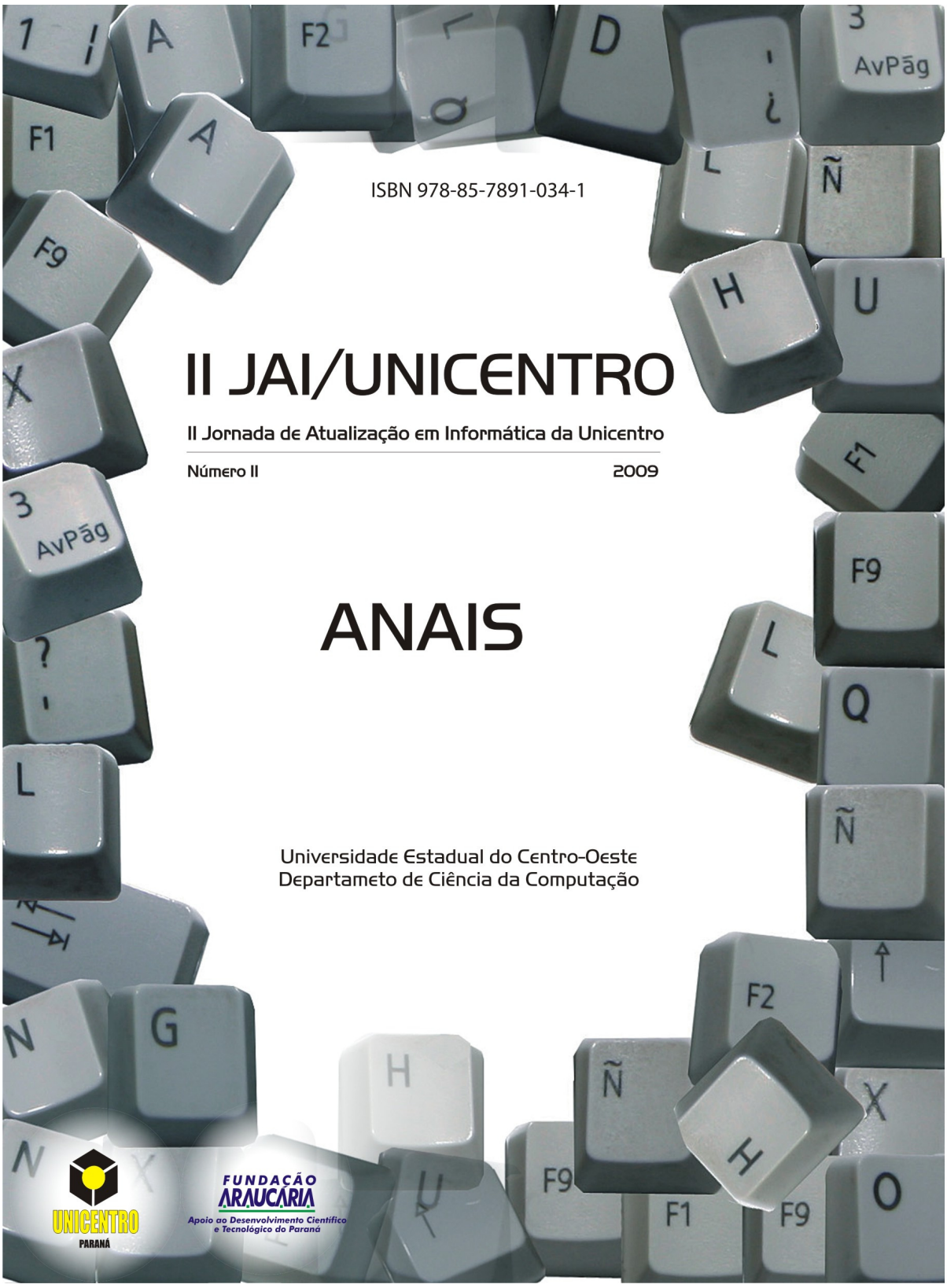

ISBN 978-85-7891-034-1

II JAI/UNICENTRO

II Jornada de Atualização em Informática da Unicentro

Número II 2009

ANAIS

Universidade Estadual do Centro-Oeste
Departameto de Ciência da Computação

UNICENTRO PARANÁ

FUNDAÇÃO ARAUCÁRIA
Apoio ao Desenvolvimento Científico e Tecnológico do Paraná

II JAI/UNICENTRO

II Jornada de Atualização em Informática da UNICENTRO

Departamento de Ciência da Computação

22 a 26 de junho de 2009

Guarapuava – PR

Catalogação na Publicação
Fabiano de Queiroz Jucá – CRB 9/1249
Biblioteca Central da UNICENTRO, Campus Guarapuava

Jornada de Atualização em Informática da UNICENTRO (2. : 2009 : Guarapuava)
J82a Anais da II Jornada... / coordenado [por] Fábio Hernandes, Lucélia de Souza. – – Guarapuava : UNICENTRO, 2009.
1 cd-rom

ISBN 978-85-7891-034-1

Evento realizado entre 22 e 26 de junho de 2009

1. Informática - Educação. 2. Software. 3. Tecnologia. 4. Internet. 5. Educação. I. Título.

CDD 004

II Jornada de Atualização em Informática da UNICENTRO
II JAI/UNICENTRO

Universidade Estadual do Centro-Oeste

Reitoria

Reitor: Prof Vitor Hugo Zanette
Vice-Reitor: Prof Aldo Nelson Bona

Pró-Reitorias

Ensino: Profª. Márcia Terezinha Tembil
Pesquisa e Pós-Graduação: Prof. Mario Takao Inoue
Extensão e Cultura: Prof. Jorge Luiz Favaro
Administração: Prof. Carlos Alberto Kuhl
Recursos Humanos: Prof. Ademir Juracy Fanfa Ribas
Planejamento: Prof. Fernando Franco Neto

Setor de Ciências Exatas e de Tecnologia

Diretor: Prof. José Raniere Mazile Vidal Bezerra
Vice-Diretora: Profª. Karina Worm Beckmann

Direção do Campus Santa Cruz

Diretor: Prof. Osmar Ambrósio de Souza
Vice-Diretor: Prof. Darlan Faccin Weide

Departamento de Ciência da Computação

Chefe: Prof. Fábio Hernandes
Vice-Chefe: Profª. Lucélia de Souza

Apresentação

A II JAI-UNICENTRO tem por objetivo divulgar o Bacharelado em Ciência da Computação da UNICENTRO para a comunidade em geral, bem como apresentar aos participantes uma visão das possibilidades de atuação do profissional da área, através de atividades que permitirão a atualização e disseminação de técnicas e metodologias, conforme as tendências do mercado.

O evento contempla a difusão de pesquisas entre os participantes, permitindo que interajam de forma a buscar o estado da arte em tópicos de interesse, troquem experiências e ampliem suas redes de contato para futuras parcerias em projetos.

Estes Anais apresentam uma coletânea de resumos, organizados em duas partes:

- Palestras e minicursos: resumos sobre os temas abordados nas palestras e minicursos proferidos durante o evento;
- Pôsteres: resumos de trabalhos de conclusão de curso, estágio supervisionado, iniciação científica e projetos de destaque desenvolvidos em disciplinas, de autoria dos acadêmicos do curso e submetidos à avaliação.

Esperamos que os assuntos aqui contidos possam contribuir para o enriquecimento técnico dos seus leitores.

Agradecemos aos que, de forma voluntária, aceitaram o convite e possibilitaram a realização do evento, incluindo palestrantes e instrutores dos minicursos. Também agradecemos o empenho dos autores que submeteram seus resumos e da Comissão Científica que os avaliou, razão da edição deste material.

Coordenação Geral e Comissão Editorial
II JAI/UNICENTRO

Sumário

Sessões Técnicas, Palestras e Minicursos

APLICAÇÃO DE METAESPAÇO DE CONHECIMENTO EM ALGORITMOS CULTURAIS

Murilo Augusto Tosatti, Sandra Mara Guse Scós Venske

Departamento de Ciência da Computação
Universidade Estadual do Centro-Oeste (UNICENTRO)
Caixa Postal 3.010 - 85.015-430 - Guarapuava - PR – Brasil

murilotosatti@gmail.com, sandravenske@gmail.com

Resumo

Métodos de computação evolucionária são ferramentas poderosas para resolução de problemas de busca e otimização, como os problemas NP - Completos. Cada método possui particularidades, vantagens e desvantagens que devem ser analisadas e ponderadas para a implementação de um algoritmo eficiente. Este estudo apresenta a arquitetura de um algoritmo cultural multipopulacional com metaespaço de conhecimento, projetado para analisar a influência da coexistência de populações heterogêneas na busca de soluções para problemas de difícil solução computacional.

Introdução

Alguns problemas ainda não possuem um método para resolução computacional em tempo polinomial, como os problemas NP - Completos [1]. Parte desses problemas, por possuírem aplicações reais, aceita soluções mesmo que essas não sejam ótimas.

Algoritmos evolucionários vêm sendo aplicados com sucesso em busca de soluções para esta classe de problemas [2], [3], [4], [5].

Algoritmos Culturais são modelos evolucionários derivados da observação de processos evolutivos da natureza humana [6]. Um Algoritmo Cultural possui três componentes principais: um espaço populacional, um espaço de crença e um protocolo de comunicação, que determina como o conhecimento é transmitido de um espaço para outro. O espaço populacional pode ser regido por qualquer modelo computacional baseado em populações, como os Algoritmos Genéticos, a evolução Diferencial ou um Enxame de Partículas.

Em [7] e [8], algoritmos culturais multipopulacionais foram usados para resolver problemas de otimização de funções e programação de horários de geradores de energia, respectivamente. Em ambos os trabalhos, as populações compartilham um mesmo espaço de busca, utilizam uma única técnica evolutiva no espaço populacional e implementam o intercâmbio de conhecimento entre as populações.

Este estudo apresenta a arquitetura de um algoritmo cultural que, além das características apresentadas, ofereça: (1) suporte a populações com diferentes técnicas evolutivas e (2) intercâmbio de conhecimento exercido por uma camada externa de metaespaço de conhecimento.

O termo algoritmo cultural simples será usado para indicar um algoritmo cultural com um único espaço populacional e uma técnica evolucionária para controle deste, sem uso do metaespaço de conhecimento.

Projeto e Arquitetura do Algoritmo

O projeto do algoritmo apresenta as seguintes características:

- usar dois algoritmos culturais simples com os conhecimentos situacional e normativo;
- um dos algoritmos com a técnica de evolução diferencial para controle do espaço populacional e o outro com enxame de partículas;
- usar um metaespaço de conhecimento para intercâmbio dos conhecimentos adquiridos entre os dois algoritmos culturais simples.

A arquitetura proposta pode ser visualizada na Figura 1.

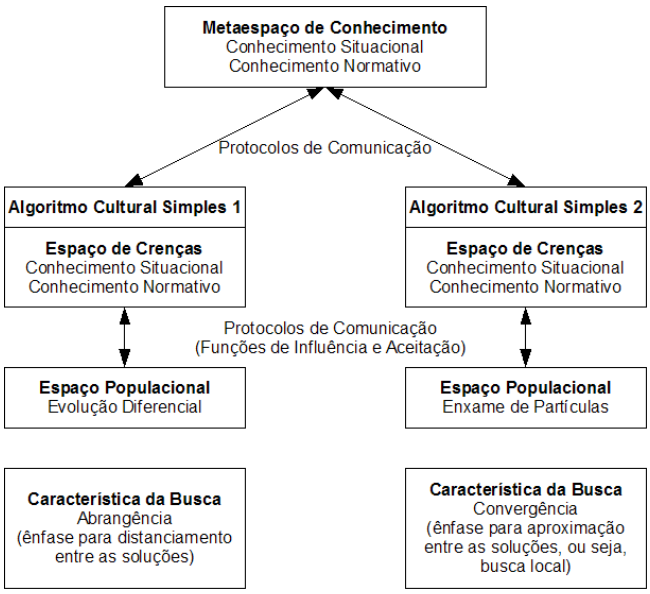

Figura 1. Arquitetura de um algoritmo cultural multipopulacional com metaespaço de conhecimento.

Os algoritmos culturais simples 1 e 2 (Figura 1) utilizam os mesmos tipos de conhecimentos (situacional e normativo). Com isso, o conhecimento adquirido pode ser transportado para o metaespaço sem passar por alterações.

As diferentes técnicas evolutivas para controle dos espaços populacionais permitem otimizações complementares para as buscas.

Conclusões e Trabalhos Futuros

A arquitetura apresentada fornece meios para a implementação de três algoritmos: (1) um algoritmo cultural simples com o espaço populacional sendo regido pela técnica de evolução diferencial; (2) um algoritmo cultural simples

com o espaço populacional sendo regido por enxame de partículas; e (3) um algoritmo cultural multipopulacional que combina os outros dois algoritmos e faz uso de um metaespaço de conhecimento para intercâmbio do conhecimento adquirido entre eles.

Essa arquitetura pode ser modificada para que as populações passem a ser regidas por técnicas evolucionárias diferentes das apresentadas (evolução diferencial e enxame de partículas). As técnicas que controlam os espaços populacionais dos algoritmos culturais determinam as características (como convergência, abrangência, aleatoriedade, entre outras) da busca realizada.

Futuramente, com a implementação dos três algoritmos descritos, pretende-se analisar a influência do metaespaço de conhecimento na busca por soluções para problemas de difícil solução computacional, executando-os sobre um problema e entradas em comum.

Referências

[1] M. Sipser. **"Introdução à Teoria da Computação"**. São Paulo: Thomson Learning, 2007.

[2] M. Li; T. Tong. **"An improved Partheno-genetic algorithm for travelling salesman problem, Intelligent Control and Automation"**. Proceedings of the 4th World Congress, Volume 4, 2002, pp. 3000–3004.

[3] Q. Wang, L. Xiong, H. Liu, J. Liang. **"Improved Particle Swarm Algorithm for TSP Based on the Information Communication and Dynamic Work Allocation"**. Asian Journal of Information Technology, Volume 5, No. 11, 2006, pp. 1191–1196.

[4] C. Groşan, M. Oltean, D. Dumitrescu. **"A new evolutionary algorithm for the multiobjective 0-1 knapsack problem"**. International Conference on Theory and Applications of Mathematics and Informatics (ICTAMI), Alba Iulia, 2003.

[5] B. Crawford, C. Lagos, C. Castro, F. Paredes. **"A Cultural Algorithm for Solving the Set Covering Problem"**. Book Series Advances in Soft Computing Publisher Springer Berlin / Heidelberg, Volume 41, 2007.

[6] R. G. Reynolds, **"An introduction to cultural algorithms"**. Proceedings of the 3rd Annual Conference on Evolutionary Programming, World Scientific Publishing, 1994, pp. 131–139.

[7] Jason G. Digalakis, Konstantinos G. Margaritis. **"A multipopulation cultural algorithm for the electrical generator scheduling problem"**. Mathematics and Computers in Simulation, No. 60, 2002, pp. 293–301.

[8] J. Alami, A. El Imrani, A. Bouroumi. **"A multipopulation cultural algorithm using fuzzy clustering"**. Applied Soft Computing, No. 7, 2007, pp. 506–519.

APLICAÇÃO *WEB* PARA CONSULTA E ACOMPANHAMENTO DE PROCESSOS

Emmanuel Damiani da Silva
emmanueldsilva@gmail.com

Tony Alexander Hild
tony_hild@yahoo.com

Universidade Estadual do Centro-Oeste

Palavras-chave: Aplicação *Web*, consulta, processos.

Resumo:
Este trabalho tem por objetivo apresentar uma proposta de desenvolvimento de uma aplicação *Web* para consulta de processos relativos a cartórios de imóveis, bem como as tecnologias envolvidas para seu desenvolvimento. O serviço visa auxiliar clientes de cartórios no acompanhamento, de maneira prática e eficiente, de seus processos, de uma forma que é possível identificar o andamento deles, bem como seus dados e documentos, utilizando a *Internet.*

Introdução

Há alguns anos as pessoas tinham de se dirigir a locais específicos para conseguir algum tipo de serviço como, por exemplo, ir a um cartório de imóveis para verificar como está o andamento de seus processos.

