmo não se corrompa durante sua transmissão, armazenamento, falha de hardware, entre outros. Segundo o autor muito pouca ênfase foi dada à recuperação dos itens.

O *BackupIT* [Loest et al., 2009] é uma ferramenta completa e funcional que faz uso de um mecanismo denominado de *Byzantine Quorum Systems* [Malkhi & Reiter, 1997] para controlar integridade e disponibilidade. No entanto, apenas há um aumento das chances de uma recuperação bem sucedida.

Colaço et al. [Colaço et al., 2008] especificou e desenvolveu um mecanismo de priorização dos arquivos mais usados por um usuário para diminuir o tempo que o sistema fica indisponível para o mesmo. A ideia central é que os arquivos mais importantes sempre estejam com uma disponibilidade maior que os demais.

O *pStore* [Batten et al., 2002] combina um algoritmo de roteamento chamado *Distributed Hash Table* (*DHT*) com quebra de arquivos em tamanhos fixos e uma técnica de controle de versão. A ideia é bastante completa, há suporte para encriptação de arquivos mantendo assim a privacidade nas máquinas clientes, bem como manter versões de arquivos já salvos, reaproveitando os pedaços de arquivos que se repetem em versões anteriores.

Samsara estimula a troca de recurso de maneira igualitária sem uso de uma terceira entidade no sistema para intermediar a comunicação [Cox & Noble, 2003]. O mecanismo utilizado para garantir a troca justa de recursos é um esquema punitivo para o *peer* da rede que por ventura venha a perder dados, atividade esta que é típica de usuários que querem utilizar e não disponibilizar recursos.

CleverSafe Dispersed Storange [Cleversafe, 2012] é uma aplicação *open source* desenvolvida para a plataforma *CentOS*. Faz uso do algoritmo de dispersão (*IDA*) e recuperação da informação injetada na rede. A ferramenta garante maior escalabilidade por não ter servidores centralizados e uma garantia de disponibilidade de acesso aos dados de 99,9999 (12 noves) por cento. Porém, a ferramenta faz uso de máquinas em ambientes controlados.

Dibs [Martinian, 2012] é uma ferramenta *open source* programada em *Python* que faz *backup* incremental, ou seja, de maneira inteligente o arquivo não é salvo novamente, apenas as modificações feitas no mesmo. Sua interação acontece mediante fechamento de contrato entre as partes, onde é comparado espaço mínimo disponível em disco.

Resilia [Meira, 2005] é um protótipo de sistema de backup que combina p2p com compartilhamento secreto e seguro de arquivos, faz uso de um mecanismo de replicação para aumentar a disponibilidade e permite a reconstrução de backups perdidos por falhas de nodos que compõe a rede.

Dropbox [Dropbox, 2012] é uma ferramenta de armazenamento de arquivos no qual faz uso da ideia de computação na nuvem, mantém um conjunto de servidores ligados em rede, com ambiente controlado. O salvamento dos arquivos dos usuários é feito por intermédio de um software instalado na máquina do cliente. Assim como o *CleverSafe*, o *Dropbox* também possui toda sua infraestrutura em ambiente controlado.

Disco virtual RNP [GTAR, 2012] é uma ferramenta Web que utiliza de servidores conectados montando um *Grid* de dados. O salvamento dos arquivos dos usuários é feito por intermédio de um software instalado na máquina do cliente ou Web.

Conforme pode ser constatado nesta sessão, diversos trabalhos vêm sendo desenvolvidos na área de salvamento e backup de dados distribuídos, A tabela abaixo

contextualiza de maneira mais detalhada os trabalhos levantados no estado da arte, onde foram tabuladas as principais características dos sistemas implementados para esse tipo de problema.

1. Garantia de disponibilidade: um mecanismo que quantifique e forneça estatisticamente que a restauração vai estar disponível em determinado horário.
2. Aumento de disponibilidade: mecanismo de redundância no qual aumente as chances de uma restauração bem sucedida.
3. Segurança: mecanismo de segurança dos dados salvos na rede de backup e compartilhamento.

   Inspeção dos dados: mecanismo de checagem que garante que os dados copiados em um *peer* não foram perdidos.
4. *Backup* Incremental: mecanismo no qual a similaridade entre diferentes versões é usada para salvar recursos de rede.
5. Ponto Central de controle: se o sistema teve sua arquitetura projetada para ser um sistema P2P híbrido.
6. Controle de espaço de armazenagem: Mecanismo no qual controla o espaço utilizado de *backup* e compartilhamento por usuário, evitando que o usuário utilize mais do que necessário e doe seus recursos.
7. Baseado em redes sociais: Método usado em busca de atingir maior índice de confiabilidade na rede. A proposta baseia-se em formar uma rede social de amigos (*friend-to-friend* [Li & Dabek, 2006]) e com isso evitar que seus arquivos sejam apagados.
8. Sistema de Armazenamento: Se o referido sistema tem em seu conceito ser utilizado como armazenamento ou backup.

3.2.7 Demais trabalhos relacionados

Além das ferramentas já apresentadas, várias são as soluções existentes para armazenamento de dados em nuvem no modelo de *DaaS*. Em geral, tais soluções são proprietárias e fechadas, o que dificulta uma análise técnica mais minuciosa sobre cada proposta. Analisando algumas das principais soluções existentes, como a *Amazon S3*, o *Megastore*, o *MSFSS* e o *Hadoop Distributed File System* (HDFS), nota-se que invariavelmente, todas utilizam uma estratégia comum baseada na replicação dos dados em diversos servidores, onde o número de servidores envolvidos na replicação de dados, varia de 3 a 7 vezes dependendo da solução. Esta replicação forçada é feita porque não há automação nos processos de replicação que possa garantir 99,9999999\% de disponibilidade [DeCandia et al., 2007].

O *Amazon S3* (*Simple Storage Service*) [Amazon, 2012] é o sistema de armazenamento por trás de muitos dos serviços da Amazon.com como o *Dynamo* [DeCandia et al., 2007], que recentemente teve seu banco de dados No-SQL lançado com o nome de *DynamoDB*[36]. O *DynamoDB* possui uma política de *backup* onde são feitas no mínimo três cópias do mesmo dado, onde duas são feitas na mesma zona e

[36] URL: http://bit.ly/xBc8mD, Acessado em 05/07/2012

uma terceira em uma zona externa, de modo a aumentar a disponibilidade da informação. Segundo dados de 2009, o sistema armazenava cerca de 40 bilhões de arquivos de 400.000 clientes. Seus desafios incluíam garantia de disponibilidade e o gerenciamento de falhas.

Outra plataforma, o *Megastore*, é um sistema desenvolvido com foco nos serviços *online* interativos [Baker et al., 2011]. Ele foi desenvolvido e é usado pela Google há muito tempo e manipula mais de 3 bilhões de escritas e 20 bilhões de transações de leitura diariamente, além de também armazenar um valor próximo a um *petabytes* de dados primários nos seus *datacenters* espalhados globalmente.

Já o *MSFSS* é um sistema de arquivos distribuído de alta escalabilidade e flexível, desenvolvido para armazenar uma grande quantidade de pequenos arquivos [Yu et al., 2007]. A sua arquitetura é dividida em três componentes principais: *single master*, *storage nodes* e os *metadata servers*. Todo arquivo armazenado no sistema recebe um identificador de 128 bits (*FID - File ID*) gerado a partir da data de criação do arquivo. Esses arquivos são armazenados no sistema de arquivos local de cada *storage node*. Dentre seus requisitos, o *MSFSS* dá suporte à replicação de dados, assegura a consistência entre as réplicas e executa rápida sincronização para identificar réplicas obsoletas. O *FSI* (*File System Interface*), biblioteca que dá acesso ao sistema para os clientes externos, aloca uma grande quantidade de *FIDs* em processos *batch* e os deixa armazenados em memória local, para uso posterior [Yu et al., 2007]. Por padrão, o *MSFSS* armazena duas réplicas por arquivo, mas esse valor pode ser configurado para garantir maior disponibilidade e confiabilidade. Ele usa qualquer uma das réplicas no processo de leitura. Já no processo de escrita, todas as réplicas devem ser atualizadas automaticamente. Para minimizar a latência, os dados são armazenados próximo ao usuário e as réplicas próximas umas das outras. Ele separa os grupos por regiões e cria 3 ou 5 réplicas para os *datacenters*.

O *HDFS* é um sistema de arquivos utilizado pelo *Hadoop* [Shvachko et al., 2010] e seus projetos relacionados. *Hadoop* é um *framework* para análise e transformação de grande quantidade de dados usando *MapReduce* [Dean & Ghemawat, 2008]. Uma das principais características do *Hadoop* é o particionamento de dados e a computação dos mesmos utilizando milhares de *hosts*. O *cluster* do *Hadoop* no *Yahoo!*, por exemplo, alcançou 25.000 servidores (com *cluster* de até 3.500 servidores) e armazenavam 25 *Petabytes* de dados [Shvachko et al., 2010].

Os trabalhos aqui apresentados apresentam como característica comum a replicação excessiva de informação. Este fato reside na necessidade de se garantir uma alta disponibilidade dos dados, aumentar a confiabilidade e desempenho da recuperação de informações. Porém, o uso de réplicas não só traz benefícios. Elas criam um tráfego extra de dados na rede, tráfego este que pode ser tão excessivo ao ponto de se tornar o principal gargalo de uma aplicação, além de gerar um custo da banda tão alto que pode tornar a aplicação inviável [Yang et al., 2006].

Nestas soluções, infere-se que as mesmas exigem a aquisição de infraestrutura dedicada para prover o serviço e para garantir a réplica dos dados. De forma a melhorar este cenário, o objetivo do usto.re é permitir a criação de uma nuvem para armazenamento de dados utilizando tecnologia P2P que se baseia na disponibilidade de cada *peer* para, dinamicamente, criar federações e definir a quantidade de replicações de

cada *chunk*[37] de cada arquivo. Esta abordagem permite que em ambientes onde os *peers* tenham maior disponibilidade (taxa menor de falhas), se tenha uma replicação menor e, no caso de um ambiente mais dinâmico como a Intranet de uma empresa, se tenha uma replicação maior. A partir da próxima seção, este artigo detalha nossa proposta.

Tabela 3.1 - Tabela comparativa entre as soluções de backup/armazenamento p2p

	pStore	Pastiche	OurBackup	OceanStore	PAST	BackupIT	Samsara	Resilia	Clever Safe	Dibs	Drop box	Disco RNP
Garantia de disponibilidade												
Aumento de disponibilidade	X	X	X	X	X	X	X	X	X	X	X	X
Segurança	X	X	X	X	X	X	X	X	X	X	X	X
Inspeção dos dados				X		X	X	X	X			
Incremental	X	X	X				X	X		X	X	
Ponto Central de controle			X									X
Controle de espaço de armazenagem		X	X	X	X		X		X		X	X
Baseado em redes sociais			X			X						
Sistema de Armazenamento				X	X				X		X	X

3.3 USTO.RE: Um Sistema P2P confiável para Data Cloud

Como vemos, o principal desafio em se implementar um sistema que realize o armazenamento e backup remoto é garantir que o os dados estarão disponíveis, pois em todas as soluções analisadas existe a possibilidade de falhas acontecerem e os servidores estarem indisponíveis, tornando a replicação e a montagem de infraestruturas redundantes e consideravelmente custosas como a única solução. Neste caso, se

[37] *Chunk* é um fragmento de informação, ou seja, uma pequena "parte" de um arquivo armazenado. Arquivos são divididos em *chunks*, que por sua vez, são replicados em pontos de armazenamento na infraestrutura da nuvem.

utilizarmos os conceitos trazidos por sistemas P2P e computação em nuvem podemos implementar um sistema de backup menos custoso desde que consigamos resolver o problema dos dados estarem disponíveis quando os usuários solicitarem.

Sendo assim, uma abordagem que permite solucionar este problema é replicar os dados em outros *peers* de forma a utilizar seus recursos ociosos sem haver a necessidade de se adquirir novos recursos.

Consequentemente temos que definir a quantidade de *peers* em que devemos replicar os dados de forma que os mesmos estejam disponíveis quando for necessário realizar a sua recuperação. Se conseguirmos prever a probabilidade de falha/indisponibilidade de um *peer* na rede poderemos calcular a quantidade utilizada de *peers* para a replicação de dados. A formalização matemática desta solução está descrita abaixo:

$$f(t)=\lambda e^{-\lambda t}$$

Equação 1: Equação exponencial

$$\lambda = \frac{1}{MTTF}$$

Equação 2: Taxa de falhas

Onde λ é a taxa de falhas e MTTF é o tempo médio entre falhas. A probabilidade de falha para função exponencial no intervalo de 0 a t é dado pela equação F(t):

$$F(t) = \int_0^t \lambda e^{-\lambda t}\, dt = 1 - e^{-\lambda t}$$

Equação 3: Probabilidade de Falha

A **confiabilidade**, ou seja, probabilidade do sistema não falhar num instante de tempo t pode ser obtido pela função R(t), fazendo o complemento da probabilidade de falha:

$$R(t) = 1 - F(t)$$

Equação 4: Formulação geral da confiabilidade

Substituindo a equação 4 em 5, se tem:

$$R(t) = 1 - \left(1 - e^{-\lambda t}\right) = e^{-\lambda t} \quad (1.5)$$

Equação 5: Confiabilidade no modelo exponencial

Uma vez que se têm as taxas de falhas contabilizadas nos módulos, *Avaibility* e *SeftAvaibility* pode-se, a partir da equação 6, saber a probabilidade de um *Peer* não falhar em determinada hora. É sabido ainda que várias máquinas podem estar ligadas e que há um número mínimo de máquinas configurável no qual um pedaço tem que ser distribuído. Para se obter um resultado geral do algoritmo, é necessário fazer as somas das probabilidades. Meyer [Meyer, 1995] define a soma das probabilidades como sendo:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Equação 6: Soma das probabilidades

Por indução matemática em 7, se chega à equação geral de soma de probabilidade:

$$P(A_1 \cup A_2 \cup \ldots \cup A_K) = \sum_{i=0}^{K} P(Ai) - \sum_{i<j=2}^{k} P(Ai \cap Aj) + \sum_{i<j<r=3}^{k} P(Ai \cap Aj \cap Ar) + \cdots + (-1)^{k-1} P(A_1 \cap A_2 \cap \ldots \cap A_K)$$

Aplicando em conjunto as equações 6 e 7 o algoritmo consegue calcular a probabilidade de um conjunto de máquina não falhar em determinado horário e com base neste calculo definir com um percentual de confiança de que a mesma esteja ligada e funcionando.

Uma vez que já foram abordadas as entradas, saídas e a fundamentação matemática do algoritmo, na sessão seguinte será apresentado o processamento de toda a informação. Foi optado por deixar o código da forma como foi implementado por ter sido usado assim na prova de conceito.

3.3.1 Arquitetura do sistema

A arquitetura do usto.re foi especificada tendo como meta atingir um conjunto de atributos de qualidade peculiares aos sistemas de armazenamento distribuído e que atendesse aos principais benefícios oferecidos pelas aplicações P2P, a saber:

- **Escalabilidade**: atendendo a possibilidade de explorar recursos de hardware de um grande número de máquinas (*hosts*) conectados à rede, principalmente por meio do uso racional de recursos ociosos em grandes corporações.
- **Otimização** de iterações (troca de mensagens): a "distância" entre os *peers* que interagem no sistema tem impacto no desempenho geral na latência das interações individuais, desta forma a carga do tráfego da rede também sofre um impacto negativo por esta latência. Neste contexto, a escolha pela estratégia do uso de federações visa agrupar os *peers* relacionados e reduzir esta latência.
- **Disponibilidade**: sistemas P2P são baseados em computadores livres para se "unir" ao sistema a qualquer momento, bem como "sair" dele. Além disso, as conexões não são gerenciadas pelo sistema ou alguma autoridade que garanta a conectividade e a qualidade do serviço, neste contexto, uma estratégia de garantia de disponibilidade do serviço deve ser implementada considerando as características básicas de um sistema de armazenamento com alta disponibilidade, bem como as restrições que uma rede P2P impõe.
- **Segurança dos dados**: em se tratando de armazenamento de dados, políticas de segurança e proteção devem ser efetivamente adotadas de modo a garantir a privacidade e autenticidade dos usuários e dados do sistema.

Para detalhar o funcionamento do usto.re, esta seção apresenta uma visão geral da arquitetura do USTO.RE, seus os principais componentes, dependências e relacionamentos, conforme ilustrado na Figura 2. O USTO.RE é composto por um conjunto de cinco componentes, dispostos em uma arquitetura P2P híbrida estruturada e em três camadas. São eles: **(i)** *Super Peers Rendezou Relay* ou simplesmente *Super Peers*, **(ii)** Servidores, **(iii)** *Proxies*, **(iv)** Bancos de Dados Relacionais e Não-SQL e **(v)** *Simple Peers*. Estes componentes possuem funcionalidades distintas e interagem como um sistema distribuído descentralizado híbrido, de modo semelhante a uma rede P2P, onde cada nó realiza tanto funções de servidor quanto de cliente. Os componentes agrupam-se dinamicamente como federações de dados, onde os grupos são montados de modo a minimizar a troca de mensagens no sistema.

A Figura 3.2 apresenta o diagrama de sequencia do USTO.RE, onde vemos de maneira geral o processo de registro de *peer*, localização de serviços e como um *simple peer* obtém os dados.

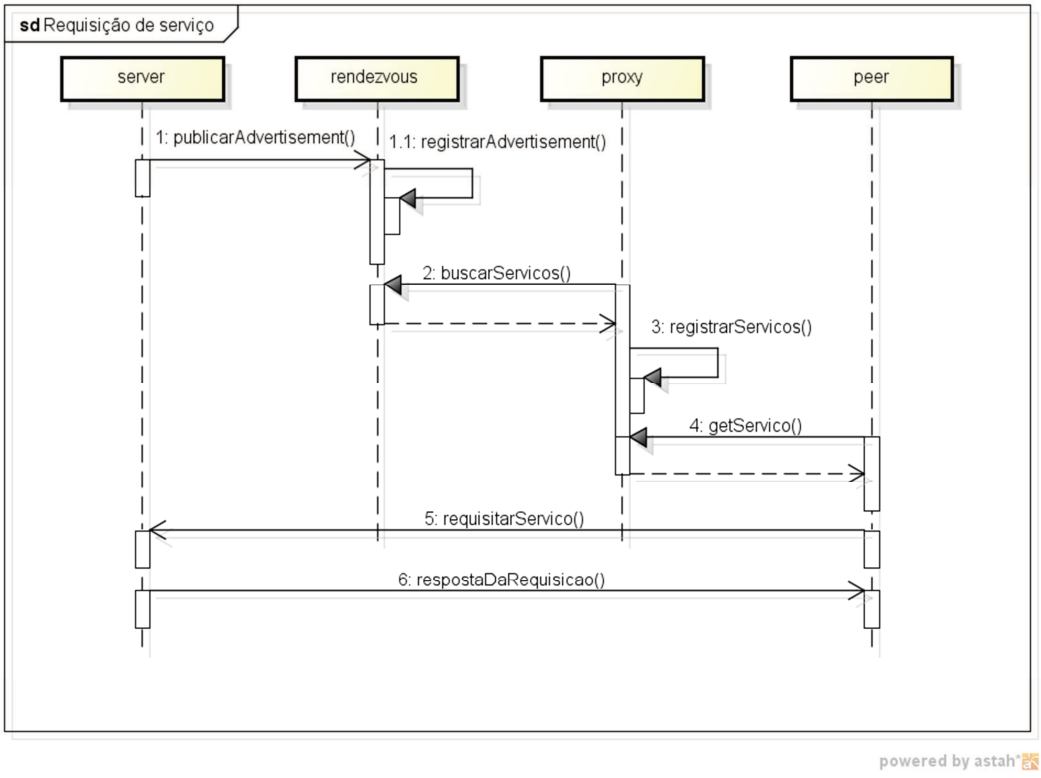


Figura 3.2 - Diagrama de sequencia USTO.RE

Na **Erro! Fonte de referência não encontrada.**.3 é apresentado este mesmo processo porém com a visão de processos isolados rodando em servidores com papeis específicos.

A organização do sistema nesta arquitetura multicamadas possibilita a distribuição do processamento, uma vez que os componentes estão fisicamente distribuídos. Entretanto, por se tratar de uma arquitetura híbrida P2P estruturada e multicamadas, o sistema possui uma distribuição dita horizontal. Nesta distribuição horizontal, em uma rede P2P, um cliente ou um servidor podem estar fisicamente divididos em partes logicamente equivalentes, onde cada um opera sobre a sua própria porção dos dados, o que propicia um balanceamento da carga.

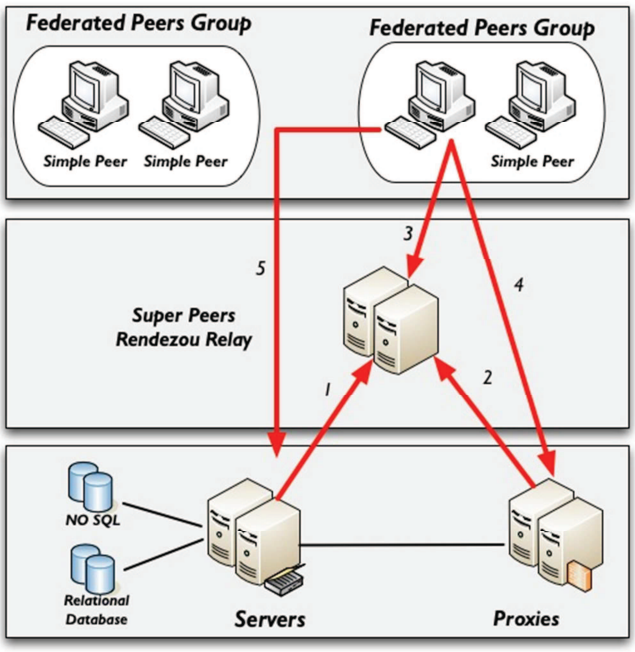


Figura 3.3 - Arquitetura do USTO.RE

Na Figura 3.4, é apresentado em detalhes cada um dos componentes de arquitetura do USTO.RE, separando cada um deles entre interfaces, bibliotecas compartilhadas, componentes da rede P2P e interfaces de consumo de serviços. Nas subseções a seguir, maiores detalhes sobre cada componente são fornecidos e para os componentes principais será acrescido o detalhamento da arquitetura.

3.3.2 Super peers

Os *super peers* funcionam como elementos de referência para os demais componentes da arquitetura, sendo a porta de entrada para a participação de servidores, *proxies* e *simples peers* no sistema. O papel do *super peer* é definir as federações de dados quando cada *peer* solicita conexão à rede. Para isso, os *super peers* devem ter sua localização previamente conhecida por todos os demais *peers* por meio de uma pré-configuração. Consequentemente, eles são os primeiros componentes a serem inicializados para o correto funcionamento do usto.re. Também como consequência, um *super peer* guarda informações sobre todos os servidores, *proxies* e *simple peers*, agrupando-os dinamicamente de acordo com o perfil de cada *peer*, a ser explicado adiante.

