Anais do XIII SEMINCO

Seminário de Computação
22 a 24 de Setembro de 2004
FURB - Campus IV - Blumenau/SC

Promoção

Universidade Regional de Blumenau - FURB
Pró-Reitoria de Extensão e Relações Comunitárias - PROERC
Centro de Ciências Exatas e Naturais - CCEN
Departamento de Sistemas e Computação - DSC
Centro Acadêmico Livre de Computação - CALCOMP

Comissão Organizadora

Prof. Everaldo Artur Grahl (Coordenador)
Prof. Alexander Roberto Valdameri
Prof. Antônio Carlos Tavares
Acadêmico Fernando dos Santos
Prof. Jomi Fred Hübner
Profa. Joyce Martins
Prof. José Roque Voltolilni da Silva
Acadêmica Karly Schubert Vargas
Prof. Roberto Heinzle
Prof. Wilson Pedro Carli

Ficha Catalográfica Elaborada pela Biblioteca da FURB

Seminário de Computação (13.: 2004 : Blumenau, SC)

Anais do XIII SEMINCO / promoção Universidade Regional de Blumenau, Departamento de Sistemas e Computação; Everaldo Artur Grahl (coordenador). - Blumenau, O Departamento, 2004. 200 p. : il.

1. Computação - Congressos. I. Grahl, Everaldo Artur. II. Universidade Regional de Blumenau. Departamento de Sistemas e Computação. III. Título.

CDD 004

Universidade Regional de Blumenau

Apresentação

A Universidade Regional de Blumenau - FURB, através do Departamento de Sistemas e Computação, realiza o XIII Seminário de Computação (XIII SEMINCO) entre os dias 22 e 24 de setembro de 2004. Este ano duas novidades podem ser destacadas: A realização do evento no Campus IV da FURB, onde ocorre regularmente o curso de Ciências da Computação e Sistemas de Informação e a criação da I Mostra de Software, evento destinado à divulgação dos melhores softwares desenvolvidos no meio acadêmico da FURB (trabalhos de graduação, projetos de pesquisa e trabalhos de conclusão de curso). Além disso, tradicionalmente temos a apresentação de palestras técnicas e artigos selecionados.

Este ano tivemos a submissão de 34 artigos provenientes de várias instituições e empresas do país, sendo que destes foram aprovados 15 artigos das seguintes organizações: ACAFE, CEFET-PB, CONSEI, FACENS, FIMES, FURB, TOTALL, UCG, UFCG, UFG, UFSC, UFU, UNED e UNIVALI.

Agradecemos a todos os envolvidos na organização do evento, bem como a Comissão de Avaliação Inter Institucional que não mediu esforços em avaliar os diversos artigos submetidos à chamada de trabalhos. Esperamos que nos três dias de realização do evento as expectativas dos participantes sejam atendidas e tenhamos um grande evento. Até o ano que vem!

Comissão Organizadora

Agradecimentos

Comissão de Avaliação Inter Institucional
Instituto Gene
Núcleo de Informática - FURB
Projeto Acredito - FURB
Sociedade Brasileira de Computação - SBC

Comissão de Avaliação Interinstitucional

Alexander Roberto Valdameri (FURB - SC)
Ana Lúcia Anacleto Reis (FURB - SC)
Anita da Rocha Fernandes (UNIVALI - SC)
Antonio Carlos Tavares (FURB - SC)
Edson Satoshi Gomi (USP - SP)
Eduardo Battistella (UNISINOS)
Everaldo Artur Grahl (FURB - SC)
Fabiane Barreto Vavassori (FURB - SC)
Fernando Santos Osório (UNISINOS - RS)
Francisco Adell Péricas (FURB - SC)
Gerson Cavalheiro (UNISINOS - RS)
Graça Marietto (USP - SP)
Gustavo G. Lugo (USP - SP)
Iraci Cristina da Silveira (UCS - RS)
Jomi Fred Hübner (FURB)
José Leomar Todesco (INE/CTC/UFSC - SC)
Joyce Martins (FURB - SC)
Marcel Hugo (FURB)
Marcello Thiry (UNIVALI - SC)
Marcos Eduardo Casa (UCS - RS)
Mauro Marcelo Mattos (FURB - SC)
Miguel Alexandre Wisintainer (FURB - SC)
Paulo Cesar Rodacki Gomes (FURB - SC)
Paulo Fernando da Silva (FURB)
Rafael Cancian (UNIVALI - SC)
Rafael Heitor Bordini (University of Durham)
Reinaldo Bianchi (USP - SP)
Roberto Heinzle (FURB - SC)
Rogerio Golçalvez Bittencourt (UNIVALI - SC)
Sérgio Crespo (UNISINOS - RS)
Sérgio Stringari (FURB - SC)
Vinícius Medina Kern (UNIVALI - SC)

Artigos Selecionados

Banco de Dados

Engenharia de Software

Informática na Educação

Inteligência Artificial

Integração Hardware/Software

Sistemas de Informação

Rede de Computadores

Mostra de Software

Personalização de Sistemas WEB utilizando *Data Mining*: um estudo de caso aplicado na Biblioteca Central da FURB

Alberto Pereira de Jesus (FURB)
albertop@ifurb.br

Evanilde Maria Moser (FURB)
emmoser@furb.br

Paulo José Ogliari (UFSC)
ogliari@inf.ufsc.br

Jacqueline Uber Silva (UFSC)
jacqueline@furb.br

Resumo. Este artigo descreve todas as etapas de implantação de *data mining* aplicado na identificação do perfil dos usuários de uma biblioteca para a personalização de sistemas WEB de recuperação e disseminação de informações.

Palavras-chave: *Data Mining*, *Data Warehouse*, Personalização de conteúdo, Bibliotecas.

1 Introdução

Com o crescimento do volume de publicações e também das necessidades de informações dos clientes, é importante, que as bibliotecas, sejam elas em papel ou em formato eletrônico, possuam sistemas de informações capazes de armazenar e indexar informações bibliográficas de forma a facilitar a recuperação e disseminação aos usuários (CARDOSO, 2000).

Neste sentido, dois sistemas têm sido desenvolvidos, sendo eles: o sistema de recuperação de informações (SRI) e o sistema de disseminação seletiva de informações (DSI). Enquanto o primeiro (SRI) trata de localizar as informações solicitadas pelo usuário, o (segundo) DSI tenta prever as necessidades desses usuários, fazendo recomendações e sugestões conforme seu interesse.

Conhecer os usuários é importante para poder fazer recomendações e ajudá-los na localização de obras. Hoje, devido à grande quantidade de usuários e publicações, precisa-se de ferramentas automatizadas que auxiliem nesse processo.

Sabendo-se que a missão das Bibliotecas, segundo Funaro et al. (2001, p. 1) "é oferecer a seus usuários informações relevantes para a realização de suas pesquisas, facilitando o acesso e localização do material necessário", os sistemas tradicionais de SRI e DSI das Bibliotecas necessitam evoluir e ser inteligentes, a fim de agregar valor ao serviço de referência. Dessa forma é necessário que se conheça o perfil do usuário, delineando suas preferências e seus interesses.

As técnicas de *data mining* permitem a identificação desse perfil, possibilitando a personalização dos processos do SRI e DSI, tornando-os objetivos e seletivos. Esta confluência de acertos caracteriza a relevância da informação. Não adianta o usuário receber uma comunicação personalizada se ela não for relevante para seus interesses e necessidades.

"O objetivo da personalização de conteúdo é garantir que a pessoa certa receba a informação certa no momento certo" (ARANHA, 2000, p. 10).

Estes sistemas, principalmente o DSI, apesar das facilidades que oferece, apresentam alguns problemas como: o não preenchimento por alguns usuários e as rápidas mudanças que ocorrem em seus interesses. Toma-se como exemplo um professor que lecionava uma disciplina de *data warehouse* e atualmente leciona a disciplina de *data mining*. Como ele preencheu seus dados com antigo perfil, continuará recebendo informações sobre seus interesses preenchidos anteriormente.

Seria prudente que o sistema reconhecesse essas alterações no ambiente e fosse capaz de se adequar às novas características. Isso é possível por meio da aplicação de técnicas de *data mining* sobre os dados contidos nos registros de transações como: empréstimos, reservas e consultas que são armazenados no banco de dados da Biblioteca e servirão para fazer um estudo do perfil do usuário. Estes registros são armazenados diariamente pelas transações de empréstimos, no entanto não são utilizados para tomada de decisão.

Mais especificamente, a aplicação de *data mining* nestes registros permitirá:

a) melhorar o SRI através da personalização das consultas, ao fazer uma busca o retorno da consulta é filtrado segundo o perfil do usuário;
b) facilitar o DSI, recomendando obras de interesse ao usuário.

2 Objetivos

O objetivo geral deste trabalho é desenvolver um sistema de recuperação e disseminação de informações, personalizado segundo o perfil de cada usuário da Biblioteca Central (BC) da Universidade Regional de Blumenau (FURB), por meio da aplicação de técnicas de *data mining*. Como objetivos específicos têm-se:

a) desenvolver um *data warehouse* para dar suporte a aplicação das técnicas de *data mining*, possibilitando também obter informações para tomada de decisões;
b) aplicar técnicas de *data mining* sobre o histórico de empréstimos e reservas dos usuários para identificar o perfil dos mesmos na BC da FURB;
c) desenvolver um sistema WEB de SRI e DSI personalizado dinamicamente para a BC da FURB.

3 Justificativa

O trabalho se justificativa, pois conhecendo as características e preferências do usuário, pode-se assim definir seu perfil que é de elevada importância para o SRI, DSI e para tomada de decisões gerenciais. Possibilitando uma maior satisfação dos usuários e melhor utilização/organização da biblioteca, redução de custos com a aquisição de materiais, bem como, facilidade no atendimento dos usuários.

4 Data Mining

A quantidade de informações produzidas versus a capacidade de armazenamento dos recursos computacionais a um baixo custo, tem impulsionado o desenvolvimento de novas tecnologias capazes de tratar estes dados, transformá-los em informações úteis e extrair conhecimentos.

Entretanto, o principal objetivo da utilização do computador ainda tem sido o de resolver problemas operacionais das organizações, que coletam e geram grandes volumes de dados que são usados ou obtidos em suas operações diárias e armazenados nos bancos de dados. Porém, os mesmos não são utilizados para tomadas de decisões, ficando retidos em seus bancos de dados, sendo utilizados somente como fonte histórica. Estas organizações têm dificuldades na

identificação de formas de exploração desses dados, e principalmente na transformação desses repositórios em conhecimento (BARTOLOMEU, 2002).

Pesquisadores de diferentes áreas estudam e desenvolvem trabalhos para obter informações e extrair conhecimentos a partir de grandes bases de dados, como tópico de pesquisa, com ênfase na técnica conhecida como *data mining*.