Com a exploração cada vez maior dos recursos da *Internet*, muitos desses serviços, que antes necessitavam da presença do cliente no local da prestadora, estão sendo implantados em servidores *Web* para atender a clientes em qualquer local que tenha conexão com a rede mundial de computadores com um dispositivo móvel para seu acesso.

O uso cada vez maior de aparelhos que visam mobilidade e praticidade na realização de tarefas, como celulares e *notebooks*, estimula as empresas a implantar serviços via *Internet* para conquistar e satisfazer uma gama cada vez maior de clientes.

Diferentemente de aplicações *desktop*, as quais todos os componentes são acessados localmente, as aplicações *Web* [1] são executadas em servidores *Web*.

Nesse trabalho será apresentada a metodologia para o desenvolvimento de um aplicativo para consulta e acompanhamento de processos via *Web* para cartórios de imóveis, bem como as tecnologias utilizadas.

Materiais e Métodos

Analisando o *Imob,* sistema *desktop* criado pela *Brainsoft Sistemas* para gerenciar as principais tarefas e serviços que um cartório de imóveis necessita, faz-se necessário tomar conhecimento sobre suas principais funcionalidades e

características, pois esse *software* é o ponto de partida para a criação da aplicação *Web* [1] para consulta de processos. Toda informação dentro de um cartório é gerenciada e manipulada por esse sistema, e qualquer outra aplicação que venha a ser implantada deve utilizar dos recursos desse complexo programa.

Para a implementação da aplicação *Web*, serão utilizadas algumas tecnologias da plataforma .NET 3.5 [2] como: *Visual Studio 2008*, conjunto de ferramentas para desenvolvimento de soluções em *software*, utilizando linguagens de programação da *Microsoft*; *Visual SourceSafe*, sistema para controle de versão em nível de arquivo e coordenação de projetos em equipe; *Microsoft SQL Server 2008*, sistema gerenciador de banco de dados; ASP.NET [1], modelo de desenvolvimento para aplicativos *Web.*

A aplicação *Web* para consulta de processos é um dos módulos constituintes de um sistema *Web* complexo e utilizará a metodologia própria e ágil de desenvolvimento de *software* [3], que visa garantir a qualidade do sistema mesmo tendo que alterá-lo várias vezes.

Este sistema *Web* será implementado utilizando o padrão de arquitetura de Três Camadas [4]: Camada de Aplicação, de Negócio e de Acesso a Dados. A Camada de Aplicação será constituída por uma Página *Web* responsável por interagir diretamente com os clientes do cartório. A Camada de Negócio é onde toda a lógica do serviço é implementada e é responsável por atender as requisições da Camada de Aplicação e se comunicar com a Camada de Acesso a Dados. A Camada de Acesso a Dados garante a consistência das informações dos processos dentro de um banco de dados.

Além disso, será implementado um *framework,* que conterá toda a lógica da aplicação e garantirá a persistência dos dados no banco, utilizando o *NHibernate* [5] e um *WebService*, que será a interface desse *framework* com o *Imob*.

O *Imob* exportará processos para o *WebService*, que será encarregado de gerenciá-los no banco de dados. A Página *Web,* por sua vez, é responsável por acessar esse banco e exibir ao cliente todas as informações de seus processos solicitados.

A Página *Web* será implementada utilizando a linguagem ASP.NET, pois possui mecanismos práticos para criar e gerenciar conteúdo dinâmico em páginas. Os processos armazenados no banco de dados são diferentes, ou seja, alguns possuem mais informações que outros. Logo, exige-se um mecanismo prático para manipular diferentes tipos de informações sem comprometer a estruturação de uma Página *Web.*

Para o desenvolvimento da aplicação, faz-se necessário também utilizar metodologias de testes, principalmente quando os módulos do sistema forem executados em conjunto. Para tanto, será utilizada a metodologia TDD (*Test-driven Development)* [3], de maneira que os testes são especificados antes do código ser implementado.

Para o funcionamento da Aplicação *Web,* é importante que outros módulos do Sistema *Web* estejam funcionando. Para demonstração de suas funcionalidades serão utilizados *Mock Objects* [6], que simulam o comportamento de outros módulos do sistema sem a real presença deles.

Considerações finais

O objetivo desta aplicação é estabelecer uma ligação entre os clientes dos cartórios de imóveis com seus processos.

Ao final do desenvolvimento da Aplicação *Web*, os clientes de cartórios de imóveis poderão ter acesso a esse serviço de qualquer local através da *Internet*, consultando uma Página *Web*. Esse serviço traz benefícios para o cliente, pois ele não precisará mais se dirigir ao seu cartório para verificar o andamento de seus processos.

Referências

[1] BUENO, Luiz Henrique. **Aplicações Web com Visual Studio.Net, ASP.NET e C#.** São Paulo : Alta Books, 2005.

[2] .NET 3.5. **Microsoft .NET de Desenvolvimento Versão 3.5.** Disponível em http://msdn.microsoft.com/pt-br/default.aspx. Acessado em 21/04/2009.

[3] QUADROS, Moacir. **Gerência de Projetos de Software: Técnicas e Ferramentas**. Florianópolis: Visual Books Editora, 2002.

[4] COCKBURN, Alistair. ***Agile Software Development: The Cooperative Game***. Boston : Addison Wesley, 2007.

[5] NHIBERNATE. ***Nhibernate for .NET***. Disponível em https://www.hibernate.org/343.html. Acessado em 21/04/2009.

[6] MOCK. **Mock Objects.** Disponível em http://www.mockobjects.com. Acessado em 03/05/2009

COMPUTAÇÃO NATURAL E SUAS APLICAÇÕES

Myriam R. Delgado[1], Carolina P. Almeida[1], Richard A. Gonçalves[1,2] e Sandra M. Venske[2]
e-mail: myriamdelg@utfpr.edu.br

1 Universidade Tecnológica Federal do Paraná - CPGEI
2 Universidade Estadual do Centro-Oeste - DECOMP

Palavras-chave: computação natural, predição da estrutura de proteínas, problema do caixeiro comprador, despacho econômico de energia.

Resumo:

Esta palestra apresenta alguns conceitos fundamentais de Computação Natural e suas aplicações em problemas de otimização combinatória e contínua. Serão abordadas as seguintes técnicas: Sistemas Imunológicos Artificiais, Inteligência Coletiva, Inteligência Computacional, Algoritmos Culturais e Algoritmos Transgenéticos, destacando a adequação das técnicas para problemas de otimização combinatória ou contínua. Dentre as aplicações que serão abordadas destacam-se a Predição da Estrutura de Proteínas, o Problema do Caixeiro Comprador e o Problema do Despacho Econômico de Energia.

Introdução

A Computação Natural (CN) agrega diversas técnicas inspiradas na natureza para o processamento de informações e tem despertado o interesse de inúmeros pesquisadores [1]. Algumas técnicas de CN que serão discutidas nessa palestra são: Sistemas Imunológicos Artificiais (SIA), Inteligência Coletiva, Inteligência Computacional (IC), Algoritmos Culturais (AC) e Algoritmos Transgenéticos (AT).

Sistemas Imunológicos Artificiais são técnicas computacionais que se inspiram no funcionamento do sistema imunológico dos seres vertebrados. Pesquisadores da área de SIA têm concentrado sua atenção em aprendizado e mecanismos de memória de sistemas imunes (como a teoria da seleção clonal e as redes imunes) e na seleção de detectores para identificar anomalias [9].

A Inteligência Coletiva aborda sistemas compostos por vários indivíduos coordenados de forma descentralizada e auto-organizada. Em particular, a Inteligência Coletiva foca no comportamento coletivo que resulta de interações locais entre os indivíduos e o ambiente, sendo os principais representantes as colônias de formigas e os exames de partículas [3].

A Inteligência Computacional combina elementos de aprendizado (características das Redes Neurais), adaptação e evolução (presentes nos algoritmos evolutivos), e tratamento de incertezas (componente principal dos Sistemas Nebulosos). Esses elementos formam a base para a criação de programas que são, de alguma maneira, inteligentes [4].

Os Algoritmos Culturais (AC) foram inicialmente desenvolvidos por Robert G. Reynolds e são uma poderosa ferramenta para resolver problemas de busca e otimização. São algoritmos baseados na evolução cultural da humanidade e servem como um complemento a metáfora evolutiva [7].

A Transgenética Computacional é uma meta-heurística evolucionária que ancora sua metáfora no processo de endossimbiose [5]. Diferente dos processos evolucionários tradicionais, o processo evolucionário na TC é realizado através de trocas de material genético e recombinações entre o DNA dos simbiontes e o DNA do hospedeiro. Como se trata de um processo de co-evolução, tanto o DNA dos simbiontes como o DNA do hospedeiro se alteram ao longo do processo evolucionário. As alterações são realizadas – operacionalizadas – por vetores de manipulação que mimetizam a ação de vetores biológicos do fluxo intracelular natural.

Será feita uma análise das técnicas supracitadas com o objetivo de identificar quais delas são indicadas para cada classe de problemas (otimização contínua ou combinatória) destacando-se os seguintes: Problema da Predição da Estrutura de Proteínas, Problema do Caixeiro Comprador e o Problema do Despacho Econômico.

O problema de dobramento de proteínas pode ser definido como um processo químico no qual a estrutura de uma proteína assume a sua configuração funcional, sendo que, neste caso, ela se encontra no mínimo da sua energia livre. Doenças - como, por exemplo, Alzheimer, Parkinson e alguns tipos de câncer - estão ligadas ao dobramento incorreto de proteínas, considerando que este mau dobramento pode causar agregação e deposição das mesmas nos tecidos. O desafio para esse problema é determinar a combinação de parâmetros de conformação (ângulos, comprimentos, torções, entre outros) que minimizam a energia livre da proteína [3].

O Problema do Caixeiro Comprador (PCC) é uma generalização do conhecido Problema do Caixeiro Viajante (PCV) e foi introduzido por Ramesh [6], podendo ser definido como a seguir: considera-se um conjunto de produtos que deverão ser adquiridos e um conjunto de mercados oferecendo certas quantidades de cada produto. São requeridas quantidades diferentes de cada produto. O preço da unidade do produto é conhecido e varia de acordo com o mercado. É conhecido também o custo da viagem entre cada par de mercados. Deve-se, então, determinar o custo mínimo (custo de viagem e custo de compra) para satisfazer a demanda de produtos.

O Despacho Econômico corresponde à alocação ótima de uma demanda desejada de energia entre os geradores de um sistema de geração termoelétrica. É importante garantir que as condições de operação sejam satisfeitas. Esse problema busca minimizar o custo de produção de energia elétrica por meio da otimização da distribuição da produção entre as unidades geradoras [8].

Considerações finais

As técnicas abordadas nesta palestra possuem inúmeras aplicações práticas, algumas das quais serão exemplificadas durante a apresentação. A

Computação Natural é uma área de grande interesse científico, agregando técnicas que se mostram promissoras na solução de diferentes problemas de otimização.

Referências

[1] DE CASTRO, Leandro **Fundamentals of Natural Computing: Basic Concepts, Algorithms, and Applications.** Chapman Hall/CRC, 2006.

[2] DILL, K. A. A. et al. The protein folding problem: when will it be solved? ***Curr Opin Struct Biol***, Department of Pharmaceutical Chemistry, University of California, San Francisco, CA 94143, USA., June 2007. ISSN 0959-440X.

[3] ENGELBRECHT, Andries **Fundamentals of Computational Swarm Intelligence**. 1ª Edição, Wiley, 2006.

[4] ENGELBRECHT, Andries **Computational Intelligence: An Introduction**. 2ª Edição, Wiley, 2007.

[5] Goldbarg, Marco; Bagi, Ligia e Goldbarg, Elizabeth Transgenetic algorithm for the Traveling Purchaser Problem. **European Journal of Operational Research**, v. 199, n. 1, p. 36-45, 2009.

[6] RAMESH, T. Traveling purchaser problem. **OPSEARCH** v. 18, p. 78-91, 1981.

[7] REYNOLDS, Robert. Tutorial on Cultural Algorithms. **IEEE Swarm Intelligence Symposium,** 2003.

[8] SINHA, N.; CHAKRABARTI, R. e CHATTOPADHYAY, P. K. Improved Fast Evolutionary Program for Economic Load Dispatch with Non-Smoth Cost Curves. **IE Journal**, v. 85, p. 110-114, 2004.

[9] TIMMIS, John. Aritificial Imune Systems - Today and Tomorrow. **Natural Computing: An International Journal**, v. 6, n. 1, p. 1-18, 2007.

BANCO DE CURRÍCULOS PARA ESTÁGIO PROFISSIONALIZANTE NA UNICENTRO.

Adamo Góes[1], Anderson Silverio (Orientador)[2]
adamoxgoes@gmail.com[1], anderson@unicentro.com.br[2]

Departamento de Ciência da Computação – DECOMP
Universidade Estadual do Centro-Oeste do Paraná – UNICENTRO
Guarapuava – PR

PALAVRAS-CHAVE: CakePHP, MVC, ORM, CRUD, benefícios, programação dinâmica.

RESUMO

Este resumo apresenta a proposta de um desenvolvimento de um banco de currículos online utilizando um *framework* chamado CakePHP. CakePHP é utilizado para um desenvolvimento ágil da aplicação, baseado na arquitetura MVC (*Model-View-Controller*) e na metodologia *ORM* (*Object-Relational-Mapping*).

INTRODUÇÃO

Atualmente nota-se que a procura de empresas por mão de obra barata vem crescendo ano a ano. Diante desse quadro, estagiários são os mais procurados para amenizar os custos. Entretanto, após a aprovação da nova lei de estágio [1] houve uma mudança importante. Conforme a lei, os benefícios de um estagiário hoje são próximos de um funcionário efetivo, logo, ainda é valido contratar estagiários do que funcionários pela questão financeira. Porém a nova lei dita também que os estagiários hoje só podem exercer funções que correspondem a sua área, por exemplo, um estagiário de computação não pode ficar carregando bolas de basquete e cones que é a função de um estagiário de Educação Física.

A Universidade Estadual do Centro-Oeste UNICENTRO é o órgão estadual com o maior número de estagiários do Paraná em números absolutos. Nos meses de fevereiro e março existe uma procura demasiada de estagiários, cabendo assim, aos Recursos Humanos da UNICENTRO fazer as verificações de cada um dos candidatos, se possui um perfil para a vaga, ou se a instituição que o estudante está possui convênio com a UNICENTRO e outros requisitos.

Sendo assim, com base nestes fatos, propõe-se um desenvolvimento de um banco de currículos online profissionalizante para a UNICENTRO. O sistema deve providenciar a integração das entidades estudante, departamento e RH.

MATERIAIS E MÉTODOS

Para o desenvolvimento da aplicação será utilizado o *framework* CakePHP [2]. Cabe ressaltar que tal *framework* é utilizado para um desenvolvimento ágil da

aplicação, baseado na arquitetura MVC (*Model-View-Controller*) e na metodologia *ORM* (*Object-Relational-Mapping*).
A metodologia *ORM* consiste em converter objetos para entidades relacionais e gera código relacional para acesso e manipulação dos objetos [3]. A Figura 1 representa o modelo objeto relacional.

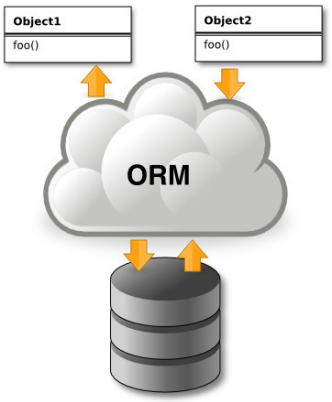

Figura 1. Modelo Objeto Relacional.

Como citado anteriormente, CakePHP possui a característica de usar as classes de *controller*, *model* e *view*. Programando no modelo MVC é possível separar a aplicação em três partes principais.

Controller: Um *controller* é usado para gerenciar a lógica para uma parte de sua aplicação. Conforme o endereço que o usuário digita é o *controller* que busca no *Model* os dados corretos para apresentar à *view.*

Model: *Models* são usados no CakePHP para acessar os dados. *Models* representam, na maioria dos casos, tabelas de banco de dados.

View: Em CakePHP, a comunicação com os usuários é realizada através das camadas das *views*. A maior parte do tempo, as *views* estarão mostrando documentos XHTML para os navegadores. A Figura 2 representa uma requisição feita em MVC no CakePHP [6].