Também é papel deste tipo de *peer*, escolher dinamicamente os *peers* e servidores das federações baseando-se em um algoritmo de proximidade [Duarte, 2010]. O agrupamento em federações permite o crescimento elástico e garante a escalabilidade do sistema, pois não existe um limite para a quantidade de federações que podem ser criadas. Crescimento elástico é a característica do sistema de crescer ou diminuir, em termos de capacidade e consumo de recursos, de maneira dinâmica e não intrusiva. Os *peers* se comunicam com a rede P2P através do protocolo *JXTA* [JXTA, 2012] e opcionalmente podem ofertar uma interface de serviço *REST* [Webber et al., 2010] para permitir a interoperabilidade com outras aplicações.


pkg
p2p_client
app
AppClient
directoryListener
utils
gui
components
AppGUI
models
dragandrop
DragAndDropGUI
renderes
P2PTray
service
chunk
dao
peer
manageFile
AbstractAgent
SimpleP2pController
messageController
BackupItem
RestoreItem
p2p_lib
service
jxta
ServiceConsumer
JxtaNode
ServiceProvider
P2pDiscoveryThread
server
ServiceConfig
util
fileutil
p2p_server
app
AppServer
SuperPeer
RMIListener
restservices
AuthenticationService
FileManagerService
UserManagerService
service
ServiceLocator
dao
bean
CassandraSingleton
PersistenceServe
server
AccessController
access
dao
CassandraAccessDAO
filetracker
FileTrackerController
AbstractController
dao
CassandraFileTrackerDAC
security
SecurityEngine
p2p_proxy
app
AppProxy
proxy
ProxyService
utils

Figura 3.4 - Visão geral da arquitetura do sistema

3.3.3 Servidores

Os *peers* servidores são aqueles que oferecem um conjunto (ou subconjunto) de uma lista existente de serviços. Na ordem de configuração do usto.re, os servidores são os componentes que devem ser executados logo após a inicialização dos *super peers* (Passo 1 na Figura 3.2). Os *Super peers* estabelecem um esquema de sincronização fazendo com que a lista de servidores em cada um deles seja atualizada quando da entrada ou saída de um *peer* servidor.

A definição de *peers* com funcionalidades especificas na rede opõem-se à algumas propostas para sistemas P2P, onde cada *peer* deveria ser capaz de desempenhar todos os papéis no sistema, promovendo a ideia de uma *DHT* (*Distributed Hash Table*) completa [Oliveira, 2007; Loest et al., 2009]. No entanto, a implementação utilizando uma DHT em sua essência é bastante custosa e de difícil escalabilidade. Sendo assim, a proposta adotada no usto.re é a criação de níveis hierárquicos que implementam serviços bem definidos e que podem crescer horizontalmente. Dentre os serviços disponibilizados pelos servidores, pode-se citar:

- ***Autenticação***: usado para que cada *peer* se autentique;
- ***Disponibilidade***: permite checar a disponibilidade de cada *peer*;
- ***Chunk***: usado para o controle de *chunks*;
- ***Erro***: permite o controle de erros como, por exemplo, um serviço indisponível;
- ***Controle de Saída***: controla a saída voluntária de um *peer*, quando este desconecta-se voluntariamente da rede;
- ***Gerência de Diretórios***: utilizado para armazenamento e recuperação de diretórios inteiros;
- ***Gerencia de Arquivos***: utilizado para armazenamento e recuperação de arquivos;
- ***Busca***: procura por um conjunto de *peers* que obedeçam ao acordo de nível de serviço (do inglês, *Service-Level Agreement - SLA*), no caso do usto.re fortemente relacionado à disponibilidade do *peer*, para o salvamento de um arquivo;
- ***Árvore de Diretórios***: utilizado para visualização de diretórios inteiros;
- ***Segurança de Acesso***: controla a permissão de acesso aos *chunks*;
- ***Rastreabilidade***: mantém uma lista de usuários e arquivos a ser consultada quando a recuperação de um arquivo é solicitada.

Como se pode observar na Figura 2, os *peers* servidores utilizam dois tipos de banco de dados para manter a consistência do sistema. Um banco de dados tradicional relacional contém dados dos usuários do sistema, e um banco de dados No-SQL [Chang et al., 2006] que permite um crescimento horizontal e a recuperação mais rápida das informações relacionadas aos arquivos e *chunks* salvos. A escolha desta separação se dá pelas questões relacionadas ao desempenho do sistema, visto que, com o aumento do

volume de arquivos e *chunks* salvos, o sistema de gerenciamento de banco de dados tende tornar-se um ponto de gargalo. A utilização de um sistema nativamente distribuído torna a solução viável e escalável [Chang et al., 2006].

Todas as informações referentes à autenticação e SLA dos *peers* são salvas em um banco de dados relacional devido à a garantia da integridade provida por este tipo de banco. Já o serviço do *File Tracker*, que permite a identificação de qual *peer* possui partes do arquivo a ser recuperado, é utilizado um banco de dados No-SQL, o que permite o seu crescimento horizontal. As instâncias dos bancos, quer seja este relacional ou Não-SQL, podem ser compartilhadas entre mais de um servidor.

Um *peer* servidor pode prover um ou mais serviços da rede, sendo assim, da mesma forma que na criação de federações de dados, pode-se iniciar *peers* servidores e *proxies* sobre demanda aumentando a escalabilidade e elasticidade do sistema. No atual estágio do projeto, este aumento deve ser feito manualmente por um humano, de acordo com o monitoramento do desempenho do sistema.

O relacionamento entre cada componente da solução descrita acima com suas interfaces e conectores está apresentada na Figura 3.5 abaixo. Nela se pode observar que um dos elementos centrais do sistema são as filas de envio e recepção de mensagens. No USTO.RE toda troca de mensagem passa por uma fila que é consumida sobre demanda e de acordo com os recursos disponíveis no sistema operacional. A quantidade de filas pode ser configurada, sendo assim, o USTO.RE pode ser utilizado em equipamentos com características diferentes, garantindo que as operações sempre ocorrerão. Quando se deseja mais performance aumenta-se a quantidade de filas desde que haja recursos disponíveis no sistema operacional.

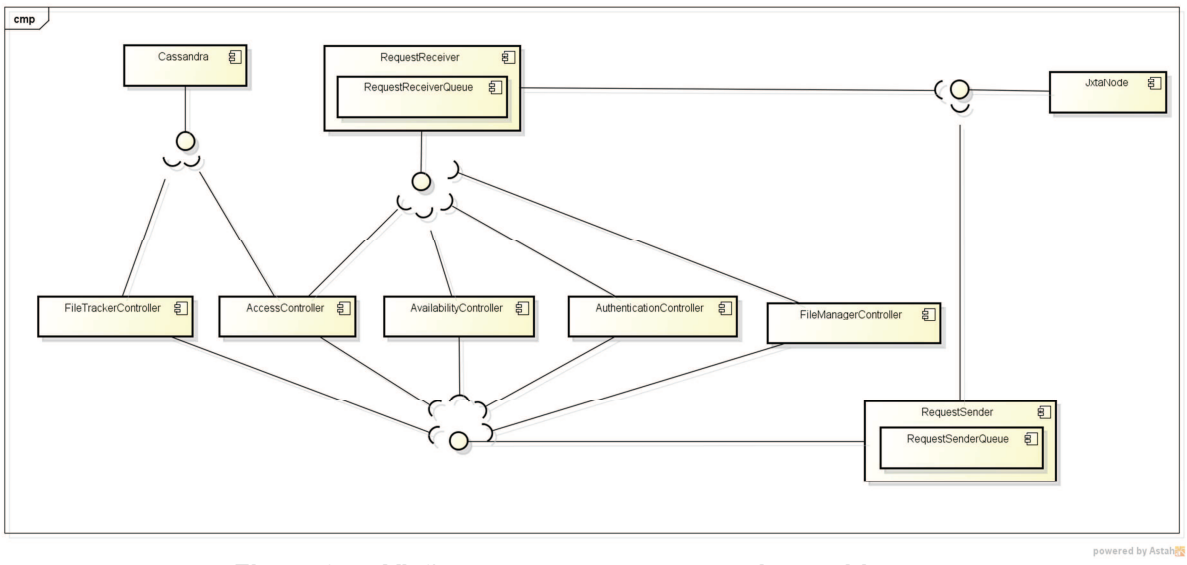


Figura 3.5 - Visão componente-conector do servidor

3.3.4 Proxies

Após a inicialização de *super peers* e servidores, o terceiro componente que precisa ser executado é o *proxy*. Cada *proxy* atua como um catálogo, um serviço de localização para serviços em execução nos diferentes servidores do usto.re. Um *proxy* ao se anunciar a um *super peer* (Passo 2 da Figura 3.2), recebe a lista de servidores

registrados. Além disso, um proxy obtém a informação de quais serviços estão disponíveis em cada servidor. Desta forma, quando um *peer* deseja a informação sobre um determinado serviço, um *proxy* pode fornecer a referência para um servidor que atenda tal requisição. Desta forma, um *proxy* estabelece uma ponte de ligação entre consumidores de serviços, tipicamente os *simple peers*, e o provedor de um serviço, no caso, os *peers* servidores.

3.3.5 Simple peers

Simple peers são aqueles responsáveis por armazenarem os *chunks* dos arquivos. Na visão alto nível da proposta, estes *peers* representam máquinas de usuários comuns que podem oferecer espaço de armazenamento para ser compartilhado entre vários usuários. Cada *simple peer* possui um perfil que define a sua disponibilidade e que lhe é atribuído quando de sua conexão com o sistema. Esta disponibilidade é relacionada ao período de tempo em que o *peer* esteja disponível para compartilhar dados. Como exemplo, um *peer* que esteja em uma *Intranet* de uma empresa pode ter em seu perfil, a disponibilidade atribuída como "***8:00 às 12:00 e 14:00 às 18:00***". Quando um *simple peer* se conecta ao sistema, ele recebe de um *super peer* a lista de *proxies* disponíveis (Passo 3 da Figura 3.2). Desta lista, o proxy busca por um serviço específico e obtém a referência de quais servidores ofertam determinado serviço (Passo 4 da Figura 3.2). Um servidor aleatório é escolhido, e o *simple peer* solicita o serviço desejado (Passo 5 da Figura 3.2). Caso um serviço não seja atendido por algum motivo, como um *timeout*, o *proxy* pode fornecer um novo servidor para o *simple peer*.

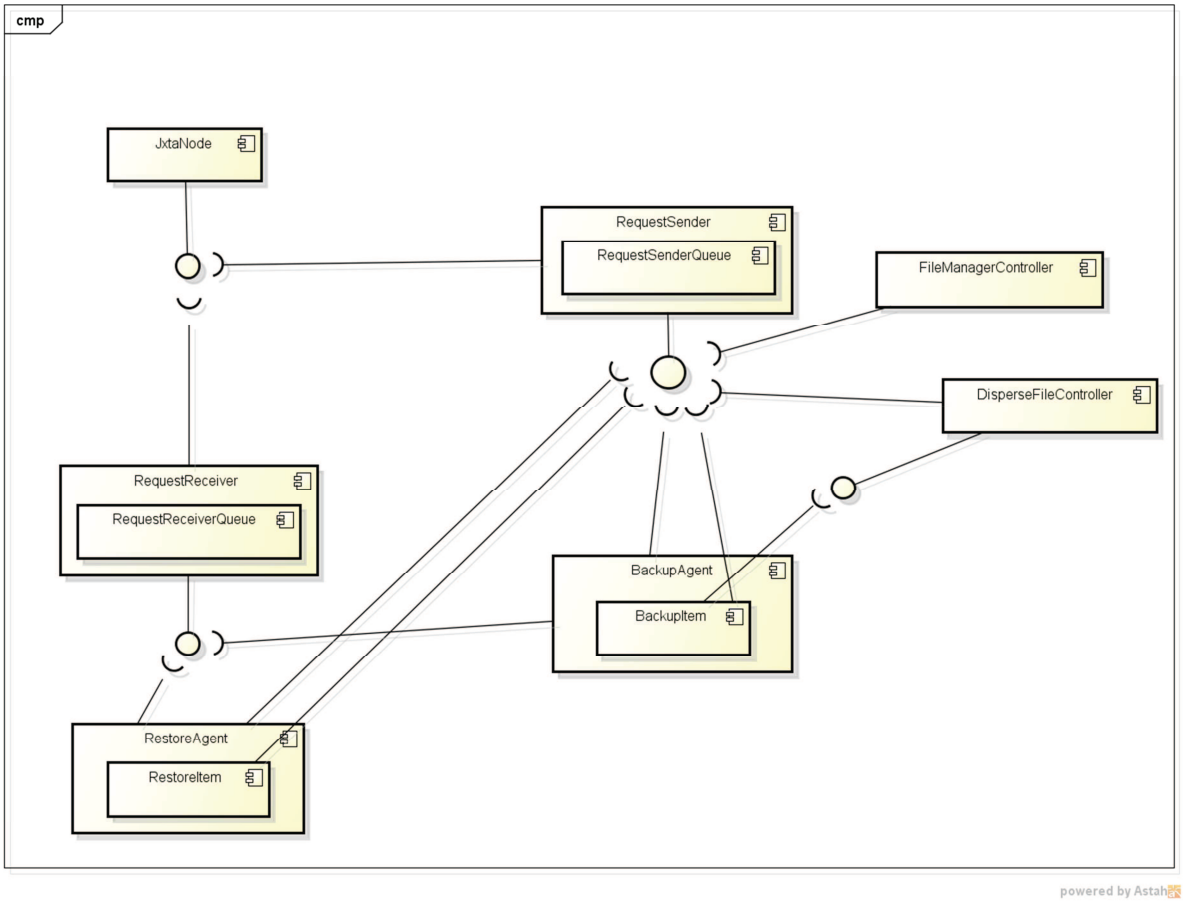


Figura 3.6 - Visão componente-conector do *simple peer*

Observando-se Figura 3.6, também se vê como elemento central do sistema o gerenciador de filas que neste caso também pode ser configurado trazendo os mesmos benefícios descritos para o servidor. Ainda como elementos centrais do sistema temos o *BackupAgent* e o *RestoreAgent*. Cada componente deste representa uma ação recuperação (*restore*) ou *backup* (salvamento). Quando uma operação é solicitada uma instancia deste componente é criada e o componente é responsável por garantir que a operação solicitada seja executada. Para evitar que haja consumo elevado de memória ou recursos do sistema, apenas duas instancias de *BeckupAgent* e *RestoreAgent* pode ser executada por vez em um *simple peer.*

Como citado anteriormente, *simple peer* são agrupados em federações de dados pelos *super peers*. O objetivo de agrupá-los desta forma é minimizar a sobrecarga na rede e em cada *peer*, além de diminuir a quantidade de mensagens trocadas e permitir que uma federação desempenhe o papel de *backup* de outra federação. O agrupamento dos *peers* em federações obedece os seguintes critérios por ordem de relevância: proximidade; perfil de cada *simple peer*; latência de rede; latência da federação; georeferenciação; capacidade de cada *peer*; e, capacidade final da federação, que é definido pelos *super peers*.

Cada *peer* possui uma interface de serviço *REST* [Webber et al., 2010] que permite a autenticação do usuário, armazenamento, recuperação e remoção dos dados salvos. Tal característica apresenta como principal vantagem a possibilidade de compatibilizar o sistema com outras interfaces existentes, como *S3* da *Amazon*. Na atual

arquitetura, o serviço de armazenamento dos dados pode ser modificado para outras alternativas (i.e. *Megastore*, *MSFSS* ou *Amazon S3*).

Para garantir o espalhamento de cada *chunk* de forma a garantir os níveis de serviço adequados, cada *peer* precisa periodicamente relatar seu estado atual com o objetivo de manter o SLA sempre atualizado. A **Erro! Fonte de referência não encontrada. (a)** apresenta o fluxo de funcionamento de um *peer* (*PL_1*), desde o momento em que ele anuncia o seu perfil à comunicação estabelecida com os demais pares por meio do servidor (*PS_1*). Periodicamente cada *peer* envia para um dos servidores uma mensagem de "*keep alive*" informando que está *online*. Desta forma, o servidor sabe se o *peer* está cumprindo com o perfil acordado (SLA) e o torna elegível para receber *chunks* no horário definido. Também periodicamente cada *peer* verifica com os demais *peers* existentes se os *chunks* que ele possui estão replicados na quantidade mínima de *peers* obedecendo o critério de disponibilidade exigida [Duarte, 2010]. Caso não esteja, ele replica o *chunk* em outro *peer*. Quando o *peer* que possui este *chunk* voltar a se conectar a rede, ele irá verificar que existe um excesso deste *chunk* e irá excluí-lo.

A **Erro! Fonte de referência não encontrada. (b)** apresenta o fluxo de funcionamento entre *peers* e servidores desde a conexão do *peer* local à rede, ao salvamento dos arquivos solicitados. Após o *peer* (*PL_1*) ao se conectar à rede *P2P_1*, o *super peer* indica um *peer servidor* para ele se autenticar. Com a autenticação bem sucedida, um processo de identificação dos pares para formar as federações é executado. Com a federação (agrupamento de *peer*) estabelecida, o sistema está apto a receber arquivos. Ao receber um arquivo ("*arq1.zip*"), o *peer PL_1* informa a necessidade de armazená-lo no sistema, para isto é feita uma segmentação do arquivo (em *chunks*) e estes segmentos são enviados para serem salvos na rede *P2P_1*. Em seguida, para efetuar o salvamento, uma rotina para medir a confiabilidade analisa o estado dos *peer* e, consequentemente, da disponibilidade dos segmentos do arquivo na rede e caso exista uma combinação de *peer* que atenda ao SLA para o armazenamento, os segmentos do arquivo são enviados para estes *peer*. Se não houverem mais segmentos de arquivos a serem salvos, o *peer servidor PS_1* comunica ao *peer PL_1*, que solicitou o salvamento do arquivo, que o mesmo foi salvo com sucesso.

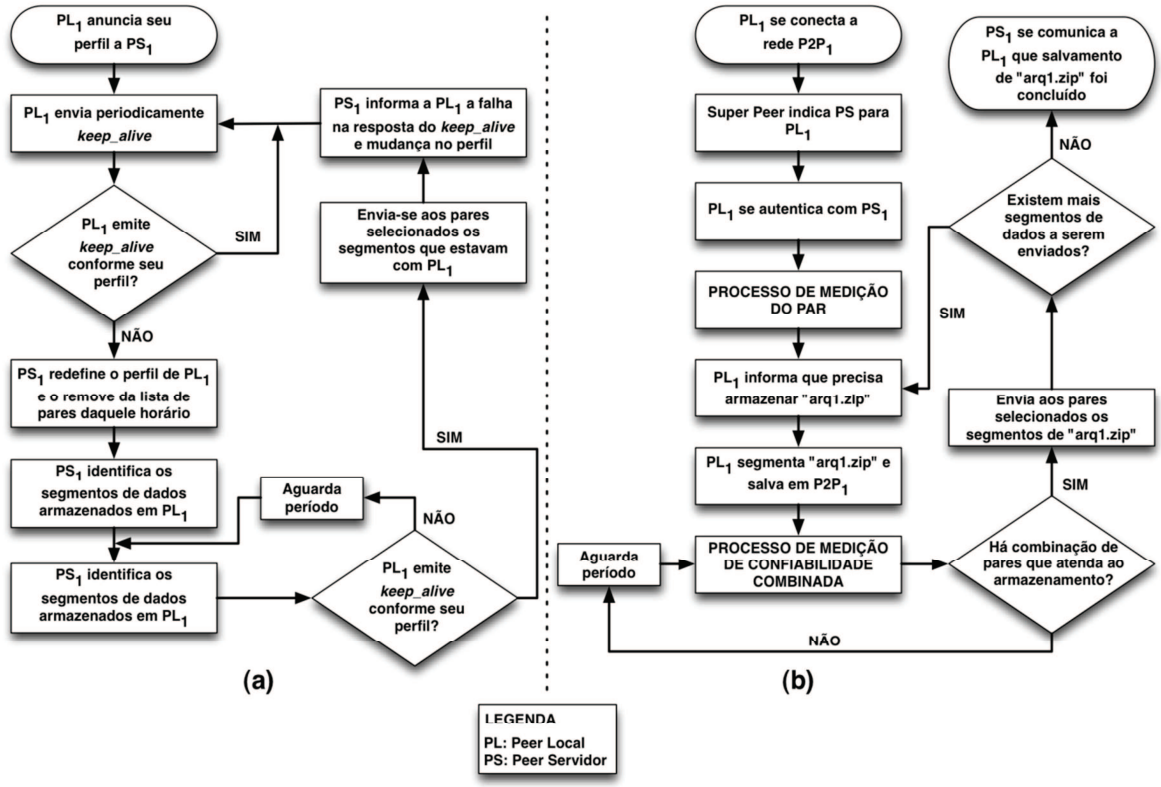


Figura 3.7 - a) Funcionamento de um *peer* b) Funcionamento *peer Server proxy*

3.3.6 Plataforma USTO.RE S3

Ainda como resultado deste projeto, utilizando os conceitos trazidos pelo reuso de software e conforme foi apresentado, pode-se enxergar a plataforma USTO.RE como sendo um sistema que permite a utilização dos recursos ociosos nos discos nos computadores que possuem o software instalado. Este sistema também pode ser visto como uma plataforma de armazenamento em nuvem. Sendo assim, se pode ofertar o uso desta plataforma como sendo um serviço (*PaaS*) na nuvem para pesquisadores que necessitam de muitos *Gigabaytes* para armazenar os dados ou backup dos seus projetos .

Neste caso será provido aos pesquisadores uma interface *SOAP/REST* compatível com as interfaces providas por outros serviços como *Allmydata*, *Amazon S3*, entre outros, de forma que seja possível utilizar os *peers* que compõe a rede como um sistema para armazenamento de dados como *PaaS*. Isto se torna possível porque a plataforma USTO.RE provê a garantia da recuperação dos dados. Neste sentido, será investigado a possibilidade de permitir aos usuários que utilizam plataforma de armazenamento a um custo mais elevado, utilizar o sistema USTO.RE S3 através de um custo mais acessível.

Pode-se afirmar que existe a diminuição de custo porque ao analisar as soluções descritas anteriormente, em todas elas existe a necessidade de se prover infraestrutura dedicada, mesmo que a baixo custo, para suportar o sistema. Neste caso, o custo diminui bastante visto que será utilizado parte do espaço disponível no disco dos usuários.

3.4 Banco de dados

Seguindo este mesmo principio a escalabilidade de banco de dados pode ser dada através da alocação de múltiplos sistemas de banco de dados em paralelo e s associando aos conceitos de sistemas multi-tenant.

Neste contexto a proposta da criação de sistemas multi-database, como adotado facebook se mostra extremamente viável.