Data mining é parte do processo de *Knowledge Discovery in Databases* (KDD), ou descoberta de conhecimentos em bancos de dados (DCBD), o qual é responsável pela extração de informações sem conhecimento prévio, de um grande banco de dados, e seu uso para a tomada de decisões (DINIZ; LOUZADA NETO, 2000).

KDD é um processo contínuo e cíclico que permite que os resultados sejam alcançados e melhorados ao longo do tempo. Na figura 1 são apresentados os passos que devem ser executados no processo de KDD. Segundo Diniz; Louzada Neto (2000) embora os passos devam ser seguidos na mesma ordem em que são apresentados, o processo é extremamente interativo e iterativo (com várias decisões sendo feitas pelo próprio usuário e loops podendo ocorrer entre quaisquer dois ou mais passos).

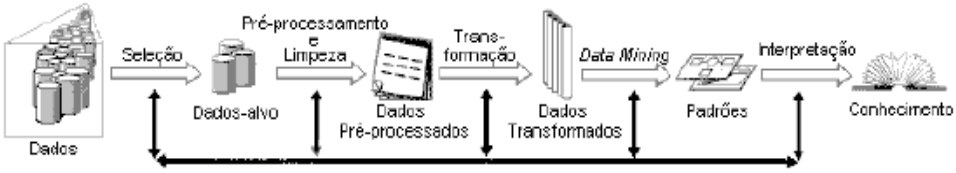


Figura 1: Passos do processo de KDD (Fonte: FIGUEIRA, 1998, p. 8.)

Para a implantação desta tecnologia é necessário que se conheça a fundo o processo, para que a mesma venha atender às expectativas do usuário. O processo de KDD começa obviamente com o entendimento do domínio da aplicação e dos objetivos finais a serem atingidos.

Segundo Harrison (1998, p. 155) "*data mining* é a exploração e análise, por meios automáticos ou semi-automáticos, das grandes quantidades de dados para descobrir modelos e regras significativas".

Deve-se destacar que cada técnica de *data mining* ou cada implementação específica de algoritmos que são utilizados para conduzir as operações *data mining* adapta-se melhor a alguns problemas que a outros, o que impossibilita a existência de um método de *data mining* universalmente melhor. Para cada particular problema tem-se um particular algoritmo. Portanto, o sucesso de uma tarefa de *data mining* está diretamente ligado à experiência e intuição do analista (Diniz; Louzada Neto 2000).

Assim, é importante que se conheçam, as tarefas desempenhadas (Classificação, Estimatição, Previsão, Agrupamento por afinidade, Segmentação e Descrição) e suas técnicas (Análise de seleção estatística, Raciocínio baseado em casos, Algoritmos genéticos, Detecção de agrupamentos, Análise de vínculos, Árvores de decisão e indução de regras, Redes neurais) a fim de dar suporte a sua escolha.

4.1 *Data Mining* aplicado a Personalização de Conteúdo WEB

A personalização de conteúdo é uma característica de sistemas automatizados que permite que usuários distintos sejam tratados de forma diferente. Estes sistemas são capazes de identificar

o usuário e direcionar a cada um conteúdo, recomendação de livros e de serviços que sejam relevantes.

Segundo REATEGUI (2002, p. 153) "o termo personalização é utilizado para designar sistemas capazes de reconhecer os usuários, armazenar dados sobre a interação com cada um e personalizar sua "comunicação" para que o usuário seja tratado de acordo com o seu perfil e seus interesses".

> *"Estratégias efetivas de personalização demandam a aplicação criteriosa e objetiva de técnicas de descoberta do conhecimento e mineração de dados, determinando padrões de comportamento a partir de variadas fontes de dados transformando esses padrões em serviços personalizados que resultem em aumento de lucratividade ou eficácia desses serviços" (MEIRA JR et al. 2002, p. 179).*

Sendo assim, aplicando-se *data mining* é possível uma personalização onde os interesses e necessidades dos usuários serão consideradas. A personalização de conteúdo pode ser utilizada em sistemas de recuperação e disseminação de informações para facilitar o acesso às informações desejadas pelos usuários. Pois segundo Meira JR et al. (2002, p. 179), "a personalização não afeta a semântica dos serviços, pois os aspectos funcionais não são alterados".

5 Metodologia

Para o processo de extração de conhecimento nos dados da biblioteca sobre o perfil dos usuários, será utilizada a metodologia referenciada por Berry; Linoff (1997). A Figura 2 apresenta o modelo proposto. A mesma será aplicada na BC da FURB.

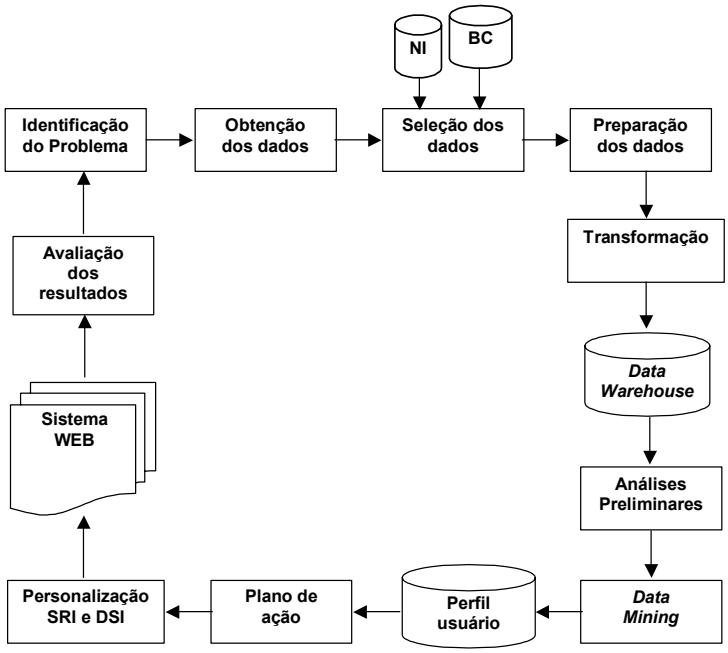


Figura 2: Modelo proposto para aplicação de *data mining* em bibliotecas

5.1 Identificação do problema

A maioria das bibliotecas não possui sistemas de recuperação e disseminação de informações capazes de ajudar no processo de localização das obras de interesse dos usuários. O mesmo é feito pelo serviço de referência com o auxílio de bibliotecários ou especialistas na área.

A Biblioteca Central (BC) da Fundação Universidade Regional de Blumenau (FURB) atualmente não apresenta um sistema informatizado de DSI aos usuários. Este ainda é feito de forma manual pelo serviço de referência. O SRI não identifica o usuário para tratá-lo de forma seletiva e personalizada. Quando é feita uma consulta, a pesquisa retorna uma grande quantidade de informações (dados) a maioria sem relevância e nenhuma ordenação, o que caracteriza uma alta revocação, mas baixa precisão (CARDOSO, 2000, p. 2).

Definido o problema, elegem-se as variáveis que serão utilizadas na investigação para a resolução do mesmo. As variáveis que tem relacionamento direto para identificação do perfil dos usuários são: usuários; obras da Biblioteca; CDD (Classificação Decimal Dewey do assunto principal da obra); transações (empréstimos, reservas).

5.2 Obtenção dos dados

Mediante a identificação das variáveis que serão utilizadas no processo de extração de conhecimento sobre o perfil do usuário, parte-se para o reconhecimento e a obtenção das mesmas nas fontes de dados. A principal fonte dos dados são os sistemas legados da BC mantidos pela seção de automação (cadastro e a circulação obras). Outra fonte é o sistema de identificação única de pessoas com vínculo na instituição, mantido pelo NI (usuários). A ligação entre o usuário e suas transações é feita através do identificador único do usuário.

5.3 Seleção dos dados

Através de uma engenharia reversa das tabelas de interesse armazenadas nos bancos de dados, torna-se possível reconhecer as variáveis de interesse para assim fazer a seleção.

Foram selecionados na amostra professores e alunos de pós-graduação da FURB. Com a integração das bases, excluem-se alguns dados como CPF, Endereço, etc, por serem usadas com finalidades operacionais que não se aplicam a esta pesquisa.

5.4 Pré-processamento dos dados

Após a seleção dos dados, faz-se a verificação da existência de inconsistências e/ou erros nas variáveis: A data de aquisição continha dados fora do formato padrão e o código CDD em alguns casos estava fora do padrão de catalogação.

5.5 Extração, transformação e carga dos dados

Os dados são estruturados para facilitar e agilizar o processo de mineração. A partir daí, foi gerado um *data mart*, que é parte de um *data warehouse*. Após identificar as variáveis de interesse, chega-se a um modelo que trata da circulação das obras.

A tabela fato é a de circulação de empréstimos, onde cada registro corresponde a uma transação que pode ser dos tipos: empréstimo e reserva. As dimensões encontradas são: a obra e o usuário que a emprestou. A partir do modelo de *data mart* foram criadas as tabelas e rotinas para carga dos dados.

Na área de biblioteconomia já foram institucionalizados alguns códigos para determinados domínios de uma variável, como é o caso da classificação dos livros. Existe uma codificação internacional, conhecida como Classificação Decimal Dewey (CDD), que é usada por diferentes

órgãos da área de biblioteconomia, a fim de organizar o acervo e facilitar a localização das obras. Assim foram criados cinco níveis da CDD. A partir da qual foram gerados os assuntos significativos (AS) através da totalização das transações segundo a CDD, reduzindo as mesmas até um nível mínimo de significância.

5.6 Análises preliminares

Em qualquer investigação é fundamental para o pesquisador ter uma visão global dos dados que estão sendo pesquisados, a seguir apresenta-se uma análise descritiva dos dados da amostra, envolvidos neste estudo.

A amostra é composta por 17421 títulos que totalizam 51011 volumes, 3906 usuários, 821 da categoria professores e 3085 da categoria de pós-graduação.

Estes usuários realizaram 68543 transações, 66769 de empréstimo e 1775 de reservas. Obteve-se uma média de 17,54 transações por usuários.

5.7 Mineração dos dados

Caracteriza-se pela transformação dos dados em conhecimento. Para encontrar o perfil do usuário utilizam-se as seguintes etapas:

5.7.1 Análise de conglomerados de assuntos significativos

A metodologia de análise de conglomerados (*cluster analysis*) é uma descoberta indireta de conhecimento a partir de algoritmos para encontrar registros de dados que são semelhantes entre si. Estes conjuntos de registros similares são conhecidos como clusters.

Segundo Velasquez et al. (2001, p. 2) “Todos os algoritmos de análise de conglomerados são baseados em uma medida de similaridade ou, ao contrário de distância, que procuram expressar o grau de semelhança entre os objetos”. Uma medida de distância muito utilizada quando os atributos são de natureza quantitativa é a distância euclidiana.