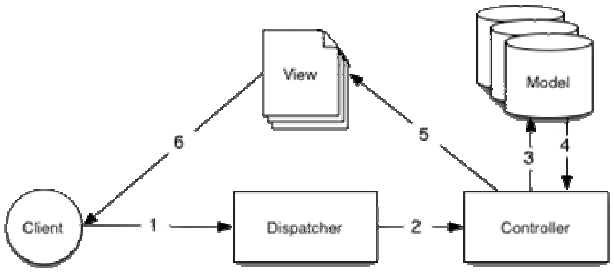

Figura 2. Requisição típica do CakePHP.

O usuário clica no *link* apontando para www.exemplo.com.br/cakes/comprar e seu *browser* faz uma requisição ao site. O *dispatcher* (expedidor) verifica a *URL* requisitada (/cakes/comprar) e redireciona ao *controller* correto. O *controller* executa a lógica específica da aplicação. Por exemplo, verifica se o usuário está logado. O *controller* também usa os *models* para acessar os dados da sua aplicação. Muitas vezes, os *models* representam as tabelas do banco de dados, mas podem representar registros *LDAP*, *feeds* de RSS ou até mesmo arquivos do sistema.

Depois que o *controller* fez sua mágica sobre os dados, ele repassa para a *view*. A *view* faz com que os dados fiquem prontos para a representação do usuário. As *views* em CakePHP normalmente vem no formato HTML, mas pode ser facilmente exibidas em PDF, documento XML, um objeto JSON ou outro formato qualquer, dependendo da necessidade.

Uma vez que a *view* tenha usado os dados provenientes do *controller* para construir a página, o conteúdo é retornado ao *browser* do usuário.

CONSIDERAÇÕES E DISCUSSÕES

Com base no levantamento bibliográfico realizado fica claro que o CakePHP torna fácil a criação e manutenção da aplicação. Elaborar tarefas divididas entre *models*, *views* e *controllers*, faz com que os desenvolvedores alterem uma parte da aplicação sem afetar outras, assim, tendo uma programação mais dinâmica aumentando a produtividade. Novas funcionalidades podem ser adicionadas dando uma nova cara às características antigas [6].

REFERÊNCIAS

[1] Nova Lei do Estágio. Disponível online em http://www.estagiarios.com/legislacaodeestagio.asp. Acessado dia 20 de maio de 2009.

[2] CakePHP. Disponível online em http://cakephp.org. Acessado dia 20 de maio de 2009.

[3] Object-Relational-Mapping (ORM). Disponível online em http://c2.com/cgi/wiki?ObjectRelationalMapping. Acessado dia 20 de maio de 2009.

[4] MVC (*Controller*). Disponível online em http://manual.cakephp.com.br/doku.php?id=controllers#introducao. Acessado dia 20 de maio de 2009.

[5] MVC (*Model*). Disponível online em http://manual.cakephp.com.br/doku.php?id=models. Acessado dia 20 de maio de 2009.

[6] Manual CakePhp. Disponível online em www.manual.cakephp.com.br/doku.php. Acessado em 20 de maio de 2009.

EXECUÇÃO DE COMANDOS LINUX UTILIZANDO RECONHECIMENTO DE VOZ

Diego Gadens dos Santos, Angelita Maria De Ré
diegogadens@gmail.com
Universidade Estadual do Centro-Oeste - UNICENTRO

Palavras-chave: redes neurais artificiais, reconhecimento de voz, linux.

Resumo:
Este trabalho apresenta uma proposta de um sistema baseado em Redes Neurais Artificiais para o reconhecimento de comandos de voz. Depois de interpretados, estes comandos são executados em um terminal Linux do mesmo modo que seriam se os mesmos estivessem sendo inseridos via teclado. Assim, esta funcionalidade tem o objetivo de prover um maior nível de acessibilidade a pessoas com necessidades especiais. A tecnologia das Redes Neurais Artificiais proporciona um alto nível de confiabilidade no reconhecimento de padrões. Utilizando-se dessa característica, pretende-se construir uma Rede Neural para reconhecer comandos em linguagem natural, e, a partir disso, executá-los em um terminal de texto.

Introdução

Poder controlar as máquinas de maneira natural é um sonho antigo dos homens. Desde o aparecimento dos primeiros inventos desta categoria a idéia de uma interação mais inteligente e natural motiva diversas pesquisas, principalmente em Inteligência Artificial (I.A.). Utilizar-se da voz humana para comandar, interagir, controlar, enfim, se comunicar com as máquinas, da mesma maneira que os humanos se comunicam entre si, é um grande desejo dos cientistas.

O objetivo deste trabalho é apresentar um sistema que utiliza uma técnica de Inteligência Artificial, as Redes Neurais Artificiais, para reconhecer e executar comandos do terminal Linux. Nessa proposta o número de comandos reconhecidos é limitado, visto que este é um estudo que serve para demonstrar e validar o proposto. E assim, a partir dos resultados obtidos, implementar os demais comandos pertinentes ao Linux.

Materiais e Métodos

Redes Neurais Artificiais

Uma Rede Neural Artificial (RNA) é basicamente um processador de dados inspirado no modelo biológico, ou seja, o cérebro humano. O elemento chave deste paradigma é a estrutura pela qual a informação é processada. Esta estrutura conta com as unidades de processamento, chamadas de neurônios, intensamente conectadas, trabalhando em conjunto para resolver um determinado problema. Assim como os seres humanos, as RNA's aprendem pelos exemplos. O aprendizado das RNA's se dá por meio do ajuste dos pesos

das conexões entre os neurônios [1]. Pode parecer que as simulações com Redes Neurais é uma técnica recente, entretanto, esse modelo foi desenvolvido antes mesmo do aparecimento dos primeiros computadores, e sobrevive até hoje devido sua alta tecnologia e confiabilidade. [2]

O funcionamento básico das Redes Neurais consiste em receber um determinado padrão numérico, realizar um processamento nestes dados e, por fim, exibir uma saída. O padrão utilizado é um dos determinantes do correto funcionamento das Redes Neurais e sua construção adequada irá influenciar fortemente no resultado do processamento. [3]

As Redes Neurais e o reconhecimento de voz

Uma questão a ser considerada quando o assunto trata do reconhecimento de voz é o fato de que os seres humanos são muito bons nesta tarefa. Sabe-se que o cérebro humano tem uma estrutura de funcionamento diferente dos computadores convencionais. Enquanto um computador convencional conta com um processador central extremamente rápido e complexo, constituído por instruções de máquina explícitas e uma memória local endereçável, o cérebro humano é constituído por bilhões de unidades de processamento pequenas e simples. Estas unidades estão densamente conectadas gerando as sinapses que armazenam o conhecimento e se modificam de acordo com as experiências [4].

Apesar de complexa a fala também conta com um padrão, e, portanto, uma Rede Neural pode ser treinada para identificá-lo. Para que o processamento da onda sonora possa ser efetuado, o som deve ser captado por um microfone e convertido em um sinal digital. Neste sinal digital, podem ser aplicados filtros de diminuição de ruídos para que os dados apresentados apresentem uma melhor qualidade [5].

Resultados e Discussão

Acessibilidade é fazer com que um software possa ser utilizado por todas as pessoas, independente de possíveis dificuldades ou deficiências que possam estar presentes. O tema da acessibilidade no ambiente GNU/Linux, não é um tema novo. Há pouco mais de uma década se deu inicio a estes trabalhos e atualmente o sistema já conta com boas opções, como leitores de tela em modo sonoro e em modo braile [6].

A utilização de uma Rede Neural Artificial será essencial para o bom resultado do reconhecimento dos padrões, pois proporcionará uma solução genérica, robusta e de fácil manipulação, atualização e modificação.

Além disso, deseja-se mostrar que a interação com as máquinas pode ser facilitada utilizando-se da voz humana. A usabilidade de equipamentos complexos pode ser aumentada ao se incluir em seu funcionamento métodos que permitam uma interação natural. Portanto, com este trabalho deseja-se criar um modelo de interação mais facilitado entre homens e máquinas.

Considerações finais

Atualmente o estágio de desenvolvimento deste projeto encontra-se na fase de pesquisa, tanto das RNA's, quanto do processamento de sinais digitais, tendo em vista que foi possível identificar a relevância e a importância do trabalho em um contexto prático, o qual visa ampliar a acessibilidade do uso do sistema GNU/Linux.

Referências

[1] CARVALHO, André Ponce de Leon F. de, Redes Neurais Artificiais, Universidade de São Paulo, Departamento de Ciência da Computação. Disponível em: http://www.icmc.usp.br/~andre/research/neural/. Acessado em: 15/05/2009

[2] Imperial College of London, Department of Computer Science. Disponível em <http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html>. Acessado em 17/05/2009

[3] Uma introdução às Redes Neurais, Universidade Estadual de Maringá. Disponível em http://www.din.uem.br/ia/neurais/#aprendizado. Acessado em 17/05/2009.

[4] Major National Resources for Study of Brain Anatomy, University of Wisconsin, Michigan State University and the National Museum of Health and Medicine. Disponível em: http://brainmuseum.org/functions/index.html. Acessado em 15/05/2009.

[5] TAFNER, Anderson M. Reconhecimento de palavras faladas isoladas usando Redes Neurais Artificiais. Florianópolis, 1996. Dissertação (Mestrado em Engenharia) - Universidade Federal de Santa Catarina, UFSC. Disponível em: http://www.eps.ufsc.br/disserta96/tafner/index/index.htm. Acessado em 12 de abril de 2009

[6] THIBAULT, Samuel, Accessibility in Linux systems. Disponível em: http://lwn.net/Articles/302159/. Acessado em 14/04/2009

ESTUDO E DESENVOLVIMENTO DE MÓDULOS NECESSÁRIOS PARA A CRIAÇÃO DE UM SISTEMA UTILIZANDO A TECNOLOGIA JAVA CARD: DESAFIOS INICIAIS.

Osvaldo Matyak Junior
E-mail: osvaldomatyak@gmail.com

Universidade Estadual do Centro-Oeste do Paraná

Palavras-chave: *Java Card, smart card, java, on-card*, operações básicas.

Resumo:
A tecnologia *Java Card* permite a programação de *Smart Cards*, cartões inteligentes utilizados atualmente em bancos e cartão GSM de celular, dentre outros dispositivos como *pendrives* e anéis (Java *Ring*). Este documento trata das dificuldades iniciais para trabalhar com essa tecnologia.

Introdução

Atualmente os cartões bancários, a certificação digital, documentos de identificação e impressos como vales alimentação e transporte são comumente utilizados pelo mercado. As pessoas carregam uma coleção desses objetos no seu dia-a-dia, o que produz diversos desafios para organizações: portabilidade, a segurança de utilização, e gerenciamento das informações. Por esse motivo escolheu-se *smart card* como o alvo de trabalho com a tecnologia Java Card.

A tecnologia *Java Card* é uma solução para unificar documentos e aplicações em um único meio digital, oferecendo maior segurança e gerenciamento mais eficiente das informações.

Java Card é um assunto que demanda atenção do mercado de tecnologia da informação e das universidades porque se trata de uma tecnologia em ascensão e, cada vez mais, adotada por segmentos como, por exemplo, financeiro e governamental. Muitas dificuldades encontradas por desenvolvedores iniciantes nessa tecnologia serão tratadas adiante.

Materiais e Métodos

Após levantamento bibliográfico, compreensão dos objetivos da tecnologia e das necessidades de hardware, o desenvolvedor que inicia na tecnologia *Java Card* deve, por exemplo, optar entre simular operações de gravação e consulta, ou então adquirir os equipamentos apropriados às operações com os cartões. Iniciou-se o trabalho com a simulação via software e, posteriormente, adquiriu-se *Java smart cards* e eleitor/gravador para os testes.

O simulador abstrai problemas como especificidades e padrões gerados pela indústria, e é uma alternativa barata para validar a programação de *applets* - pequenos programas embarcados nos cartões. Também é possível testar as

“mensagens” para a comunicação com equipamentos reais. O simulador utilizado foi o JCWDE, fornecido pelo *Java Card Development Kit* [1].

Existem *smart cards* e *java smart cards*, a diferença entre os dois é que os *Java smart cards* suportam a tecnologia *Java card*. A tecnologia Java não pode ser utilizada em todos os smart cards. As tecnologias alternativas são *Smart card for Windows* e *Multos* [2].

Após adquirir certa experiência utilizando-se do simulador, a programação real utilizando cartões reais é iniciada, existe um conjunto grande de restrições (formas de acesso e fechamento das especificações de cartões) e de programas próprios impostos pela indústria de cartões para tal atividade, alternativamente existem alguns padrões de uso livre, que podem ser utilizados. Encontrar *Java smart cards* e equipamentos é um desafio para estudantes, poucas empresas se interessam por vender poucas unidades, o que eleva seu custo unitário.

Observa-se que a fase de simulação é proveitosa para o contato inicial com a tecnologia, existindo poucos paralelos entre as atividades realizadas na simulação e os trabalhos com dispositivos reais (*scripts* para instalação, remoção, listagem e comunicação).

As atividades básicas para se programar e se comunicar com um *Java smart card* utilizando a tecnologia *Java Card* são: programação da *applet,* criação de script para a instalação *on-card* (consequentemente de remoção e listagem de *applets*) e de comunicação.

Existe um protocolo para a comunicação entre o cartão e a máquina, o APDU (Application Protocol Data Unit). Os formatos gerais dessas mensagens são exibidos na Figura 1.

Command APDU						
Cabeçalho Obrigatório				Corpo Opcional		
CLA	INS	P1	P2	LC	Data field	LE

Response APDU	
Corpo Opcional	Cola Obrigatória
Data field	SW1 SW2

Figura 1: formato geral de apdu`s.

Resultados e Discussão

Sobre os problemas enfrentados pelo desenvolvedor iniciante na tecnologia *Java Card* cita-se: bibliografia para estudo prático da tecnologia e dos dispositivos periféricos utilizados como *smart cards, java rings* e etc.; opção por realizar testes com simuladores ou diretamente com os dispositivos; escolha de dispositivos para adquirir e a dificuldade de encontrá-los; diversidade de dispositivos criados pela indústria com características próprias; ambiente de desenvolvimento que deve ser instalado; materiais que contribuam para o desenvolvimento real de aplicações; limitações dos dispositivos; questões de segurança, privacidade de usuários finais e criptografia; grupos de pesquisa fechados; falta de listas de discussão e fóruns sobre o tema; orientação de trabalhos na Universidade; falta de apoio da indústria de cartões.

O diálogo sobre como se deve iniciar nessa área motiva e facilita a compreensão de toda a tecnologia envolvida, fazendo com que as curvas de aprendizado e produtividade sejam mais ascendentes.

Considerações finais

Para se compreender teoricamente a tecnologia *Java Card* existem diversos materiais, fornecidos inclusive pela SUN [1], da mesma forma para a fase de testes em simuladores, onde casos simples são demonstrados e explicitados.

O principal problema inicia-se com o desenvolvimento real, não existe bibliografia consistente para o assunto. Isso se deve por diversos fatores, como grupos de pesquisa fechados, especificações diversas na indústria e interesses particulares em consultoria no assunto.

A indústria influi diretamente na produção de materiais, como por exemplo, os livros, pois focalizam o seu conteúdo em vários padrões, em um padrão ou em nenhum, situação que influência conteúdos vagos.

Referências

[1] Java Card Technology**. Site: http://java.sun.com/javacard/.**

[2] Ministério de ciência e tecnologia. **Sistemas operacionais para cartões inteligentes.** Disponível em http://www2.dem.inpe.br/ijar/SistOperSmartCard.pdf. Acessado em 22 de maio de 2009.

[3] CHIKASAWA, Ricardo. **Java Card e Smart Card**. Disponível em http://blogs.sun.com/chikasawa. Acessado em 22 de maio de 2009.

AUTOMATIZAÇÃO DO MAPEAMENTO OBJETO-RELACIONAL EM BANCO DE DADOS

Marcos Latchuk
marcos_latchuk@hotmail.com

UNICENTRO - Universidade Estadual do Centro-Oeste

Palavras-chave: mapeamento objeto-relacional, *impedance mismatch*, Nhibernate.