Os objetivos deste modelo de funcionamento são:

Redistribuir e crescer sistemas de banco de dados dinamicamente sem a necessidade de alteração no código da aplicação.

a) Diminuição da carga no SGBD's
b) Crescimento elástico
c) Sistema de provisionamento baseado nos princípios de cloud computing

Como funciona:

I. A aplicação ao iniciar consulta a lista de banco de dados existentes
   a. Esta lista contém a URL de conexão do BD e qual informação ele possui
II. O pool de conexão conecta nestes bancos
III. Baseado no identificador de segmentação o sistema ao realizar o getConnection para o pool de conexão informa o identificador e baseado nesta informação o pool seleciona o DB
IV. Quando administrador do SGBD encontrar a necessidade de dividir o banco ele envia o comando através da interface de gerenciamento dos banco de dados provida pela aplicação
V. O banco de dados e colocado em read-only mode e um snapshot e feito. As consultas continuam porém novas transações serão rejeitadas
   a. Um mecanismo de tratamento de exceção precisa ser implementado/alterado para suportar o erro gerado
VI. Um novo servidor de SGBD é solicitado para o gerenciador de cloud
VII. O snapshot do BD em read-only é copiado para este novo banco
VIII. Queries de limpeza do Banco rodam para limpar os dois bancos novos.
IX. O sistema de particionamento atualiza o banco de dados com os hashes
X. O sistema de particionamento notifica a aplicação para atualizar seus hasehs.
XI. O pool de conexão referente ao servidor que foi particionado é ressetado
   a. Assim as conexões existentes na aplicação gerarão um erro forçando outro getConnection a acontecer

A representação deste fluxo esta descrito na Figura 3.8

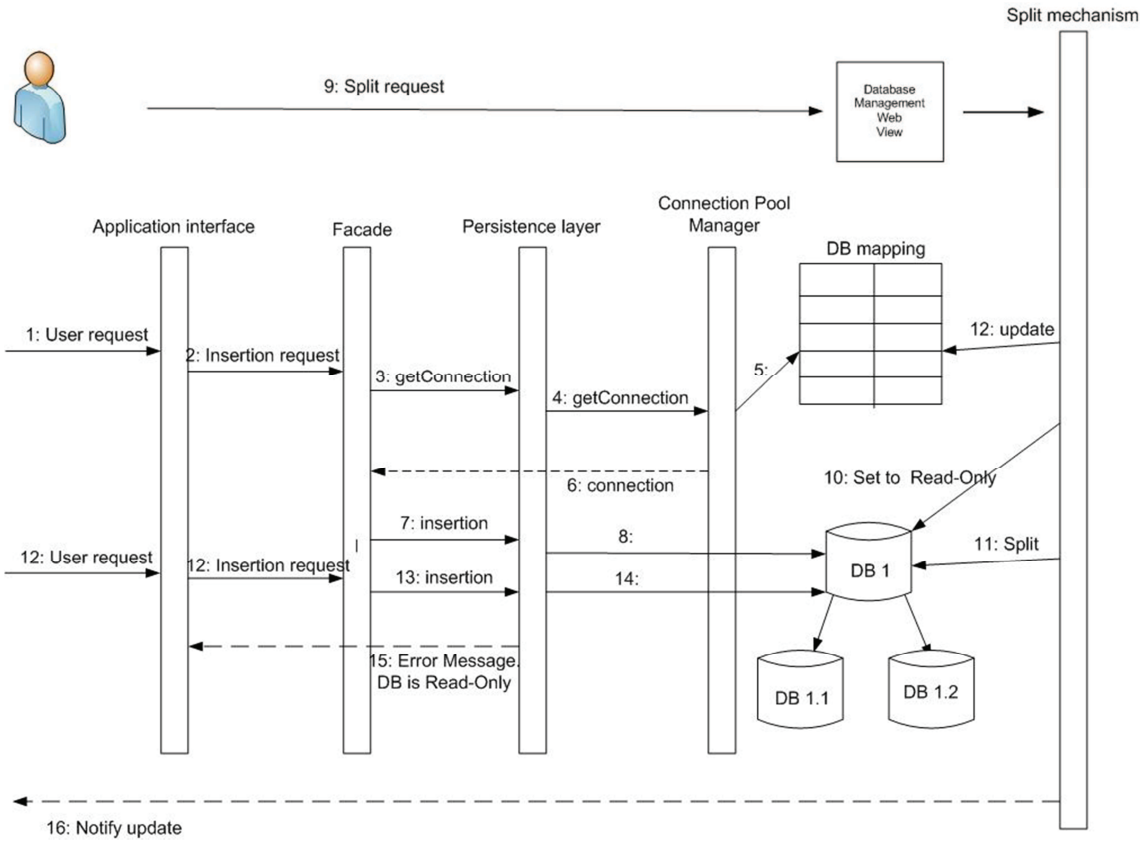


Figura 3.8 - Sistemas Multi-Database

3.5 Bigdata

Os conceitos relacionados a BIGDATA estão intrinsicamente ligados as necessidades de busca, recuperação e recomendação em uma base de dados desestruturados. Este tipo de dado, normalmente visto como um arquivo, agora esta salvo em sistemas de cloud storages como descrito nas sessões anteriormente.

Sendo assim é necessário que os mesmos conceitos ofertados por sistemas de arquivos tradicionais, como EXT4 do Linux e NTFS do Windows sejam portados para os sistemas de nuvem e a eles acrescidos informações extras que permitam a realização das operações desejadas nas plataformas de BIGDATA.

O começo da solução deste problema, esta na definição de metadados que contenham as informações mínimas de sistemas de arquivo, acrescidas de informações para a busca e recuperação, como índices e palavras chave.

3.5.1 Trabalhos Relacionados

Tanenbaum (2007) lista alguns atributos que podem compor os metadados de um sistema de arquivos para ajudar a superar os desafios existentes. Segundo o autor supracitado, nenhum sistema existente dispõe de todos os atributos, porém cada um deles está presente em algum sistema. Na lista aparecem atributos ligados à proteção do arquivo e indicam quem poderá acessá-lo, bem como a senha necessária para isso, caso seja necessário. Há ainda flags para indicar se o arquivo é somente para leitura, oculto,

se já foi feita cópia de segurança, além disso há campos para indicar datas de criação, último acesso e tamanho do arquivo.

O Microsoft Windows 8 utiliza como sistema de arquivos o ReFS[38](Resilient File System) que foi desenvolvido utilizando como base o NTFS. O ReFS utiliza árvores B+ para armazenar todos os dados e metadados do mesmo. A reutilização de código foi amplamente utilizada para manter uma alta compatibilidade com um subconjunto de funcionalidades do NTFS, como alguns atributos dos metadados (padrão de informações, nome do arquivo, valores de algumas flags, etc), interface do sistema de arquivos (leitura, escrita, abrir, fechar, modificar, notificar, etc), manter arquivos na memória, imposições de segurança e memória cache. Muitos dos atributos dos metadados utilizados no ReFS vem do NTFS, como os atributos para nome, dados, lista de controle de acesso, informações padrões (datas de criação, modificações e último acesso). Mas há atributos exclusivos do novo sistema de arquivos como o INTEGRITY_STREAM e o NO_SCRUB_DATA. O atributo INTEGRITY_STREAM é criado unicamente para gerenciar a integridade de diretórios e arquivos pertencentes a aplicativos que preferem gerenciar o armazenamento de arquivos cuidadosamente e, para isso, dependem de um determinado layout de arquivo no disco. Como os fluxos de dados realocam blocos sempre que o conteúdo de um arquivo é alterado, o layout do arquivo fica muito imprevisível para esses aplicativos. Para combater os casos de corrupções de disco e falhas no armazenamento, o ReFS periodicamente depura todos os metadados e dados em um volume do ReFS. O atributo de arquivo NO_SCRUB_DATA é utilizado para indicar que o depurador deve ignorar o arquivo. Esse atributo é útil para os aplicativos que mantém suas próprias informações de integridade[39].

O EXT4 é um sistema de arquivos de journaling para Linux desenvolvido como uma extensão para o EXT3 com o objetivo de revolver problemas de escalabilidade, desempenho e confiabilidade. O EXT3 possui usa 32 bits para representar blocos de números e tem por padrão um bloco de 4K, isso faz com que o sistema de arquivos fique limitado a 16 TB, o que hoje, devido à enorme massa de dados crescente existente, é uma capacidade de armazenamento pequena. O EXT4 ampliou o bloco de número para 48 bits, o que, em teoria, permite que o sistema de arquivos suporte até 1 Exabytes (1 milhão de TB) (Mathura, 2007). Além disso, o mesmo possui uma alta compatibilidade com versões futuras e antigas do sistema de arquivos, como o EXT3, e aprimoramentos na resolução e no intervalo do registro de data e hora[40]. Os metadados do EXT4 são compostos pelo número do inodes que o diretório aponta, tamanho do diretório, nome, tipo de dado armazenado no inode (arquivo, diretório, *socket*, *link* simbólico), além da lista de controle de acesso (que é guardada separadamente, em um *inode* especial) e atributos para a configuração do sistemas de arquivos, como o EXT4_INDEX_FL que indica se os diretórios serão armazenados em árvores B ou em um *array* linear[41].

O Hadoop Distributed File System (HDFS) é um sistema de arquivos distribuídos projetado para rodar sob hardwares comuns. Diferentes de alguns outros

[38]http://bit.ly/KzTYdZ
[39]http://bit.ly/KMEEQV
[40]http://ibm.co/KQQBB3
[41] http://bit.ly/LrsF6p

sistemas de arquivos, o HDFS é altamente tolerante a falhas e foi projetado para ser executado em hardwares de baixo custo. Todos os servidores são totalmente conectados e a comunicação acontece através de protocolos baseados em TCP.

Diferentemente de outras abordagens de sistemas de arquivos distribuídos, o armazenamento e processamento do HDFS é feito em cada nó do sistema. Assim como outros sistemas de arquivos, o HDFS guarda os metadados separados dos arquivos da aplicação. O metadados são armazenados em um servidor dedicado chamado NameNode e os dados da aplicação são armazenados em outros servidores, chamados DataNodes. O NameNode possui inodes, que representam os arquivos e diretórios. Além disso, os inodes guardam atributos como permissões, data de acesso e modificações, namespace onde os arquivos estão e espaço disponível em cada disco (Shvachko, 2011).

O XtreemFS[42] é um sistema de arquivos globalmente distribuído e replicado. Ele é desenvolvido como parte do projeto XtreemOS EU, o qual tem o objetivo de criar um sistema operacional open source para ser utilizado em ambientes de grid. XtreemFS é baseado em objetos, com metadados e dados armazenados em diferentes tipos de nós. Além disso, ele é compatível com o padrão POSIX, e possui sistema de tratamento de falhas. Ele replica os objetos para tolerância a falhas e faz cache dos dados e metadados para melhorar o desempenho no caso de alta latência (Hupfeld, 2007 e 2008). O XtreemFS faz a replicação a partir do arquivo completo, ao invés de utilizar as partes já divididas do arquivo. Isso faz com que haja um aumento considerável na comunicação interna para manter as réplicas sincronizadas. O XtreemFS utiliza o BabuDB como banco de dados para armazenar os metadados do sistema de arquivos. Os atributos utilizados nos metadados incluem uma identificação única para arquivos e diretórios, nome, tipo, além de atributos relacionados a data de criação e modificação, tamanho do arquivo, localização do conteúdos do arquivos, localização das réplicas[43], dono, controle de acesso e atributos estendidos.

O Google File System (GFS) é um sistema de arquivos distribuídos escalável para grandes aplicações distribuídas com uma grande quantidade de dados. Ele provê tolerância a falha e um acesso confiável, eficiente ao dados utilizando um conjunto de hardwares comuns. Este sistema de arquivos é otimizado para trabalhar com grandes arquivos, normalmente 100MB ou mais, sendo muito utilizado para arquivo com tamanho superior a 1GB. Sua arquitetura consiste de múltiplos nós, sendo eles de dois tipos: um nó Master e um grande ChunkServer. Cada arquivo armazenado é divido em chunks de tamanho fixo, os quais são armazenados no ChunkServer. Cada chunk criado é replicado para outros servidores do mesmo tipo, no mínimo, três vezes. O nó Master armazena todos os metadados associado com os chunks, como uma tabela onde há o mapeamento da localização dos chunks com os arquivos, a localização das cópias dos chunks, quais processos estão acessando os chunks (Ghemawat, 2003).

O Amazon S3[44](*Simple Storage Systems*) é um de armazenamento oferecido pela Amazon Web Service que funciona através de interfaces *web services* baseadas em *REST*, *SOAP* e *BitTorrent*.

[42] http://www.xtreemfs.org

[43]http://www.xtreemfs.org/arch.php

[44]http://aws.amazon.com/s3/

O Amazon S3 é projetado para fornecer escalabilidade, alta disponibilidade e baixa latência. Ele é formado por *buckets*, que é similar a um diretório or um *container*. Cada objeto armazenado é composto pelo nome, um *blob* com o conteúdo (até 5 Gb) e os metadados do mesmo, composto por atributos que representam o tipo de dado armazenado, associação com usuários e permissões de acesso, datas de criação, acesso e modificação, localização de objetos relacionados, classificação e comentários feitos pelo usuário, rótulos de área, assunto e geolocalização[45].

O Amazon S3 só permite buscas limitadas a consultas simples baseadas no nome dos objetos e dentro de um único *bucket*, não permite buscas baseadas em metadados e nem no conteúdo dos objetos (Palankar, 2008).

Existem outros sistemas focados em backup nas nuvens como Dropbox, SugarSync, Google Drive, mas os mesmos são soluções proprietárias e não possuem informações relacionadas com metadados disponíveis, o que dificulta uma análise técnica aprofundada sobre os mesmos, mas é possível dizer que os mesmos não possuem uma indexação que permita que consultas sejam realizadas baseadas no conteúdo de cada arquivo armazenado.

Os diversos sistemas mostrados acima são utilizados para diferentes fins: alguns são utilizados em ambientes tradicionais (disco local), outros são focados em ambientes de rede ou distribuídos e outros são direcionados para serem utilizados em ambientes de computação nas nuvens, por isso, eles possuem uma estrutura de metadados que varia de acordo com a finalidade desejada. Os metadados descritos possuem atributos utilizados para diversos fins, como: identificar os arquivos e diretórios, controlar o acesso aos mesmos, indicar a forma de armazenar os dados, localizar os *chunks* e réplicas, armazenar as datas de criação, acesso e modificação, controlar a integridade. Como pode ser observado, não há uma estrutura definida para metadados voltada para sistemas de *cloud storage* que atenda aos requisitos apresentados anteriomente na seção 2 (motivacao), principalmente, no que se refere a indexação do conteúdo armazenado.

3.5.2 Solução Proposta

Um sistema de arquivos é um conjunto de estruturas lógicas e de rotinas, que permitem ao sistema operacional controlar o acesso ao disco rígido[46] e, para isso, Tanembaum (2007) define que é necessário que estas característas estejam presentes:

- Segurança ou permissões
- Listas de controle de acesso (ACLs, em inglês, *Access Control Lists*)
- Mecanismo para evitar a fragmentação
- Capacidade de enlaces simbólicos (*symbolic links*) ou duros (*hard links*)
- Integridade do sistema de arquivos (*Journaling*)
- Suporte para arquivos dispersos
- Suporte para quotas de discos
- Suporte de crescimento do sistema de arquivos nativo

[45]http://bit.ly/JeaAZg
[46]http://bit.ly/JebDZe

Já para o desenvolvimento de um sistema de data cloud, como o usto.re, outras característas diferentes das mencionadas acima foram identificadas. Um sistema de *cloud storage* deve ser preocupar em como armazenar e recuperar os dados, além de identificar qual a máquina utilizada, indexação do conteúdo, como provisionamento sob demanda (habilidade de alocar espaço de acordo com a necessidade), snapshots (criar uma imagem instantânea do disco sem criar uma cópia completa), replicação e clonagem[47].

Portanto, o usto.re, deve ser visto como um sistema de *cloud storage* que implementa metadados como forma de facilitar o gerenciamento e acesso aos arquivos armazenados.

3.5.3 METASTORE

Assim como no modelo tradicional, onde um disco local possui o sistema de arquivo, é necessário que seja definido um conjunto de metadados associados ao arquivo que contém as informações sintáticas.

No usto.re os metadados foram descritos utilizando JSON (*JavaScript Object Notation*) devido a facilidade de representação de objetos e ao fato de consumirem menos recursos para serem processados, favorecendo assim a troca de dados, quanto comparado com o XML. Os atributos dos metadados foram definidos tendo como base os requisitos definidos anteriormente na seção motivacao, os atributos levantados na seção de trabalhos relacionados, além de atributos ligados às necessidades da computação nas nuvens para identificação e recuperação de informações.

A partir da análise dos sistemas de arquivos na seção de trabalhos relacionados observa-se que para o usto.re atender aos requisitos funcionais especificados são necessários atributos para identificação do dono do arquivo, caminho relativo do arquivo para o diretório monitorado, código hash para identificar se modificações foram feitas, a data e hora da última modificação. Além disso, também foi preciso criar um perfil para identificar a máquina a partir da qual o usuário acessa o sistema no momento do backup, para que o sistema pudesse identificar se o usuário está autenticado a partir de uma máquina diferente. Isso é necessário devido ao requisito de monitoramento de diretórios, já que uma máquina diferente, possivelmente, não possui a mesma organização hierárquica de arquivos e diretórios, podendo, assim, fazer com que o sistema tentasse monitorar um diretório que não existe. O perfil da máquina é composto pelo nome da máquina, sistema operacional e arquitetura do mesmo.

Também é necessário incluir atributos para identificar quais as opções de segurança e compartilhamento relacionadas ao arquivo. Estas opções de segurança dizem respeito à forma como os outros usuários verão este arquivo (público ou privado) e como interagirão com ele (somente leitura, escrita, controle total). Através desse atributo é possível também determinar que o arquivo fique acessível apenas para o grupo ao qual o usuário pertença.

Além das informações relacionadas ao arquivo e sobre a maquina que o usuário está autenticado, outra característica identificada como necessária é a indexação do

[47]http://bit.ly/KVuGsU

arquivo de forma a facilitar a localização de arquivos. Esta funcionalidade de indexação de dados em sistemas de data cloud é uma das funcionalidaes mais desejadas (Buyya, 2009), (Leung, 2009) e (Grossman,2008), porém de difícil implementação, visto que os arquivos não são salvos integralmente, mas sim quebrados em pedaços menores (*chunks*) e cada pedaço salvo em um servidor da nuvem de dados diferente.

Na implementação do usto.re, os *peers* que recebem os arquivos e os salvam na *data cloud*, são responsáveis também por indexá-los, antes que os mesmos sejam quebrados em partes menores para serem armazenados, com o objetivo de extrair o conteúdo dos mesmos e salvá-lo em um arquivo separado, formando, assim, um índice onde as buscas serão realizadas.

Após a geração dos arquivos necessários para armazenar o índice, um serviço provido pelo sistema é responsável por copiar esta informação para o servidor, onde esses dados ficarão disponíveis para a realização de buscas.

Para a indexação de arquivos foi utilizado o Apache Lucene[48], por ser um motor de buscas de alto desempenho escrito em Java. O Lucene contém apenas o núcleo do "motor" de busca. Por isso, ele não inclui um *Web crawler* ou um *parser* para diferentes formatos de documentos. Para resolver o problema do \textit{parser} foi utilizado o Apache Tika[49], que é um detector e extrator de conteúdo de metadados e texto estruturado, podendo ser utilizado com arquivos de diversos formatos como: HTML, XML, OLE2 e OOXML do Microsoft Office, OpenDocument Format, PDF, ePub, RTF, arquivos compactados e empacotados.

Sendo assim, quando um usuário realizar uma busca pelo conteúdo de um arquivo no usto.re ele poderá consultar todos os arquivos do sistema, porém:

- O usuário somente poderá acessar os arquivos pertencentes ou compartilhados com o mesmo;
- Somente poderá visualizar que um ou mais arquivos tem a informação procurada, desde que no metadado do mesmo seja informado que o arquivo poderá ser indexado;
- Caso um usuário deseja acessar um arquivo que não lhe pertence deverá solicitar ao dono que o compartilhe;
- Poderá existir um número N de arquivos que o usuário não tem acesso que contém esta informação;

Assim, a estrutura dos metadados para o usto.re é composta pelos seguintes atributos:

- Nome do arquivo;
- Código *Hash*;
- Data da última modificação do arquivo;

[48]http://lucene.apache.org/
[49]http://tika.apache.org/

- *Flag* para identificar se o arquivo deve ser indexado ou não;
- Atributos de segurança e compartilhamento;
- Dono do arquivo;
- Grupo ao qual o dono pertence;

Detalhes da Implementação do METASTORE

A Figura 3.3 representa o processo de backup, geração de metadados e indexação. No usto.re quando a confirmação de um backup chega ao *appClient*, é executado um método para gerar os metadados do arquivo salvo. Com a posse desses dados, o *appClient* os adiciona ao arquivo que representa o mapeamento do diretório do qual faz parte e onde se encontram todos os metadados dos arquivos cujo *backup* foi concluído com êxito. Após isso, o *appClient* envia esse mapeamento para que seja salvo no banco de dados juntamente com as informações de qual é o caminho completo para o diretório monitorado, caminho relativo e código *hash* do mapeamento.

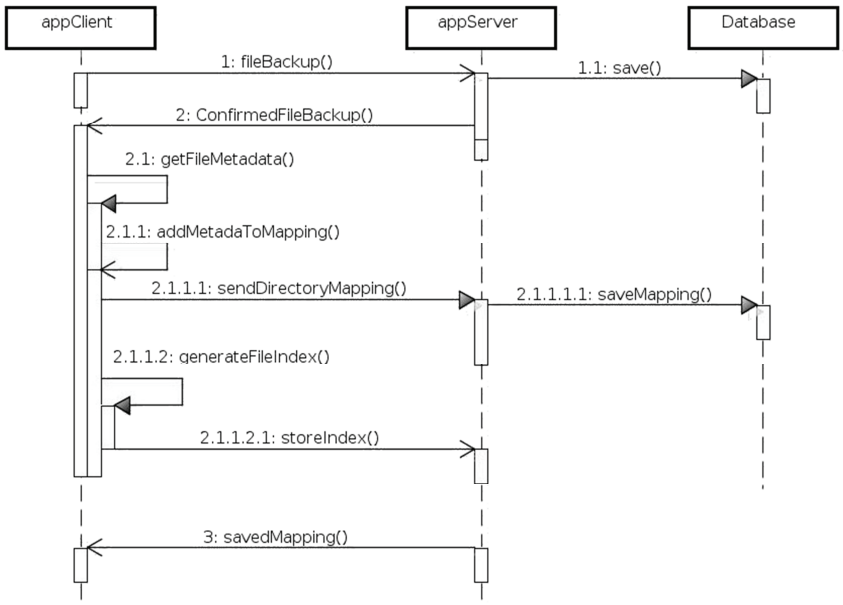


Figura 3.9 - Diagrama de sequencia metastore

Paralelamente à atividade descrita anteriormente é feita a indexação dos dados do arquivo e os dados gerados são enviados para o diretório *index* no servidor. Esta indexação extrai o conteúdo dos arquivos utilizando o Apache Tika e junta isso com a informação, repassada pelo appClient, de qual o nome do usuário que está fazendo backups no momento, o qual representa o dono dos arquivos e o nome junto com o caminho relativo para cada arquivo.

Na Figura 3.9 é apresentada a sequência executada quando é necessário efetuar uma busca por um arquivo no usto.re. A requisição feita ao *appClient* é repassada para o *appServer*. Nele o método *searcher* executa a busca utilizando a indexação gerada pelo Apache Lucene para isso. A busca pode ser realizada através do nome do arquivo, parte do conteúdo ou nome do dono. Se a busca for efetuada utilizando o nome do arquivo ou

parte do conteúdo como parâmetro, o método retornará uma lista de arquivos contendo o nome dos mesmo e a quem pertence cada um deles. A depender das permissões de acesso do usuário, alguns arquivos podem deixar de ser exibidos na listagem ou podem aparecer arquivos aos quais o usuário não poderá acessar sem a permissão do dono.