Formam-se agrupamentos das obras em grandes áreas de conhecimento, ou seja, grupos de livros os quais são utilizados por usuários para estudo de determinando assunto ou área. Assim foram analisados alguns métodos estatísticos de agrupamento hierárquico, como o do vizinho mais próximo, do vizinho mais distante, e de Ward. Optou-se pelo método Ward com distância euclidiana, pois o mesmo apresentou melhores resultados e por ser indicado por Aranha (2000) em seu trabalho. O método Ward baseia-se no agrupamento de indivíduos dentro dos conglomerados a partir da soma dos quadrados dos desvios das observações. A cada estágio a soma é minimizada a partir da combinação de dois agrupamentos do estágio anterior. Outra característica desse método é a união de conglomerados que têm um pequeno número de observações. (HAIR, 1995).

Afirmam Velasquez et al. (2001, p. 2) que “Nos métodos hierárquicos o número de classes não é fixado a priori, mas resulta da visualização do dendrograma, um gráfico que mostra a seqüência das fusões ou divisões ao longo do processo iterativo”.

Foi aplicada esta técnica aos dados de transações dos usuários segundo suas transações por AS, contidos no *data mart* resultando no dendograma apresentado na Figura 3.

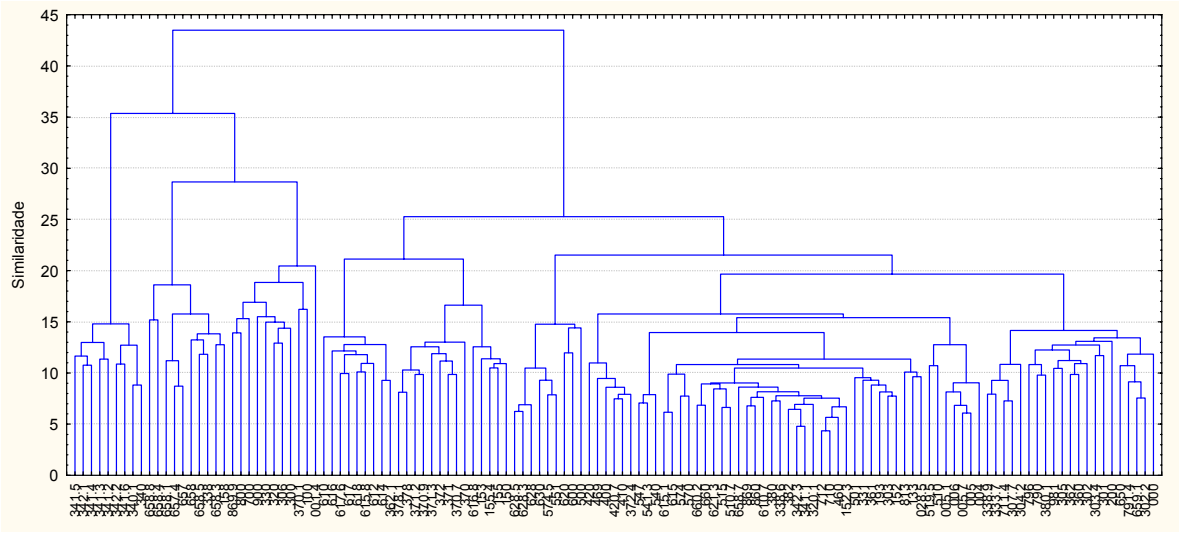


Figura 3: Dendograma com transações dos usuários por AS

Através da análise do dendograma foram gerados 34 grupos de grandes áreas como o apresentado na Tabela 1.

Tabela 1: Exemplo da tabela de grupos de grandes áreas de interesse

Grupo	Descrição do Grupo	AS	Descrição do AS
1	Direito	341.5	Direito penal
		342.1	Direito civil
		341.2	Direto constitucional
		342.2	Direito comercial
		341.6	Direito do trabalho
		340.1	Filosofia do Direito
		340	Direito

5.7.2 Classificação do acervo em grandes áreas

A classificação é uma tarefa muito utilizada em *data mining*. Consiste em examinar os aspectos de um objeto e atribuí-lo a um dos conjuntos de classes existentes. Assim, classificam-se as obras do acervo da biblioteca em grandes áreas do conhecimento segundo a tabela gerada através da analise de *cluster* apresentada anteriormente.

5.7.3 Descrição do perfil dos usuários

Segundo Harrison (1998 p.181) “as vezes o propósito de executar *data mining* é simplesmente descrever o que está acontecendo em um banco de dados complicado de maneira a aumentar o conhecimento das pessoas, produtos ou dos processos que produziram os dados”.

A descrição do comportamento do usuário da biblioteca, através da análise de suas transações, objetiva identificar seu perfil de utilização de obras, podendo interagir com o mesmo através dos sistemas de SRI e DSI de forma personalizada.

No estudo do perfil, o primeiro nível de descrição seria a maior grande área de interesse, assim determinando a grande área de interesse do usuário. O próximo nível de descrição seria

formado por três subáreas de interesse, no quarto nível da CDD, identificados através de uma análise das três principais áreas de transações do usuário. Toma-se por exemplo as transações de um usuário segundo as grandes áreas (Figura 4) e segundo CDD (Figura 5).

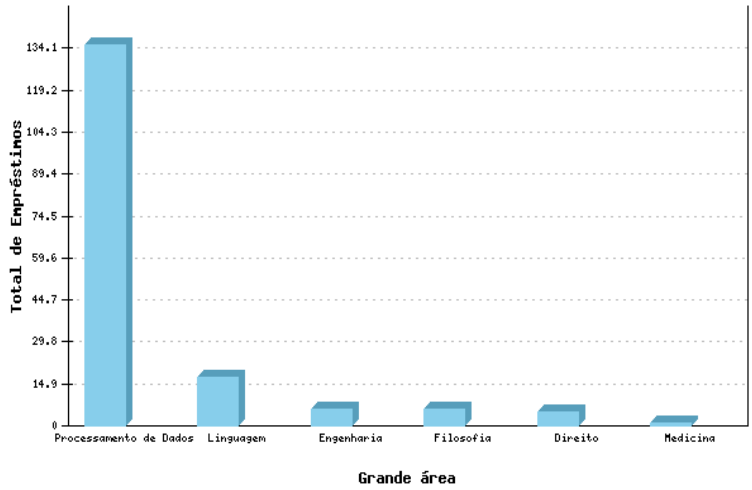


Figura 4: Transações usuários por grandes áreas

Pode-se verificar na figura 4 que o primeiro nível de descrição apresenta "processamento de dados" como a grande área de interesse do usuário em estudo.

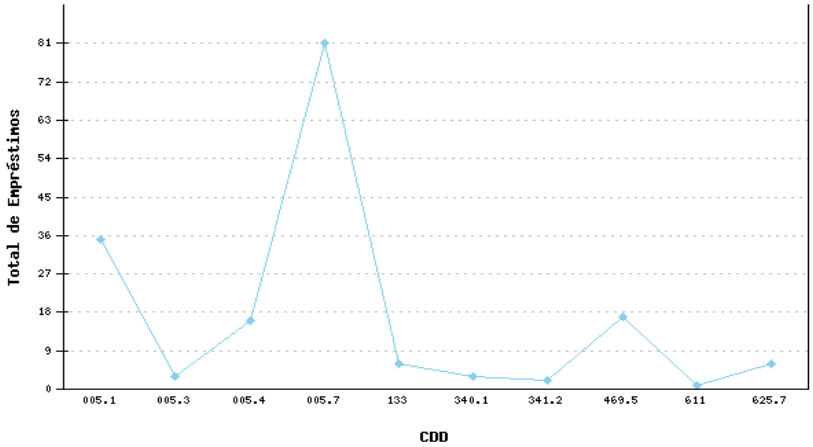


Figura 5: Transações usuários por CDD nível quatro

Observa-se na figura 5, o segundo nível de descrição apresenta as três principais subáreas de interesse do usuário que são: 005.1, 005.7 e 469.5.

5.8 Plano de ação

Identifica-se o perfil do usuário, tornando possível personalizar os sistemas de recuperação e disseminação de informações. Para tanto, utiliza-se um sistema WEB (de SRI e DSI) que foi desenvolvido e personalizado dinamicamente ao perfil de cada usuário.

O sistema desenvolvido fica esperando requisições do servidor WEB quando a recebe processa e retorna páginas HTML com o conteúdo ao usuário personalizado. A Figura 6, mostra a arquitetura do sistema desenvolvido.

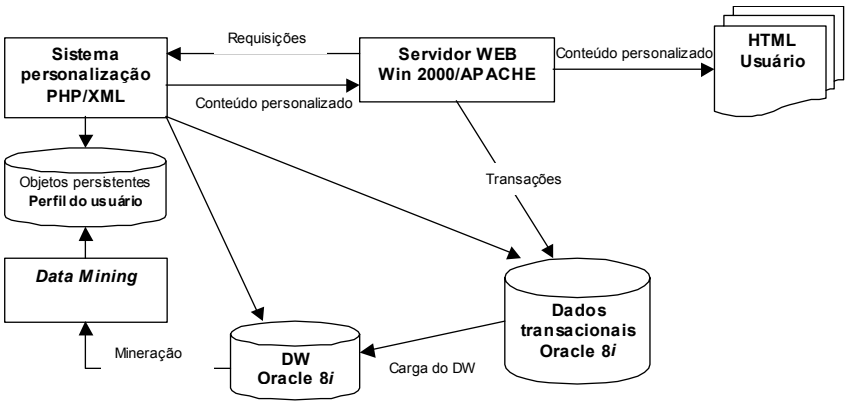


Figura 6: Arquitetura do sistema de personalização

O sistema conta com um banco de dados onde estão contidos os dados transacionais sobre as obras, usuários e suas transações, com um *data warehouse* onde estão os dados que serviram de fonte para a aplicação do *data mining*, e um objeto persistente o qual recebe os dados sobre o perfil do usuário. Quando o usuário faz uma requisição ao servidor WEB este recebe e a repassa para o sistema de personalização que recebe a requisição processa e envia a resposta de volta ao usuário personalizada.

5.9 Sistema WEB

Ao entrar no sistema é apresentada a tela de *login* (Figura 7) onde devem ser informados o código e senha do usuário na biblioteca, após validação são carregados os dados do perfil do usuário para uma sessão no servidor, que funciona como objeto persistente ficando ativo até que o usuário saia do sistema.

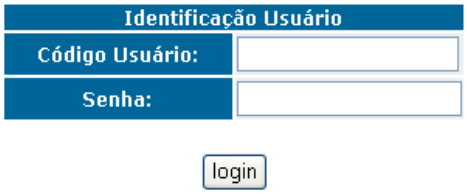


Figura 7: Tela de *Login*

A tela principal do sistema (Figura 8) é divida em três partes: menu superior, menu lateral e corpo principal.