Resumo:
A incompatibilidade entre as linguagens orientadas a objetos e os bancos de dados relacionais acarretam sérios problemas para os programadores. Esses têm que deixar de lado o paradigma orientado a objetos, baixar seu nível de abstração e atuar de forma manual sobre o banco de dados a cada inovação ou modificação elementar do seu código. Uma solução parcial para este problema foi encontrada com o desenvolvimento dos ORMs (*Object Relational Mapping*). Esses possibilitam fazer o mapeamento dos objetos para a forma relacional a partir de arquivos de configuração criados pelo usuário. Muito esforço tem sido empregado em maneiras para facilitar este mapeamento. A proposta deste trabalho é a criação de uma biblioteca que torne possível o mapeamento de forma automática, sem a necessidade da criação manual de arquivos para a configuração do mesmo.

Introdução

Com o surgimento das linguagens orientadas a objetos, ocorreram mudanças não somente no paradigma das práticas de programação, mas também no comportamento dos sistemas gerenciadores de banco de dados (SGBDs) com essas novas linguagens.

Atualmente se pode encontrar alguns bancos de dados orientados a objetos, atuando com o mesmo paradigma da linguagem orientada, mas esses ainda são pouco utilizados. Os bancos mais utilizados, e que tomam quase todo o mercado, ainda são os bancos de dados relacionais[1].

A utilização de linguagens orientadas a objetos e bancos de dados relacionais compõe um grande problema, pois existe uma diferença no modo de comunicação entre essas tecnologias. Essa diferença gera várias dificuldades técnicas que dificultam a construção de sistemas que adotem a utilização dessas. O termo *impedance mismatch* (tipos incompatíveis) é utilizado para descrever tais diferenças e dificuldades [1].

Para tentar solucionar o problema de *impedance mismatch* surgiram os ORMs, que fazem o mapeamento objeto-relacional e possibilitam a comunicação entre a linguagem e o banco. Um ORM é uma técnica que

[1] Pesquisa realizada em abril de 2009 em *sites* referentes aos bancos de dados mais utilizados: Oracle, SQL Server, MySQL, Db4O, PostGreSQL e FireBird.

converte os dados incompatíveis de uma linguagem de programação orientada a objetos em dados compatíveis para um sistema de banco de dados relacional e vice-versa.

O ORM cria um efeito de virtualização de um banco de dados orientado a objetos, deixando o programador livre para utilizar apenas recursos de sua linguagem, sem se preocupar em como será feito o acesso aos recursos do banco. Contudo, para que esta virtualização seja possível, arquivos de configuração e mapeamento devem ser criados pelo programador e passados ao ORM [2].

Materiais e Métodos

O NHibernate é a versão do ORM Hibernate da plataforma Java, portado para a plataforma .NET. Ele trabalha realizando a persistência de objetos em um banco relacional subjacente. Esse será utilizado para o mapeamento objeto-relacional no desenvolvimento do trabalho.

Os arquivos de configuração do NHibernate são, geralmente, escritos em linguagem XML. Esses dizem como cada objeto deve ser mapeado para o modo relacional, como deve ser a construção dos campos das tabelas do banco, as ligações entre elas, chaves primárias e estrangeiras, e todo o restante da configuração de cada tabela e do próprio banco. Pode-se pensar então em maneiras para se automatizar a criação desses arquivos, escrevendo marcações dentro do código somente quando alguma configuração particular for necessária. O restante pode ser mapeado com uma configuração padrão, que também pode ser configurada pelo usuário.

Dado um arquivo XML que descreva os atributos de cada entidade e seu relacionamento com as demais entidades, o NHibernate gera automaticamente o *script* SQL para carregar e armazenar os objetos no banco. Em sua última versão existe a possibilidade de se descrever o mapeamento dentro do próprio código-fonte, em forma de marcações, que funcionam como metadados de cada atributo dentro do código [3].

Resultados e Discussões

As marcações para o mapeamento por atributos são feitas entre *tags* ou caracteres especiais da própria linguagem, que descrevem que o trecho de código será analisado por outro programa. Essas marcações definem toda a configuração da classe e dos atributos, por isso devem ser feitas uma a uma, classe a classe, atributo por atributo, definindo também as relações entre os objetos, que por sua vez, são definidas no banco pelo NHibernate.

O NHibernate é um projeto *openSource*, ou seja, todo seu código pode ser baixado e alterado livremente, assim como modificado, aprovado por seus moderadores e publicado aos usuários da comunidade.

A automatização do mapeamento pode ser feita utilizando o próprio mapeamento por atributos, já implementado no Nhibernate, estendendo o código existente desse mapeamento e criando novas classes que agregariam

as novas funcionalidades e, ainda, aproveitando as marcações já definidas e documentadas pelo mesmo.

Considerações finais

As novas funcionalidades fornecidas pelas classes implementadas possibilitariam um mapeamento mais automatizado. Esse seria feito de uma forma padrão para todos os atributos do objeto, ou seja, não seria mais necessária uma marcação para cada atributo. Esse padrão poderia ser modificado e definido pelo usuário em uma marcação única, feita no início do código-fonte.

As marcações para atributos poderiam então apenas definir algumas particularidades, ou seja, somente quando se quiser que certo atributo tenha alguma configuração específica dentro do banco. O restante dos atributos seria mapeado automaticamente seguindo a configuração padrão definida no início da classe, facilitando ainda mais o mapeamento e diminuindo o esforço manual do usuário.

Referências

[1] PATERSON, Jim. **Simple Object Persistence with the db4o Object Database**. Disponível em: <http://www.onjava.com/pub/a/onjava/2004/12/01/db4o.html>. Acesso em: 23/04/2004

[2] KUATÉ, Pierre H., HARRIS Tobin. NHibernate in Action. Disponível em: <http://www.manning.com/kuate/>. Acesso em: 23/04/2008

[3] MAULO, Fábio. Site oficial do Nhibernate, Disponível em: <https://www.hibernate.org/343.html>. Acesso em: 23/04/2009.

SISTEMA DE CONTROLE, CADASTRO E AGENDAMENTO DE TRANSPORTES INTERMUNICIPAIS DE PACIENTES DO SISTEMA ÚNICO DE SAÚDE.

Vênyton Nathan Leandro Izidoro
venyton@yahoo.com.br

Universidade Estadual do Centro-Oeste - UNICENTRO

Palavras-chave: Java, DB4o, Interação Humano-Computador.

Resumo:
Este resumo descreve o desenvolvimento de um sistema de software que será utilizado na Secretaria Municipal da Saúde da cidade de Guarapuava. O sistema será desenvolvido em JAVA e fará uso do banco de objetos DB4o, servirá para o cadastro de pacientes que utilizam o transporte intermunicipal fornecido pela Prefeitura de Guarapuava para consultas médicas. O sistema também fará controle de vagas e das datas para as viagens oferecidas.

Introdução

Com a evolução da tecnologia da informação, empresas e órgãos públicos de todo o mundo estão informatizando seus recursos.

A Prefeitura Municipal de Guarapuava ainda está em processo de informatização, sendo que em muitos de seus setores o trabalho ainda é realizado de forma manual.

O objetivo deste projeto é o desenvolvimento de um sistema de software para informatizar os serviços fornecidos em uma das unidades da Secretaria Municipal da Saúde de Guarapuava, serviços esses que ainda são prestados manualmente, acarretando na falta de praticidade, acúmulo de documentos, e muitas outras dificuldades que podem ser evitadas com o emprego da tecnologia.

Materiais e Métodos

Esta ferramenta de software começou a ser desenvolvida em fevereiro de 2009. Ela visa automatizar o agendamento de viagens intermunicipais de pacientes do SUS (Sistema Único de Saúde) da cidade de Guarapuava.

O sistema será implantado no TFD (Tratamento Fora de Domicílio), que é um órgão da Secretaria Municipal da Saúde de Guarapuava que, entre outras obrigações, é responsável pelo encaminhamento de pacientes de Guarapuava para exames e tratamentos em hospitais de outros municípios. Esse setor faz em média de 650 agendamentos de viagens e 150 cadastros de pacientes por mês.

O objetivo do software é efetuar o cadastro dos pacientes e o agendamento de viagens para os mesmos, como também o controle do número de vagas por viagem nos veículos disponíveis, e gerar as tabelas diárias, mensais e anuais

de viagem. Serão utilizados conceitos de Engenharia de Software, Bancos de objetos e Interação Humano-Computador (IHC).

A ferramenta deve funcionar em uma rede de computadores que conta com um banco de objetos em comum. Ela está sendo desenvolvida em linguagem JAVA que será empregada com o banco de objetos DB4o, que é o banco de objetos de código aberto que possibilita aos desenvolvedores Java e .Net reduzir o tempo, o custo de desenvolvimento e alcançar altíssima perfumasse, [1].

A interface com o usuário deverá ser simples, robusta, e utilizará as boas práticas de Interação Humano-Computador, devido ao fato de que os usuários finais são, em geral, leigos com relação à informática.

A tela principal do sistema em sua fase atual pode ser observada na Figura 1.

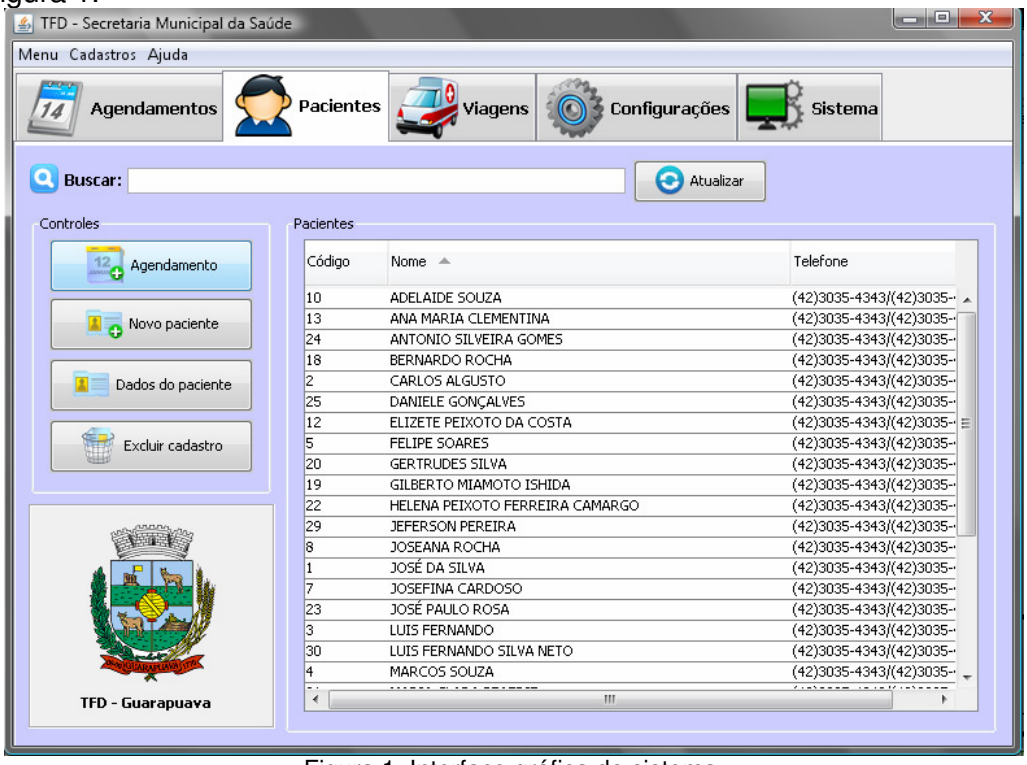

Figura 1. Interface gráfica do sistema.

Testes com os usuários finais e avaliação das interfaces também serão feitos no decorrer do projeto.

Resultados e Discussão

O sistema encontra-se em fase de desenvolvimento, as telas de interação com o usuário, comunicação com o banco de objetos e as funcionalidades gerais, como cadastros, relatórios e exclusões, estão em funcionamento.

Estão ainda em evolução o controle de acessos concorrentes à base de objetos, assim como os testes do sistema, testes com usuários e as avaliações de interface gráfica.

Considerações finais

Os usuários demonstraram satisfação com relação ao projeto que está em andamento, o sistema deverá facilitar o trabalho no setor e tornar ágil o cadastro e controle dos pacientes que utilizam o serviço.

Referências

[1] Monteiro, E. "**DB4o: Banco OO**". Disponível em http://imasters.uol.com.br/artigo/5056/bancodedados/db4o_banco_oo. Acessado em 16 de abril de 2009.

[2] Gamma, E., Helm, R., Johnson, R. e Vlissides, J. (1994) "**Padrões de Projeto: Soluções reutilizáveis de software orientado a objeto**".

[3] Amstel ,F. (2005) "**Afinal, o que é usabilidade?**". Disponível em http://usabilidoido.com.br/afinal_o_que_e_usabilidade.html.

[4] Ricarte I, M, L (2001) "**Programação Orientada a Objetos: Uma Abordagem com Java**". Disponível em: http://www.dca.fee.unicamp.br/cursos/PooJava/Aulas/poojava.pdf

ALGORITMO IMUNOFUZZY: UM NOVO MODELO DE COMPUTAÇÃO FLEXÍVEL

Luiz Antonio Carraro, Angelita Maria De Ré
e-mail: lacarraro@gmail.com

Universidade Estadual do Centro-Oeste

Palavras-chave: sistemas imunológicos artificiais, *fuzzy*, computação flexível.

Resumo:
Os Sistemas Imunológicos Artificiais (SIA) são constituídos de técnicas inspiradas na teoria do sistema imunológico do corpo humano. Essa metodologia foi comprovada como capaz de prover soluções para problemas em diversos domínios. Já os sistemas *fuzzy* são amplamente utilizados para a resolução de problemas que envolvem variáveis com valores incertos. Com o objetivo de tornar SIAs capazes de computar com variáveis incertas, este trabalho apresenta uma proposta para a criação de um novo modelo de computação flexível, através da hibridização entre SIAs e *fuzzy.*

Introdução

A computação flexível (*Soft Computing*) consiste em um novo paradigma de computação, usualmente visto como a fusão de um ou mais sistemas inteligentes para a solução de problemas. Diferente da computação convencional, esta é tolerante à imprecisão, incerteza, verdade parcial, e aproximação [1],[2]. Essa combinação ou hibridização de técnicas tornou possível a criação de várias soluções sofisticadas e bem sucedidas para problemas complexos do mundo real [3]. Cada técnica para solução de problemas possui suas características particulares como, por exemplo, redes neurais que possuem uma ampla capacidade de aprendizado e generalização. Zadeh [2] afirma que a computação flexível não consiste de uma mistura de técnicas, mas sim de uma parceria em que cada qual contribui com uma metodologia diferente para o domínio do problema, de forma a complementar a estratégia de solução.

O objetivo deste trabalho é apresentar uma proposta para a criação de um modelo de computação flexível baseado em Sistemas Imunológicos Artificiais e Sistemas *fuzzy*, ou seja, um algoritmo *imunofuzzy.*

Materiais e Métodos

Os Sistemas Imunológicos Artificiais (SIAs) podem ser definidos como sistemas computacionais inspirados pela imunologia teórica (mais especificamente, a imunologia dos seres humanos), com o objetivo de resolver problemas [1]. Atualmente, os SIAs vêm sendo utilizados em diversas áreas, como reconhecimento de padrões, detecção de falhas e anomalias, segurança,

otimização, controle, robótica, análise de dados, aprendizagem de máquina, dentre outras [2][4].

Entre as fases existentes na atividade do sistema imunológico deve-se destacar a maturação de afinidade. A afinidade se refere ao grau de ligação entre o anticorpo e o antígeno. Quanto maior a afinidade, mais forte é a ligação e, consequentemente, melhor é o reconhecimento e a resposta imunológica [5]. Considerando esse fator, Castro [3] argumentou que o reconhecimento de um antígeno pelo sistema imunológico é aproximado, portanto uma resposta imunológica pode ser ativada mesmo quando a ligação entre um antígeno e um anticorpo não é perfeita, ou seja, uma ligação aproximada foi encontrada. Isso demonstra que a imprecisão está presente no Sistema Imunológico Artificial e que a inclusão do *fuzzy* deve ser adequada para modelar alguns aspectos do sistema imunológico.

Considerando que o sistema imunológico é conceito para o desenvolvimento de soluções inteligentes, alguns algoritmos foram propostos como modelos para a solução de problemas em domínios mais específicos, como por exemplo: aiNet [6], CLONALG [7] e o RAIN [8].