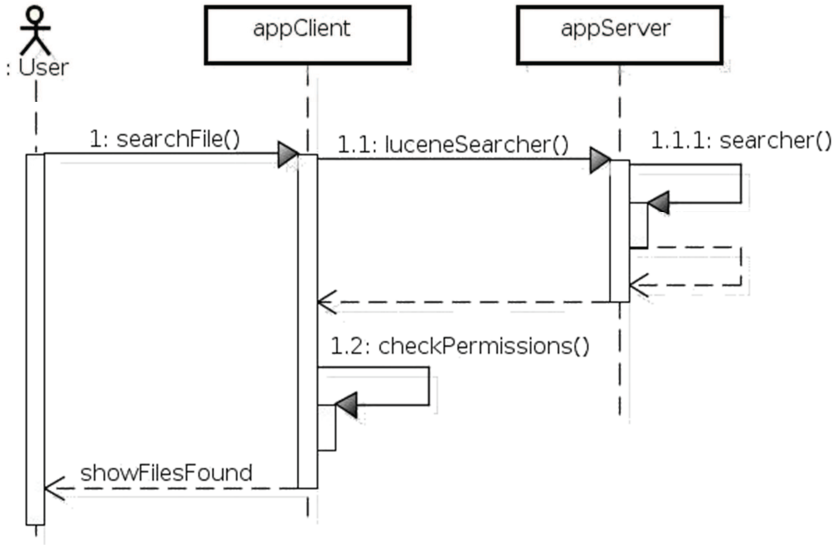


Figura 3.10 - Sincronização de diretorio

A sincronização de diretório só é possível porque o banco de dados armazena o caminho completo para os diretórios monitorados, isso permite chegar até os diretórios e, a partir disso, passar a monitorá-los. Quando um usuário loga, o usto.re faz uma consulta ao banco de dados pelos arquivos que o mesmo possui e quais são os diretórios monitorados. O usto.re traz o mapeamento armazenado no banco de dados e compara com os arquivos existentes nos diretórios monitorado, caso haja alguma diferença entre eles, será feita uma sincronização de arquivos, o que pode incluir adicionar e, até mesmo, remover arquivos.

Caso o usuário esteja em um cliente que não é sua máquina de trabalho (ou seja, que não possui seus arquivos) o usto.re criará uma pasta chamada "usto.re" dentro da pasta do usuário logado e realizará a recuperação de todos os arquivos do usuário. Isso permite que o mesmo tenha acesso a todos seus arquivos e diretórios independente do cliente utilizado.

3.6 Conclusão

Computação nas nuvens (Cloud Computing), consiste de um conjunto de tecnologias que agrupadas trazem vantagens significativas para a gerenciamento dos ambientes de Tecnologia da Informação, consequentemente a diminuição dos custos.

No entanto, a idéia de que qualquer aplicação pode ser migrada para a nuvem sem a necessidade de alterações, consiste de uma visão simplista do problema. Esta afirmativa, é verdadeira desde que os gestores de tecnologia abdiquem das questões relacionadas a performance (no caso de

computação nas nuvens o crescimento elástico) e segurança, caso estejam em execução em ambientes públicos.

Este capítulo de livro apresentou como se define e implementa sistemas de armazenamento de massa em nuvens, cloud storages, como se pode definir e implementar sistemas de banco de dados escaláveis e uma abordagem para a incorporação dos conceitos de BIGDATA em sistemas de armazenamento de dados em nuvens.

3.7 Referências

[Aidouni et al., 2009] F. Aidouni, M. Latapy, and C. Magnien, "Ten weeks in the life of an eDonkey server," Proceedings of HotP2P'09, 2009, pp. 1-5.

[Armbrust et al., 2009] M. Armbrust et al. Above the Clouds: A Berkeley View of Cloud Computing. EECS Department, University of California, Berkeley. Technical Report No. UCB/EECS-2009-28. February 10, 2009.

[Amazon EC2, 2012] Amazon Elastic Compute Cloud (Amazon EC2), URL: http://aws.amazon.com/ec2/, Acessado em 05/07/2012.

[Amazon, 2012] Amazon. Amazon Simple Storage Service (Amazon S3), URL: http://aws.amazon.com/pt/s3/, last access 05/07/2012.

[Baker et al., 2011] J. Baker, C. Bond, J. Corbett, J. J. Furman, A. Khorlin, J. Larson, J.-M. Leon, Y. Li, A. Lloyd, and V. Yushprakh. Megastore: Providing scalable, highly available storage for interactive services. In CIDR'11, pages 223–234, 2011.

[Balakrishnan et al., 2003] Balakrishnan, H., Kaashoek, M.F., Karger, D., Morris, R., and Stoica, I. Looking up data in P2P systems. Communications of the ACM 46, 2 (2003), 43.

[Bass et al., 2000] L. Bass, C. Buhman, S. Dorda, F. Long, J. Robert, R. Seacord, K. Wallnau, Market Assessment of Component-Based Software Engineering, Software Engineering Institute (SEI), Technical Report, Vol. 01, May, 2000, pp. 41.

[Batten et al., 2002] Christopher Batten, Kenneth Barr, Arvind Saraf, Stanley Trepetin. pStore: A secure peer-to-peer backup system. Technical Memo MIT-LCSTM-632, Massachusetts Institute of Technology Laboratory for Computer Science, October 2002.

[Baumgarten & Chuí, 2009] Jason Baumgarten and Michael Chui, E-government 2.0, mckinseyquarterly.com, July 2009.

[Bell, 2008] M. Bell, Service-Oriented Modeling, 2008, pp. 366.

[BitTorrent, 2012] BitTorrent. Site Ofical. http://www.bittorrent.com/. Acessado em 09/07/2012.

[Bughin et al., 2010] Jacques Bughin, Michael Chui, and James Manyika, Clouds, big data, and smart assets: Ten tech-enabled business trends to watch, August 2010.

[Chang et al., 2006] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: a distributed storage system for structured data. In Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7, pages 15–15. USENIX Association, 2006.

[Clements & Northrop, 2001] P. Clements, L. Northrop, Software Product Lines: Practices and Patterns, Addison-Wesley, 2001, pp. 608.

[Cleversafe, 2012] CLEVERSAFE. Site Oficial. Cleversafe dispersed storage project. http://www.cleversafe.org/dispersed-storage. Acessado em 09/07/2012.

[Colaço et al., 2008] COLAÇO, Eduardo José Moreira ; OLIVEIRA, Marcelo Iury ; SOARES, A.; BRASILEIRO, F ; GUERRERO, D. . Using a file working set model to speed up the recovery of Peer-to-Peer backup systems. ACM SIGOPS Operating Systems Review, v. 42, p. 64-70, 2008.

[Cox & Noble, 2003] Cox, L. P. and Noble, B. D. 2003. Samsara: honor among thieves in peer-to-peer storage. In Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles (Bolton Landing, NY, USA, October 19 - 22, 2003). SOSP '03. ACM, New York, NY, 120-132.

[Dean & Ghemawat, 2008] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. Commun. ACM, 51:107–113, Jan. 2008.

[DeCandia et al., 2007] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: amazon's highly available key-value store. SIGOPS Oper. Syst. Rev., 41:205–220, Oct. 2007.

[Dropbox, 2012] Dropbox, URL: https://www.dropbox.com/. Acessado em 09/07/2012.

[Duarte, 2010]M. DUARTE. Um algoritmo de disponibilidade em sistemas de backup distribuído seguro usando a plataforma peer-to-peer. Dissertação de mestrado, Centro de Informática, Universidade Federal de Pernambuco, Recife-PE, Brazil, 2010.

[Erl, 2008] T. Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2008, pp. 760.

[FIPS, 1995] FIPS 180-1. Secure Hash Standard. U.S. Department of Commerce/NIST, National Technical Information Service, Springfield, VA, Apr. 1995.

[Galán & Sampaio, 2009] F. Galán, A. Sampaio et al. Service Specification in Cloud Environments Based on Extensions to Open Standards. In 4th International Conference on Communication System Software and Middleware (COMSWARE 2009), 2009.

[Grobauer et al., 2011] B. Grobauer, T. Walloschek, and E. Stocker. Understanding cloud computing vulnerabilities. IEEE Security and Privacy, 9:50–57, 2011.

[Gnutella, 2012] Gnutella, Site Oficial, http://rfc-gnutella.sourceforge.net/, Acessado em 09/07/2012.

[Google, 2012] Google Web Applications for Communication and Collaborations. http://www.google.com/apps. Acessado em 09/07/2012.

[GTAR, 2010]GTAR, DVN: Disco Virtual Nacional. URL: http://bit.ly/PCP3kj, Acessado em 09/07/2012.

[JXTA, 2012] JXTA. Jxta protocol, 2012. URL: http://java.net/projects/jxta/, Acessado em 05/07/2012.

[Karger et al., 1997] Karger, D., Lehman, E., Leighton, F., Levine, M., Lewin, D., and Panigrahy, R. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. 29th Annual ACM Symposium on Theory of Computing, (1997), 654-663.

[Kubiatowicz et al., 2000] John Kubiatowicz, David Bindel, Yan Chen, Steven Czerwinski, Patrick Eaton, Dennis Geels, Ramakrishna Gummadi, Sean Rhea, Hakim Weatherspoon, Chris Wells, and Ben Zhao. 2000. OceanStore: an architecture for global-scale persistent storage. SIGOPS Oper. Syst. Rev. 34, 5 (November 2000), 190-201.

[Landers et al., 2004] Landers, M., Zhang, H., and Tan, K. PeerStore: better performance by relaxing in peer-to-peer backup. Proceedings of the Fourth International Conference on Peer-to-Peer Computing, (2004), 72-79.

[Leavitt, 2009] N. Leavitt, Is Cloud Computing Really Ready for Prime Time? IEEE Computer, 2009. 42(1): p. 15-20.

[Li & Dabek, 2006] J. Li and F. Dabek. F2F: Reliable storage in open networks.In Intl Workshop on Peer-to-Peer Systems, Santa BarbaraCA, Feb. 2006.

[Loest et al., 2009] S. R. Loest, M. C. Madruga, C. A. Maziero, and L. C. Lung. Backupit: An intrusion-tolerant cooperative backup system. In Proceedings of the 2009 Eigth IEEE/ACIS International Conference on Computer and Information Science, pages 724–729. IEEE Computer Society, 2009.

[Malkhi & Reiter, 1997] D. Malkhi and M. Reiter. Byzantine quorum systems. In ACM Symposium on Theory of Computing, 1997.

[Martinian, 2012] Emin Martinian. Site Oficial. Distributed internet backup system (dibs). http://www.mit.edu/~emin/source_code/dibs/index.html. Acessado em 09/07/2012.

[Mattos, 2005] Mattos, Antonio Carlos M. Sistemas de Informação: uma visão executiva. São Paulo. Saraiva. 2005.

[McIlroy, 1969] M. D. McIlroy, "Mass Produced Software Components," in Software Engineering: Report on a conference sponsored by the NATO Science Committee, 1969, pp. 138--155.

[Meira, 2005] Meira, Fernando. Resilia: A safe & secure backup-system. Final year project, Engineerng Faculty of the University of Porto, May 2005

[Meyer, 1995] Meyer, Poul L. (1995) “ PROBABILIDADE Aplicações à Estatística”. Livros Técnicos e Científicos Editora. 2 Edição. Rio de Janeiro.

[Microsoft, 2012] Microsoft Skydrive service, http://skydrive.live.com/, 2012.

[Napster, 2012] Napster, Site Oficial, http://www.napster.com, Acessado em 09/07/2012.

[Oliveira, 2007] M. Oliveira. OurBackup: A P2P backup solution based on social networks, MSc Thesis, Universidade Federal de Campina Grande, Brazil, 2007

[Osterweil, 2007] Osterweil, L. J. 2007. A Future for Software Engineering?. In 2007 Future of Software Engineering (May 23 - 25, 2007). International Conference on Software Engineering. IEEE Computer Society, Washington, DC, 1-11

[Papazoglou et al., 2007] Michael P. Papazoglou, Paolo Traverso, Schahram Dustdar, and Frank Leymann. 2007. Service-Oriented Computing: State of the Art and Research Challenges. Computer 40, 11 (November 2007), 38-45.

[Ratnasamy et al., 2001] Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S. 2001. A scalable content-addressable network. In Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols For Computer Communications (San Diego, California, United States). SIGCOMM '01. ACM, New York, NY, 161-172.

[Rowstron & Druschel, 2001] Rowstron, A. and Druschel, P. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. Lecture Notes in Computer Science, November 2001 (2001), 329-350.

[Schollmeier & Rudiger, 2002] Schollmeier, Rudiger. “A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications”. In: IEEE Internet Computing, 2002.

[Shvachko et al., 2010] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), pages 1–10. IEEE Computer Society, 2010.

[Smith et al., 2009] D. M. Smith et al. Gartner Report ID Number G00163522, Predicts 2009: Cloud Computing Beckons, 2009.

[Stal, 2002] M. Stal, Web Services: Beyond Component-Based Computing, Communications of the ACM, Vol. 45, No. 10, October, 2002, pp. 71-76.

[Stoica et al., 2001] Stoica, I., Morris, R., Karger, D., Kaashoek, M., and Balakrishnan, H. Chord: A scalable peer-to-peer lookup service for internet applications. Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, (2001), 160.

[Sun, 2009] Sun Microsystems, Introdution to cloud computing architecture whitepaper, June 2009.

[Turner et al., 2003] M. Turner, P. Brereton, and D. Budgen, "Turning Software into a Service," Computer, vol. 36, 2003, pp. 38-44.

[Vignatti et al., 2009] VIGNATTI, T. ; BONA, L. C. E. ; VIGNATTI, A. ; SUNYE, M. . Long-term Digital Archiving Based on Selection of Repositories Over P2P Networks. In: Ninth International Conference on Peer-to-Peer Computing (P2P'09), 2009, Seattle. Proceedings of Eight IEEE International Conference on Peer-to-Peer Computing, 2009.

[XDriver, 2012] XDriver Box service, http://www.box.net/xdrive, Acessado em 05/07/2012

[W3C, 2000] W3C, A Little History of the World Wide Web, 2000. URL: http://www.w3.org/History.html, Acessado em 05/07/2012.

[Webber et al., 2010] J. Webber, S. Parastatidis, and I. Robinson. REST in Practice: Hypermedia and Systems Architecture. O’Reilly Media, 2010.

[Yang et al., 2006] Q. Yang, W. Xiao, and J. Ren. Prins: Optimizing performance of reliable internet storages. In Proceedings of the 26th IEEE International Conference on Distributed Computing Systems, pages 32–. IEEE Computer Society, 2006.

[Yu et al., 2007] L. Yu, G. Chen, W. Wang, and J. Dong. Msfss: A storage system for mass small files. In W. Shen, Y. Yang, J. Yong, I. Hawryszkiewycz, Z. Lin, J.-P. A. Barthes, M. L. Maher, Q. Hao, and M. H. Tran, editors, 11th International Conference on Computer Supported Cooperative Work in Design (CSCWD), pages 1087–1092, Los Alamitos, CA, USA, April 2007. IEEE Computer Society Press.

[Zhang et al., 2007] Liang-Jie Zhang, Jia Zhang, Hong Cai, Services Computing, Springer and Tsinghua University Press, 2007, ISBN: 978-3-540-38281-2, July 2007

[Zhang et al., 2008] Liang-Jie Zhang, Carl K Chang, Ephraim Feig, Robert Grossman,Keynote Panel, Business Cloud: Bringing The Power of SOA and Cloud Computing, pp.xix, 2008 IEEE International Conference on Services Computing (SCC 2008), July 2008.

[Zhang and Zhou, 2009] Liang-Jie Zhang and Qun Zhou. 2009. CCOA: Cloud Computing Open Architecture. In Proceedings of the 2009 IEEE International Conference on Web Services (ICWS '09). IEEE Computer Society, Washington, DC, USA, 607-616.

[Zao et al., 2004] Ben Y. Zhao, Ling Huang, Jeremy Stribling, Sean C. Rhea, Anthony D. Joseph, and John D. Kubiatowicz. Tapestry: A Resilient Global-Scale Overlay for Service Deployment. IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 22, NO. 1, JANUARY 2004 41

[Grossman 2008] R. Grossman and Y. Gu. Data mining using high performance data clouds: experimental studies using sector and sphere. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD ’08, pages 920–927, New York, NY, USA, 2008. ACM.

[Ghemawat 2003] ,S. Ghemawat, H. Gobioff, and S. Leung. The Google file system. In ACM SIGOPS Operating Systems Review, volume 37, pages 29–43. ACM, 2003.

[Buyya 2009] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems, 25(6):599 – 616, 2009.

[Palankar 2008] M. R. Palankar, A. Iamnitchi, M. Ripeanu, and S. Garfinkel. Amazon s3 for science grids: a viable solution? In Proceedings of the 2008 international workshop on Data-aware distributed computing, DADC ’08, pages 55–64, New York, NY, USA, 2008. ACM.

[Ghemawat 2003] S. Ghemawat, H. Gobioff, and S. Leung. The Google file system. In ACM SIGOPS Operating Systems Review, volume 37, pages 29–43. ACM, 2003.

[Mathur 2007] A. Mathur, M. Cao, S. Bhattacharya, A. Dilger, A. Tomas, and L. Vivier. The new ext4 filesystem current status and future plans. 2007.

[Steinacker 2001] A. Steinacker, A. Ghavam, and R. Steinmetz. Metadata standards for web-based resources. IEEE Multimedia, 8:70–76, 2001.

[Shvachko 2011] K. V. Shvachko. Apache Hadoop: The Scalability Update. Pages 7–13, 2011.

[XtreemFS 2008]F. Hupfeld, T. Cortes, B. Kolbeck, J. Stender, E. Focht, M. Hess, J. Malo, J. Marti, and E. Cesario. The XtreemFS architecture: a case for object-based file systems. Concurrency and computation: Practice and experience, 20(17):2049–2060, 2008.

Capítulo 4

Análise de Informações Contextuais através de Técnicas de Aprendizagem de Máquina

Fábio Santos da Silva, Kátia Cilene Neles da Silva e Graça Bressan

Abstract

This chapter discusses how to employ some techniques from machine learning to facilitate the task of analyzing contextual information on context-sensitive systems. In addition, the chapter presents a proposed approach to this task. Finally, describes how to employ the techniques of machine learning in the proposed approach through the use of the Weka API.

Resumo

Este capítulo aborda como empregar algumas técnicas de aprendizagem de máquina para viabilizar a tarefa de análise de informações contextuais em sistemas sensíveis ao contexto. Além disso, o capítulo apresenta a proposta de uma abordagem para realização desta tarefa. Finalmente, descreve como empregar as técnicas de aprendizagem de máquina na abordagem proposta a através do uso da API Weka.

4.1. Introdução

O estudo da exploração das informações implícitas presentes na interação entre o ser humano e sistemas computacionais com o objetivo melhorar esta interação é uma linha de pesquisa que tem despertado cada vez mais atenção de pesquisadores (Dey e Abowd, 2001;Vieira et al., 2009; Alves, 2009; Silva, 2012). Uma categoria de informações abstraídas deste processo de interação é denominada de informações contextuais. Tais informações podem ser utilizadas por um sistema para automaticamente, em contextos diversos, adaptar seu serviço ou conteúdo, ou fornecer informações relevantes e personalizadas aos usuários. Este tipo de sistema está sendo referenciado na literatura como Sistema Sensível ao Contexto (do inglês, *Context-Sensitive System*) (Vieira et al., 2009) ou Sistema Ciente de Contexto (do inglês, *Context-Aware System*) (Dey e Abowd, 2001).

Segundo Vieira et al. (2009) as áreas da Computação Ubíqua e Inteligência Artificial foram às pioneiras nos estudos da exploração das informações contextuais em

sistemas computacionais. Pesquisas recentes (Vieira et al., 2009; Silva, 2012; Neles, 2012) indicam que as informações contextuais estão sendo utilizadas em outras áreas como: *Hipermídia Adaptativa*, *Interface Humano-Computador*, *Sistemas de Recomendação* e *aplicações médicas.*

No entanto, o desenvolvimento de um sistema sensível ao contexto não é uma tarefa trivial, exige a superação de vários desafios técnicos como compreender, coletar, representar, analisar ou processar as informações contextuais. Desta forma, vem aumentando o interesse na concepção de metodologias de engenharia de software (Vieira et al., 2009), e no desenvolvimento de ferramentas para suporte a construção de sistemas sensíveis ao contexto tais como *toolkits* (Dey e Abowd, 2001), infraestruturas (Silva, 2011; Alves, 2009), *middlewares* (Gu; Pung; Zhang, 2005), entre outros.

Dentre os principais desafios técnicos se destaca a análise ou processamento de informações contextuais que consiste no uso de um conjunto de técnicas que permitem a partir de uma base de informações contextuais coletadas, ou um histórico de contexto, a inferência ou a geração de novas informações mais refinadas e relevantes que serão posteriormente empregadas para prover serviços como personalização e adaptação de conteúdo, recomendação de conteúdo, adaptação de interfaces, entre outros.

Atualmente, diversas técnicas vêm sendo utilizadas na implementação da tarefa de análise de informações contextuais tais como raciocínio baseado em regras, ou regras contextuais, raciocínio baseado em casos, tarefa de classificação, técnicas de aprendizagem de máquina, entre outras (Vieira et al., 2009). Em determinados tipos de sistemas sensíveis ao contexto (Silva, 2012; Alves, 2009) as técnicas de aprendizagem de máquina, estão se destacando como uma abordagem promissora para análise de informações contextuais.

Assim, este capítulo propõe-se apresentar os conceitos introdutórios de técnicas de aprendizado de máquina como classificador Bayesiano, Árvore de Decisão, Rede Neural Artificial e Máquina de Vetores de Suporte (Witten; Franck, 2005). Além disso, discutir uma abordagem para análise de informações contextuais baseada em tais técnicas, e finalmente demonstrar a tarefa de análise contextual por meio de exemplos práticos.

Neste capítulo, será utilizada a API da ferramenta Weka (Hall et al., 2009), desenvolvida na Universidade de Waikato na Nova Zelândia, que contempla, um conjunto de técnicas de aprendizagem de máquina implementado. Também será discutido um exemplo de uso da análise de contexto para recomendação personalizada de conteúdo em ambientes como Web, plataformas móveis e TV Digital.

O restante deste trabalho está estruturado como segue. A seção 4.2 apresenta uma visão geral sobre Computação Sensível ao Contexto, e discute os conceitos básicos sobre contexto e sistemas sensíveis ao contexto. A seção 4.3 apresenta uma visão geral sobre aprendizagem de máquina, e destaca algumas técnicas tradicionais empregadas nesta tarefa. A seção 4.4 descreve a abordagem de analise de informações contextuais baseada em aprendizagem de máquina. Na seção 4.5 um exemplo prático de implementação e emprego da abordagem é apresentado. Finalmente, a seção 4.6 apresenta as considerações finais deste capítulo.

4.2. Computação Sensível ao Contexto

A Computação Sensível ao Contexto ou Computação Ciente de Contexto é uma área de estudo que engloba pesquisas em diversas áreas tais como Inteligência Artificial, Computação Ubíqua, Redes Sensíveis a Serviços, Rede de Sensores Sem fio, TV Digital entre outros. Para todas essas áreas, o estudo dos aspectos contextuais envolvidos possibilita o aumento e melhoria da interação homem-máquina e máquina-máquina, com o objetivo de prover uma experiência de interação mais transparente e natural.

A Computação Sensível ao Contexto começou a ser discutida em meados de 1994 por Schilit e Theimer quando estes afirmaram que esta diz respeito à capacidade de um produto de software "se adaptar ao usuário de acordo com a sua localização, as pessoas e os objetos próximos e às mudanças destes ao longo do tempo". Schilit e Theimer expandiram o conhecimento acerca das aplicações sensíveis ao contexto afirmando que estas além de fornecerem informações sobre o contexto também tem a capacidade de se adaptar a este (Schilit e Theimer 1994 Apud Dey 2001).

Ainda Dey e Abowd (2000) apresentam uma definição mais abrangente e mais genérica que afirma que *"Um sistema é sensível ao contexto quando usa o contexto para fornecer informações e/ou serviços importantes para o usuário e essa importância depende da atividade ou da tarefa do usuário"*. Essa afirmação inclui todas as aplicações sensíveis ao contexto desde aquelas que se adaptam ao contexto quanto àquelas que o apresenta ao usuário final.