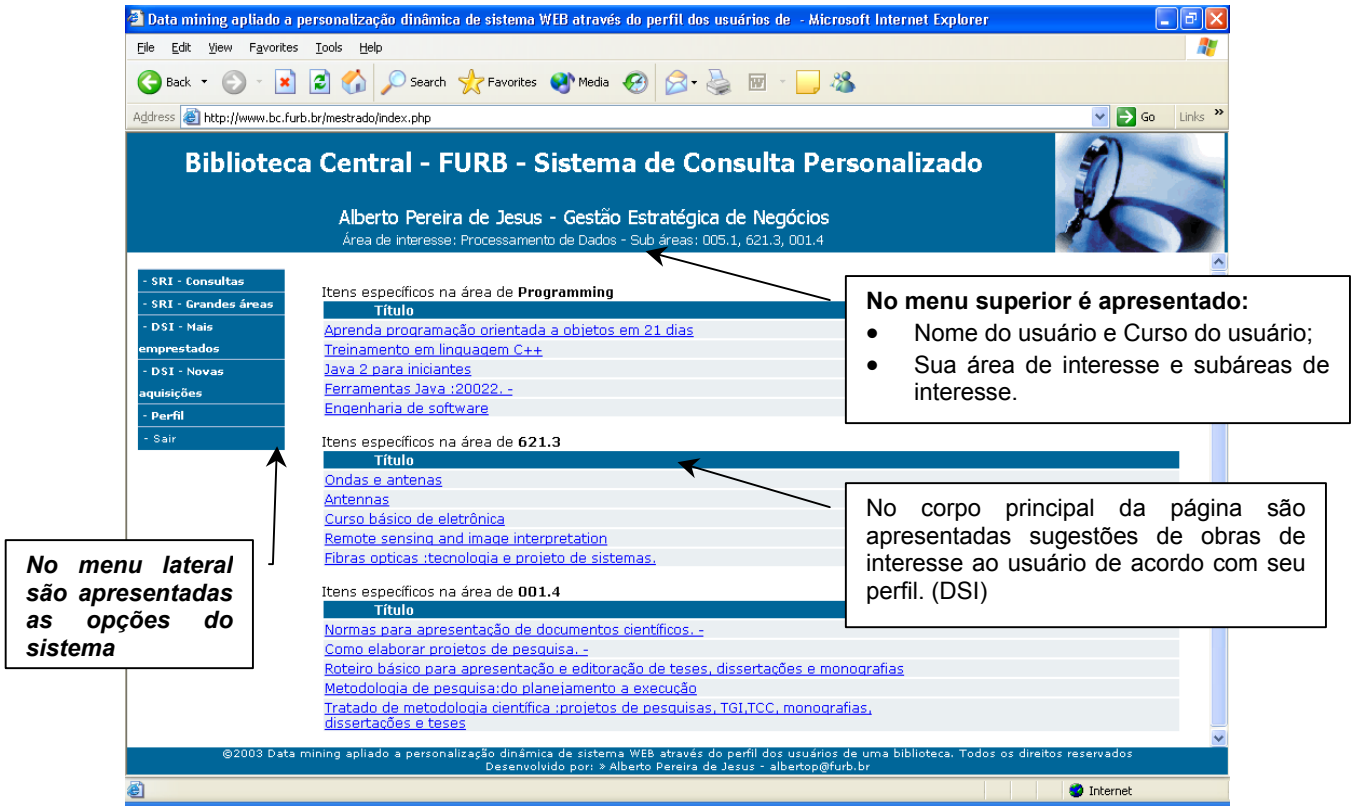


Figura 8: Tela principal do sistema

A tela de resultado da consulta (Figura 9) retornará os títulos encontrados no acervo segundo a expressão de busca determinada, ordenados conforme o perfil do usuário.

Resultado da consulta

Título	CDD	Relevância	Proximidade
Projeto & engenharia de software :teste de software	005.1	1	0
Qualidade & teste de software :engenharia de software, qualidade de software, qualidade de produtos de software, teste de software, formalização do processo de teste, aplicação prática dos testes	005.1	1	0
Guia completo ao teste de software	005.14	1	0.04
O teste gestáltico Bender para crianças	150.1982	999	145.0982
Tecnicas de exame psicologico e suas aplicacoes no Brasil :testes de aptidoes	155.28	999	150.18
Testes para admissão em empresas e empregos públicos	155.28	999	150.18

Figura 9: Tela resultado da consulta

5.10 Avaliação dos resultados

Através do modelo proposto e do protótipo desenvolvido foi possível melhorar o processo de recuperação e recomendações de obras, através da identificação da relevância da mesma ao

usuário através do perfil identificado pela análise de conglomerados. Este foi comparado ao perfil obtido pela tabela de classificação das obras desenvolvida por bibliotecários especialistas na área. O mesmo apresentou resultados excelentes. Aleatoriamente, foram sorteados alguns usuários e verificou-se o perfil obtido com seus respectivos interesses. Cabe-se ressaltar que um próximo passo nessa pesquisa seria uma avaliação quantitativa desses resultados através de uma pesquisa aos usuários.

6 Conclusões

Com a revisão bibliográfica pôde-se conhecer a tecnologia de *data mining,* necessária para dimensionar a sua aplicação na Biblioteca Central da FURB. A tecnologia de *data mining* ainda é pouco aplicada em bibliotecas. Destacam-se dois trabalhos realizados por Aranha[1999 e 2000], que aplica técnicas de *data mining* gerando listas de recomendações de itens. Diferencia-se dos trabalhos citados anteriormente, pela utilização de *data mining* aliada às técnicas de personalização de sistemas WEB, gerando SRI e DSI mais eficientes.

Cabe ressaltar dois resultados importantes:

a) o modelo para aplicação de *data mining* em bibliotecas;

b) a aplicação do modelo proposto para sua validação.

A definição de um modelo facilitou a realização do estudo, pois o mesmo determinou os passos que deveriam ser realizados para obtenção dos resultados com sucesso.

O modelo proposto baseou-se na aplicação das técnicas de *data mining* sobre a classificação CDD das obras, possibilitando determinar o perfil dos usuários quanto aos seus interesses bibliográficos. A CDD é um padrão utilizado em várias bibliotecas, tornando fácil a aplicação deste modelo em outras bibliotecas que utilizem este padrão.

Os objetivos do trabalho foram alcançados. Quanto ao principal, o desenvolvimento de um sistema WEB de recuperação e disseminação de informações personalizado ao usuário, mostrou-se funcional ao seu propósito, facilitando o processo de recuperação de informações através da ordenação do resultado das pesquisas, disseminando informações de interesse ao usuário.

Quanto aos objetivos específicos:

a) o *data warehouse* desenvolvido foi eficiente para aplicação das técnicas de *data mining*, possibilitando também informações para tomada de decisões através da criação de relatórios com ferramentas apropriadas;

b) a aplicação das técnicas de *data mining* sobre os assuntos significativos gerou grupos com correlações entre livros implícitas. A classificação dos livros possibilitou descrever o perfil dos usuários, possibilitando conhecer melhor os seus hábitos e interesses na biblioteca;

c) o sistema desenvolvido de SRI e DSI personalizado, facilitou o processo de recuperação de informações e tornando eficiente a disseminação seletiva de informações.

Quanto à tecnologia envolvida, acredita-se que está apenas nascendo e passará a fazer parte do nosso dia-a-dia. O mercado está em ampla expansão e com possibilidades de grandes negócios, pois a maioria das empresas possui grandes bancos de dados gerados a partir de seus sistemas legados, sem nenhuma utilização para tomada de decisões.

Referências

ARANHA, Francisco. **Perfil de usuários da biblioteca Karl A. Boedecker**: geração de valor para pesquisadores por meio de cooperação indireta. São Paulo: EAESP/FGV/NPP, 1999. 59p.

ARANHA, Francisco. **Análise de redes em procedimentos de cooperação indireta**: utilização no sistema de recomendações da Biblioteca Karl A. Boedecker. São Paulo: EAESP/FGV/NPP, 2000. 71p.

BARTOLOMEU, Tereza Angélica. **Modelo de investigação de acidentes do trabalho baseado na aplicação de tecnologias de extração de conhecimento**. 2002. 302f. Tese (Doutorado em Engenharia de Produção) – EPS. Universidade Federal de Santa Catarina, Florianópolis, 2002.

BERRY, Michael J. A, LINOFF, Gordon. **Data mining techniques**: for marketing, sales, and customer support. New York : J. Wiley E Sons, 1997. 454 p.

CARDOSO, Olinda Nogueira. Paes. Recuperação de Informação. **INFOCOMP Revista de Computação da UFLA**, Lavras, v.1, 2000. Disponível em: <http://www.comp.ufla.br/infocomp/e-docs/a2v1/olinda.pdf> Acesso em: 23 out. 2003.

DINIZ, Carlos Alberto R., LOUZADA NETO, Francisco. **Data mining**: uma introdução. São Paulo: ABE, 2000. 123p.

FIGUEIRA, Rafael. **Mineração de dados e bancos de dados orientados a objetos**. 1998. 96f. Dissertação (Mestrado em Ciências da Computação) – UFRJ, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 1998.

FUNARO, Vânia Martins B. O., CARVALHO, Telma de, RAMOS, Lúcia Maria S. V. Costa. **Inserindo a disseminação seletiva da informação na era eletrônica**. São Paulo: Serviço de Documentação Odontológica de Faculdade de Odontologia da USP, 2001. 17p.

HAIR, Josep F. **Multivariate data analysis with readings**. 4.ed. Uper Saddle River: Prentice Hall, 1995. 745p.

HARRISON, T. H. **Intranet data warehouse**: ferramentas e técnicas para a utilização do data warehouse na intranet. Berkeley Brasil: São Paulo, 1998.

MEIRA JR, Wagner et. al. **Sistemas de comércio eletrônico:** projeto e desenvolvimento. Rio de Janeiro: Campus, 2002. 371 p.

REATEGUI, Eliseo. **Data mining e personalização dinâmica.** Criciúma: X Escola de Informática da SBC-Sul, 2002.

VELASQUEZ,Roberto M.G. et. al. Técnicas de Classificação para Caracterização da Curva de Carga de Empresas de Distribuição de Energia - Um Estudo Comparativo. **V Congresso Brasileiro de Redes Neurais**, 2001, Rio de Janeiro. Disponível em: <http://bioinfo.cpgei.cefetpr.br/anais/CBRN2001/5cbrn-6ern/artigos-5cbrn/>.

Criação do Sistema Integrado de Bibliotecas do Sistema ACAFE: utilizando JAVA e XML

Alberto Pereira de Jesus (FURB)
albertop@furb.br

Jefferson José Gomes(ACAFE)
jeffer@acafe.org.br

Resumo. Este artigo descreve a construção do Sistema Integrado de Bibliotecas do Sistema ACAFE (SINBAC), com o objetivo de integrar os metadados das instituições, possibilitando uma interface WEB de consulta centralizada para realização de empréstimos entre bibliotecas. Para que esta integração e interoperabilidade aconteçam nas Bibliotecas das IES foram utilizadas as tecnologias XML e JAVA, para tanto, definiu-se um padrão de metadados em XML, criou-se um sistema JAVA para indexar os dados e desenvolveu-se um sistema de recuperação de informações em JSP para busca integrada e a gerência dos serviços de empréstimos entre Bibliotecas.