A partir dos conceitos apresentados neste trabalho, o próximo passo será a definição de um algoritmo *imunofuzzy*, isto é, um algoritmo utilizando os conceitos do Sistema Imunológico Artificial em conjunto com *fuzzy* para permitir uma flexibilidade computacional para a solução de problemas que envolvam variáveis incertas e prover também uma maior similaridade com o sistema imunológico teórico.

Resultados e Discussão

Até o presente momento os SIAs têm demonstrado uma grande aplicabilidade em problemas, tais como: reconhecimento de padrões, detecção de falhas e anomalias, segurança, otimização, controle, robótica, análise de dados, aprendizagem de máquina, dentre outras. Além disso, os Sistemas *fuzzy* já estão consolidados como soluções eficientes para problemas com variáveis incertas. Sendo assim, acredita-se que a partir da integração do *fuzzy* com Sistemas Imunológicos Artificiais será possível criar sistemas híbridos que são mais plausíveis biologicamente e, ainda, que podem computar com parâmetros incompletos ou incertos.

Considerações finais

Atualmente o estágio de desenvolvimento deste projeto encontra-se na etapa de estudo sobre os Sistemas Imunológicos Artificiais, aplicações já desenvolvidas e sobre a futura integração com *fuzzy*. Com os resultados obtidos até o momento, constatou-se a existência de trabalhos correlatos desenvolvidos [3], o que possibilitou a identificação de uma proposta e uma justificativa para a integração das técnicas já citadas e o desenvolvimento do algoritmo *imunofuzzy*.

Referências

[1] INSTITUTO HUMANITAS UNICINOS. **Máquinas com rosto humano. Artigo de Javier Sampedro.** Disponível em www.unisinos.br/ihu/index.php. Acessado em 20/05/2009.

[2] ZADEH L. A. **What is soft computing?** Disponível em http://www.soft-computing.de/def.html. Acessado em 20/05/2009.

[3] CASTRO, L. N.; TIMMIS, J. I, **Artificial immune systems as a novel soft computing**. Springer-Verlag 2003.

[4] CASTRO, L. N.; ZUBEN, Fernando J. **Engenharia Imunológica: Desenvolvimento e Aplicação de Ferramentas Computacionais Inspiradas em Sistemas Imunológicos Artificiais**. Tese de doutorado – Engenharia Elétrica, UNICAMP, 2001.

[5] CASTRO, L. N., ZUBEN, Fernando J. **Artificial Immune Systems: Part 1 – Basic Theory And Applications.** Technical Report TR – DCA 01/99. UNICAMP, 1999.

[6] CASTRO, L. N., ZUBEN, Fernando J**. aiNet: An Artificial Immune Network for Data Analysis**. Idea Group Publishing, USA. Disponível em ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/lnunes/DMHA.pdf. Acessado em 20/05/2009.

[7] CASTRO L.N, Timmis J. **Artificial immune Systems: A New Computational Intelligence Paradigm**, Springer-Verlag, 2002.

[8] TIMMIS, J. **Artificial Immune Systems: A Novel Data Analysis Technique Inspired by the Immune Network Theory**, Ph.D. Dissertation, University of Wales, 2000.

DESENVOLVIMENTO NA PLATAFORMA LINUX COM MONO, MONODEVELOP E GTK#

Tony Alexander Hild
e-mail: tony_hild@yahoo.com

Universidade Estadual do Centro-Oeste

Palavras-chave: linux, mono, monodevelop, gtk#, .NET framework

Resumo:

Neste resumo será apresentada a plataforma de desenvolvimento Mono, com seu *framework*[2], *sua* IDE[3] MonoDevelop e a sua biblioteca gráfica GTK# (Gimp Tool Kit) para criação de *widgets*[4]. A intenção deste resumo é apenas explanar brevemente o conteúdo que será visto no minicurso que será ministrado pelo autor como o mesmo título na II JAI/UNICENTRO.

Introdução

O desenvolvimento de software multiplataforma ainda é um desafio devido à heterogeneidade dos sistemas operacionais e hardwares disponíveis.

O projeto Mono, com sua implementação livre do *framework* .NET, tem uma proposta bastante interessante. Além de ser compatível com a versão 2.0 do *framework* .NET, e implementar várias funcionalidades da versão 3.5, oferece inúmeras outras funcionalidades, bibliotecas e ferramentas para o desenvolvimento de software multiplataforma.

Com a IDE MonoDevelop, o desenvolvedor pode criar rapidamente aplicações utilizando tecnologias como C#, Visual Basic.NET, Asp.NET, ADO.NET, além de criar aplicações GTK#.

Materiais e Métodos

Mono

Mono é a plataforma de desenvolvimento de fonte aberta baseada no framework .NET que permite aos desenvolvedores construir aplicações multiplataforma.

Sua implementação é baseada nos padrões ECMA para o C# (ECMA 334) e para a Common Language Infraestructure (ECMA 335).

---

[2] Em desenvolvimento de software, um framework ou arcabouço é uma abstração que une códigos comuns entre vários projetos de software provendo uma funcionalidade genérica.

[3] IDE, do inglês Integrated Development Environment ou Ambiente Integrado de Desenvolvimento.

[4] Um widget é um componente de uma interface de usuário gráfica (GUI), que inclui janelas, botões, menus, ícones, barras de rolagem, etc.

Patrocinado pela Novell, o projeto tem uma comunidade ativa, e inclui tanto ferramentas de desenvolvimento quanto a infraestrutura necessária para executar aplicações .NET. Sua ambição é tornar-se a principal escolha para desenvolvimento de aplicações Linux.

Também inclui compiladores sendo: uma *runtime engine* [5] compatível com o padrão ECMA (*Common Language Runtime*, ou CLR), e várias bibliotecas compatíveis com o Microsoft .NET (incluindo ADO.NET, System.Windows.Forms e ASP.NET), além de bibliotecas próprias e de terceiros.

MonoDevelop

O MonoDevelop é uma IDE projetada principalmente para a linguagem C#, mas que também dá suporte a outras linguagens .NET. Ele permite que desenvolvedores escrevam rapidamente aplicações *desktop* e ASP.NET. Além disso, permite importar projetos criados no Visual Studio .NET, mantendo assim uma base de código única.

Suas principais características são:

- Edição avançada de texto – suporte para complementação de código para C# 3.0; modelos de código; *code folding*[6];
- Bancada de trabalho configurável – Janelas totalmente customizáveis; teclas de atalho definidas pelo usuário; ferramentas externas;
- Suporte multilinguagens – C#, Visual Basic.NET, C/C++, Vala, Boo, e muitas outras;
- Depurador integrado – para depurar aplicações nativas ou Mono;
- Designer Visual GTK# - pode-se criar facilmente aplicações GTK#;
- ASP.NET – pode-se criar projetos *web*;
- Outras ferramentas – controle de versão, integração com makefile, testes de unidade, empacotamento e distribuição, localização.

GTK#

O Gtk# é um *Toolkit*[7] de Interface de Usuário Gráfica para Mono e .NET. O projeto faz uma amarração com o toolkit gtk+ e várias outras bibliotecas do GNOME, permitindo o desenvolvimento de aplicações GNOME nativas usando os frameworks Mono e .NET.

Algumas de suas características são:

---

[5] Software que certas aplicações necessitam para serem executadas no computador.

[6] *Code folding*, habilidade para esconder blocos de código ou texto em arquivos de código.

[7] Um conjunto de ferramentas e blocos de construção básicos para interfaces gráficas de usuário.

- Multiplataforma (UNIX, Windows, MacOS);
- Grande variedade de Widgets/Controles;
- Internacionalização;
- Disponível para C#, Java, Python, VB.Net, etc.
- Suporte a construtor de Interface com Usuário;
- FOSS[8].

Considerações finais

O projeto Mono, além de ser uma alternativa a outras plataformas de desenvolvimento não baseadas em NET, como Java ou QT, oferece o necessário para desenvolver aplicações de propósito geral com produtividade e qualidade.

Por ser um software totalmente aberto, utilizando licenças aprovadas pela OSI (*Open Source Initiative*), a voz da comunidade tem um grande peso nas direções do projeto.

Sua utilização está em constante crescimento, permitindo que haja uma grande interoperabilidade entre software que antes somente rodavam em ambiente Windows, e que agora podem ser portados para diversas plataformas além do Linux, como OS/2, MacOS, Solaris, *BSD e *NUX.

Referências

[1]MONO. Sítio do projeto Mono. Disponível em: <http://www.mono-project.com>. Acesso em: 28 mai. 2009.

[2]MONODEVELOP. Sítio da IDE MonoDevelop. Disponível em: <http://www.monodevelop.com>. Acesso em: 28 mai. 2009.

[3]GTK#. Sítio da biblioteca GTK#. Disponível em: <http://http://www.mono-project.com/GtkSharp>. Acesso em: 28 mai. 2009.

[8] Free and Open Source Software ou Software Livre e de Código Aberto.

APLICAÇÃO DE PADRÕES DE PROJETO EM UM MODELO DE CONTROLE HOLÔNICO

Vênyton Nathan Leandro Izidoro, Marcos Antonio Quináia
venyton@yahoo.com.br, quinaia@unicentro.br

Universidade Estadual do Centro-Oeste - UNICENTRO

Palavras-chave: Padrões de Projeto, *Observer,* Controle Holônico.

Resumo:

Este artigo propõe a aplicação do padrão de projeto conhecido como *Observer* em um Modelo de Controle Holônico proposto por Jean Marcelo Simão quem tem o objetivo de tornar ágeis os sistemas modernos de produção. Também é apresentado um simulador feito em JAVA para demonstrar o funcionamento do padrão na arquitetura. O artigo traz também uma contextualização da arquitetura, o uso de padrões de projeto, o funcionamento do padrão *Observer* e os benefícios e consequências de sua aplicação.

Introdução

A idéia de padrões foi apresentada por Christopher Alexander em 1977 no contexto de arquitetura de prédios e cidades. “Cada padrão descreve um problema que ocorre repetidamente de novo e de novo em nosso ambiente, e então descreve a parte central da solução para aquele problema de uma forma que você pode usar esta solução um milhão de vezes, sem nunca implementá-la duas vezes da mesma forma”, [1].

A tese proposta por Simão, [5], apresenta um Modelo de Controle Holônico, este possui um mecanismo de notificação onde a Aplicação do padrão de projeto *Observer* é necessária.

O projeto propõe uma maneira de aplicar o padrão *Observer* no modelo e implementar um simulador em JAVA que demonstra esta aplicação.

Materiais e Métodos

A tese propõe um mecanismo de inferência (Figura 1), baseado na colaboração entre entidades desacopladas, que, quando alteradas, devem notificar as entidades das camadas seguintes, como mostra a Figura 1.

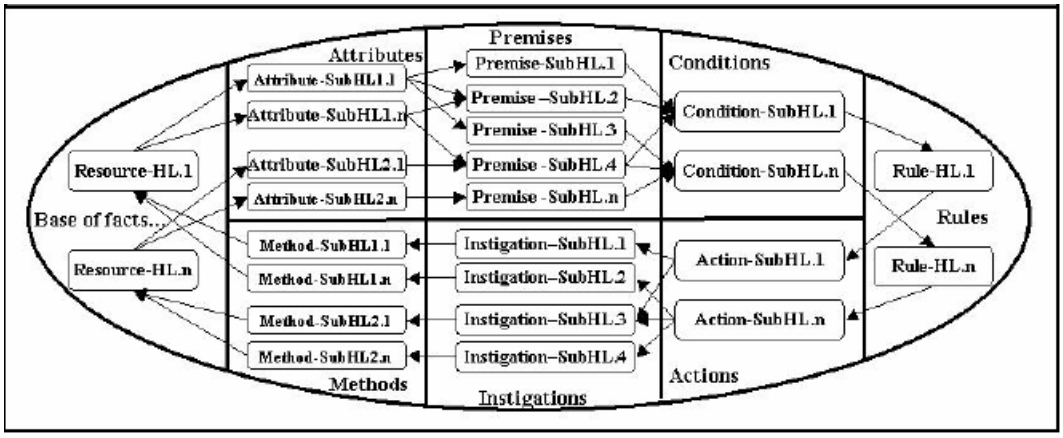

Figura 1. Mecanismo de notificação, [5].

Cada sub-holon, ao sofrer uma mudança de estado, ao invés de notificar todos os seus sucessores, notifica apenas os sucessores interessados em suas mudanças de estado, limitando, assim, a redundância temporal e tornando o mecanismo mais rápido e eficiente.

É nesse mecanismo de notificação que o *Observer* é aplicado. Uma vez que é necessária a atualização de um grupo de objetos quando ocorrem mudanças no ambiente.

O projeto encontra-se em sua fase final, o simulador foi desenvolvido em JAVA e encontra-se em fase de testes.

Resultados e Discussão

A aplicação do padrão no modelo deve garantir a funcionalidade do mesmo, visto que deve haver um mecanismo de atualização instantânea dos sub-holons da arquitetura.

Considerações finais

O padrão *Observer* mostrou-se adequado para o projeto, tornando a implementação do mecanismo de notificação proposto na arquitetura possível.

Referências

[1] Alexander, C., Ishkawa, S., Silverstein, M., Lacobson, M., Fiksdahl-King, I. e Angel, S. (1977) “A Pattern Language” - Oxford University Press, New York.

[2] Deschamps, F. (2004) “Padrões de Projeto: Uma introdução” - Universidade Federal de Santa Catarina – UFSC.

[3] Gamma, E., Helm, R., Johnson, R. e Vlissides, J. (1994) “Padrões de Projeto: Soluções reutilizáveis de software orientado a objeto”.

[4] Sauvé, J. P. (2000) “Análise e Projeto de Sistemas Orientados a Objeto”, http://www.dsc.ufcg.edu.br/~jacques/cursos/apoo/html/pat/observer.htm, Abril 2009.
[5] Simão, J. M. (2005) "A contribution to the development of a hms simulation tool and proposition of a meta-model for holonic control" – CEFET-PR.

FILTROS DIGITAIS PARA IMAGENS MÉDICAS

Thaysa Kozlik[1], Flávio C. Prodossimo[1], Carolina P. Almeida[2], Richard A. Gonçalves[1,2]
thaysakozlik@yahoo.com.br,frovas@gmail.com,carolpa@cpgei.cefetpr.br, richardgoncalves@cpgei.cefetpr.br

[1]Departamento de Ciência da Computação - UNICENTRO - Guarapuava, PR
[2]Programa de Pós Graduação em Engenharia Elétrica e Informática Industrial – UTFPR Curitiba, PR

Palavras-chave: Processamento Digital de Imagens, Imagens Médicas, Filtros Digitais.

Resumo:
O processamento de imagens médicas é de suma importância para o tratamento de diversas doenças, tais como o câncer. Este trabalho propõe o processamento de imagens médicas para serem utilizadas no tratamento radioterápico. Através da utilização de filtros busca-se definir os contornos da imagem para que o planejamento do tratamento radioterápico seja eficiente. Os filtros utilizados foram: *Sobel*, *Prewitt*, *Sharpen*, *Smoth* e Roberts. Com a execução de testes buscou-se o filtro ou combinação de filtros que melhor realçasse as características de cada imagem, permitindo diferenciar um tumor de um órgão sadio na imagem. Os resultados preliminares mostram que não existe um filtro, ou combinação destes, que se adapte a qualquer imagem a ser processada, necessitando de mais pesquisas para determinar uma metodologia de seleção de filtros.

Introdução

Atualmente, muitas pessoas sofrem de algum tipo de câncer [1]. Um tratamento comum é o uso da radioterapia [2]. Um passo importante do tratamento é o planejamento radioterápico, o qual determina quais são as áreas que estão sendo afetadas pelo tumor e quais são as áreas sadias, para que, dessa forma, os feixes de radiação possam ser emitidos na posição e dosagem corretas [3]. Este planejamento faz uso de imagens médicas. Para a automação do planejamento faz-se necessário pré-processar essas imagens de modo que todas as informações possam ser identificadas e tratadas separadamente [3]. Um planejamento eficaz melhora a qualidade de vida dos pacientes, pois evita que regiões sadias recebam altas doses de radiação, assim como permite que altas doses sejam depositadas no tumor.

No trabalho em andamento o pré-processamento das imagens médicas consiste na aplicação de filtros digitais para realçar seus contornos.