A Computação Sensível ao Contexto é um tema em expansão e seu estudo fornece uma base importante na construção de sistemas que envolvem a interação personalizada com o usuário e um comportamento inteligente. As próximas subseções destacam os principais conceitos relacionados à Computação Sensível ao Contexto.

4.2.1. Definição de Contexto

O contexto é um conceito que está presente de forma intrínseca nas atividades diárias das pessoas e durante o processo de interação humana este é empregado de maneira natural e implícita para melhorar a qualidade da comunicação (Silva, 2011). Em sistemas computacionais o conceito de contexto possibilita que haja maior aproximação deste com o usuário. Na literatura é possível encontrar diversas definições oriundas de diferentes áreas para o conceito de contexto. A seguir serão apresentadas algumas importantes definições deste conceito frequentemente referenciadas na literatura.

Uma definição clássica é apresentada em Dey (2001) em que "*Contexto é qualquer informação que pode ser utilizada 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 a aplicação em si*".

Zimmer (2004) e Ranganathan (2003) afirmam que o termo "Contexto" se refere a uma coleção de informações que caracterizam a interação entre o usuário e a aplicação. Tempo, localização, temperatura, luz, som e atividade são exemplos de informação contextual ou contexto. Também se relaciona com qualquer informação que pode ser usada para caracterizar circunstâncias, objetos ou condições pelos quais o usuário é rodeado.

A partir dessas definições tem-se que o contexto assume maior importância uma vez que este é característico de sistemas computacionais que visam interagir com o usuário de maneira inteligente e proativa.

4.2.2. Dimensões Semânticas – 5Ws+1H

As informações que constituem um contexto podem ser identificadas por meio de um conjunto de categorias de informações denominadas de dimensões semânticas. Tais informações contextuais são abstraídas durante o processo de modelagem do contexto de uma aplicação.

Dey (2001) e Abowd (2000) Apud Silva (2011) destacam que a identificação das informações contextuais pode ser facilitada por meio da utilização das dimensões semânticas ou 5Ws+1H: *when* (tempo), *where* (localização), *who* (identificação), *what* (atividade) e *why* (utilizada para se indicar o porquê da execução de uma determinada atividade, qual a motivação) (Dey, 2001; Abowd, 2000). Resumidamente essas dimensões semânticas podem ser observadas na Tabela 4.1.

Tabela 4.1 Dimensões semânticas ou 5Ws+1H

Dimensão	Descrição
Who	Quem realiza uma determinada atividade, quem pode alterar o contexto ou quem pode ser notificado caso o contexto seja alterado.
Where	Onde a pessoa está. Esta é uma das dimensões mais usadas devido ao grande interesse de sistemas baseados em localização.
When	A informação temporal para determinar quanto tempo uma entidade está dentro de um contexto. Esta dimensão associada com a dimensão *where* permite rastrear os caminhos que uma entidade tomou durante um período.
What	O que o usuário está fazendo neste momento. Geralmente necessita de sensores para determinar qual é a atividade, o que torna isso uma tarefa complexa.
Why	Determinar o porquê o usuário está realizando determinada atividade, essa é umas das tarefas mais difíceis por envolver questões de inteligência artificial.
How	Determina a forma como as informações contextuais serão capturadas.

Em Troung (2001), observa-se que devido à dificuldade na obtenção de informações relativas à dimensão *Why*, geralmente associa-se a dimensão *What* com uma sexta dimensão *How* (como) para inferir informações sobre *Why*. Para aplicações específicas, como captura e acesso, a dimensão *How* é importante e pode definir como os dados foram capturados e como eles podem ser acessados.

4.2.3. Sistemas Sensíveis ao Contexto

Sistemas sensíveis ao contexto são sistemas que reagem e se adaptam às mudanças, podendo fornecer serviços e informação de acordo com o contexto do usuário. Além disso, tais sistemas agem automaticamente, reduzindo o excessivo envolvimento do usuário e provendo assistência inteligente e proativa. Para Dey (2001) um sistema é dito sensível ao contexto se este utiliza o contexto para apresentar informação e serviço relevantes para o usuário dependendo da tarefa desempenhada.

A Figura 4.1 Silva (2011) destaca uma comparação entre os sistemas tradicionais e sistemas sensíveis ao contexto. Nos sistemas tradicionais há uma entrada explícita e clara que após processamento fornece uma saída que segue um padrão. Já nos sistemas sensíveis ao contexto há além da entrada explícita, a percepção do

contexto, e a inferência de conhecimento a partir de uma base de conhecimento para então fornecer serviços adaptados ao usuário. Este processo de inferência pode ser implementado, por meio de técnicas de aprendizagem de máquina.

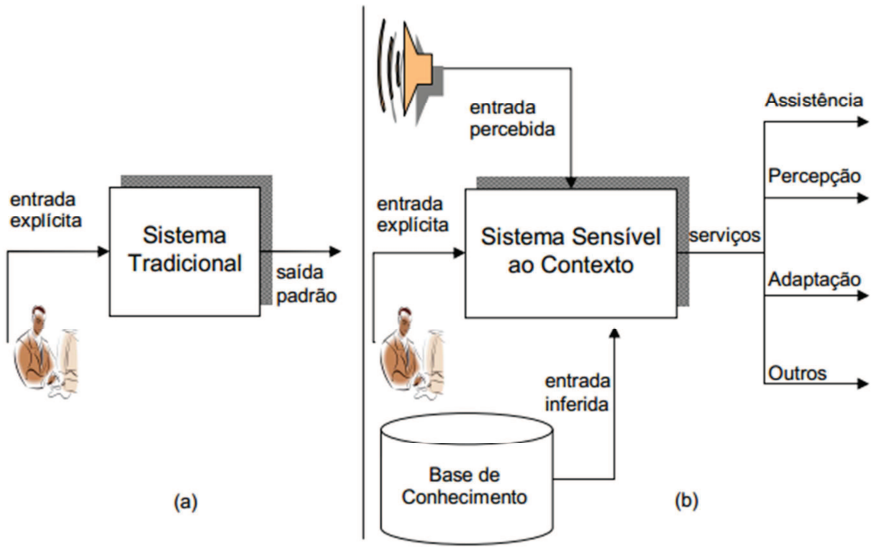


Figura 4.1. Sistema tradicional (a) vs. Sistema sensível ao contexto (b). (Vieira et al, 2009 Apud Silva 2011)

4.2.4. Exemplos de Sistemas Sensíveis ao Contexto

Em Pessoa (2006) é possível encontrar diversas referências de sistemas sensíveis ao contexto, conforme destacadas abaixo:

- Solar system: é uma arquitetura distribuída em componentes, organizada de maneira semelhante ao sistema solar, que oferece apoio à realização da Computação Sensível ao Contexto por parte das aplicações. Foi desenvolvido em Dartmouth Colleg por (Chen et al., 2002);
- Socam (*Service-Oriented Context-Aware Midleware):* é uma arquitetura orientada a serviços que utiliza um modelo forma de contexto com o objetivo de prover prototipação rápida de serviços e aplicações sensíveis ao contexto em ambientes de Computação Pervasiva. Essa arquitetura foi desenvolvida na Computing School of Singapure National University (Guet al., 2005);
- GaiaOS: trata-se de uma plataforma que introduz um ambiente de computação composto por um espaço físico qualquer enriquecido com dispositivos eletroeletrônicos capazes de realizar algum tipo de computação útil aos frequentadores do ambiente (Roman, 2002);
- Aura: é uma plataforma resultante de trabalhos desenvolvidos na Carnegie Mellon University, tem como intuito prover a usuários móveis uma infraestrutura de computação virtual, particular e repleta e serviços capazes de se adaptar em ambientes propensos a falhas e com variabilidade de recursos.

- CoBrA: arquitetura com distribuição híbrida, composta de um componente central denominado ContextBroker. Além do componente central (*broker*), essa arquitetura também incorpora outros agentes configuráveis para notificar as aplicações acerca de mudanças de contexto.Foi desenvolvida na University of Maryland (Chen, 2004);
- Wasp (Web *Architectures for service plataforms*): arquitetura que tem como característica particular o uso da tecnologia de distribuição de serviços Web sendo executada sobre uma rede 3G. Foi desenvolvida na Universtity of Twente, na Holanda (Dockhorn, 2003; Santos, 2004).

Em Vieira et al (2009) outros exemplos de sistemas sensíveis ao contexto que usam pares de chave-valor para representar o contexto incluem o Context Toolkit (Dey et al., 2001) e o CXMS (Context Management System) (Zimmermann et al., 2005).

4.2.4.1. Personalização e Adaptação de Conteúdo baseadas em Contexto

O conceito de sensibilidade ao contexto tem uma estreita relação com a capacidade das aplicações serem personalizáveis ou adaptáveis. Esta capacidade relaciona-se com o comportamento, suas interfaces de apresentação ao usuário, a interação que esta pode ter com sistemas externos, ou ainda em seu conteúdo.

Neste sentido propostas de têm sido apresentadas buscando empregar as vantagens da sensibilidade ao contexto para a melhoria da comunicação entre dos sistemas para com o usuário final. Um exemplo de estudo voltado para personalização e adaptação de conteúdo pode ser observado no trabalho de Goularte (2003) no qual propõe técnicas com suporte à sensibilidade a contexto para adaptação e personalização de conteúdo em TV Interativa. As técnicas desenvolvidas são baseadas em esquemas de descrição compatíveis com o padrão MPEG-7, e na segmentação de conteúdo em objetos MPEG-4. Também foi definida e implantada uma infraestrutura para produção, distribuição, e consumo de programas de TV.

Já o trabalho de Zimmermann (2005) destaca o sistema *Listen* que rastreia o usuário com uma solução que permite identificar se o usuário está olhado para um detalhe de uma obra de arte em uma galeria ou apenas ao lado da obra de arte.

4.2.4.2. Recomendação Sensível ao Contexto de Conteúdo

Os sistemas de recomendação são desenvolvidos como forma de auxiliar os usuários na identificação de itens de seu interesse. Com o contínuo aumento da quantidade de serviços e informações disponíveis e por conseguinte a necessidade de acesso, torna-se cada vez mais necessária a tarefa de recomendação de conteúdos. Conceitualmente, um sistema de recomendação é um tipo de sistema que apresenta ao usuário de maneira personalizada os objetos úteis ou de seu interesse a partir de diversas opções possíveis. É também descrito como uma técnica inteligente para lidar com o problema da sobrecarga de informação (Vozalis e Margaritis 2003, Burke, 2002, Apud Silva 2011).

Silva (2011) destaca que a qualidade das recomendações geradas pelos tradicionais sistemas de recomendações pode ser melhorada por meio da exploração do contexto. O autor ressalta que informações contextuais podem ser empregadas para inferir preferências contextuais do usuário posteriormente utilizadas no processo de recomendação visando retornar itens que estão mais próximos dos interesses do usuário.

Para isso torna-se necessária a utilização de técnicas cada vez mais complexas que viabilizem as tarefas de aquisição, representação, análise de informações contextuais e filtragem sensível ao contexto.

No trabalho de Silva (2011) é proposta uma infraestutura de software alinhada aos padrões de TV Digital Interativa para suporte ao desenvolvimento e execução de sistemas recomendação sensíveis ao contexto para TV Digital Interativa. Foram implementadas esquemas para representação de contexto e conteúdo compatíveis com os padrões MPEG-7 e TV-Anytime e técnicas para análise de contexto e recomendação de conteúdo. Assim, os desenvolvedores de sistemas de recomendação concentram esforços na lógica de apresentação de seus sistemas, deixando questões de baixo nível para infraestrutura gerenciar.

4.2.4.3. Monitoramento remoto e sensível ao contexto da saúde humana

Nas aplicações na área de saúde, a sensibilidade ao contexto tem sido o foco em diversas pesquisas, especialmente naquelas relacionadas à mobilidade dos pacientes e dos profissionais de saúde.

Este tipo de aplicação emprega o conceito de sensibilidade ao contexto a partir da interação entre os sensores empregados por meio de uma rede de sensores. Uma vez que estes ocorrem à distância é importante que os especialistas médicos considerem o contexto em que o paciente está inserido para então direcionar tratamento e diagnóstico. As informações contextuais são capazes de alterar inclusive as informações fisiológicas dos pacientes monitorados (Neles, 2012).

Neste tipo de aplicação, os contextos podem ser classificados como de baixo e de alto nível (Abowd, 2000).

- Contexto de baixo nível: obtido diretamente de sensores ou através de processamento simples;
- Contexto de alto nível: obtido através da composição de informações contextuais de baixo nível ou a partir da análise de informações contextuais de baixo nível por meio de técnicas de Inteligência Artificial. Exemplo: uso de informação de tempo, localização e agenda do usuário como auxiliares da definição de sua situação social: “em reunião”, “recebendo medicação”, “cozinhando”.

Nesta relação, o contexto de baixo nível fornece dados oriundos de sensores e perfis ao contexto de alto nível que através de um processador provê a informação contextual (onde, que, quando, por que) que será usada de maneira ativa ou passiva pela aplicação, possibilitando assim a adaptação do seu comportamento

4.2.5. Requisitos para Implementação de Sistemas Sensíveis ao Contexto

Na literatura e possível encontrar um conjunto de requisitos que são bastante comuns na implementação de sistemas sensíveis ao contexto (Raatikainen et al. 2002), (Strang; Linnhoff, 2004), (Henricksen et al. 2005), e (Vieira et al., 2006). Este conjunto de requisitos é genérico, e visa orientar a implementação deste tipo de sistema. De acordo com o domínio e aplicação os requisitos possuem graus de relevância diferentes. A seguir serão descritos alguns desses principais requisitos.

4.2.5.1. Abstração de Informações Contextuais

Este requisito consiste em identificar quais informações contextuais serão exploradas pela aplicação. Para facilitar esta tarefa, as informações contextuais podem ser identificadas por meio do emprego das dimensões semânticas básicas, referenciadas como 5Ws+1H conforme descrito na seção 4.2.2. O desenvolvedor pode utilizar de modelos formais que viabilizem a representação em alto nível das principais entidades, suas respectivas informações contextuais e seus relacionamentos. Por meio desta representação se torna possível a utilização e a compreensão semântica das informações contextuais por usuários e sistemas computacionais.

Segundo Vieira et al. (2009), na escolha de um modelo formal também se deve levar em consideração fatores como interoperabilidade, extensibilidade, compartilhamento e reusabilidade. Na literatura é possível encontrar alguns trabalhos que apresentam propostas de técnicas para representação de contexto como *par chave-valor*, *linguagem de marcação*, *orientação a objetos*, *mapas de tópicos* e *ontologias* (Strang; Linnhoff-Popien, 2004; Vieira et al. 2009).

4.2.5.2. Separar Aquisição da Utilização das Informações Contextuais

A separação destas tarefas, viabiliza que os sistemas utilizem as informações contextuais sem se preocupar com os detalhes de como tais informações foram adquiridas e tratadas.O objetivo é tornar cada tarefa o mais independente possível, pois as informações contextuais podem ser obtidas a partir de fontes heterogêneas por meio de sensores físicos ou lógicos, de bases de dados, agentes inteligentes, ou até mesmo em último caso fornecidas explicitamente pelo próprio usuário.

Além disso, a independência dos componentes de aquisição em relação às aplicações permite que diversos sistemas possam fazer uso destes, de forma compartilhada. Deste modo, por meio desta separação de tarefas, a complexidade da aquisição e o tratamento da qualidade das informações contextuais são abstraídos para as aplicações.

4.2.5.3. Análise das Informações Contextuais

A análise, processamento ou interpretação das informações de contexto é um dos requisitos mais importantes de sistemas sensíveis ao contexto. Consiste no emprego de um conjunto de métodos e processos que viabilizam a aprendizagem, o raciocínio, a derivação, a predição de tendências, padrões ou preferências sobre as informações contextuais. O objetivo é obter informações de alto nível que permitam a melhorar a compreensão de um determinado contexto pela aplicação de modo a modificar o seu comportamento ou auxiliá-la na tomada de decisões.

Desta forma, é possível obter informações contextuais de alto nível que serão posteriormente exploradas pelas aplicações para diversas finalidades. Por exemplo, predizer a melhor rota de ambulâncias para hospitais através de informações contextuais sobre as condições do trânsito; descobrir qual gênero de música poderá ser recomendado para um usuário de acordo com seu o humor e ambiente.

Na literatura é possível encontrar a utilização de diversas técnicas na implementação do requisito de análise de informações contextuais tais como raciocínio baseado em regras, ou regras contextuais, raciocínio baseado em casos, tarefa de

classificação, técnicas de aprendizagem de máquina, entre outras (Vieira et al., 2009). Em determinados trabalhos (Silva, 2012; Alves, 2009) as técnicas de aprendizagem de máquina, se destacaram como uma abordagem promissora para análise de informações contextuais.

4.2.5.4. Comunicação Distribuída e Transparente

Em termos de arquitetura, os sistemas sensíveis ao contexto são modulares e distribuídos. Assim, a comunicação entre seus componentes dever ser implementada de forma flexível e padronizada. Para isso, protocolos de comunicação, padrões para integração de componentes devem ser utilizados em sistemas sensíveis ao contexto. Além disso, questões também relacionadas à segurança, privacidade e escalabilidade também devem ser consideradas.

Podem ser adotadas tecnologias como Serviços Web (do inglês, *Web Services*) (W3C, 2002). Serviço Web oferece um conjunto de padrões que propiciam uma comunicação padronizada entre diferentes aplicações, promovendo a integração entre elas. A descrição dos principais serviços é feita por meio da WSDL (*Web Service Definition Language*). Já a comunicação é viabilizada por meio da troca de mensagens XML baseada em SOAP (*Simple Object Acess Protocol*) (W3C, 2003). Tais mensagens XML são transportadas por meio de protocolos da família TCP/IP (W3C, 2002).

4.2.5.5. Aquisição Continua de Informações Contextuais

A aquisição de informações contextuais deve ser realizada de forma continua e não intrusiva, sem que usuário tenha que ser questionado insistentemente sobre o contexto em que se encontra. Para isso, sensores eletrônicos e lógicos podem ser empregados para o monitoramento do contexto do usuário. A disponibilidade constante destes componentes viabiliza a aquisição de informações em diversos contextos.

Algumas informações contextuais não podem ser obtidas diretamente por meio de componentes de aquisição. Torna-se necessária a utilização de técnicas de raciocínio para derivar novas informações a partir da composição de informações contextuais primárias ou de baixo de nível. Por exemplo, a informação sobre qual atividade o usuário está realizando "estudando" ou "trabalhando" pode ser derivada a partir de condições de teste de uma árvore de decisão sobre as informações contextuais de tempo e localização.

4.2.5.6. Armazenamento de Informações Contextuais

A necessidade da disponibilidade constante das informações contextuais; o compartilhamento destas informações entre os sistemas; e a tarefa de análise ou raciocínio sobre tais informações demanda o armazenamento das informações contextuais em repositórios contextuais. Torna-se necessário armazenar de forma eficiente um grande volume de informações contextuais levando em consideração o dinamismo do contexto. Para isso, as abordagens tradicionais para armazenamento de dados devem ser analisadas com o objetivo de identificar as abordagens mais adequadas para armazenamento e recuperação de tais informações.

4.2.5.7. Descoberta de Componentes ou Recursos

Em determinados sistemas pode ser necessário descobrir quais componentes estão disponíveis para apoiar a realização de determinadas tarefas. Por exemplo, se um componente que fornece determinada informação não estiver funcionando, torna-se necessário encontrar automaticamente outros componentes similares, ou até mesmo efetuar uma composição de componentes de forma a fornecer uma resposta adequada. Assim, automatizar a descoberta de recursos é um requisito importante em sistemas sensíveis ao contexto.

Devido ao aumento do número de componentes disponibilizados, surge um problema para os serviços de descobertas, que é o de localizar o componente mais apropriado para determinada operação (Forstadius et al. 2005). A descoberta de recursos pode ser implementada por meio da *computação orientada a serviços* (Papazoglou, 2003), o qual utiliza os serviços Web para composição de componentes considerados elementos fundamentais na construção de aplicações distribuídas.

4.2.6. Gerenciamento de Contexto

Conforme já foi destacado, os sistemas sensíveis ao contexto são aqueles que utilizam contexto para execução de uma tarefa como prover informações personalizadas e adaptadas ou serviços relevantes. No entanto, compreender e identificar o contexto e executar tarefas de acordo com as mudanças deste requer a superação de requisitos técnicos. Torna-se preciso lidar com questões como: que tipo de informação considerar como contexto, em qual formato representá-las, como podem ser adquiridas e analisadas ou processadas, e como serão efetivamente utilizadas para alterar o comportamento do sistema.

Diante deste cenário, o gerenciamento de contexto ocupa um papel muito importante em sistemas sensíveis ao contexto, pois envolve conforme ilustrado na Figura 4.2 proposta por (Vieira et al., 2009) as tarefas de aquisição, processamento ou análise de informações contextuais, e disseminação.

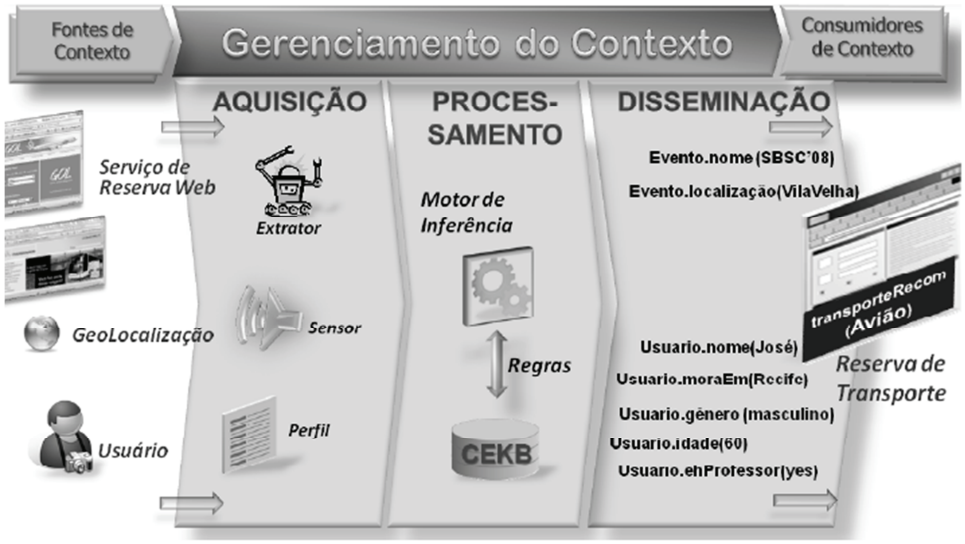


Figura 4.2. Principais elementos envolvidos no gerenciamento do contexto. (Vieira et al., 2009)

A tarefa de aquisição refere-se ao processo de captura de informações contextuais, que podem ser obtidas de diferentes fontes de contexto. A tarefa de processamento emprega um conjunto de técnicas para raciocínio, aprendizagem, predição com o objetivo de inferir preferências, produzir informações refinadas, derivar novas informações contextuais de alto nível. Por fim, é na disseminação que o resultado do processamento é empregado.

Em um sistema de recomendação, por exemplo, o resultado pode ser uma classe de conteúdo que deverá ser recomendado para o usuário (Silva, 2011). Segundo Samaras (1996) quando uma aplicação emprega a gerência de contexto essa gerência mantém controle sobre vários contextos, permite a aplicação criar um conjunto de contextos para trabalho e permite que essa aplicação altere o contexto quando apropriado. A gerência de contexto compartilha a noção de contexto com a gerência de recursos e a aplicação.