Palavras-chave: Sistemas distribuídos, XML, JAVA, JSP e Bibliotecas.

1 Introdução

A era da informação exige das bibliotecas universitárias uma nova atitude que implica no rompimento das barreiras de acesso à informação. A Internet tornou-se uma ferramenta significativamente poderosa, permitindo o surgimento de possibilidades, oportunidades e desafios.

A utilização destes serviços em rede gera mudança de hábitos e rotinas bibliotecárias. Para Steele (apud MARCHIORI, 1997),

> *a pouco menos de dez anos, ninguém poderia predizer o impacto fenomenal da interconectividade global que, em conjunto com os desenvolvimentos de sistemas abertos e do poder dos microcomputadores, modificaria o gerenciamento das bibliotecas. Pela primeira vez, em uma centena de anos, estas enfrentam o grande desafio de rever e "redesenhar" seus serviços.*

Para as bibliotecas universitárias a Internet vem se tornando, uma ferramenta extremamente útil e necessária para seus serviços, dando uma maior amplitude, visibilidade e possibilitando disseminar as informações contidas em seus acervos.

A Internet, mais especificamente as redes de telecomunicações, possibilita a integração de sistemas, compartilhando dados e serviços, agregando valor e facilidades às redes de bibliotecas e de empresas. No meio da biblioteconomia já existem várias redes como a Rede Bibliodata/Calco entre outras que pregam a cooperação entre as instituições.

> *Cooperação bibliotecária é qualquer atividade realizada entre duas ou mais bibliotecas com objetivo de facilitar, promover e melhorar os processos da biblioteca, o uso de recursos ou os serviços aos usuários (CARVALHO apud MARKUSON, 1999, p. 148.)*

Assim, em Santa Catarina tem-se a Associação Catarinense das Fundações Educacionais (ACAFE), com sede em Florianópolis(SC), que tem como missão "promover a integração, a

cooperação e o desenvolvimento das instituições de ensino superior (IES) filiadas visando o fortalecimento da educação superior comunitária no Estado de Santa Catarina".

As bibliotecas universitárias das IES do Sistema ACAFE, formam uma rede de bibliotecas, a qual é mantida pela Câmara Setorial de Bibliotecas que foi criada em 15 de outubro de 2001, com objetivo de promover a integração e a melhoria dos serviços prestados pelas bibliotecas do Sistema ACAFE. Vários serviços já são oferecidos pela Câmara.

Entre as Fundações Educacionais participantes, há 11 universidades, 2 centros universitários e 3 faculdades, que em pesquisa realizada em 2002 as IES mantinham 66 bibliotecas com um acervo aproximado de mais de 530.000 títulos de livros e 13.500 títulos de periódicos. Para o atendimento ao público contavam com 64 bibliotecários, 214 auxiliares e 217 bolsistas. A maioria das bibliotecas oferece, um conjunto de serviços que inclui o empréstimo domiciliar, consulta local, consulta via Internet, levantamento bibliográfico, comutação bibliográfica, visitas orientadas, consulta local e remota a bases de dados, capacitação de usuários e acesso público à Internet.

Dessa forma, as bibliotecas das IES do Sistema ACAFE possuem, em sua maioria, um grande acervo. Porém, nada comparável à disponibilização de um único sistema de pesquisa, oportunizando que as bibliotecas possibilitem o acesso aos acervos e a diversos serviços como empréstimo entre bibliotecas, comutação bibliográfica, entre outros que possibilitarão e facilitarão a busca das informações que a comunidade acadêmica necessita.

Diante deste anseio, surgiu o Sistema Integrado de Bibliotecas da ACAFE (SINBAC), o qual congregará os acervos de todas IES, será inédito para o Estado e região sul do país, haja vista, que muitas bibliotecas universitárias ainda nem conseguiram informatizar seus próprios acervos. É um desafio instigante que, foi desenvolvido pela Câmara Setorial de Bibliotecas e a Câmara Setorial de Tecnologias da Informação e Comunicação com o apoio da ACAFE. Oportunizando novos serviços e criando um diferencial competitivo para IES do Sistema ACAFE.

A construção deste Sistema Integrado neste ambiente de diversos sistemas legados distintos foi possível através da utilização de metadados em XML; da linguagem JAVA para agrupar e indexar esses dados para recuperação; e de um sistema WEB de recuperação integrado desenvolvido em JSP.

Este artigo descreve as tecnologias utilizadas para o desenvolvimento deste projeto.

2 Objetivos

Construir um catálogo coletivo dos acervos das bibliotecas do Sistema ACAFE.

2.1 Objetivos específicos

a) identificar tecnologias de integração e interoperabilidade dos sistemas legados das IES;
b) integrar metadados bibliográficos das IES do sistema ACAFE;
c) disponibilizar um sistema de recuperação centralizado dos acervos das IES numa interface WEB;
d) facilitar a recuperação e a localização física e geográfica dos materiais bibliográficos;
e) promover a comutação e o empréstimo de materiais bibliográficos entre as IES do Sistema ACAFE.

3 Justificativa

Por meio dessa cooperação será permitida a ampliação e manutenção da infra-estrutura básica para o desenvolvimento da pesquisa, otimizando investimentos, disponibilizando acervos de excelência em todas as áreas de conhecimento, proporcionando a utilização em sua totalidade.

4 Metadados

Metadados são freqüentemente descritos como "dados sobre dados". Metadados não são mais do que informações adicionais (além da informação espacial e tabular) que é necessária para que os dados se tornem úteis. É informação essencial para que se possa fazer uso dos dados. Em resumo, metadados são conjuntos de características sobre os dados que não estão normalmente incluídas nos dados propriamente ditos. (ROSSETO; NOGUEIRA, 2002)

Os elementos metadados podem ter diferentes níveis de especificidade, estrutura e complexidade. E seu propósito primário é: descrever, identificar e definir um recurso eletrônico com o objetivo de modelar e filtrar o acesso, termos e condições para o seu uso, autenticação, avaliação, preservação e interoperabilidade.

Os metadados são importantes para a identificação, organização e recuperação da informação digital. Sua finalidade é facilitar, globalmente, a localização e recuperação das informações eletrônicas para os usuários. Neste sentido, utiliza-se os procedimentos técnicos de indexação e classificação dos conteúdos informacionais, possibilitando a integração de fontes diversificadas e heterogêneas de informação.

O uso de elementos metadados podem ser comparado ao dos elementos de descrição de registros bibliográficos contidos no catálogo da biblioteca, e até mais amplamente considerado o próprio processo de catalogação de uma biblioteca. Desta forma, o catálogo pode ser exemplificado como um tipo de metadados que emprega, basicamente, regras de catalogação e um formato de intercâmbio da informação, como o formato MARC.

Com a idéia inicial de metadados relacionados a catálogos bibliográficos, pode-se entender que as fichas bibliográficas de livros são metadados, proporcionando informação básica sobre as obras de um autor e relacionando-a com outras obras do mesmo autor ou com informações similares.

O que se denominava descrição bibliográfica ou registro bibliográfico no ambiente convencional das bibliotecas, no ambiente WWW passa a ser denominado metadados, tendo por finalidade descrever recursos informacionais, armazenados na Internet.

O esforço para desenvolver, organizar e padronizar o uso de metadados, é feito através de vários programas cooperativos na Internet, orientados pelo World Wide Web Consortium - W3C[1], órgão que regula o desenvolvimento técnico da Internet. Este órgão é responsável pela normalização evolutiva da linguagem HTML em todas as suas versões, e na implementação de linguagens derivadas como o XML (eXtensible Markup Language) (W3C, 2004).

4.1 XML (eXtensible Markup Language)

XML significa E**x**tensible **M**arkup **L**anguage (Linguagem de Marcação Extensível). Linguagens de marcação possibilitam a formatação de elementos por meio de atributos e tags como o HTML e XHTML (SILVA, 2001).

Segundo Silva (2001), o grande diferencial de XML é ser extensível, possibilitando a criação de elementos, assim, você mesmo pode criar suas tags conforme suas necessidades. Sua finalidade

[1] www.w3.org

é descrever informações. Assim, podem-se criar padrões, que sejam de interesse de um grupo de pessoas ou empresas, facilitando o processo de interpretação das informações para os mais variados sistemas.

Afirma Silva (2001, p.18), "um dos usos mais difundidos do XML é o armazenamento e transação de dados entre empresas". Desenvolvedores encontram em XML uma poderosa ferramenta para representação, modelagem e interoperação de dados.

XML conta com conjunto de tecnologias para a manipulação, transformação, localização e visualização:

a) manipulação via programação em DOM e SAX;
b) transformação: XSLT e Xpath;
c) localização e extração: XLink, XQuery e Xpointer;
d) visualização: XSL-FO.

4.2 JAVA e XML

A Internet permite integrar aplicações localizadas em plataformas e sistemas operacionais diferentes. Sendo Java uma linguagem de programação independente de plataforma, a qual disponibiliza várias classes para manipulação de XML. Aliado a sua facilidade para desenvolvimento de aplicações em rede, assim, é possível escrever aplicações distribuídas que reforçam o conceito de interoperabilidade utilizando JAVA e XML (VELOSO, 2003).

5 Estudo de caso

As Bibliotecas do Sistema ACAFE apresentam as seguintes situações:

a) encontram-se distribuídas fisicamente em todo o estado de Santa Catarina;
b) apresentam sistemas legados diferentes, algumas com sistemas próprios e outras sistemas comerciais;
c) utilizam diversos formatos de entrada dos dados bibliográficos;
d) algumas não possuem serviços WEB;
e) integram o sistema RCT-2, assim estão ligadas à Internet;
f) diversificado parque tecnológico.

Diante da situação atual encontrada e do problema a respeito da integração do acervo chegaram-se as possíveis soluções para a integração dos acervos:

a) desenvolvimento de um sistema de busca distribuída;
b) centralização dos dados, através de metadados e busca.

Adotou-se como solução a centralização dos dados, devido à situação das bibliotecas. Pois, as mesmas estavam em diferentes estágios de automação. Isso exigiria uma adequação à tecnologia, trazendo gastos com a aquisição de softwares e computadores. Levou-se em conta, também, o tempo de processamento e os constantes problemas de fluxo da Internet.