Materiais e Métodos

Imagens médicas requerem um pré-processamento, pois geralmente possuem ruídos [4]. A forma mais utilizada de se realizar este pré-processamento é através da aplicação de filtros [5]. A seguir os filtros utilizados até o momento, implementados na linguagem Java, serão brevemente descritos [5]:

- ***Sharpen:*** o filtro *Sharpen* dá uma nitidez maior à imagem. Ele define as áreas da imagem onde as mudanças de cor são mais significativas. A aplicação desse filtro é realizada através da *convolução* da imagem de entrada com uma máscara;
- ***Smooth:*** tem a finalidade de suavizar os contornos de uma imagem. Ao contrário do *Sharpen*, ele deixa as fronteiras menos nítidas, dificultando a detecção dos contornos.

- ***Sobel*****:** utilizado para a detecção das bordas verticais e horizontais da imagem de entrada, facilitando a visualização de seus componentes. No filtro *Sobel* são utilizadas 2 máscaras. Procura-se definir melhor as bordas da imagem através da sua *convolução* com as máscaras do filtro. A primeira é aplicada para a obtenção das bordas horizontais, e a segunda aplicada em seguida, obtém as bordas verticais e então é somada com o valor da primeira *convolução*, formando a fronteira completa das regiões da imagem;

- ***Prewitt*****:** possui a função de realçar as bordas. Sua aplicação é feita da mesma forma que o filtro *Sobel*, diferenciando-se apenas pelas máscaras;

- ***Roberts*****:** é um filtro mais simples, também utilizados para a detecção das bordas dos componentes de uma imagem. Na sua aplicação são utilizadas duas máscaras. Os pixels da imagem são multiplicados na diagonal pelos elementos da máscara. Assim como o *Sobel* e o *Prewitt*, primeiramente é utilizada uma máscara, depois é aplicada a outra e soma-se o resultado de ambas.

Resultados e Discussão

Foram utilizadas duas imagens médicas de crânios para a realização dos testes. Pode-se observar que a aplicação do filtro *Sobel* na primeira imagem facilitou a identificação dos contornos das suas regiões. Contudo, a aplicação do filtro *Prewitt* foi mais eficiente, deixando as fronteiras mais nítidas.

Já, para a segunda imagem, a aplicação dos filtros *Prewitt* e *Sobel* não foi eficiente quando utilizados individualmente devido a maior incidência de ruídos e a complexidade da segunda imagem com relação à primeira. Assim, testou-se várias combinações de filtros, sendo que aquela que obteve melhores resultados foi a aplicação conjunta dos filtros *Smooth* e *Sobel*.

Considerações finais

Dentre os filtros utilizados até o momento o que teve um melhor resultado, considerando o realce das bordas em imagens médicas, foi o filtro *Sobel*. Mas em imagens que possuem muitos ruídos o *Sobel,* sozinho, não produziu um resultado satisfatório. Para resolver esse problema a solução foi combinar filtros.

Pretende-se estudar outros modelos de filtros tais como os filtros morfológicos [6] e automatizar a metodologia de seleção de filtros para a identificação das regiões de imagens médicas utilizadas no planejamento radioterápico.

Referências

[1] INCA. Estimativa 2008: Incidência do Câncer no Brasil. Disponível em: http://www.inca.gov.br/estimativa/2008/versaofinal.pdf.

[2] CAMARGO, A. C. H. Introdução Sobre o Câncer. Disponível em: Http://www.hcanc.org.br/index.php?page=12 - Acessado em: 11/03/2009.

[3] ELBERN, A. Noções Básicas de Radioterapia – 1. Notas de Aula. www.prorad.com.br/pro/Radioterapia-1.pdf - Acessado em 12/03/2009.

[4] DOUGHERTY, G. **Digital Image Processing for Medical Applications**, Cambridge University Press, 2009.

[5] GONZALEZ, R.; WOODS, R. **Processamento de Imagens Digitais**. 1ª Edição, Edgard Blücher, 2000.

[6] SOILLE, P. **Morphological Image Analysis: Principles and Applications**. Springer, 2007.

ARQUITETURA DE UMA PLATAFORMA PARA SISTEMAS COMERCIAIS EM EJB 3.0

Julio César Araújo Galvão Filho
juliogalvaofilho@yahoo.com.br

Universidade Estadual do Centro-Oeste - UNICENTRO
Departamento de Ciência da Computação

Palavras-chave: Sistemas comerciais, JEE, EJB 3.0

Resumo:
Atualmente, um diferencial em sistemas comerciais é a característica de uma aplicação distribuída multicamada, onde as informações em fase de processamento podem transitar entre diferentes camadas físicas e lógicas. A plataforma Java, em sua versão empresarial (JEE), provê importantes recursos tecnológicos para a implementação de tais sistemas, como J*ava Server Faces* (JSF) e *Enterprise JavaBeans* (EJB). Este trabalho tem o objetivo de demonstrar a arquitetura de uma plataforma para sistemas comerciais baseada na tecnologia EJB 3.0.

Introdução

A plataforma JEE[9] utiliza um modelo de aplicativo multicamada distribuído para aplicativos empresariais [1]. Esse modelo permite clara separação entre componentes relacionados às camadas de persistência, lógica de negócios e apresentação de dados, resultando em menor acoplamento e maior escalabilidade. Sua característica distribuída consiste na capacidade de trabalhar com componentes situados em diferentes localizações físicas. Entende-se por componente uma unidade de software funcional independente, que pode comunicar-se com outros componentes.

A tecnologia *Enterprise JavaBeans* (EJB) é a arquitetura de componentes JEE relacionados ao lado servidor [4]. Um *enterprise bean* implementa a tecnologia EJB e é executado em um contêiner, que por sua vez integra o *middlewar* denominado servidor de aplicação. Os *enterprise beans* encapsulam a lógica de negócios da aplicação. O contêiner é o responsável por tratar serviços de sistema, como segurança e controle de concorrência.

O uso de EJB simplifica o desenvolvimento de aplicações distribuídas. Suas principais vantagens são: o foco na lógica de negócio, o reuso de componentes e o gerenciamento através de contêiner [3].

Este trabalho demonstra a arquitetura de uma plataforma tecnológica para múltiplos sistemas comerciais, conforme detalhado na seção seguinte.

[9] http://java.sun.com/javaee

Materiais e Métodos

Para a melhor integração de diversos módulos comerciais é necessária uma base comum. Esta base pode ser definida através da arquitetura de uma plataforma tecnológica, onde funcionalidades gerais devem ser providas.

A plataforma proposta tem como principal objetivo a separação lógica das camadas de persistência e de negócios. Para isso, utiliza-se *enterprise beans*. A Figura 1 demonstra a arquitetura física e lógica da plataforma em seu *back-end*. A arquitetura *front-end* é detalhada em [2]. A seguir, conceitua-se a divisão lógica definida.

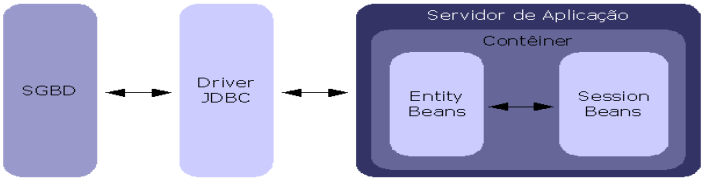

Figura 1. Arquitetura física e lógica da plataforma (*back-end*)

Em relação à camada de persistência, utilizam-se os *entity beans*, ou *beans* de entidade. Estes representam objetos de negócios que constam em meio de armazenamento persistente, geralmente bancos de dados. Nesse caso, cada classe de objeto é relacionada a uma tabela de um banco de dados relacional, onde cada tupla gera uma instância. Trata-se de objetos de domínio onde são definidos apenas atributos e métodos acessadores / modificadores. O relacionamento entre atributos e campos da base de dados ocorre através de anotações definidas pela *Java Persistence API* (JPA) [5], enquanto o *driver* JDBC [6] é responsável pela comunicação entre o SGBD e a plataforma Java.

A lógica de negócios é encapsulada em *session beans*, ou *beans* de sessão. Cada instância de um *session bean* tem a função de atender a requisição de um cliente por funcionalidades específicas, que dizem respeito aos *entity beans*. Essas funcionalidades permitem as operações básicas de banco de dados (inserção, busca, atualização e remoção) em nível de objeto, além da navegação (primeiro, último, anterior e próximo).

Os *session beans* implementados, por não conter variáveis de instância, não necessitam armazenar o estado corrente. Assim, qualquer cliente pode ser atendido por qualquer *session*, caracterizando *beans* de sessão *stateless*.

Resultados e Discussão

A utilização de *entity beans* permite maior flexibilidade quanto ao meio físico de armazenamento. Os objetos de domínio podem ser migrados sobre diferentes bases de dados, com pouca ou nenhuma alteração. Dessa forma, a lógica de negócios permanece inalterada, mesmo com mudanças sobre o modelo de dados. Isso constitui uma importante vantagem no contexto de uma plataforma tecnológica, pois garante que diversos módulos sejam integrados sem impactos drásticos na aplicação.

Os *session beans*, apesar de implementar a lógica de negócios, não são considerados os únicos componentes de controle. Outros *frameworks* podem operar simultaneamente, como implementações JSF. Assim, pode-se separar a

lógica de negócios do servidor da lógica do cliente, bem como operar de maneira específica para diferentes tipos de camada de apresentação, como clientes *web*, *desktop* e *mobile*.

Considerações finais

O presente trabalho, em fase de desenvolvimento, apresenta resultados parciais satisfatórios. O modelo de arquitetura encontra-se bem definido e os componentes comuns aos módulos integrantes da plataforma estão sendo implementados. O primeiro módulo comercial a ser integrado refere-se às funcionalidades de nota fiscal eletrônica (NF-e).

Referências

[1] BODOFF, Stephanie et al. **Tutorial do J2EE Enterprise Edition 1.4.** Tradução da 2ª Edição, Editora Ciência Moderna, 2005.

[2] FORBECK, Felipe Ribas. Especificação da camada de apresentação de uma arquitetura para sistemas comerciais Utilizando JSF 1.2. Guarapuava: Universidade Estadual do Centro-Oeste, 2009.

[3] FRANTZ, Rafael Zancan. **Enterprise JavaBeans 3.0.** Disponível em http://www.j2eebrasil.com.br/mostrar/101. Acessado em 21/05/2009.

[4] SUN. **Enterprise JavaBeans Technology**. Disponível em http://java.sun.com/products/ejb - Acessado em 22/05/2009.

[5] SUN. **Java Persistence API**. Disponível em http://java.sun.com/javaee/technologies/persistence.jsp - Acessado em 08/06/2009.

[6] SUN. **JDBC Overview**. Disponível em http://java.sun.com/products/jdbc/overview.html - Acessado em 07/06/2009.

RECURSOS PARA O DESENVOLVIMENTO DE RELATÓRIOS EM JAVA

Laila Maria Gomes Gechele (UNICENTRO),
lailamariah@hotmail.com

Universidade Estadual do Centro Oeste do Paraná/ Departamento de Ciência da Computação
Guarapuava – PR

Palavras-chave: relatórios em Java, JasperReports, iReport.

Resumo:
Apresenta-se uma síntese das tecnologias mais utilizadas para auxiliar na programação de relatórios em Java. O *framework* JasperReports [1] é uma ferramenta *open source* escrita em Java para geração de relatórios. Permite gerar dinamicamente relatórios em diversos formatos, entre eles: PDF, HTML, XLS, CSV e XML. É facilmente usado em conjunto com o iReport, que é um "desenhador" visual de relatórios, um gerador de layout, também escrito em Java, que gera descrições XML no formato esperado pelo JasperReports. Essas duas ferramentas juntas são de grande valia para auxiliar na confecção de relatórios eficientes e altamente profissionais.

Introdução

A maioria dos desenvolvedores de sistemas de informação empresarial habituou-se a contar com geradores de relatórios visuais em seus ambientes de desenvolvimento RAD desde que o Visual Basic 3 incorporou uma versão reduzida do *Crystal Reports*, entretanto algumas IDEs populares como o Eclipse ainda não possuem ferramentas do gênero, de modo que muitos desenvolvedores acreditam que este tipo de recurso não está disponível em Java [2].

Parte do problema se originou do fato de a maioria dos geradores de relatórios populares serem aplicações independentes escritas na linguagem C e, na maioria das vezes, suporta apenas o sistema operacional Windows, sendo assim incompatíveis com a portabilidade proporcionada pelo Java. Outro motivo para essa confusão provém do fato de a maioria dos geradores de relatórios estarem disponíveis para aquisição em separado e utilizar chamadas a *jni* ou requisições *http*, sendo possível serem utilizados como parte de qualquer sistema, independente da linguagem de programação na qual foram escritos.

Mas isto não quer dizer que não existam geradores de relatórios escritos em Java capazes de atender à maioria dos desenvolvedores, inclusive opções livres. Na verdade existem muitos, inclusive vários premiados em publicações especializadas. Utilizar um gerador escrito em Java tem muitas vantagens além da portabilidade, por exemplo, usar como fonte de dados coleções de objetos Java recuperadas via *ejbs*, *hibernate*, *jaxb* ou *web services*. Geradores escritos

em outras linguagens necessitarão de acesso direto ao banco de dados, e não poderão interagir de forma simples e clara com objetos Java.

Este artigo descreve duas ferramentas que se complementam mutuamente: o JasperReports e o iReport.

Um dos projetos do mundo do software livre que vem ganhando mais destaque nos últimos tempos é o JasperReports, uma vez que gerar relatórios com informações consolidadas de alguma parte do sistema é um dos passos fundamentais de qualquer aplicação e fazer isso de forma eficiente, com resultado profissional e sem custo, é o que este *framework* proporciona [3].

Por fazer uso de XML, sua flexibilidade e possibilidade de integração com outras ferramentas são garantidas e, gerar relatórios diretamente dos dados de um determinado banco de dados, necessita de poucos códigos.

O iReport é uma ferramenta que permite definir o design do relatório dentro de um ambiente gráfico, contendo "todos" os recursos que a biblioteca Jasper oferece. É possível definir relatórios com designers modernos e complexos sem ter que escrever o código XML, pois é gerado automaticamente. O ambiente oferece atalhos para tarefas de compilação e visualização do relatório, permitindo a realização de testes.

Materiais e Métodos

O JasperReports recebe como entrada uma descrição estruturada do relatório na forma de um documento XML, como localização dos campos a serem preenchidos e seus respectivos nomes, para futuro mapeamento. O arquivo *jasperreports.dtd* gera uma saída tanto diretamente na impressora (incluindo uma pré-visualização baseada em Java Swing), quanto como um documento *pdf*, *html*, *xls* ou *csv*.

Usando o XML o designer pode definir textos estáticos, imagens, linhas e formas geométricas como retângulos e elipses, e suas localizações dentro do relatório. Além disso, pode-se ainda definir os campos que serão preenchidos dinamicamente a partir de uma base de dados.

O arquivo XML é compilado gerando um arquivo **.jasper**, onde as expressões Java existentes dentro do XML serão verificadas em tempo de compilação. Diferentes objetos JasperReports são usados para representar as etapas do processo de geração de relatórios, dentre eles estão [3]:

- **JasperDesign**: Representa a definição do relatório. A partir de um template XML é criado um JasperDesign.
- **JasperReport**: Representa o JasperDesign compilado. O processo de compilação verifica o design do relatório e compila o design em um JasperReport.
- **JasperPrint**: Representa o relatório gerado. É criado um JasperPrint a partir de um JasperReport, contendo o relatório preenchido.

Para simplificar o desenvolvimento o iReport oferece um *plug-in*, fazendo a integração do iReport a IDE NetBeans, o que permite o uso de uma única IDE quando o aplicativo é usado. Edson Gonçalves resume *"...O iReport é uma ferramenta que está preparada para construir visualmente os mais complexos relatórios dentro dos limites de JasperReports..."* [4].

Resultados e Discussão

Para produzir um relatório precisa-se fornecer dados ao Jasper. Esses dados podem ser tanto consultas SQL inseridas no código XML, como podem ser gerados por uma classe Java, a partir de um objeto ResultSet, que é passado às classes do Jasper para o preenchimento do relatório. O JasperReports suporta vários tipos de fonte de dados (*datasources*) através de uma interface específica chamada JRDataSource. Há uma implementação padrão desta interface para objetos ResultSet, chamada JRResultSetDataSource.