4.2.7. Técnicas de Representação de Contexto

A representação de contexto visa fornecer uma visão de alto nível, para que seja possível o compartilhamento, a utilização e a compreensão semântica das informações contextuais por usuários e sistemas computacionais. Para isso, uma representação de contexto apropriada deve especificar as principais entidades e informações contextuais que deverão ser processadas pelo sistema.

No trabalho de Vieira et al. (2009), é apresentada uma análise comparativa das técnicas mais empregadas para representação de contexto, onde são resumidas as vantagens e desvantagens de cada uma delas, assim como, a dos métodos utilizados para processamento e recuperação de contexto de cada uma das abordagens. Algumas dessas principais técnicas são sumarizadas na Tabela 4.2.

Além das técnicas apresentadas na Tabela 4.2, pode ser adicionada a técnica baseada em Orientação a Objetos, que oferece como vantagens: encapsulamento, herança, e reusabilidade. No cenário da TV Digital, em (Goularte, 2003), foi utilizada a linguagem de marcação (XML) e Esquema XML para criar uma biblioteca extensível de elementos contextuais estruturados. Tal biblioteca foi utilizada para o desenvolvimento de um serviço de adaptação e personalização de conteúdo para TV Digital.

Cada técnica de representação possui recursos próprios que podem facilitar a representação de contexto. Deste modo, não há uma técnica que seja considerada a ideal para todos os sistemas sensíveis ao contexto, uma vez que diferentes sistemas impõem diferentes restrições.

Tabela 4.2. Técnicas para representação de contexto.

Técnica	Vantagens	Desvantagens	Processamento e Recuperação
Par chave-valor	Estrutura simples, de fácil implementação e uso.	Não considera hierarquia. Inadequado para aplicações com estruturas complexas.	Busca linear com casamento exato de nomes.
Linguagem de marcação	Baseado em XML. Prevê hierarquia. Esquema de marcação implementa o próprio modelo. Utilização típica em *perfis*.	Incompletude e ambiguidade na informação devem ser tratadas pelo sistema. Inadequado para representar estruturas complexas.	Linguagem de consulta baseada em marcação.
Mapas de tópicos	Facilita a navegação entre os contextos. Facilita a modelagem por humanos.	Estágio inicial. Tecnologia imatura. Faltam exemplos reais.	Navegação por redes semânticas.
Ontologias	Contextos modelados como conceitos e fatos. Viabiliza formalização, compreensão e compartilhamento por humanos e computadores.	Tecnologia de manipulação imatura.	Motor de inferência, linguagem de consulta baseada em OWL
Modelos Gráficos	Facilita a especificação dos conceitos e definição do comportamento do CSS.	Não permite processar os conceitos: mapeamento para estruturas de dados.	Pode ser traduzido para XML e usa processamento em XML.

4.3. Aprendizagem de Máquina

A Aprendizagem de Máquina (do inglês, *Machine Learning*) é uma subárea da Inteligência Artificial que se concentra no desenvolvimento de técnicas que permitem sistemas computacionais aprender e predizer padrões ou tendências a partir de um grande volume de dados (Witten e Franck, 2005; Han e Kamber, 2006; Tan et al., 2009). Suas técnicas são exploradas, principalmente, pelas áreas de Recuperação de Informação, Mineração de Dados, Reconhecimento de Padrões, Robótica e Jogos.

Segundo Han e Kamber (2006) as principais abordagens de aprendizagem de máquina são:

- **Aprendizagem supervisionada**: consiste na aprendizagem de uma função a partir de um conjunto de registros com rótulos de classes conhecidos. Tais rótulos correspondem ao valor da classe que se espera ser descoberta pela função da técnica de aprendizagem sempre que receber registros com rótulos desconhecidos;
- **Aprendizagem não-supervisionada**: se refere à aprendizagem de padrões a partir de um conjunto de registros em que não são fornecidos os rótulos de classes.

Para a abordagem geral de análise de informações contextuais discutida neste trabalho foi empregada a aprendizagem supervisionada. O objetivo desta abordagem de análise é predizer os rótulos de classes de registros de testes a partir de registros de treinamento compostos por atributos preditivos e de classe correspondentes as dimensões semânticas 5Ws+1H. A implementação desta abordagem tem como base a tarefa de classificação discutida na próxima seção.

4.3.1. Tarefa de Classificação

A tarefa de classificar objetos em uma dentre as diversas classes pré-definidas, é um problema amplamente discutido na literatura (Witten e Franck, 2005; Han e Kamber, 2006; Tan et al., 2009). Por meio da classificação é possível analisar um conjunto de dados para obter modelos que podem ser usados para predizer classes, padrões ou tendências futuras e apoiar a tomada de decisão.

Por exemplo, um gerente de *marketing* de uma loja virtual pode utilizar um algoritmo de classificação para analisar o histórico de compras de um cliente para predizer se ele desejará comprar um determinado tipo de computador. Em outro exemplo, um pesquisador pode empregar algoritmo de classificação para analisar dados obtidos de pacientes sobre um determinado tipo de câncer para predizer dentre os tipos de tratamentos qual o seu paciente deverá receber.

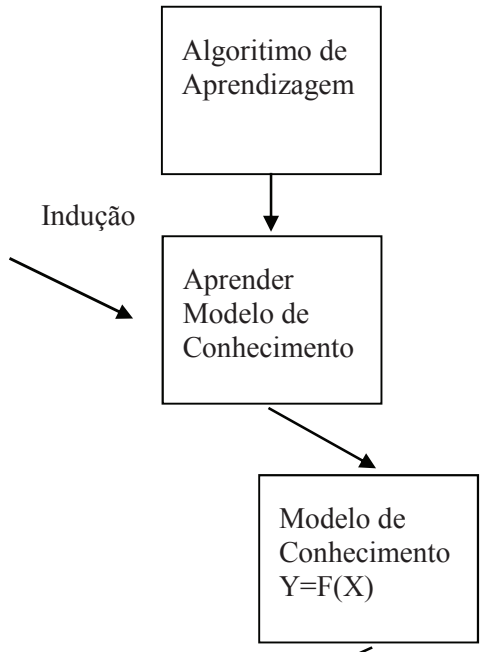
Em cada um destes exemplos, a tarefa de análise consiste na criação de um modelo a partir de um conjunto de dados aprendidos. Tal modelo permite ao sistema ao predizer um rótulo de classe, tais como "computador A" ou "computador B" para o segundo exemplo, ou ainda "tratamento A" ou "tratamento B" ou "tratamento C" para o segundo exemplo por meio de dados dos pacientes. Os dados de entrada da tarefa de classificação são denominados de registros, tuplas ou instâncias. Cada registro pode ser definido por uma dupla (x,y), onde *x* corresponde ao conjunto de atributos preditivos e *y* é um atributo especial, conhecido como atributo-classe (Tan et al., 2009). O conjunto de registros pode ser obtido a partir de uma base de dados, de um histórico de uso, entre outros.

Segundo Han e Kamber (2006) a tarefa de classificação é realizada em duas etapas que são: aprendizagem e predição. A aprendizagem é a etapa onde o algoritmo de aprendizagem constrói o *modelo de conhecimento* por meio da aprendizagem do conjunto de registros de treinamento com seus rótulos de classes conhecidos. O *modelo de conhecimento* obtido pode ser compreendido como uma função, $y=f(X)$, que permite predizer o rótulo, *y,* de uma classe pré-definida para o atributo-classe de um dado *registro de teste*. Uma vez aprendida essa função, a mesma pode ser aplicada posteriormente a vários registros de testes com rótulos de classe desconhecidos de forma a predizer os rótulos, *y,* de classe. Esta etapa é a chamada *predição de classe* ou *etapa de predição*.

A Figura 4.3 apresenta uma visão geral da tarefa de classificação aplicada para o problema de jogar futebol. O objetivo é prever a classe que determina se os jogadores devem jogar futebol. Neste exemplo, o atributo *Jogar* é denominado de atributo-classe, cujos possíveis valores são "sim" ou "não". Por meio do *modelo de conhecimento* obtido a partir do conjunto de treinamento é possível obter para os registros de testes as classes desconhecidas.

Conjunto de Registros de Treinamento

Tempo	Temperatura	Umidade	Jogar
Sol	30	80	Sim
Sol	26,6	30	Não
Chuva	20	80	Sim
Chuva	16	70	Não
Sol	42	90	Não
Nublado	23	80	Sim
Nublado	9	70	Não



Conjunto de Registros de Testes

Tempo	Temperatura	Umidade	Jogar
Sol	30	80	?
Chuva	20	80	?
Nublado	23	70	?

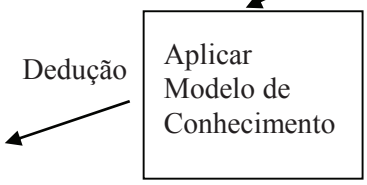


Figura 4.3. Visão Geral da Tarefa de classificação. Adaptado de (Tan et al., 2009)

4.3.2 Introdução a Técnicas de Classificação

Nesta seção será apresentada uma breve introdução de quatro técnicas de classificação de dados: Árvore de Decisão, classificador Bayesiano, Rede Neural Artificial e Máquina de Vetores de Suporte (Tan et al., 2009). Tais técnicas são foram utilizadas em trabalhos como (Silva, 2012; Alves, 2009), e implementadas na abordagem de análise de informações contextuais. De acordo com Tan et al.(2009) uma técnica de classificação é uma abordagem sistemática para criação dos *modelos de conhecimento* ou *modelos de classificação* a partir de um conjunto de registros de entrada. Cada técnica emprega um algoritmo de aprendizagem de máquina específico para criar um *modelo de conhecimento* que possa prevê com maior precisão os rótulos, *y,* de classes dos registros.

4.3.2.1. Indução da Árvore de Decisão

Uma Árvore de Decisão é uma técnica de classificação simples que explora um conjunto de questões e as suas possíveis respostas por meio dos atributos de um registro de teste visando obter o rótulo desconhecido de classe. Conforme ilustrado na Figura 4.4, tais questões e as suas possíveis respostas são organizadas por meio de uma estrutura hierárquica composta por nodos e arestas direcionadas.

A Árvore de Decisão possui nodos terminais ou não terminais. Os nodos terminais ou nodos folha são os que possuem um rótulo de classe, onde cada nodo possui uma aresta que chega e nenhuma que sai. Enquanto, que os nodos não terminais incluem o nodo raiz que é o mais alto na hierarquia da Árvore de Decisão e os demais nodos internos. Cada um desses nodos possui condições de teste sobre os atributos que correspondem às questões e as arestas as possíveis respostas. Desta forma, dado um registro, *X*, de teste para o qual o rótulo, *y,* da classe é desconhecido, os valores dos atributos preditivos do registro são testados contra a árvore de decisão. Na Figura 4.4 um caminho é traçado indicando os resultados das condições de teste sobre os atributos a partir do nodo raiz até um nodo folha, que possui o rótulo, *y,* da classe para este registro de teste.

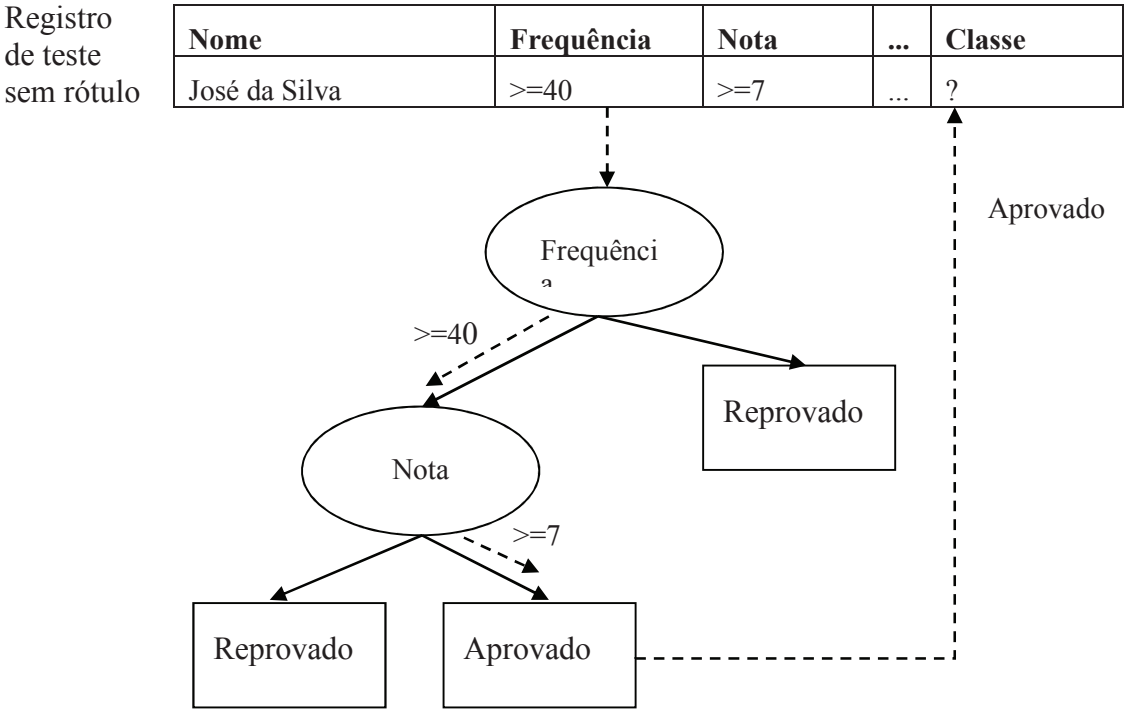


Figura 4.4. Árvore de Decisão classificando um registro de teste sem rótulo. Adaptado de (Tan et al., 2009)

Algoritmos eficientes de Árvore de Decisão têm sido desenvolvidos para tarefa de classificação, e empregados em muitas áreas de aplicação, tais como medicina, análise financeira, astronomia, produção, manufatura, entre outros. O ID3 (*Interactive Dichotomiser*), CART (*Classification and Regression Trees*) e o C4.5 são os algoritmos de árvore de decisão questão entre os mais tradicionais (Han e Kamber, 2006).

4.3.2.2. Classificador Bayesiano

Um classificador Bayesiano é fundamentado em um teorema estatístico conhecido como o teorema de Bayes, que pode ser empregado para resolver o problema da previsão (Tan et al., 2009). Desta forma, é uma técnica de classificação que se destaca por empregar o cálculo das probabilidades para classificar um registro de teste com rótulo, *y,* de classe desconhecido.

Suponha que *X* denote um registro formado por um conjunto de atributos, e *Y* o rótulo de uma determinada classe. Se *X* e *Y* possuem um relacionamento não determinístico, então podem ser tratadas como variáveis randômicas, e calculada sua probabilidade condicional *P(Y|X)* também chamada de probabilidade posterior de *Y*, por meio do emprego da Fórmula 1, conhecida como teorema de Bayes:

$$P(Y \mid X) = \frac{P(X \mid Y) \times P(Y)}{P(X)} \quad (1)$$

O teorema Bayes permite obter a probabilidade condicional *P(Y|X)* por meio da combinação da probabilidade condicional da classe *P(X|Y)*, probabilidade anterior *P(Y),* e a evidência *P(X)*. A probabilidade anterior *P(Y)* é obtida a partir do conjunto de treinamento calculando-se a fração dos registros que pertence a cada classe, já *P(X)* é sempre constante para todas as classes e pode ser ignorada. Com relação à probabilidade condicional de classe *P(X|Y)* podem ser empregadas duas implementações de algoritmos de classificação Bayesianos: o método Bayes simples ou ingênuo e a rede de crenças Bayesiana (Han e Kamber, 2006;Tan et al., 2009).

O processo de aprendizagem consiste em calcular as probabilidades condicionais *P(Y|X)* para cada combinação *X* e *Y* dos registros obtidos do conjunto de treinamento. A partir das probabilidades obtidas, um registro, *X'*, de teste pode ser classificado de acordo com a classe *Y'* que maximiza a probabilidade posterior *P(Y'|X')* (Tan et al., 2009).

Para ilustrar essa técnica, suponha que se deseja prever se uma pessoa que assiste TV deseja assistir um determinado tipo de programa de TV. A Tabela 4.3 apresenta um conjunto de treinamento composto por registros com os seguintes atributos: *Lugar, Dia, Período, Programa de TV* e *Assistir-Programa de TV*. Os programas de TV que devem ser de assistidos pelo usuário são classificados como *Sim*, caso contrário são classificados como *Não*.

Tabela 4.3. Conjunto de treinamento de programas de TV.

Lugar	Dia	Período	Programa de TV	Assistir-Programa de TV
Casa	Domingo	Noite	Filme	Sim
Escritório	Segunda	Manhã	Desenho	Não
Casa	Sábado	Tarde	Desenho	Sim
Carro	Sábado	Noite	Filme	Não
Escritório	Segunda	Manhã	Economia	Sim

Para classificar um registro, *X*, de teste composto pelos seguintes atributos: *X*=(Lugar=Casa, Dia = Domingo, Período = Manhã, Programa de TV= Esportes), deve-se, calcular as probabilidades posteriores *P(Sim|X)* e *P(Não|X)* a partir dos dados dos registros do conjunto de treinamento. Se a probabilidade posterior *P(Sim|X)*> *P(Não|X)*, então o registro de teste é classificado como *Sim*, senão é classificado como *Não*.

4.3.2.3. Rede Neural Artificial

Redes Neurais Artificiais (RNAs) são algoritmos inspirados nos princípios de funcionamento dos neurônios biológicos para realização de tarefas tais como classificação, previsão de séries temporais, reconhecimento de padrões, entre outras (Tan et al., 2009). O modelo básico de uma rede neural artificial ilustrado na Figura 4.5 é conhecido como *perceptron,* e foi proposto por McCulloch e Pitts (1943).

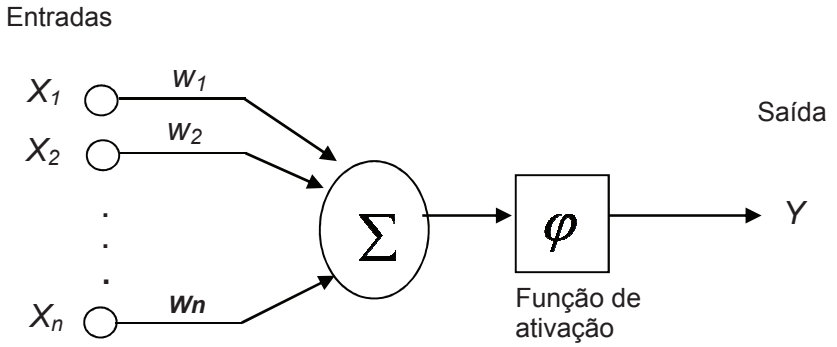


Figura 4.5. Modelo Básico de um perceptron (Han e Kamber, 2006)

Um *perceptron* é composto pelos os seguintes elementos: (i) um conjunto de n unidades x_1, x_2, ..., x_n, que representam os atributos de entrada de um registro com os seus pesos multiplicativos correspondentes (w_n); (ii) uma unidade somadora $\sum$ para acumular o produto de cada entrada com o peso multiplicativo; (iii) uma função de ativação φ que determina a saída efetiva Y do neurônio.

Para ilustrar o funcionamento de um *perceptron*, suponha que as suas entradas x_1, x_2, ..., x_n recebam valores booleanos (0 ou 1), que os pesos multiplicativos w_1, w_2, ..., w_n possuem valores reais e que tenha um fator t de tendência. O *percepron* poderá obter o valor de saída, Y, por meio uma função de ativação, que subtraia o valor obtido da unidade somadora $\sum$ por um fator t de tendência e então analisa o valor do resultado. Este modelo pode ser expressado matematicamente da seguinte forma:

$$Y = 1, \text{ se } w_1x_1 + w_2x_2 + \ldots w_nx_n - t > 0; \qquad (2)$$
$$Y = -1, \text{ se } w_1x_1 + w_2x_2 + \ldots w_nx_n - t < 0.$$

Na literatura é possível encontrar propostas de funções de ativação tais como lineares, tangentes hiperbólicas, sigmóides , dentre outras (Han e Kamber, 2006;Tan et al., 2009). Com relação ao processo de aprendizado de um *perceptron,* consiste em ajustar os pesos multiplicativos por meio de algoritmos específicos até que as saídas obtidas sejam consistentes aos rótulos de classes dos registros de treinamento (Tan et al., 2009).

Conforme ilustrado na Figura 4.6 por meio da combinação de *perceptrons* é possível forma uma estrutura mais complexa de rede neural artificial. A arquitetura de uma rede neural pode ser baseada em uma ou múltiplas camadas. Por exemplo, uma rede neural com três camadas (*Perceptron Multicamadas*) é composta por uma camada de entrada, em quem que os neurônios recebem os estímulos (ou sinais); a camada intermediária, onde é realizado o processamento; e a camada de saída, que apresenta o resultado final. Na literatura é possível encontrar diferentes propostas de algoritmos voltados para rede neural artificial como o *backpropagation* (Han e Kamber, 2006).

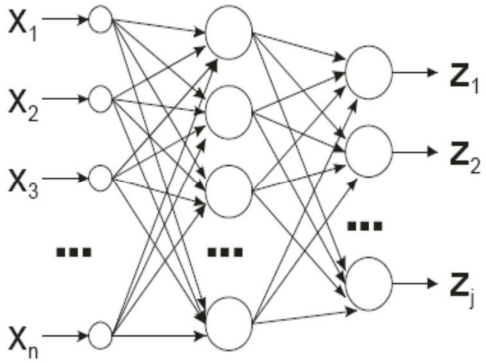


Figura 4.6. Rede Neural Artificial Perceptron Multicamadas

4.3.2.4. Máquina de Vetores de Suporte

Máquina de Vetores de Suporte (SVM, do inglês, *Support Vector Machine*) é uma técnica de classificação que é embasada na Teoria da Aprendizagem Estatística (TAE), desenvolvida por Vapnik (1995). Esta técnica tem recebido cada vez mais atenção por parte de pesquisadores por apresentar resultados promissores em muitas aplicações como reconhecimento de padrão e classificação de textos (Tan et al., 2009). Uma Máquina de Vetores de Suporte tem como base o emprego de condições matemáticas oriundas da TAE que permitem classificar um conjunto de dados em duas classes diferentes. A fronteira entre as duas classes de dados é denominada de limite de decisão ou fronteira de decisão, que pode ser linear ou não linear, e determina o tipo de Máquina de Vetores de Suporte.

Uma classificação de dados por meio de uma Máquina de Vetores de Suporte do tipo linear consiste na aprendizagem de uma função ou classificador baseado em um hiperplano ou uma representação espacial para dividir linearmente um conjunto de dados em duas classes conforme ilustrado na Figura 4.7 (a). O classificador é obtido por meio de um conjunto de treinamento durante a etapa de aprendizado. Na etapa de predição, dado um conjunto de dados de testes em que os dados podem assumir os valores (+,-) o classificador separa estes dados de modo que um dos lados do hiperplano contenha somente dados da classe (+), e outro lado dados da classe (-).

A classificação de um conjunto de dados com dados não separáveis linearmente, como ilustra a Figura 4.7 (b) deve ser realizada por meio de uma Máquina de Vetores de Suporte do tipo não linear. Para isso, a Máquina de Vetores de Suporte é construída a partir da transformação do conjunto de dados inicial em um novo conjunto de dados linearmente separáveis por meio de uma série de princípios do teorema de Cover da separabilidade de padrões (Han e Kamber, 2006). Com relação aos algoritmos de Máquina de Vetores de Suporte, o algoritmo de Otimização Mínima Sequencial (SMO, do inglês, *Sequential Minimal Optimization*) é um dos que estão entre os mais tradicionais (Platt, 1998).