Após decidir-se quanto as medidas a serem adotadas, partiu-se para o levantamento das informações que deveriam ser contempladas pelo catálogo, as quais deveriam possibilitar ao usuário pesquisar a obra, gerar uma referência bibliográfica, identificar a sua localização, bem como solicitá-la para empréstimo. As variáveis levantadas foram: biblioteca, campus, obra que contém título, subtítulo, autores, assuntos, local de publicação, editora, data de publicação, formato, ISBN no caso de livros e ISSN no caso de periódicos, idioma, edição, código da obra no catálogo do Bibliodata Calco.

Diante dos dados levantados e da solução adotada, tem-se um servidor central que recebe, armazena e indexa os dados das bibliotecas. A definição quanto às tecnologias adotadas com vista a custo, facilidade, agilidade foram:

a) XML para suporte à importação e exportação de dados;
b) JAVA para desenvolvimento do servidor de importação de dados;
c) JSP/Servlets para desenvolvimento do sistema de consulta;
d) SQL Server para dar suporte ao armazenamento de informações e à indexação.

Quanto à infra-estrutura necessária:

a) servidor central de banco de dados e WEB, que fica localizado na sede da ACAFE, em Florianópolis, SC. Recebendo os arquivos de dados das bibliotecas processando estes arquivos armazenando e indexando. Para após serem consultados;
b) nas bibliotecas, os sistemas de catalogação geram um arquivo XML diariamente com os dados sobre seus acervos e disponibilizam em uma URL para acesso pelo servidor central;
c) os usuários através de um navegador acessam o sistema de consulta.

A figura 1 apresenta o modelo de funcionamento do sistema de recuperação de obras e a figura 2 o modelo de funcionamento dos pedidos de obras.

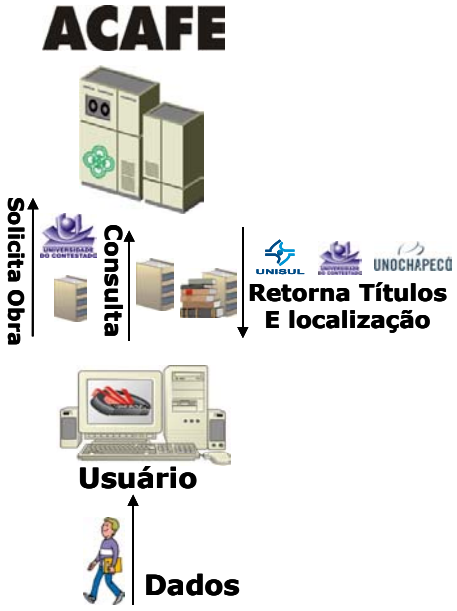


Figura 1: Modelo de funcionamento do sistema de recuperação de obras

Figura 2: Modelo de funcionamento do pedido de obras

5.1 Implementações

As seguir são apresentadas as etapas para execução e implementação do projeto que são: definição dos metadados, servidor de integração e sistema de recuperação de informações.

5.1.1 Definição dos metadados

Cada Biblioteca possui seu sistema legado para o controle do acervo. Sendo que todas trabalham com o formato de catalogação MARC. Para a integração dos dados das bibliotecas foi definido um metadado no formato XML e um mapeamento do formato MARC para XML.