O iReport possui toda a biblioteca JasperReport, portanto usar o iReport implica usar o JasperReport.

Considerações finais

Ao utilizar as ferramentas acima citadas percebe-se um enorme ganho em produtividade e facilidade no desenvolvimento de relatórios em aplicações Java, principalmente pela facilidade encontrada com a comunicação entre o JasperReport e as estruturas consultadas no Banco de Dados.

Referências

[1] JASPERFORGE - Disponível em http://jasperforge.org/ - Acessado em 19/03/2009.

[2] LOZANO, Fernando. Java Magazine: Relatórios Corporativos. Nº 13, 2008.

[3] Departamento de Sistemas e Computação da Universidade Federal de Campina Grande. Jacques Sauvé. Disponível em http://www.dsc.ufcg.edu.br/~jacques/cursos/daca/html/documentviews/relatorios.htm - Acessado em 11/03/2009.

[4] GONÇALVES, Edson. Desenvolvendo Relatórios Profissionais com iReport com NetBeans IDE. 1ª Edição. Editora Ciência Moderna, 2009.

CURSO PILOTO DE CONSTRUÇÃO DE APLICAÇÕES COM INTERFACES RICAS (RIA) NA WEB USANDO JAVAFX

Rafael Garcia Valle Galego, Marcos Aurélio Pedroso Leandro
Rafaelgalego@gmail.com
Marcos_unicentro@yahoo.com.br

Universidade Estadual do Centro-Oeste

Palavras-chave: JavaFX, *Rich Internet Application*, *Script*, *Mobile*.

Resumo:

Este resumo apresenta as principais características sobre a tecnologia desenvolvida pela Sun Microsystems denominada JavaFX, além de resultados obtidos até o momento, problemas encontrados e os próximos passos a serem tomados no desenvolvimento do trabalho.

Introdução

A crescente utilização de aplicações *web* fez surgir à idéia de um ambiente *on-line* dinâmico e colaborativo entre os usuários para a organização de seu conteúdo. Além disso, há uma preocupação em relação à interface com o usuário, buscando melhorar sua experiência na utilização de serviços *web* [5].

O JavaFX é uma tecnologia que permite o desenvolvimento de aplicativos RIA (*Rich Internet Application*) que visam melhorar a experiência do usuário na utilização de serviços *web*.

Tendo em vista que a tecnologia é recente e o número de pessoas que tiveram acesso é pequeno, optou-se pela elaboração do curso voltado a pessoas que dominem Orientação a Objetos com o intuito de apresentar, enriquecer e ampliar os conhecimentos sobre o JavaFX.

Materiais e Métodos

O JavaFX (acrônimo para "*Java Effects*" [2]) é uma família de produtos e tecnologias da Sun Microsystems, apresentado inicialmente na conferência de desenvolvedores *JavaOne* em maio de 2007. Seu objetivo é permitir o desenvolvimento e implantação de aplicativos em dispositivos utilizados por diversos tipos de usuários, tais como: telefones, televisores, navegadores e sistemas de automóveis [1].

A RIA busca levar a liberdade e a riqueza visual dos componentes *desktops* para as aplicações *web* através de interfaces mais robustas e níveis mais altos de interatividade [2]. Além disso, são aplicações implementadas no servidor e que tiram vantagem da tecnologia cliente para prover uma nova classe de *websites* interativos com a sofisticação de aplicações *desktop*, mas

que não comprometem a facilidade de desenvolvimento, implementação e manuseio dos aplicativos *web* [3].

A família do JavaFX é composta por duas tecnologias: JavaFX *Mobile* e JavaFX *Script* [1]. O JavaFX *Mobile*, lançado em março de 2009, é um sistema operacional baseado em Java para dispositivos móveis que possui o Java SE (Java *Standard Edition*) e o Java ME (Java *Micro Edition*) sendo executados sob o *kernel* Linux. O principal foco são os *smartphones* com interfaces elaboradas (*touch-screen*) [2].

Já o JavaFX *Script* é uma DSL (*Domain Specific Languages*) para definição e programação de RIAs. Ela é estaticamente *tipada* (toda variável e parâmetro tem um tipo fixo escolhido pelo programador [4]) e compilada, além de ter acesso a todos os pacotes existentes na plataforma Java [2].

O estudo dá-se através de *e-books*, artigos, tutoriais e materiais de aula (*slides*) encontrados na Internet. Os tutoriais são utilizados para fixar os conteúdos, uma vez que é possível praticar o que é visto em teoria.

Resultados e Discussão

Os resultados até o momento são irrisórios, uma vez que somente a atividade de estudo da tecnologia está em desenvolvimento, porém observa-se que o número de pessoas interessadas em tal tecnologia é grande, uma vez que os fóruns de discussões sobre o assunto sempre trazem dúvidas de desenvolvedores novatos.

Os problemas encontrados até o momento dizem respeito aos materiais existentes na Internet, pois a maioria não traz explicações por completo, ou seja, demonstra apenas alguns passos a serem seguidos no desenvolvimento, e os demais devem ser deduzidos ou buscados em outros *sites*.

Considerações finais

As próximas etapas do desenvolvimento visam à elaboração do curso a ser ministrado ao fim do estudo, e envolve atividades tais como: definição de carga horária, número de alunos por turma, planejamento de conteúdos, metodologia de ensino, entre outras atividades pertinentes.

Referências

[1] WEAVER, James L. **JavaFX™ Script: Dinamic JavaTM Scripting for Rich Internet/Cliente-Side Applications**. New York: First Press, 2007.

[2] SANTOS, Wanderson. **JavaFX: Aplicações com Interface Rica na Plataforma Java**. Disponível em http://www.slideshare.net/rodrigofm/java-fx-aplicaes-com-interface-rica-na-plataforma-java-presentation. Acessado em 24/04/2009.

[3] COSTA, Henry F. D. **Internet Rica (RIA)**. Disponível em http://www.henry.eti.br/pagina.php?idPagina=248. Acessado em 22/03/2009.

[4] RAMALHO, Franklin. **PLP – Valores e Tipos (parte II)**. Disponível em http://www.dsc.ufcg.edu.br/~franklin/disciplinas/2005-2/PLP/AULA_02_PLP2005-2.pdf. Acessado em 24/04/2009.

[5] FOLHA de São Paulo. **Entenda o que é a web 2.0**. Disponível em HTTP://www1.folha.uol.com.br/folha/informática/ult123u20172.shtml. Acessado em 22/03/2009.

DESENVOLVIMENTO DE UM *WEBSERVICE* PARA CONTROLE DE PROCESSOS

João Paulo Minoru Kobayashi Katayama
e-mail: minoru.koka@gmail.com

Tony Alexander Hild
e-mail: tony_hild@yahoo.com

Universidade Estadual do Centro-Oeste

Palavras-chave: desenvolvimento, *webservice,* Imob.

Resumo:
A empresa BrainSoft desenvolve um sistema chamado Imob para automação de cartórios. Para facilitar a centralização e replicação de informações será desenvolvido um *webservice.* Essas informações serão consumidas por uma aplicação *web* permitindo aos clientes dos cartórios acessá-las de qualquer local e acompanhar o andamento dos processos.

Introdução

Com o crescimento de usuários da internet e a facilidade de acesso a esse meio, empresas têm investido cada vez mais em serviços *on-line.* Os serviços *on-line* tem o objetivo de facilitar as tarefas realizadas pelos usuários sem que eles tenham que sair de casa. A internet também possibilitou que sistemas se comuniquem e troquem informações entre si.

O *webservice* é um componente programável, com um conjunto de operações acessíveis via protocolos utilizados na *internet* [4], permitindo que sistemas distintos, com diferentes tecnologias e plataformas, possam se conectar de maneira padronizada e executem procedimentos remotos sobre o protocolo *http*, mesmo através de *firewalls* [4, p.19].

O Imob é um sistema para cartórios desenvolvido pela empresa Brainsoft. Ele é responsável por gerenciar as principais tarefas realizadas pelos cartórios, também desenvolve algumas tarefas de segurança. Uma das tarefas realizadas pelos cartórios é o gerenciamento de processos. Os clientes dos cartórios necessitam fazer o acompanhamento dos processos com o objetivo de finalizar uma eventual exigência ou pagamento. Para realizar esse acompanhamento eles, muitas vezes, necessitam se locomover até os respectivos cartórios. O desenvolvimento e implementação de um *webservice* permitirá, em uma arquitetura cliente/servidor, que seja replicado em lote[10] os dados dos processos, possibilitando aos clientes dos cartórios acompanhá-los utilizando uma página *web*.

[10] Em um período de tempo específico, um lote de processos é enviado para o servidor.

O objetivo deste trabalho é apresentar o desenvolvimento de um *webservice* para controle de processos. Nas sessões seguintes será feita uma descrição dos materiais e métodos utilizados.

Materiais e Métodos

Para desenvolver o *webservice* será utilizado o ambiente de desenvolvimento da Microsoft [6], com os seguintes softwares: Visual Studio 2008 – conjunto de ferramentas de desenvolvimento; Microsoft Visual SourceSafe – sistema de controle de versão em nível de arquivo e Microsoft SQL Server 2008 – sistema gerenciador de banco de dados. A linguagem de programação Visual Basic e os *softwares* da Microsoft serão utilizados em virtude do ambiente de desenvolvimento da empresa.

O Nunit e NMock serão utilizados para testes. Nunit é um *framework* de testes de unidades para todas as linguagens .Net [8]. Nmock é um *framework* para criação de *Mock Objects,* que são objetos que simulam o comportamento de objetos reais de forma controlada [3].

A arquitetura do sistema é baseada no modelo de 3 camadas, sendo elas: camada de apresentação, negócio e dados. Esse modelo de arquitetura permite que as camadas sejam independentes, ou seja, a troca, alteração ou correção de uma dessas camadas não afeta as demais [5].

O *webservice* é uma interface do Imob com o servidor de replicação de dados, onde se encontra a camada de negócios e acesso a dados. A camada de acesso a dados utilizará o NHibernate [7] para mapeamento de objeto relacional.

O banco de dados de replicação será alimentado pelo Imob e acessado por uma aplicação ASP.NET [1] (a ser desenvolvida), por meio da camada de negócios.

Para a implementação do *webservice* será utilizada a técnica de Desenvolvimento Orientado a Testes, que incentiva a simplicidade, aumenta a confiança, ajuda na documentação e facilita a refatoração [2].

Considerações finais

Atualmente, a competitividade do mercado exige que as empresas procurem meios para conquistar cada vez mais clientes. Dentre esses meios estão os sistemas *Web* que disponibilizam serviços *on-line.* O desenvolvimento do *webservice* permitirá que os clientes dos cartórios acessem com facilidade e comodidade quaisquer informações relativas aos seus processos.

Esse trabalho irá proporcionar um aprendizado em linguagens e ferramentas da plataforma .NET, desenvolvimento de *webservices*, aplicações *web* e segurança.

Referências

[1] ASP.NET. **Microsoft ASP.NET.** Disponíveis em http://www.asp.net/. Acessado em 22 mai. 2009.

[2] BECK, Kent. **Test Driven Development:** By Exemple. Addison-Wesley Professional, 2002.

[3] BURK, Eric M., COYNER, Brian M. **Java Extreme Programming Cookbook.** O'Reilly, 2003.

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

[5] QUADRADOS, Moacir. **Gerência de projeto de software**. Florianópolis: Visual Books, 2002.

[6] MICROSOFT. **Microsoft Developer Network.** Disponível em http://msdn.microsoft.com. Acessado em 21/05/2009.

[7] NHIBERNATE. **Nhibernate for .NET.** Disponível em https://www.hibernate.org/343.html. Acessado em 21/05/2009.

[8] Nunit. Nunit. Disponível em http://www.nunit.org/index.php. Acessado em 21 mai. 2009.

ESPECIFICAÇÃO DA CAMADA DE APRESENTAÇÃO DE UMA ARQUITETURA PARA SISTEMAS COMERCIAIS UTILIZANDO JSF 1.2

Felipe Ribas Forbeck
e-mail: fforbeck@gmail.com

Universidade Estadual do Centro-Oeste

Palavras-chave: plataforma tecnológica, sistemas comerciais, camada de apresentação, *java server faces.*

Resumo:
Existem diversas tecnologias existentes para desenvolvimento e implementação da camada de apresentação no ambiente *web.* Assim, determinar qual tecnologia utilizar em um projeto requer um estudo mais profundo e detalhado. Este trabalho apresenta a tecnologia escolhida para desenvolver a camada de apresentação de uma plataforma tecnológica para a empresa Teorema Informática. Sendo esta plataforma desenvolvida inicialmente em um ambiente *web,* espera-se simular o ambiente e visual *desktop* com a tecnologia aqui abordada e especificada.

Introdução

Quando se trata de mudanças em sistemas antigos a situação é de risco, porque há dependência não só das regras de negócio da empresa, no caso Teorema Informática, mas também de fatores como leis, economia, entre outros.

Assim, pensa-se em modelar e conceber um protótipo de uma arquitetura funcional, ou seja, uma arquitetura distribuída e de alta granularidade. Tal arquitetura atuará como modelo de desenvolvimento para a empresa. Ademais, será construída utilizando tecnologia Java EE 5 [1] e permitirá a integração de diferentes módulos de sistemas comerciais.

Além de buscar por um modelo de arquitetura de desenvolvimento que opere como uma plataforma tecnológica, a empresa Teorema Informática preocupa-se em otimizar não só o desenvolvimento de seus próprios sistemas, mas também a usabilidade dos mesmos, pensando nos clientes.

O objetivo é migrar todo o sistema convencional *Delphi* da empresa para a plataforma *Java* e, inicialmente, definiu-se que será em ambiente *web.* Assim, é importante aproximar a aparência *desktop,* com uma interface mais intuitiva, rica e amigável, sem perder funcionalidades. Levando em consideração esses aspectos, e outros descritos a seguir, foram realizadas pesquisas e estudos sobre as tecnologias disponíveis no mercado para tal propósito.

Definiu-se que a camada de apresentação *web* será toda desenvolvida utilizando a tecnologia *ICEfaces* [2], que é uma implementação do *framework Java Sever Faces* (JSF) proposto por *Sun Microsystems* e incorporado a especificação JEE 5 [1], designado com o intuito de facilitar o desenvolvimento de aplicações *web* através de componentes de interface de usuário (GUI), conectando esses componentes a objetos lógicos. JSF é classificado como

paradigma de programação visual de *User-interfaces* aplicado à *web* e de fácil aprendizado.

Vale ressaltar que a modelagem e desenvolvimento do protótipo da arquitetura é baseado no *design pattern* MVC, sendo aqui abordada a camada *View* e que [3] relata as funcionalidades do *back-end* da mesma aplicação, tratando das camadas MVC aqui não descritas: *Model* e *Controller*.

Materiais e Métodos

Priorizando o aspecto visual da aplicação, solicitado pela empresa supracitada, para manter o visual *desktop* na *web*, busca-se por tecnologias que ofereçam este atributo. Por meio de pesquisas na internet e materiais de referência como em CONCEIÇÃO [4], percebeu-se que a tecnologia *ICEfaces* é a melhor e vem sendo utilizada por grandes corporações como: NASA, HP, Siemens e Cisco, sendo que estas e demais corporações podem ser encontradas em ICESOFT [2].

A tecnologia oferece interfaces ricas (RIA), um *framework* para desenvolvimento, proporciona integração com outras bibliotecas de componentes, é *open-source*, mecanismo seguro, possui diversos materiais de suporte e auxílio, bem como documentação e API de seus componentes, além de trabalhar com a tecnologia *Java* opera com o mecanismo *Ajax Push,* que provê mais eficiência na *renderização* das páginas *web* (utilizando JSF *lifecycle*)*,* monitoramento das requisições e compartilhamento da conexão do *browser.*

Vale ressaltar que o *framework* JSF oferece recursos não abordados em outras tecnologias estudadas, por exemplo: se tornou um padrão; uso de componentes (GUI); conceito de evento para os componentes (muito parecido com os componentes *swing*); gerenciamento dos estados dos componentes. O *framework* incorpora alguns módulos como: componentes, eventos, validação e conversões, navegabilidade e *backbeans.*

Um fator importante e crucial para a escolha da tecnologia é a capacidade de reutilizar códigos existentes, no caso: componentes e facilidade de manutenção. Características estas, encontradas na ferramenta *ICEfaces.*

Outro diferencial é a aplicabilidade do conceito de navegação orientada à estados e não à páginas, que a ferramenta oferece. Tal conceito consiste em diminuir o número de páginas da aplicação e manter o fluxo de navegação centralizado, diminuindo a complexidade do projeto e ganhando assim, em performance, usabilidade para o usuário, bem como, explorando eficientemente os recursos *Ajax* em cada página.