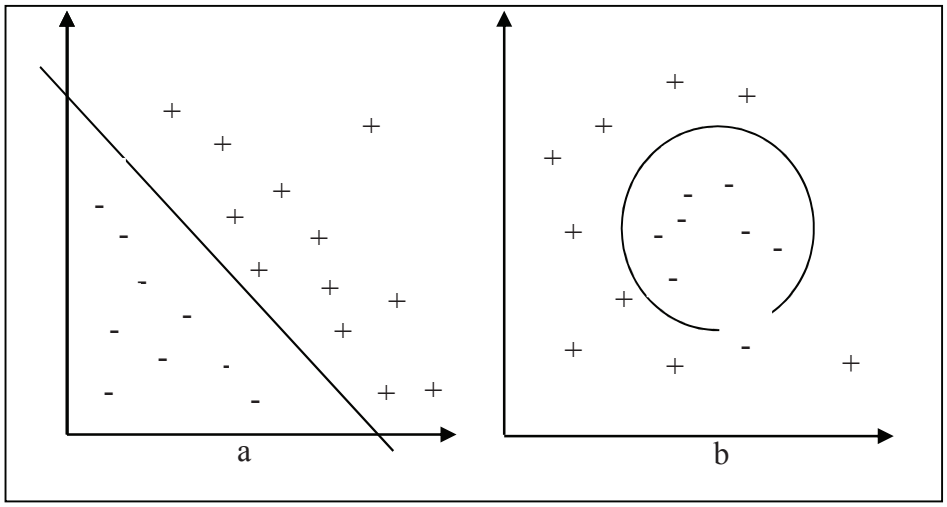


Figura 4.7. (a) Classificação de dados com limite de decisão linear; (b) Classificação de dados com limite de decisão não linear

4.4. Abordagem Geral para a Análise de Informações Contextuais

A abordagem de análise apresentada nesta seção, tem como base o emprego da tarefa de classificação por meio de etapa de aprendizagem supervisionada para criação de *modelos de conhecimento*, posteriormente utilizados na etapa de predição de rótulos de classes. Desta forma, são utilizadas as técnicas de classificação da área de aprendizagem de máquina, e o conceito de *registro contextual,* que será apresentado na subseção a seguir.

4.4.1. Registro Contextual

Um *registro contextual* é definido, neste trabalho, como sendo o registro a ser utilizado nas tarefas de aprendizagem supervisionada e predição, sendo formado por uma combinação de atributos preditivos e de classe, ou seja, o atributo alvo com ou sem o rótulo de classes pré-definidas que correspondem as dimensões semânticas 5Ws+1H.

Cada classe pré-definida pode se referir a um determinado tipo de conteúdo, uma atividade do usuário, um tipo de necessidade, um nível de satisfação de um serviço, uma avaliação de um produto etc. Por exemplo, em um sistema de recomendação de conteúdo para TV Digital, uma classe pode corresponder a um gênero de programa de TV que será recomendado para o usuário. Um rótulo de classe pré-definida, pode ser atribuído ao atributo-classe pelo usuário ou inferida para o mesmo dependendo da etapa de classificação que esteja sendo realizada com o *registro contextual.* Assim, um *registro contextual* pode conter ou não um rótulo de classe no atributo-classe.

No que se refere à obtenção do *registro contextual* pode ser empregada a técnica realimentação de relevância (do ingles, *feedback relevance*) que tem como base a interação explícita e implícita do usuário com o sistema computacional (Blanco-Fernandez, 2007). Toda vez que o usuário interage com o sistema, como assistir ou avaliar um programa de TV, acessar informações sobre um produto, ou solicitar recomendações de conteúdo, um *registro contextual* poderá ser gerado.

A Tabela 4.4 apresenta alguns atributos preditivos e de classe que podem ser combinados para compor um *registro contextual*, destacando a dimensão semântica contextual correspondente e possíveis valores dos mesmos. Torna-se importante destacar que de acordo com os requisitos do sistema computacional, tais atributos podem ser complementados ou até mesmo substituídos por outros de maior expressividade. Por questões de privacidade, não aparece um atributo preditivo de identificação do usuário (ID do usuário) referente à dimensão *Who* no *registro contextual,* pois esta informação é opcional no processo de predição.

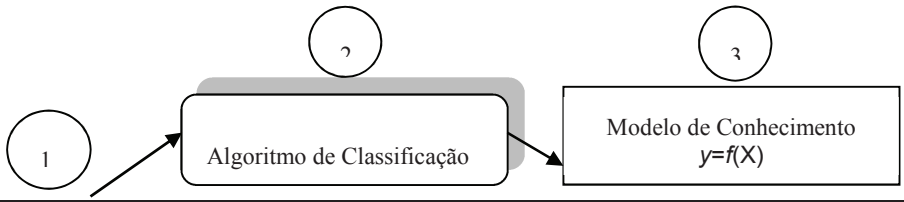
Tabela 4.4. Exemplos de atributos preditivos e atributos-classe que podem ser combinados para compor um registro contextual.

Dimensão Semântica	Atributos	Valores candidatos
Where	**Preditivos**	Casa, escola, shopping, escritório, ônibus, restaurante
	Lugar	
	Zona	Norte, sul, leste, oeste, centro
	Estado	AM, SP, RJ, RR, MG, RS
When	Dia	Segunda, terça, quarta, quinta, sexta, sábado, domingo
	Período	Manhã, tarde, noite, madrugada
	Tipo do dia	Fim de semana, dia de semana, feriado
How	Dispositivo	Tablet, smartphone, TV, set-top box, notebook, PC
	Tipo de dispositivo	Fixo, móvel, portátil
	Aplicação	Web, móvel, desktop
What	**Atributos-classe**	Policial, musical, ação, romântico, comédia, ficção
	Gênero	
	Necessidade	Hedônica, utilitária
	Conteúdo	Esporte, notícias, entretenimento, musica, educação, compras

4.4.2. Aprendizagem Supervisionada baseada em Contexto

Esta é a etapa onde o algoritmo de classificação constrói o *modelo de conhecimento* por meio da análise do histórico de contexto composto por *registros contextuais* dos usuários com o atributo-classe instanciado, ou seja *registros contextuais* pré-classificados. No processo da classificação os *registros contextuais* dos usuários podem ser referenciados como tuplas, instâncias, exemplos, ou casos de testes. Uma tupla, *X*, pode ser representada por meio de vetor de atributos n-dimensional, $X=(x_1, x_2, x_3,..., x_n)$. A Figura 4.7 ilustra o processo de aprendizagem supervisionada baseada em contexto.

O histórico de contexto (1) é obtido por meio da técnica de realimentação de relevância, e posteriormente analisado pelo algoritmo de classificação (2) para criação do *modelo de conhecimento* também conhecido como modelo de classificação (Tan et al., 2009). Vale ressaltar que o *modelo de conhecimento* (3) é construído de acordo com a técnica de classificação empregada, e que pode ser compreendido como uma função, $y=f(X)$, que permite predizer o rótulo, *y,* de classe do atributo-classe de um dado *registro, X, contextual* de um usuário.



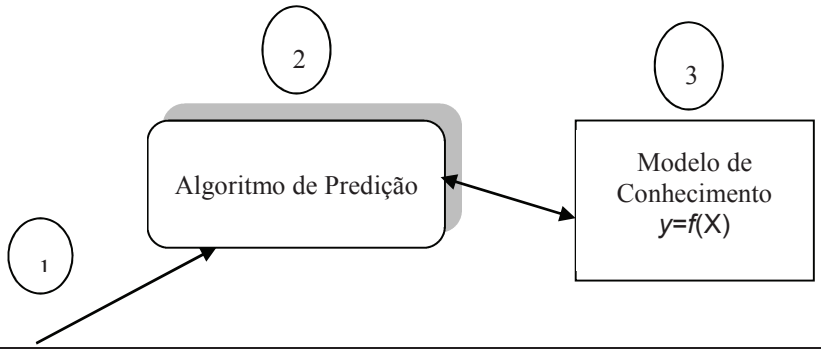
Registros Contextuais (Histórico de Contexto)				
Atributos Preditivos				**Atributo-classe**
Lugar (*Where*)	Dia (*When*)	Período (*When*)	Dispositivo (*How*)	Conteúdo (*What*)
Casa	Domingo	Noite	TV	Entretenimento
Escritório	Segunda	Manhã	Notebook	Economia
Casa	Sábado	Tarde	TV	Esporte
Carro	Sábado	Noite	Navegador GPS	Música
Shopping	Domingo	Tarde	Smartphone	Compras

Figura 4.7. Etapa de Aprendizagem Supervisionada. Adaptado de (Han e Kamber, 2006)

4.4.3. Predição de Classes Contextuais

Conforme ilustrado na Figura 4.8 esta é a segunda etapa da abordagem, onde dado um *registro, X, contextual* com um atributo-classe, não instanciado (1), ou seja, um registro contextual sem o rótulo, *y,* de classe, deseja-se predizer um rótulo, *y,* de classe para o atributo-classe deste registro. Vale lembrar que um atributo-classe pode ser um tipo de conteúdo, um gênero, uma necessidade, dentre outros. Para que a predição ocorra, é necessário que o algoritmo de predição (2) utilize o *modelo de conhecimento* $y=f(X)$ (3) obtido na etapa anterior, que permite predizer o rótulo, *y,* de classe do atributo-classe de um dado *registro, X, contextual de teste.*

O *registro, X, contextual de teste* é obtido automaticamente partir de uma ação explícita ou implícita do usuário. Por exemplo, ao solicitar uma recomendação de conteúdo, ou ao acessar uma lista de produtos. Assim, a partir do rótulo, *y,* de classe que fora obtido se torna possível a sua utilização posterior na fase de disseminação de contexto como parâmetro de entrada para serviços como personalização e adaptação de conteúdo, filtragem de informação, entre outros.



Registro Contextual de Teste				
Atributos Preditivos				**Atributo-classe**
Lugar (*Where*)	Dia (*When*)	Período (*When*)	Dispositivo (*How*)	Conteúdo (*What*)
Casa	Domingo	Noite	TV	Classe y?

Figura 4.8. Etapa de Predição. Adaptado de (Han e Kamber, 2006)

4.5. Implementação da Abordagem para Análise de Informações Contextuais

Nesta seção propõe-se apresentar como implementar a abordagem de análise de informações contextuais sobre um histórico de contexto ou base de dados. O objetivo é criar um *modelo de conhecimento* que classifique o conteúdo de acordo com o contexto do usuário. Por meio do rótulo de classe inferido será possível posteriormente obter uma lista de conteúdos correspondentes a partir de técnicas de filtragem de informação como a técnica de filtragem baseada em conteúdo (Adomavicius e Tuzhilin, 2005), implementada em sistemas de recomendação.

Para a implementação das técnicas de classificação (Árvore de Decisão, classificador Bayesiano, Rede Neural Artificial e Máquina de Vetores de Suporte) exploradas pela abordagem apresentada pode ser empregado o modo programado da ferramenta Weka desenvolvida na Universidade de Waikato na Nova Zelândia (Hall et al., 2009). Desta forma, deve-se utilizar a API Weka que contempla uma série de técnicas de aprendizagem de máquina implementadas em Java.

4.5.1. O Ambiente de Desenvolvimento

Para programar aplicações baseadas em Weka, deve-se ter instalado a plataforma de programação Java (J2SE, *Java 2 Standard Edition*). Assim, é necessário instalar o *J2SE Development Kit* (J2SE, 2012), e um ambiente de desenvolvimento integrado (do inglês, IDE *Integrated Development Environment*) como o Eclipse (Eclipse, 2012) usado na programação das aplicações.

Além disso, para que o Weka seja utilizado nas aplicações, deve-se adicionar a API Weka no projeto da aplicação, por meio da *tab libraries* da caixa de dialogo *Java Settings*. Deve ser usado o botão *Add External JARs* para adicionar o arquivo *weka.jar*. Todos os exemplos apresentados neste trabalho foram implementados por meio de tais ferramentas.

4.5.2. Criação da Base de Dados para Tarefa de Classificação

A Weka utiliza um arquivo no formato ARFF (*Attribute-Relation File Format*) que contém um conjunto de registros que são utilizados na tarefa de classificação pelas técnicas de aprendizagem de máquina implementadas na API Weka. No caso desta abordagem o conjunto de registros ou base de dados corresponde ao histórico de contexto (Figura 4.8) composto por *registros contextuais* com atributos preditivos e atributos-classe instanciados, ou seja, com rótulos de classes.

Tais registros são tratados pela API do Weka como instâncias. Na Figura 4.9 é apresentado um trecho de um arquivo ARFF. Vale ressaltar que API Weka também fornece recursos que permitem que as instâncias sejam obtidas a partir de um sistema de gerenciamento de banco de dados. Para isso, algumas configurações devem ser feitas no arquivo DatabaseUtils.props para definir a conexão com o banco de dados. Informações detalhadas sobre o acesso por meio de banco de dados podem ser encontradas em (Weka, 2012).

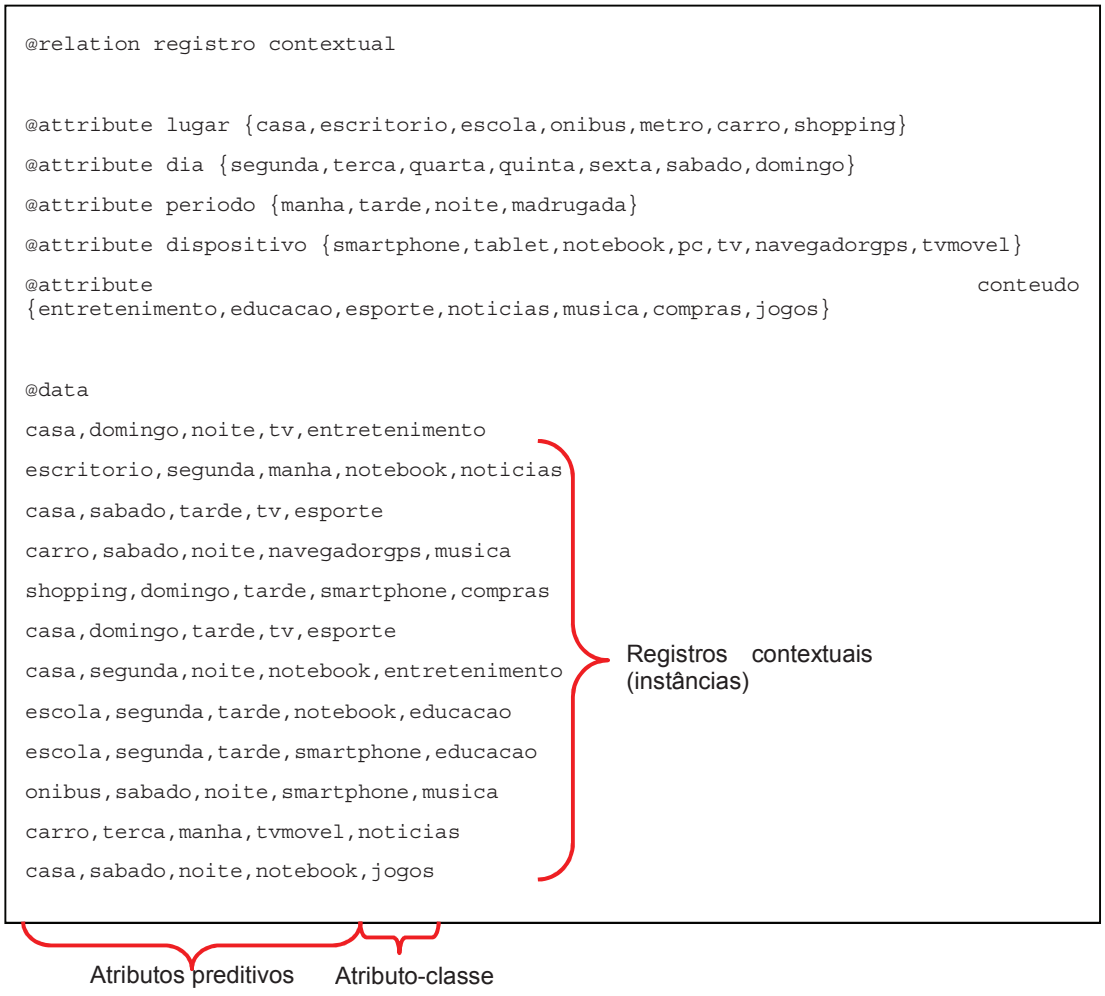


Figura 4.9. Arquivo ARFF com alguns registros contextuais

O arquivo no formato ARFF é um arquivo de texto dividido em três partes:

- Relação (*@relation*), se refere ao termo utilizado para identificar a relação de registros;
- Atributos (*@Attribute*), correspondem aos atributos preditivos que compõe um registro. Para cada atributo é definido seu nome e os seus possíveis valores. Geralmente, o último atributo é o atributo-classe com seus rótulos de classes pré-definidas;
- Dados *(@data),* cada linha corresponde a uma instância, ou seja um *registro contextual*, com os valores separados por virgula e seguindo a ordem dos atributos.

A seguir serão descritas as implementações das principais etapas da abordagem de análise de informações contextuais por meio de exemplos práticos das técnicas de classificação: Árvore de Decisão, classificador Bayesiano e Rede Neural e SVM.

4.5.3. Obter Instâncias de Registros Contextuais

Esta tarefa consiste em recuperar e converter os registros do arquivo ARFF para um formato mais apropriado visando uso posterior nas etapas de aprendizagem supervisionada e predição. Com isso, os registros do arquivo ARFF são carregados e convertidos como instâncias (ou vetores de atributos), e o atributo-classe também é definido. O trecho de código a seguir apresentado na Figura 4.10 ilustra a implementação de Java que utiliza classes da API Weka.

```
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

(...)

public Instances getInstanciasRegistrosContextuais(){
                Instances instancias = null;
                try
                {
                        DataSource  fonteDeDados = new DataSource(ARFF);
                        instancias = fonteDeDados.getDataSet();

instancias.setClassIndex(instancias.numAttributes() - 1);
                        System.out.println(instancias);
                }
                catch(Exception e)
                {
                        System.out.println("O arquivo arff "+ARFF+" nao
existe...");
                        e.printStackTrace();
                }
```

Figura 4.10. Exemplo de código para obter instâncias de registros do ARFF

A API Weka fornece, por meio do pacote *weka.core.converters*, um conjunto de classes para pré-processamento de registros do arquivo ARFF. No exemplo, *fonteDeDados* é uma variável de instância da classe *weka.core.converters.ConverterUtils.DataSource*. Por meio de *fonteDeDados* é possível chamar o método *getDataSet()* para obter do arquivo ARFF um conjunto de registros tratados como instâncias da classe *weka.core.Instances*. A localização do arquivo ARFF

é passado como argumento para o contrutor *DataSource()*. Por fim, é definido o atributo-classe das instâncias por meio do método *setClassIdex()* que recebe como argumento o índice que identifica qual será o atributo-classe usado na aprendizagem e predição. Por padrão no arquivo ARFF o atributo-clase é o ultimo atributo, o que explica o motivo para o índice está definido como *numAttributes-1*.

4.5.4. Criação da Instância do Registro Contextual de Teste

A API Weka trata o registro de teste como um vetor de atributos preditivos sem o atributo-classe instanciado, ou seja uma instância sem o rótulo, *y*, de classe. Assim, torna-se necessária a criação de uma instância correspondente ao *registro contextual de teste* que será utilizado posteriormente na etapa de predição. Para isso, deve-se utilizar o pacote *weka.core,* pois fornece a classe *weka.core.Instance* responsável pela codificação de uma instância. O trecho de código a seguir apresentado na Figura 4.11 ilustra a criação de uma instância a partir de atributos preditivos e atributo-classe de um *registro contextual* de teste.

No exemplo, a classe *weka.core.Instance* é instanciada por meio do construtor *Instance()* que recebe como argumento a quantidade de atributos que a instância deve ter, incluindo o atributo-classe da classe. No exemplo, um objeto do tipo *weka.core.Instance* é referenciado pela variável *instancia*. Por meio de *instancia* é possível chamar o método *setDataSet()* para definir o conjunto de instâncias (obtido na etapa de pré-processamento.) que a *instancia* pertence. Em seguida, por meio do método *setValue()* é possível definir cada atributo preditivo que pertence ao registro de teste. Com isso, uma instância será criada posteriormente utilizada na tarefa de predição conforme será descrito a seguir.

```
import weka.core.Instance;

(...)

public Instance getInstanciaDeTeste(String lugar, String dia, String
periodo, String dispositivo,
                    Instances instancias){
                  Instance instancia = new Instance(5);
                  instancia.setDataset(instancias);
                  instancia.setValue(0, lugar);
                  instancia.setValue(1, dia);
                  instancia.setValue(2, periodo);
                  instancia.setValue(3, dispositivo);
                  System.out.println("A instancia de TESTE e -->
"+instancia);
              return instancia;
       }
```

Figura 4.11. Criação da Instância de um Registro Contextual de Teste

4.5.5. Aprendizagem Supervisionada baseada em Árvore de Decisão

O trecho de código a seguir apresentado na Figura 4.12 ilustra a implementação da etapa de aprendizagem por meio da técnica Árvore de Decisão. Para isso, deve-se utilizar o pacote *weka.classifiers.trees* que fornece classes que implementam diversos algoritmos de Árvore de Decisão.

```
import weka.classifiers.trees.J48;

(...)

public void arvoreDeDecisao_Aprendizagem(Instances
instancias){
                                try
                {
                                j48 = new J48();

j48.buildClassifier(instancias);
                }
                catch(Exception e)
                {
                                e.printStackTrace();
                }
        }
```

Figura 4.12. Implementação da Árvore de Decisão

No exemplo, é utilizado o algoritmo C4.5 (Witten e Franck, 2005; Tan et al., 2009) por meio da implementação J48 disponível na API Weka. Assim, *j48* é uma instância da classe *weka.classifiers.trees.j48*. Por meio do objeto *j48* é possível chamar o método *buildClassifier()* para construir um *modelo de conhecimento* (baseado em Árvore de Decisão) usado na classificação a partir do conjunto de instâncias recebidas como argumento.

4.5.6. Predição baseada em Árvore de Decisão

Para implementação da tarefa de predição baseada em Árvore de Decisão deve-se utilizar além do pacote *weka.classifiers.trees*, também o pacote *weka.core* , pois fornece a classe *weka.core.Instance* responsável pela codificação de uma instância (*registro contextual*), ou seja, um vetor de atributos, e a classe *weka.core.Attribute* responsável pela codificação de um atributo. O trecho de código a seguir apresentado na Figura 4.13 ilustra a implementação da tarefa de predição por meio da técnica Árvore de Decisão.

```
import weka.core.Instance;
import weka.core.Attribute;
import weka.classifiers.trees.J48;

(...)

public String arvoreDeDecisao_Predicao(Instance instancia){

                String classeInferida = null;
                try {
                                int indice =
Double.valueOf(j48.classifyInstance(instancia)).intValue();
                                Attribute atributoClasse = instancia.classAttribute();
                                classeInferida = atributoClasse.value(indice);

                } catch (Exception e) {
                                e.printStackTrace();
                }
                return classeInferida;
        }
```

Figura 4.13. Implementação da Predição baseada em Árvore de Decisão

No exemplo, por meio do objeto *j48* é possível chamar o método *classifyInstace()* usado para classificar uma instância com rótulo, *y*, de classe desconhecido, ou seja, obter uma classe para um *registro contextual de teste*. Para isso, o método *classifyInstace()* recebe como argumento a instância que deve ser classificada, e retorna um índice que posteriormente será utilizado para obter o rótulo, *y*, de classe inferido. O objeto *atributoClasse,* cujo o tipo é *weka.core.Attribute* se refere ao atributo-classe obtido via o método *classAttribute()* da instância de teste. Por meio do método *value()* do objeto *atributoClasse* é possível obter o rótulo, *y*, de classe inferido.