A partir da definição dos metadados, as bibliotecas começam a gerar metadados do seu acervo. O sistema legado da biblioteca fica controlando as alterações, inclusões e exclusões e a noite gera o XML no formato especificado disponibilizando o mesmo em uma URL a qual é acessada pelo servidor responsável por armazenar e indexar estes dados. Na figura 3 um exemplo do XML gerado.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<biblioteca>
 <id_ies>1</id_ies>
 <obra>
 <titulo>Compreensão e produção de textos</titulo>
  <autor tipo="P">Souza, Luiz Marques de,</autor>
  <autor tipo="S">Carvalho, Sergio Waldeck de</autor>
  <assunto>Redação</assunto>
  <assunto>Língua portuguesa Composição e exercícios</assunto>
  <editor>Petrópolis : Vozes</editor>
  <data>2003</data>
  <tipo>1</tipo>
  <formato>164p.</formato>
  <identificador>8532614906 (broch)</identificador>
  <edicao>8. ed.</edicao>
  <id_campus id_legado="267437">5</id_campus>
  <cod_mov>1</cod_mov>
 </obra>
</biblioteca>
```

Figura 3: Metadados XML gerado

5.1.2 Servidor de integração

Para a integração dos dados foi desenvolvido um sistema em JAVA que busca o XML, com os metadados bibliográficos, em cada biblioteca, e, através de uma URL processa os mesmos armazenando, indexando e controlando as redundâncias de livros. Quando encontra livros iguais, estes são agrupados em uma mesma entidade, gerando somente relacionamentos para as bibliotecas que possuem a obra.

O processamento é realizado todos os dias pelo servidor, gerando logs do processamento dos arquivos XML. Estes logs são enviados por e-mail às bibliotecas de origem dos dados.

5.1.3 Sistema de recuperação de informações

Para a recuperação dos dados indexados pelo servidor de integração, (descrito no item 5.1.2) foi desenvolvido um sistema WEB em JSP e Servlets para a recuperação de informações e empréstimo entre bibliotecas.

Na figura 4 é apresenta um macro fluxo do sistema desenvolvido e descreve as principais funcionalidades.

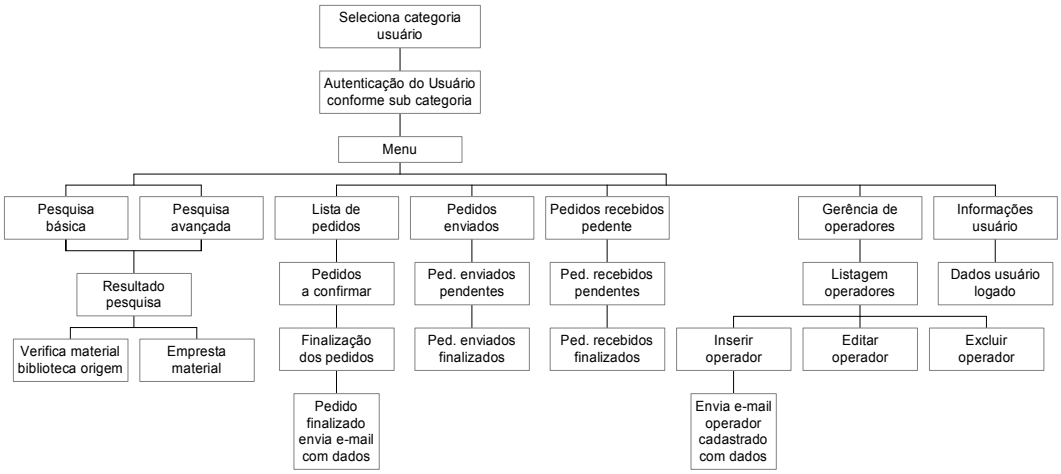


Figura 4: Macro fluxo do sistema

O sistema contempla a pesquisa integrada no acervo das IES da ACAFE, bem como, o empréstimo entre bibliotecas, proporcionando um gerenciamento e controle dos empréstimos.

Para acessar o sistema é necessário a identificação do usuário (figura 5). Após a identificação, realiza-se a autenticação é mostrada a tela principal do sistema (figura 6), à direita o menu, as opções do sistema:

a) pesquisa básica: encontra registros buscando por palavras-chave através de campos específicos como autor, título, assunto ou todos que busca por qualquer palavra da obra que foi indexada;

b) pesquisa avançada: encontrar registros através de palavras-chave e campos específicos utilizando-se de filtros como: tipos de materiais específicos; regiões do estado; instituições; campus;

c) lista de pedidos: apresenta as obras selecionadas pelo usuário que aguardam uma confirmação para pedido;

d) pedidos enviados: gerencia os pedidos que foram enviados;

e) pedidos recebidos: gerencia os pedidos que foram solicitados;

f) altera senha: possibilita a alteração da senha do usuário;

g) operadores: cadastro de novos operadores;

h) logout: sair do sistema.

Figura 5: Tela de login

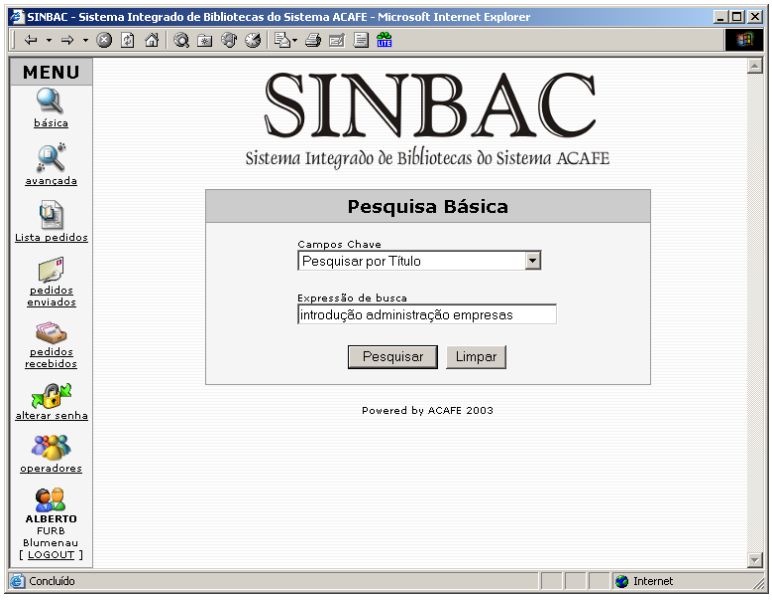


Figura 6: Tela principal do sistema

Para realizar uma busca pode-se optar pela pesquisa básica ou avançada, digitando uma expressão de busca. A consulta trará os resultados de todas as bibliotecas que integram o Sistema ACAFE.

O resultado da pesquisa é apresentado em páginas contendo 10 registros através da aplicação de uma folha de estilo XSL sobre os dados que são retornados em XML. Os registros são apresentados em forma de tabela contendo informações básicas, e necessárias para a pesquisa do material com a possibilidade de gerar uma referência. O resultado da consulta apresenta as bibliotecas que possuem a obra, conforme apresentado na figura 7.

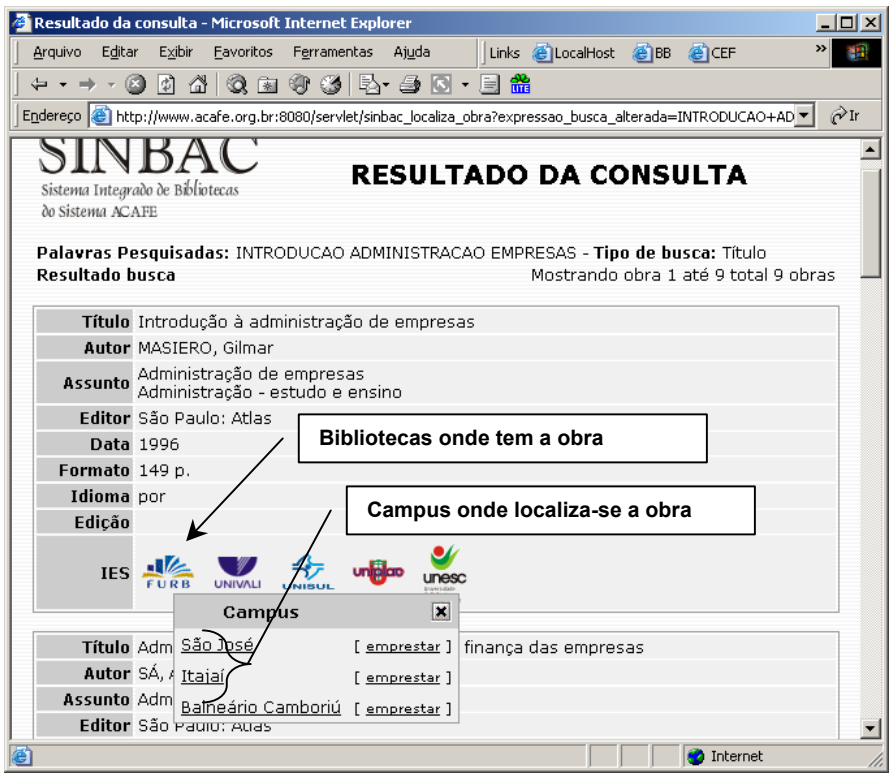


Figura 7: Tela resultado da busca

Selecionando emprestar os livros são inseridos em uma lista de pedidos (figura 8), após confirmação do pedido, é enviado a biblioteca solicitada. Recebem um aviso via e-mail os responsáveis pelo empréstimo entre bibliotecas e o solicitante.

Figura 8: Tela de confirmação do pedido

A gerência dos pedidos enviados, como dos recebidos é feita nos itens: pedidos enviados e pedidos recebidos. Possibilita o acompanhamento dos estados dos pedidos, o controle de comunicações com o solicitante, mudança do status do pedido e cadastrar informações sobre o mesmo, conforme apresentado na figura 9 e figura 10.

Figura 9: Tela de gerência de pedidos enviados

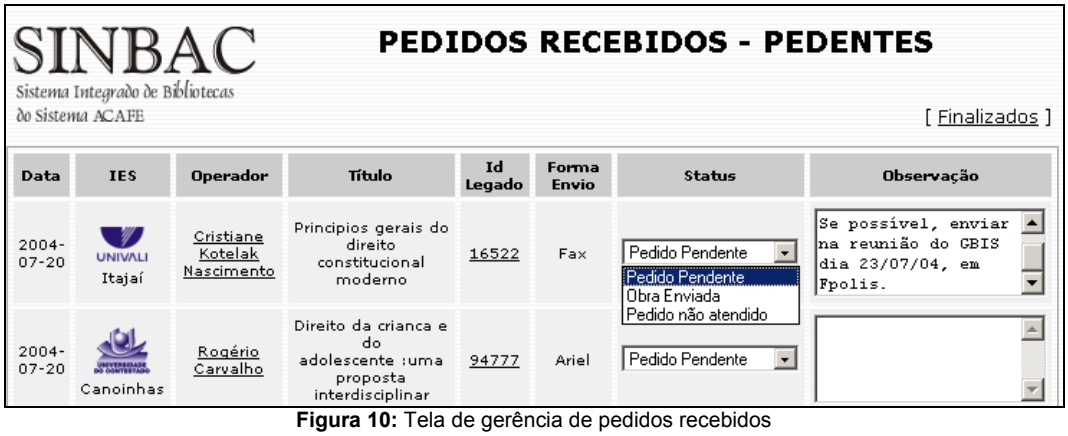


Figura 10: Tela de gerência de pedidos recebidos

6 Conclusões

As tecnologias de Java e XML permitem integrar aplicações localizadas em plataformas e sistemas operacionais diferentes. Pois Java é uma linguagem independente de plataforma e XML é totalmente baseado em texto. Assim foi possível integrar as bibliotecas do Sistema ACAFE. Onde existiam vários sistemas legados.

Com a definição de um padrão de metadados em XML, será possível que novas bibliotecas venham participar do projeto, pois o padrão foi totalmente baseado nas normas de catalogação.

Os objetivos do trabalho foram alcançados. A construção de um catálogo coletivo dos acervos das bibliotecas do Sistema ACAFE mostrou-se funcional ao seu propósito, facilitando a recuperação de obras e o empréstimo entre bibliotecas. O sistema já esta sendo utilizado.

Este projeto inovador e único do estado de Santa Catarina, possibilita um diferencial as IES do Sistema ACAFE, bem como possibilitará as bibliotecas universitárias compartilhar acervos e serviços, colocando em prática o papel de agentes disseminadores de informações.

Referências

CARVALHO, Maria Carvalho Romcy. **Compartilhamento de recursos e acesso à informação no Brasil:** um estudo nas áreas de química e engenharia química. 1999. Tese (Doutorado em Ciência da Informação) – Universidade de Brasília, Brasília, 1999.

MARCHIORI, P. Z. “Ciberteca” ou biblioteca virtual: uma perspectiva de gerenciamento de recursos de informação. **Ciência da Informação**, Brasília, v. 26, n. 2, p.115-124, 1997.

ROSETTO, Marcia; NOGUEIRA, Adriana Hypólito.**Aplicação de elementos metadados dublin core para descrição de dados bibliográficos on-line da biblioteca digital de teses da USP**. SNBU , 2002.

SILVA, Osmar J. **XML**: aplicações práticas. São Paulo: Érica, 2001. 276p.

VELOSO, Renê Rodrigues. **Java e XML**: processamento de documentos XML com Java. São Paulo: Novatec, 2003. 96p.

W3C, World Wide Web Consortium <http://www.w3.org/XML/> Acesso em 22/07/2004.

Reutilização de soluções com *patterns* e *frameworks* na camada de negócio

Márcio Carlos Grott (Totall.com)
grott@totall.com.br

Fabio Cordova de Sousa (Consei)
fabio@consei.com.br

Marcel Hugo (FURB)
marcel@furb.br

Resumo: Este artigo apresenta os padrões de projeto (*design paterns*) e *frameworks* para o desenvolvimento de um *framework* para cálculo de impostos incidentes em vendas de mercadorias. Como resultado, chegou-se a um conjunto de informações que podem ser utilizados por desenvolvedores interessados em estudar reutilização e aplicação de padrões de projeto e *frameworks* para a melhoria de seu desenvolvimento de software. Além disto, foi desenvolvido um *framework* em Java para cálculo de impostos, aplicando diferentes padrões de projeto: *Singleton, Factory Method, Flyweight e Strategy*.

Palavras chaves*: framework, patterns,* reutilização, impostos

1 Introdução

Os projetistas de sistemas devem encontrar os objetos e fatorá-los em classes no nível correto de granularidade, definindo as interfaces das classes e as hierarquias de herança, estabelecendo as relações chaves entre eles, criando projetos específicos para resolver o problema, mas também genérico o suficiente para atender futuros problemas e requisitos (GAMMA, 2000). Os objetos bem modelados fazem com que os programadores ganhem mais tempo na implementação. Depois de modeladas as classes bases e testadas, os desenvolvedores precisam apenas se preocupar com a implementação da especialização da subclasse criada (ALUR, 2002).

Para se poder reutilizar uma maior quantidade de soluções, não basta ter escrito classes que descrevem toda a complexidade do projeto. Os projetistas novatos tendem a usar técnicas não orientadas a objetos pela sobrecarga de opções disponíveis. Sistemas cada vez mais complexos tendem a ter um número de classes cada vez maior, dificultando o desenvolvedor na codificação. Com o passar dos anos de experiência dos arquitetos em projetar soluções, observou-se que a essência dos problemas é semelhante e que quando especializavam as classes bases, o que é difícil conseguir na primeira vez, ganhavam maior produtividade no desenvolvimento, fazendo a reutilização de soluções corretas e que comprovadamente funcionavam (GAMMA, 2000) (SUN MYCROSYSTEM, 2002).

Observando que na essência os problemas assemelham-se, os projetistas começaram a documentar as soluções criadas durante os anos. Essa documentação foi sendo aprimorada formando padrões (*patterns*) de desenvolvimento. Os padrões, segundo Alur (2002), permitem documentar um problema conhecido e recorrente (que surgem várias vezes durante o projeto) e sua solução em um contexto específico e comunicar esse conhecimento para outras pessoas. Cada padrão tem uma aplicação específica dentro da arquitetura do software composto de multicamadas que separam a lógica de negócio, da interface do usuário e do acesso aos dados persistentes.

Os padrões podem ser agrupados em um *framework* (PREE, 1995). Booch (1994) define "*frameworks* como coleção de classes que fornece um conjunto de serviços para um domínio particular". Um *framework* pode conter um conjunto de classes no qual foram aplicados um ou vários padrões na sua modelagem. Os *frameworks* proporcionam um alto grau de reutilização, pois a partir do mesmo, outros sistemas podem herdar funcionalidades reutilizando a solução desenvolvida e testada em vários outros sistemas já implementados.

Este artigo tem como objetivo apresentar o desenvolvimento de um *framework* em Java para cálculo de impostos incidentes durante o processo de aquisição de uma mercadoria, aplicando os padrões de projeto *Singleton*, *Factory Method*, *Flyweight* e *Strategy* melhorando a qualidade, a reutilização e facilitando a manutenção que são constantes devido a modificações na legislação fiscal. A utilização de Java para o desenvolvimento do *framework* deve-se à sua freqüente utilização no mundo da orientação a objetos. Por tratar-se de uma linguagem fortemente orientada a objetos e de alta legibilidade, tem-se um aumento significativo na facilidade de entendimento do código-fonte.

O artigo divide-se em cinco seções, iniciando com os conceitos envolvidos no desenvolvimento de software com a utilização de padrões e *frameworks*, a implementação do *framework* para cálculo de impostos demonstrada através de um protótipo e ao final apresenta as conclusões alcançadas.

2 Patterns