Resultados e Discussão

A integração das tecnologias utilizadas na camada *View* com o projeto todo, nem sempre é trivial. Neste caso, não houve problemas, pois *ICEfaces* foi projetada para trabalhar com *Java,* tecnologia utilizada no desenvolvimento das outras camadas do projeto. Possui um funcionamento muito simples, trabalhando com navegação de páginas orientadas à estados, e separando claramente as regras pertinente ao cliente do resto da aplicação.

Ademais, facilita o desenvolvimento de interfaces *web* oferecendo recursos que se assemelham ao desenvolvimento de interfaces *desktop* com *Swing*. A ferramenta utiliza uma linguagem própria e muito semelhante ao *html* por usar *tags.* Possui extensões para utilizar *Expression Language* que facilita obter propriedades de componentes / objetos da aplicação.

Considerações finais

Assim, por meio dos recursos apresentados e ofertados pela tecnologia, que, principalmente, aproxima o visual do ambiente *web* para o *desktop* e, além disso, pode ser integrada com a tecnologia Java, ICEfaces é utilizada no projeto de desenvolvimento da camada de apresentação da plataforma tecnológica. Em fase de *prototipação* das interfaces gráficas, a tecnologia apresenta resultados parciais satisfatórios.

Referências

[1] SUN, Microsystems. ***The Java EE 5 tutorial*.** Disponível em http://java.sun.com/javaee/5/docs/tutorial/doc/. Acessado 23 de maio de 2009.

[2] ICESOFT. ***ICEFaces Open Source Ajax, JSF Java Framework*.** Disponível em http://www.icefaces.org. Acessado em 23 de maio de 2009.

[3] FILHO, Júlio César Araújo Galvão. **Arquitetura de uma plataforma para sistemas comerciais em EJB 3.0**. Guarapuava: Universidade Estadual do Centro-Oeste, 2009.

[4] CONCEIÇÃO, Rodrigo Menezes da. ***Java Server Faces* (JSF): Um estudo comparativo entre bibliotecas de componentes**. Universidade Tiradentes, 2009.

DESENVOLVIMENTO DE UM SISTEMA *WEB* UTILIZANDO TECNOLOGIA RIA E PRÁTICAS ÁGEIS

Andre Brito Fonseca
andrebritofonseca@gmail.com

UNICENTRO – Universidade Estadual do Centro-Oeste

Palavras-chave: *Rich Internet Application*, práticas ágeis, usabilidade.

Resumo:
Usuários com diversos níveis de conhecimento sobre computadores interagem com sistemas *web*. Sendo assim, uma boa opção é desenvolver este tipo de sistema como uma *Rich Internet Application* – aplicações que possuem características de interface gráfica parecidas com as interfaces de sistemas *Desktop* e que são fáceis de utilizar. Essas aplicações são construídas utilizando algumas tecnologias, dentre as quais é necessário citar a Adobe Flex, que está sendo utilizada no desenvolvimento de uma rede social para pessoas com interesse em gastronomia. Além disso, para ser um projeto de qualidade, determinadas práticas devem ser adotadas, como realizar o desenvolvimento direcionado a testes. Dessa forma, o usuário terá em mãos um sistema de fácil utilização e o desenvolvedor estará participando de um projeto que possui código limpo e de fácil extensão. Este resumo relata como um sistema *web* está sendo desenvolvido, utilizando tecnologia de *Rich Internet Application* e práticas ágeis.

Introdução

Determinadas aplicações que se adaptam ao conceito de *Web* 2.0[11] podem ser de grande complexidade, não somente para serem construídas, mas também para serem utilizadas. Para que estas sejam bem aceitas e permitam interação natural do usuário com a aplicação, são necessárias interfaces gráficas que sejam fáceis de utilizar e flexíveis.

Segundo [5], "*Rich Internet Applications*[12] (RIA) combinam o alcance da Internet com uma interface gráfica rica e que fornece um alto nível de satisfação para o usuário". Não obstante, existem diversas tecnologias que o desenvolvedor pode utilizar para construir uma RIA. Dentre estas se destaca a Adobe Flex[13], fornecendo ferramentas que agilizam e simplificam o desenvolvimento da aplicação.

No entanto, é necessário ressaltar que, se as práticas corretas não forem adotadas no desenvolvimento, o resultado final não será de alta qualidade. Dessa forma, opta-se pela escolha de práticas ágeis, como *test-driven development* (TDD) [2] e refatoração constante, utilizadas em diversas

[11] Diversas definições de Web 2.0: http://web2.0br.com.br/conceito-web20/
[12] Aplicações Ricas para a Internet.
[13] Disponível em http://www.adobe.com/products/flex/

metodologias ágeis, como *Extreme Programming* (XP). Além disso, a metodologia *Kanban* [1] está sendo utilizada para o desenvolvimento.

Materiais e Métodos

Aplicações ricas para a Internet "possuem características muito próximas às aplicações *Desktop*" [3]. Dessa forma, uma aplicação rica é muitas vezes bem aceita pelo usuário pelo fato de que este se sente mais próximo de aplicações *Desktop,* as quais ele está mais acostumado. Além disso, quando se utiliza somente *HTML*[14], diversas requisições são feitas ao servidor de aplicação e o visual do sistema é *renderizado* muitas vezes. Da mesma, forma uma aplicação rica faz diversas requisições, porém "somente partes dessa interface gráfica são atualizadas" [3], e como não é necessário fazer *renderizações* na página inteira "a aplicação apresenta os dados com tempo de resposta menor ao usuário final" [5].

A tecnologia Adobe Flex fornece ferramentas para a construção de RIA, fazendo com que a camada da apresentação seja bastante flexível e fácil de utilizar. Além disso, "Flex dá a liberdade ao desenvolvedor de escrever códigos tanto em *MXML* quanto em *ActionScript*" [3], duas linguagens bastante simples. No entanto, é necessário ressaltar que algumas validações são tratadas na parte *front-end* da aplicação com a utilização somente de *ActionScript.*

Refatorar significa "melhorar a estrutura do código existente sem modificar o comportamento externo" [4], melhorando a qualidade do projeto e mantendo as funcionalidades, o código simples, limpo, sem duplicações e legível.

Segundo [2], "TDD é um conjunto de técnicas de testes que são adequadas para projetos de qualquer tamanho". Dentre estas técnicas destacam-se os testes unitários, que "validam unidades do sistema" [2]. Além disso, os testes são aplicados com base em um ciclo: escrever um teste automatizado que falhe (ou seja, que o código existente retorne dados incorretos), escrever o código para tratar aquele caso que falhou, *refatorar* e testar novamente. Este ciclo é feito até que todos os casos sejam testados. Assim, quando uma nova funcionalidade for adicionada, os testes devem ser executados e o código deve ser *refatorado*.

Em metodologias ágeis de desenvolvimento de *software*, uma determinada parte do sistema é incorporada (ou entregue) em termos de iterações. Segundo [1], "uma iteração é um curto período de tempo (não mais que 6 semanas), e, no final dessa iteração, o que foi desenvolvido é incorporado no projeto como um todo". Com base nisto, a metodologia de desenvolvimento escolhida é *Kanban.* De acordo com [1], "esta possui três elementos de sucesso: reduzir trabalho em progresso, balancear capacidade com demanda e dar prioridades a determinadas atividades". Dessa forma, com a utilização de um quadro para a organização de tarefas (que são representadas em cartões), é fácil saber o que precisa ser desenvolvido. Além

[14] Linguagem de marcação utilizada para construir páginas para a Internet.

disso, o quadro deve ser atualizado no final de uma iteração (estipulada em 2 semanas).

Considerações finais

Um sistema com diversas funcionalidades, mas difícil de usar, é geralmente abandonado pelos usuários. Sendo assim, desenvolver um sistema com boa usabilidade é crucial para o seu sucesso. Ferramentas para a construção de RIA fornecem todos os recursos necessários para a construção de uma interface gráfica amigável e simples, aumentando as chances de aceitação deste sistema.

Empresas que desenvolvem *software* de qualidade utilizando metodologias ágeis buscam por profissionais de qualidade. Estes profissionais estão acostumados a utilizar práticas ágeis. Dessa forma, é interessante para o acadêmico aprender essas práticas ainda na graduação.

Referências

[1] ANDERSON, David J. **Kanban in Action.** Disponível em www.agilemanagement.net/Articles/Weblog/KanbaninAction.html – Acessado em 04/05/2009.

[2] BECK, Kent. **Test-Driven Development By Example.** Addison Wesley, Novembro, 2002.

[3] BROWN, Charles E. **The Essential Guide to Flex 3**. Friends of Ed Adobe Learning Library, 2008.

[4] FOWLER, Martin. **Refactoring: Improving the Design of Existing Code.** Addison Wesley, 2002.

[5] STALEY, Tad. **Planning for RIA success.** Disponível em www.adobe.com/devnet/flex/articles/planning_ria/planning_ria.pdf - Acessado em 10/05/2009

UMA PROPOSTA DE SOLUÇÃO PARA O PROBLEMA DO CAMINHO MÍNIMO *FUZZY*

Wagner Santos de Oliveira
darkghost199@hotmail.com

Fábio Hernandes
hernandes@unicentro.br

Universidade Estadual do Centro-Oeste

Palavras-chave: Teoria dos grafos, problema de caminho mínimo, teoria dos conjuntos *fuzzy*.

Resumo
Considerando que o problema de caminho mínimo *fuzzy* possui diversas aplicações nas áreas da Engenharia e da Computação, neste trabalho é abordado um algoritmo que trata deste problema, apresentado como solução um conjunto de caminhos não-dominados.

Introdução

O problema de caminho mínimo em grafos com parâmetros incertos é um dos mais estudados da programação matemática *fuzzy*, visto que possui aplicações nas mais diferentes áreas (por exemplo: telecomunicações, transportes, manufaturas, etc).

Na literatura há diversos trabalhos de caminho mínimo *fuzzy* [1,2,3], porém verifica-se que esses possuem desvantagens que necessitam ser contornadas, tais como: encontram custos sem caminhos associados; apresentam um conjunto solução de caminhos, sem informar ao usuário uma ordenação dos mesmos; dentre outras.

Com isso, é proposto um algoritmo, baseado no trabalho de Ford-Moore-Bellman, para contornar tais desvantagens supracitadas.

Algoritmo

Este algoritmo tem por objetivo encontrar um conjunto solução de caminhos não-dominados, semelhante ao trabalho de Hernandes *et al* [1], utilizando o método das múltiplas etiquetas, com a relação de ordem proposta por Okada e Soper [2] e abordando os custos nos arcos como números *fuzzy* trapezoidais.

Definição: Sejam $\tilde{a} = (a_1, a_2, a_3, a_4)$ e $\tilde{b} = (b_1, b_2, b_3, b_4)$ dois números *fuzzy* trapezoidais, então $\tilde{a} \prec \tilde{b}$ ($\tilde{a}$ domina $\tilde{b}$) se, e somente se $a_1 \leq b_1$, $a_2 \leq b_2$, $a_3 \leq b_3$, $a_4 \leq b_4$, *e* $\tilde{a} \neq \tilde{b}$.

Informações sobre o algoritmo:
N: conjunto dos nós;
it: contador de iterações;
$\tilde{c}_{ji}$: custo do arco (j,i);
$\tilde{c}^{it}_{(i,k)}$: custo do caminho entre o nó 1 e o *i* com a etiqueta *k* na iteração *it;*
M: número grande substituir o infinito; e
Γ_i^{-1} : conjunto dos nós predecessores de *i*.

Algoritmo Abordado

PASSO 0: Início.

- $\tilde{c}^{0}_{(1,1)} \leftarrow (0,0,0,0)$.
- $\tilde{c}^{0}_{(j,1)} \leftarrow (M, M+1, M+2, M+3),\ j = 2,3,...,r;$, sendo *r o* número de nós.
- $it \leftarrow 1$.

PASSO 1: Construção dos caminhos, seleção das etiquetas e verificação da dominância.

- $\tilde{c}^{it}_{(1,1)} \leftarrow (0,0,0,0)$.
- $\forall j \in \Gamma_i^{-1},\ i = 1,2,3,...,r,$ faça:

  $\tilde{c}^{it}_{(i,k1)} \leftarrow \tilde{c}^{it-1}_{(j,k2)} \oplus \tilde{c}_{ji}$ (construção dos custos dos caminhos)
- Verificação da dominância entre as etiquetas. Para todas as etiquetas do nó *i* faça:
  - se $\tilde{c}^{it}_{(i,k1)} \succ \tilde{c}^{it}_{(i,k2)} \Rightarrow$ elimine a *k*1 – *ésima* etiqueta;
  - se $\tilde{c}^{it}_{(i,k1)} \prec \tilde{c}^{it}_{(i,k2)} \Rightarrow$ elimine a *k*2 – *ésima* etiqueta.

PASSO 2: Critério de parada.

- Se $\left(\tilde{c}^{it}_{(i,k1)} = \tilde{c}^{it-1}_{(i,k1)},\ \forall i \in N\right)$ ou (*it*=*r*) faça:
  - se $it = r$ e $\tilde{c}^{it}_{(i,k1)} \neq \tilde{c}^{it-1}_{(i,k1)},\ \forall i \in N \Rightarrow$ Passo 5 (circuito negativo)
  - senão $\Rightarrow$ Passo 3
- Senão: $it \leftarrow it+1 \Rightarrow$ volte ao Passo 1.

PASSO 3: Composição dos caminhos não-dominados.

Encontre os caminhos não-dominados entre os nós 1 e *i*.

PASSO 4: FIM.

Resultados computacionais

O algoritmo foi executado na rede da Figura 1 e apresentou os resultados contidos na Tabela 2.

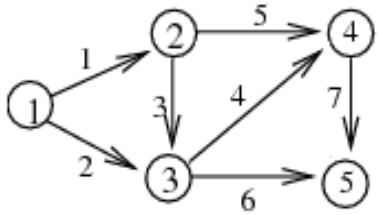

Figura 1. Rede exemplo executada pelo algoritmo

Arcos	Custos
1	(25, 30, 35, 40)
2	(20, 22, 30, 35)
3	(20, 23,25, 28)
4	(10, 12, 15, 18)

Arcos	Custos
5	(8, 9, 10, 12)
6	(15, 18, 20, 23)
7	(13, 17, 20, 23)

Tabela 1. Custos dos arcos da rede da Figura 2.

Caminhos	Custos
1 → 2	(25, 30, 35, 40)
1 → 3	(20, 22, 30, 35)
1 → 2 → 4	(33, 39, 45, 52)
1 → 3 → 4	(30, 34, 45, 53)
1 → 3 → 5	(40, 44, 55, 63)

Tabela 2. Caminhos encontrados e seus respectivos custos.

Considerações finais

Neste trabalho foi adaptado um algoritmo para o problema de caminho mínimo *fuzzy* que utiliza o método das múltiplas etiquetas e a dominância entre caminhos, apresentando como solução um conjunto de caminhos não-dominados e contorna as desvantagens abordadas na introdução. O mesmo utilizou nos parâmetros números *fuzzy* trapezoidais, inédito na literatura.

Pretende-se usar outros critérios de dominância em trabalhos futuros.

Referências

[1] HERNANDES, Fábio *et al*. **The shortest path problem on networks with fuzzy parameters**. *Fuzzy Sets and Systems, 158, 1561-1570.* 2007.

[2] OKADA, Shinkoh. **Fuzzy shortest path problems incorporating interactivity among paths.** *Fuzzy Sets and Systems,* 142**,** 335-357, 2004.

[3] OKADA, Shinkoh e SOPER, Timothy. **A shortest path problem on a network with fuzzy arc lengths**. *Fuzzy Sets and Systems*, 109, 129-140, 2000.