4.5.7. Aprendizagem Supervisionada baseada em um Classificador Bayesiano

O trecho de código a seguir apresentado na Figura 4.14 ilustra a implementação da tarefa de aprendizagem por meio de um classificador Bayesiano. Para isso, deve-se utilizar o pacote *weka.classifiers.bayes* que fornece classes que implementam diversos algoritmos de classificadores Bayesianos.

No exemplo, é utilizada uma implementação do algoritmo Naive Bayes (Han e Kamber, 2006;Tan et al., 2009). Assim, *naiveBayes* é uma instância da classe *weka.classifiers.bayes.NaiveBayes*. Por meio de *naiveBayes* é possível chamar o método *buildClassifier()* para construir um *modelo de conhecimento* (baseado no teorema de Bayes) usado na classificação a partir do conjunto de instâncias recebidas como argumento.

```
import weka.classifiers.bayes.NaiveBayes;

(...)

public void classificadorBayesiano_Aprendizagem(Instances
instancias){
                                try
                {
                        naiveByes = new NaiveBayes();
                        naiveByes.buildClassifier(instancias);
                }
                catch(Exception e)
                {
                       e.printStackTrace();
                }
        }
```

Figura 4.14. Implementação do Classificador Bayesiano

4.5.8. Predição baseada em um Classificador Bayesiano

Para implementação da tarefa de predição baseada em Naive Bayes deve-se utilizar além do pacote *weka.classifiers.bayes,* também o pacote *weka.core,* pois fornece a classe *weka.core.Instance* responsável pela codificação de uma instância (*registro contextual*), ou seja, um vetor de atributos, e a classe *weka.core.Attribute* responsável pela codificação de um atributo. O trecho de código a seguir apresentado na Figura 4.15 ilustra a implementação da tarefa de predição por meio do algoritmo Naive Bayes.

```
import weka.core.Instance;
import weka.core.Attribute;
import weka.classifiers.bayes.NaiveBayes;

(...)

public String classificadorBayesiano_Predicao(Instance instancia){

String classeInferida = null;
       try {
       int indice =
Double.valueOf(naiveBayes.classifyInstance(instancia)).intValue();
       Attribute atributoClasse = instancia.classAttribute();
       classeInferida = atributoClasse.value(indice);
       } catch (Exception e) {
                     e.printStackTrace();
       }
                           return classeInferida;
       }
```

Figura 4.15. Implementação da Predição baseada em Naive Bayes

No exemplo, por meio do objeto *naiveBayes* é possível chamar o método *classifyInstace()*usado para classificar uma instância com rótulo de classe desconhecido, ou seja, obter um rótulo, *y*, de classe para um *registro contextual de teste*. Para isso, o método *classifyInstace()* recebe como argumento a instância que deve ser classificada, e retorna um índice que posteriormente será utilizado para obter o rótulo, *y*, de classe inferido. O objeto *atributoClasse,* cujo o tipo é *weka.core.Attribute* se refere ao atributo-classe obtido via o método *classAttribute()* da instância de teste. Por meio do método *value()* do objeto *atributoClasse* é possível obter o rótulo, *y*, de classe inferido.

4.5.9. Aprendizagem Supervisionada baseada em Rede Neural Artificial

O trecho de código a seguir apresentado na Figura 4.16 ilustra a implementação da tarefa de aprendizagem por meio de uma Rede Neural Artificial. Para isso, deve-se utilizar o pacote *weka.classifiers.functions* que fornece uma miscelânea de classes que implementam diversas técnicas de aprendizagem, como Multlayer Perceptron (Han e Kamber, 2006;Tan et al., 2009)

```
import weka.classifiers.functions.MultilayerPerceptron;

(...)

public void redeNeural_Aprendizagem(Instances instancias){

   try {
                        multilayerPerceptron = new
MultilayerPerceptron();

multilayerPerceptron.buildClassifier(instancias);
                }
                catch(Exception e)
                {
                           e.printStackTrace();
                }
```

Figura 4.16. Implementação da Rede Neural Artificial

No exemplo, é utilizada a implementação do algoritmo Multlayer Perceptron da API Weka. Assim, *multlayer Perceptron* é uma instância da classe *weka.classifiers.functions.MultlayerPerceptron*. Por meio de *multlayerPerceptron* é possível chamar o método *buildClassifier()* para construir um *modelo de conhecimento* (baseado em Rede Neural Artificial Multlayer Perceptron) usado na classificação a partir do conjunto de instâncias recebidas como argumento.

4.5.10. Predição baseada em Rede Neural Artificial

Para implementação da tarefa de predição baseada em rede neural artificial deve-se utilizar além do pacote *weka.classifiers.functions,* também o pacote *weka.core*, pois fornece a classe *weka.core.Instance* responsável pela codificação de uma instância (*registro contextual*), ou seja, um vetor de atributos, e a classe *weka.core.Attribute* responsável pela codificação de um atributo. O trecho de código a seguir apresentado na Figura 4.17 ilustra a implementação da tarefa de predição por meio da técnica rede neural artificial.

No exemplo, por meio do objeto *multilayerPerceptron* é possível chamar o método *classifyInstace()*usado para classificar uma instância com rótulo de classe desconhecido, ou seja, obter um rótulo, *y*, de classe para um *registro contextual de teste*. Para isso, o método *classifyInstace()* recebe como argumento a instância que deve ser classificada, e retorna um índice que posteriormente será utilizado para obter o rótulo, *y*, de classe inferido. O objeto *atributoClasse,* cujo o tipo é *weka.core.Attribute* se refere ao atributo-classe obtido via o método *classAttribute()* da instância de teste. Por meio do método *value()* do objeto *atributoClasse* é possível obter o rótulo, *y*, de classe inferido.

```
import weka.core.Instance;
import weka.core.Attribute;
import weka.classifiers.functions.MultilayerPerceptron;

(...)

public String redeNeural_Predicao(Instance instancia){

              String classeInferida = null;
              try {
                     int indice =
Double.valueOf(multilayerPerceptron.classifyInstance(instancia)).intValue();
                     Attribute atributoClasse = instancia.classAttribute();
                     classeInferida = atributoClasse.value(indice);
              } catch (Exception e) {
                     e.printStackTrace();
              }
              return classeInferida;
       }
```

Figura 4.17. Implementação da Predição baseada em Rede Neural Artificial

4.5.11. Aprendizagem Supervisionada baseada em Máquina de Vetores de Suporte

O trecho de código a seguir apresentado na Figura 4.18 ilustra a implementação da tarefa de aprendizagem por meio de uma Máquina de Vetores de Suporte. Para isso, deve-se utilizar o pacote *weka.classifiers.functions* que fornece uma miscelânea de

classes que implementam diversos algoritmos de aprendizagem, como o algoritmo de Otimização Mínima Sequencial (SMO) (Platt, 1998).

```
import weka.classifiers.functions.SMO;

(...)

public void svm_Aprendizagem(Instances instancias) {

            try {
                  smo = new SMO();
                  smo.buildClassifier(instancias);
            } catch (Exception e) {
                                    e.printStackTrace();
            }

}
```

Figura 4.18. Implementação de uma Máquina de Vetores de Suporte

No exemplo, é utilizada a implementação do algoritmo SMO da API Weka. Assim, *smo* é uma instância da classe *weka.classifiers.functions.SMO*. Por meio de *smo* é possível chamar o método *buildClassifier()* para construir um *modelo de conhecimento* (baseado em Máquina de Vetores de Suporte) usado na classificação a partir do conjunto de instâncias recebidas como argumento.

4.5.12. Predição baseada em Máquina de Vetores de Suporte

Para implementação da tarefa de predição baseada em Máquina de Vetores de Suporte deve-se utilizar além do pacote *weka.classifiers.functions,* também o pacote *weka.core*, pois fornece a classe *weka.core.Instance* responsável pela codificação de uma instância (*registro contextual*), e a classe *weka.core.Attribute* responsável pela codificação de um atributo. O trecho de código a seguir apresentado na Figura 4.19 ilustra a implementação da tarefa de predição por meio do algoritmo SMO da API Weka.

No exemplo, por meio do objeto *smo* é possível chamar o método *classifyInstace()*usado para classificar uma instância com rótulo de classe desconhecido, ou seja, obter um rótulo, *y*, de classe para um *registro contextual de teste*. Para isso, o método *classifyInstace()* recebe como argumento a instância que deve ser classificada, e retorna um índice que posteriormente será utilizado para obter o rótulo, *y*, de classe inferido. O objeto *atributoClasse,* cujo o tipo é *weka.core.Attribute* se refere ao atributo-classe obtido via o método *classAttribute()* da instância de teste. Por meio do método *value()* do objeto *atributoClasse* é possível obter o rótulo, *y*, de classe inferido.

```
import weka.core.Instance;
import weka.core.Attribute;
import weka.classifiers.functions.SMO;

(...)

public String svm_Predicao(Instance instancia){

            String classeInferida = null;
            try {
                  int indice =
Double.valueOf(smo.classifyInstance(instancia)).intValue();
                  Attribute atributoClasse =
instancia.classAttribute();
                  classeInferida = atributoClasse.value(indice);
            } catch (Exception e) {
                  e.printStackTrace();
            }
            return classeInferida;
      }
```

Figura 4.19. Implementação da Predição baseada em Máquina de Vetores de Suporte

4.5.13. Disseminação do Rótulo de Classe Inferido

A disseminação pode ser implementada de acordo com o tipo do serviço oferecido pelo sistema sensível ao contexto. Deste modo, o rótulo de classe inferido pode ser empregado, por exemplo, como parâmetro para serviços de adaptação e personalização de conteúdo, para modificar o comportamento de um sistema ou prover recomendação de conteúdo por meio de técnicas de adaptação ou filtragem de informação da área de Recuperação de Informação como a técnica de filtragem baseada em conteúdo (Adomavicius e Tuzhilin, 2005), implementada em sistemas de recomendação.

Em cenários como Web, TV Digital Interativa, e Computação Ubíqua, a exploração do contexto pode enriquecer a experiência de interação do usuário, principalmente quando empregado em aplicações interativas para prover conteúdos e/ou informações de forma personalizada (Silva, 2012; Alves, 2009, Golularte, 2003). A Figura 4.20 apresenta a implementação do método *main* da classe *AnalisadorDeContexto* responsável gerenciamento das etapas de aprendizagem e predição.

```
import weka.classifiers.bayes.NaiveBayes;
import weka.classifiers.trees.J48;
import weka.classifiers.functions.MultilayerPerceptron;
import weka.core.Attribute;
import weka.core.Instance;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

public class AnalisadorDeContexto {

(...)

public static void main(String[] args) {

AnalisadorDeContexto analisadorDeContexto = new
AnalisadorDeContexto();

            Instances instancias =
analisadorDeContexto.getInstanciasRegistrosContextuais();


      analisadorDeContexto.setAprendizagemSupervisionada(NAIVEBAYE
S, instancias);
            Instance instancia =
analisadorDeContexto.getInstanciaDeTeste("casa","domingo","noite",
"tv", instancias);
            System.out.println("Classe inferida --->
"+analisadorDeContexto.getPredicao(NAIVEBAYES,instancia));
                        }
\
```

Figura 4.20. Implementação do método main da classe AnalisadorDeContexto

No exemplo, o método *main* executa sistematicamente os métodos responsáveis pela implementação das etapas de aprendizagem e predição. O método *getInstanciasRegistrosContextuais()* recupera e trata como instâncias os registros armazenados no arquivo ARFF. Em seguida, o método *setAprendizagemSupervisionada()* gera o modelo de conhecimento a partir das instâncias e da técnica de aprendizagem passadas como parâmetros.

Finalmente, a etapa de predição é executada por meio do método *getPredicao()* que retorna o rótulo de classe inferido. Para isso, o método *getPredicao()* recebe como parâmetros a instância correspondente ao *registro contextual de teste* e a técnica que será utilizada na tarefa de predição. A instância do *registro contextual de teste* é obtida por meio do método *getInstanciaDeTeste ()* que recebe como parâmetros as informações contextuais correspondentes aos atributos preditivos. Por fim, o analisador de contexto de posse do rótulo de classe obtido pode passá-lo como parâmetro para um método que implementa, por exemplo, uma filtragem de conteúdo.

4.6. Considerações Finais

A Computação Sensível ao Contexto é uma área de pesquisa que vem despertando cada vez mais a atenção e interesse de pesquisadores. O objetivo desta área é desenvolver sistemas computacionais que sejam mais adaptáveis, flexíveis, e fáceis de usar de modo

a diminuir a quantidade de intervenções por parte dos usuários na solicitação de serviços ou na buscar de informação.

Para isso, tais sistemas devem ser capazes de automaticamente identificar o usuário e o seu contexto corrente para oferecer informações personalizadas ou serviços adaptados às necessidades dos usuários. Pesquisadores estudam diversas técnicas, abordagens e métodos para implementar os principais requisitos de software de sistemas sensíveis ao contexto. No entanto, ainda não há um consenso de quais são as melhores ferramentas para suporte ao desenvolvimento deste tipo de sistema.

Desta forma, este capítulo apresentou uma visão geral sobre as áreas Computação Sensível ao Contexto e Aprendizagem de Máquina. Foi destacado que o desenvolvimento de sistemas sensíveis ao contexto exige a superação de vários desafios técnicos como aquisição automática, representação, privacidade e segurança, e especialmente a análise ou processamento de informações contextuais.

Neste sentido, o trabalho discutiu a proposta de uma abordagem para a tarefa de análise de informações contextuais através da tarefa de classificação implementada por meio de técnicas de classificação que utilizam aprendizagem máquina. Com isso, foram apresentados os conceitos introdutórios sobre as principais técnicas de classificação experimentadas na abordagem proposta neste trabalho.

Para o desenvolvedor foram apresentados exemplos práticos de como empregar as técnicas de classificação para aprendizagem de máquina e predição de modo programático através do uso da API Weka. Torna-se importante ressaltar que o assunto discutido neste trabalho não está esgotado, e que outras técnicas para aprendizagem de máquina como Raciocínio Baseado em Casos, ou ainda Classificador de Vizinho mais Próximo (Tan et al., 2009) podem ser experimentadas por meio da abordagem proposta para análise de informações contextuais.

O principal objetivo deste texto foi apresentar uma visão geral sobre Computação Sensível ao Contexto e os conceitos introdutórios sobre Aprendizagem de Maquina que serviram de base para concepção da abordagem de análise de informações contextuais discutida. Para o desenvolvedor de sistemas sensíveis ao contexto este texto pode auxiliar na aprendizagem de novas técnicas que viabilizem o desenvolvimento de sistemas cada vez mais atrativos. Na Computação Sensível ao Contexto, muitos assuntos estão em aberto. Com isso, surge à demanda por realização de novos estudos, e oportunidades de pesquisas para alunos e professores.

4.7. Referências

Abowd, G. D.; Mynatt, E. D. (2000) “Charting past, present, and future research. In: Ubiquitous Computing”, ACM Transactions on Computer-Human Interaction (TOCHI), v. 7, n.1, p. 29-58.

Alves, L. G. P.; Silva, F. S.; Bressan, G. (2009) “CollaboraTVware: A Context-Aware Insfrastructure with Support for Collaborative Participation in Interactive Digital Environment”, In: International Journal of Advanced Media and Communication (IJAMC), v. 3, n.4.

Adomavicius, G.; Tuzhilin, A. (2005) “Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions”, In: IEEE Transactions on Knowledge and Data Engineering, v.17, n.6, p.734-749.

Blanco-Fernandez, Y. (2007) “Propuesta Metodológica para el Razonamento Semántico en Sistemas de Recomendación Personalizada y Automática. Aplicación Al Caso de Contenidos Audiovisuales”, Tese (Doutorado) - Departamento de Enxeneria Telemática E.T.S.E. de Telecomunicación, Universidade de Vigo.

Chen, G.; Kotz, D. (2002) “ SOLAR: A pervasive-Computing Infrastructure for Context-Aware Mobile Applications”. Technical Report TR2002-421, Dartmouth College.

Chen, G.; Kotz, D. (2004) “Na Intelligent Broker Architecture for Pervasive Context-Aware Systems”, Ph. D. Thesis, University of Maryland, USA.

Dey, A.K. and Abowd, G.D. (2000) “Towards a better understanding of context and context-awareness”, In the Workshop on the What, Who, Where, When and How of Context-Awareness, affiliated with the 2000 ACM Conference on Human Factors in Computer Systems (CHI 2000).

Dey, A. K., Abowd, G. D. (2001) “A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications”, In: Human-Computer Interaction, v. 16, n. 2-4, p.97-166.

Dey, A. K. (2001) “ Understanding an Using Context”, In: ACM Personal and Ubiquitous
Computing Journal, v.5, n.1, p.4-7.

Dockhorn Costa, P. D. (2003) “Towards a Services Plataform for Context-Aware Applications”, Master Thesis, University of Twente, The Netherlands.

Eclipse (2012). The Eclipse Foundation open source community. Disponível em: <http://www.eclipse.org/downloads/>. Acesso em: 19 agos 2012

Forstadius, J.; Lassila, O.; Seppanen, T. (2005) “RDF-Based Model for Context-Aware Reasonig in Rich Service Environment.” In: Proceedings of the 3rd International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), p.15-19, 2005.

Goularte, R. (2003) “Personalização e adaptação de conteúdo baseadas em contexto para TV Interativa**”,** Tese (Doutorado) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-23092004-153330/pt-br.php>

Gu, T.; Pung, H. K.; Zhang, D. Q. (2005) “A Service-Oriented Middleware for Building Context-Aware Services” In: Journal of Network and Computer Applications, v.28, n.1, p.1-18.

Han, J.; Kamber, M. Data Mining: Concepts and Techniques. Morgan & Kaufmann, San Francisco, 2006.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reautmann, P., Witten, I. (2009) “The WEKA data mining software: an update”, ACM SIGKDD Explorations Newsletter, v. 11, n.1, p. 10-18, 2009.

Henricksen, K.; Indulska, J.; Mcfadden, T.; Balasubramaniam, S. (2005), “Middleware for distributed context-aware systems”, In: *Lecture Notes in Computer Science*, 3760:846–863.

J2SE (2012). *J2SE Development Kit.* Disponível em: <http://www.oracle.com/technetwork/java/javase/index-jsp-135232.html>. Acesso em: 19 agos 2012

McCulloch, W. S.; Pitts, W. (1943) "A Logical Calculus of the Ideas Immanent", in Nervous Activity**.** Bulletin of Mathematical Biophysics**,** n.5, p.115-133, 1943.

Neles, K.; Malvezzi, W.; Bressan, G. (2012) "Dealing with uncertainties in the monitoring of patients through sensors networks", Health Care Exchanges (PAHCE), 2012 Pan American, p. 50-57, doi:10.1109/PAHCE.2012.6233439.

Papazoglou, M. P. Service-Oriented Computing: Concepts, Characteristics and Directions. In: Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE'03), p.3-12, 2003.

Pessoa, R. M. (2006) " Infraware: Um Middleware de Suporte a Aplicações Sensíveis ao Contexto", Master Thesis, Universidade Federal do Espírito Santo (UFES).

Platt J. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In B. Schoelkopf and C. Burges and A. Smola, editors, Advances in Kernel Methods - Support Vector Learning, 1998.

Raatikainen, K.; Christensen, H. B. T.; Nakajime, T. (2002), " pplication requirements for middleware for mobile and pervasive system**s**", In: *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):16-24.

Ranganathan, A.; Cambell, R.H. (2003) "A Middleware for Context-Aware Agents in Ubiquitous Computing Environments", ACM/IFIP/USENIX Int'l Middleware Conf., LNCS 2672, Springer-Verlag.

Samaras, G.; Kshemkalyani, A.D.; Citron, A. (1996) "Context management and its applications to distributed transactions", Proceedings of the 16th International Conference on Distributed Computing Systems.

Santos, L. O. B da S. (2004) " Semantic Services Support for Context-Aware Platforms" Dissertação de Mestrado, Programa de Pós Graduação em Informática, Universidade Federal do Espírito Santo.

Silva, F. S.; Alves, L. G. P.; Bressan, G. (2012) " PersonalTVware: An Insfrastructure to Support the Context-Aware Recommendation for Personalized Digital TV", In: International Journal of Computer Theory and Engineering (IJCTE), v. 4, n.2.

Silva, F. S. (2011) "PersonalTVware: uma infraestrutura de suporte a sistemas de recomendação sensíveis ao contexto para TV Digital Personalizada", Tese (Doutorado em Sistemas Digitais) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2011. Disponível em: <http://www.teses.usp.br/teses/disponiveis/3/3141/tde-31052011-171129/pt-br.php>.

Strang, T.; Linnhoff-Popien, C. (2004), "A context modeling survey", In: First International Workshop on Advanced Context Modelling, Reasoning And Management, Nottingham, England.

Tan, P-N., Steinbach, M., Kumar, V. Introdução ao Data Mining, Ciência Moderna,Rio de Janeiro, 2009.

Truong, K. N.; Abowd, G. D.; Brotherton, J. A. (2001) "Who, What, When, Where, How: Design Issues of Capture &Access Applications.", In: Proceedings of Ubicomp 2001: Ubiquitous Computing. p. 209-224.

Vapnik, V. N., The nature of Statistical learning theory. Springer-Verlag, New York, 1995.

Vieira,V.; Souza, D.; Salgado, A. C.; Tedesco, P. (2006), "Uso e Representação de Contexto em Sistemas Computacionais", Mini-curso apresentado no Simpósio de Fatores Humanos em Sistemas Computacionais (IHC 2006), Natal, Brasil.

Vieira, V.; Tadesco, P.; Salgado, A. C. (2009) "Modelos e Processos para o Desenvolvimento de Sistemas Sensíveis ao Contexto", Mini-curso apresentado no XXIX Congresso da Sociedade Brasileira de Computação (CSBC 2009), Bento Gonçalves-RS, Brasil.

W3C. Web Services Activity. World Wide Web Consortium (W3C) Recommendation. Disponível em:<http://www.w3.org/2002/ws>. Acesso em: 18 agos. 2012.

W3C. Simple Object Acess Protocol (SOAP) 1.2. World Wide Web Consortium (W3C) Recommendation. Disponível em:<http://www.w3.org/TR/soap12-part0/>. Acesso em: 18 agos. 2012

Weka (2012), Use Weka in your Java Code, Disponível em:<http://weka.wikispaces.com/Use+Weka+in+your+Java+code>. Acesso em: 19 agos 2012

Witten, I. H.; Franck, E. Data Mining: Practical Machine Learning Tools and Techniques", Elsevier, 2005.

Zimmer, T. (2004) "Towards a Better Understanding of Context Attributes". Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, p.23, March 14-17.

Zimmermann, A.; Specht, M.; Lorenz, A.(2005) "Personalization and Context Management", User Modeling and User – adapted Interaction. Volume 15, Numbers 3-4 (2005), 275-302, DOI: 10.1007/s11257-005-1092-2

Índice de Autores

PROMOÇÃO

ORGANIZAÇÃO

PATROCÍNIO

WebMedia em cooperação com