Na área da informática, a evolução das capacidades computacionais e os seus requisitos crescem em ritmo acelerado. Dado que a capacidade de processamento duplica a cada 18 meses, segundo a lei de Moore, Pressman (1995) pergunta como a Engenharia de Software é capaz de responder a este crescente desenvolvimento

A primeira solução aparece com a reutilização de código e de soluções anteriormente implementadas, com a intenção de aumentar a rapidez com que são criadas novas aplicações e de aprimorar sua qualidade final. Mas mesmo assim, não é evitado um problema: "Quem desenvolve software tem muitas vezes a sensação de que está reinventando a roda". Dentro da mesma empresa pode estar um desenvolvedor tentando resolver um problema, enquanto que o colega no cubículo ao lado já tem a solução há meses.

A solução encontrada originou-se da adaptação dos conceitos de padrões de arquitetura, desenvolvida por Christopher Alexander, à área da Engenharia de Software. Os padrões aparecem como um método para facilitar essa comunicação de conhecimentos entre entidades e indivíduos. Cada padrão descreve um problema que ocorre repetidas vezes em nosso ambiente, e então descreve o núcleo da solução para aquele problema, de tal maneira que se pode usar essa solução milhões de vezes sem nunca fazê-la da mesma forma duas vezes segundo Christopher Alexander (apud GAMMA, 2000).

Os padrões abrangem diferentes níveis de abstração, resultantes da enorme aplicabilidade prática que possuem, podendo classificá-los em diversas categorias, de modo a facilitar a sua recuperação e utilização num determinado problema. No entanto, esta classificação é um pouco flexível, podendo haver padrões que nitidamente se encaixam em várias categorias. Na tabela 1 são descritas as categorias mais importantes de padrões, segundo Alur (2002).

Padrão	Explicação
Padrões de processo	Definem soluções para os problemas encontrados nos processos envolvidos na engenharia de Software: desenvolvimento, controle de configuração, testes, etc.
Padrões arquiteturais	Exprimem a organização das estruturas fundamentais de sistemas de software ou hardware.
Padrões de padrão (do Inglês, *patterns on patterns*)	São padrões que descrevem como um padrão deve ser escrito, ou seja, padrões que uniformizam o modo como estes devem ser apresentados aos seus utilizadores. Padrões de análise descrevem soluções para os problemas de análise de sistemas, incluindo conhecimento sobre o domínio específico de aplicação.
Padrões de projeto	Definem soluções para diversos problemas de projeto de software. Padrões de interface definem soluções para problemas comuns no projeto de interfaces de sistemas. É um caso particular dos padrões de projeto.
Padrões de programação	Descrevem soluções particulares de programação numa determinada linguagem ou regras gerais de estilo de programação.
Padrões de persistência	Descrevem soluções para problemas de armazenamento de informações em arquivos ou bancos de dados.
Padrões de Hipertexto	Descrevem soluções para problemas que se encontram no projeto de aplicação direcionadas para a World Wide Web (WWW).

Tabela 1: Categorias de padrões mais importantes

Para facilitar a utilização e divulgação dos padrões de projeto, existe uma catalogação que difere entre alguns autores sob o aspecto de propósito e intenção de utilização. Um catálogo de padrões pode oferecer um esquema de classificação e recuperação dos seus padrões, já que eles estão subdivididos em categorias, e adiciona uma certa quantidade de estrutura e organização a uma coleção de padrões, mas tipicamente não vai além de mostrar apenas as estruturas e relações mais superficiais (quando o faz). Na figura 1 tem-se a classificação feita por Gamma (2000), classificado pelo seu escopo e um determinado propósito.

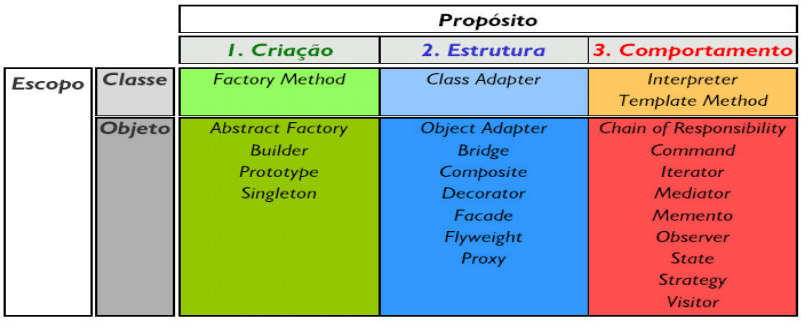


Figura 1: Catálogo de padrões de projeto feito por Gamma(2000).

2.1 Aplicação de Patterns no desenvolvimento de software

Segundo Buschmann (1998), os padrões não definem um novo método para o desenvolvimento de software que substitua os já existentes. Eles apenas complementam os

métodos de análise e projeto gerais e independentes do problema, por exemplo, Booch, OMT, Shlaer/Mellor, etc, com diretrizes para resolver problemas específicos e concretos. Buschmann (1998) sugeriu os seguintes passos para desenvolver um sistema usando padrões de software:

a) utilizar o método preferido para o processo de desenvolvimento de software em cada fase do desenvolvimento;

b) utilizar um sistema de padrões adequado para guiar o projeto e a implementação de soluções para problemas específicos, isto é, sempre que se encontrar um padrão que resolva um problema do projeto presente no sistema, utilizar os passos da implementação associados a esse padrão. Se esses se referirem aos outros padrões, aplicam-se recursivamente;

c) se o sistema de padrões não incluir um padrão para o problema do projeto, tenta-se encontrar um padrão em outras fontes conhecidas;

d) se nenhum padrão estiver disponível, aplica-se as diretrizes de análise e projeto do método que se está usando. Essas diretrizes fornecem pelo menos algum apoio útil para resolver o problema do projeto em questão.

Segundo Buschmann (1998), essa abordagem simples evita que se criem outros métodos de projeto. Ela combina a experiência de desenvolvimento de software captada pelos métodos de análise e projeto com as soluções específicas para problemas de projeto descritas pelos padrões.

3 Framework

Framework é definido por Coad (1988) como um esqueleto de classes, objetos e relacionamentos agrupados para construir aplicações específicas como um conjunto de classes abstratas e concretas que provê uma infra-estrutura genérica de soluções para um conjunto de problemas. Essas classes podem fazer parte de uma biblioteca de classes ou podem ser específicas da aplicação. *Frameworks* possibilitam reutilizar não só componentes isolados como também, toda a arquitetura de um domínio específico.

Uma característica importante dos *frameworks* é que os métodos (funções e procedimentos) definidos pelo desenvolvedor para especializá-los são chamados de dentro do próprio *framework*, ao invés de serem chamados do código da aplicação do desenvolvedor. A reutilização em nível de *framework* leva a uma inversão de controle. Quando se utiliza uma biblioteca convencional de sub-rotinas, escreve-se o corpo principal da aplicação e chama-se o código que quer reutilizar. Quando se utiliza *framework* reutiliza-se o corpo principal e escreve-se o código que este chama (GAMMA, 2000).

O desenvolvimento “ideal” de aplicações a partir de *frameworks* consiste em completar ou alterar procedimentos e estruturas de dados neles presentes. Sob esta ótica, uma aplicação gerada por um *framework* não deveria incluir classes que não fossem subclasses das classes do *framework*. Todavia, como um *framework* nunca é uma descrição completa de um domínio, é possível que a construção de aplicações por um *framework* leve à obtenção de novos conhecimentos do domínio tratado, indisponíveis durante a sua construção, ensejando a criação de novas classes.

O desenvolvimento de aplicações inicia com o entendimento do sistema contido no projeto do *framework* e segue no detalhamento das particularidades da aplicação específica, o que é definido por seu usuário. Assim, a implementação de uma aplicação a partir do *framework* é feita pela adaptação de sua estrutura de classes, fazendo com que esta inclua as particularidades da aplicação.

4 Desenvolvimento do framework de cálculo de impostos

A implementação de *frameworks* para cálculos de impostos, devido à complexidade existente nas leis tributárias vigentes e sua constante modificação, constitui-se uma interessante área de aplicação de *patterns* para facilitar a modelagem, o entendimento e manutenção da implementação de um *framework*.

O desenvolvimento de um *framework* requer uma modelagem que abranja de forma concisa e precisa o domínio da utilização de *framework*. Uma modelagem abrangente geralmente é constituída de várias classes utilizadas para realização de tarefas específicas do domínio da aplicação. Com a interação de várias classes em uma modelagem inicia-se a aplicação de padrões de projetos. Os padrões de projeto utilizados no decorrer deste tópico são extraídos do livro de Gamma (2000), denominados popularmente de padrões GoF (*gang of four*).

4.1 Aplicação de alguns padrões para implementação do *framework*

4.1.1 Singleton

O padrão *Singleton* deve garantir que uma classe tenha somente uma instância e fornecer um ponto de acesso para a mesma. Na tabela 2 demonstra-se o componente relevante do *pattern Singleton*.

Itens	Descrição
Nome	*Singleton* – criação de Objetos
Motivação	Deve-se garantir que uma classe tenha somente uma instância. Por exemplo, embora possam existir inúmeras impressoras em um sistema, deveria haver somente um *spooler* de impressora.
Conseqüências	a) acesso controlado à instância única; b) espaço de nomes reduzido; c) permite refinamento de operações e representação; d) permite um número variável de instancias; e) mais flexível do que operações de classes;
Implementação	a) garantir a existência de uma única instância; b) criando subclasses da classe Singleton.

Tabela 2: Itens relevantes na composição do padrão Singleton. Fonte: Gamma (2000)

No *framework* de cálculo de Impostos somente existe a necessidade de criação de uma única instância da classe Emissor, pois a mesma uma vez instanciada será utilizada pelo restante da aplicação. O diagrama de classes da classe Emissor pode ser visto na figura 2 demonstrando a utilização do padrão *Singleton* na implementação do *framework*.

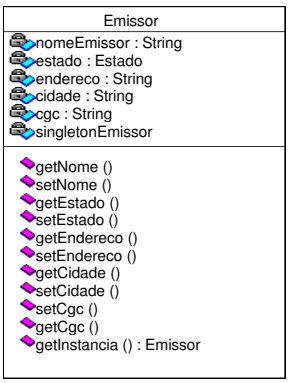


Figura 2: Diagrama de classes da classe Emissor implementando o padrão *Singleton*

4.1.2 Factory Method

O padrão *Factory Method* é uma interface para instanciação de objetos que mantém isoladas as classes concretas usadas na requisição da criação destes objetos. A separação de classes em uma "família" dotadas de uma mesma interface ("produtos") e uma classe ("fábrica") que possui um método (o *factory method*) que cria tais objetos. Na tabela 3 demonstra-se o componente relevante do padrão *Factory Method*.

Itens	Descrição
Nome	*Factory Method* – criação de classes.
Motivação	Os *frameworks* usam classes abstratas para definir e manter relacionamentos entre objetos. Um *framework* é freqüentemente responsável também pela criação destes objetos. O *pattern Factory Method* oferece uma solução. Ele encapsula o conhecimento sobre a subclasse que deve ser criada e move este conhecimento para fora do *framework*.
Conseqüências	a) fornece ganchos para subclasses; b) conecta hierarquias de classes paralelas.
Implementação	a) duas variedades principais: - a classe Creator é uma classe abstrata e não fornece uma implementação para o método fabrica que ela declara. - quando a classe Creator é uma classe concreta e fornece uma implementação por omissão para o método fabrica. b) Métodos *factory* parametrizadas.

Tabela 3: Itens relevantes da composição do Factory Method. Fonte: Gamma (2000)

O *framework* de cálculo de imposto utilizará quando fizer a criação das classes de imposto como IPI, ICMS, etc. Quando da instanciação da classe de *Imposto* será passado um parâmetro que identificará qual classe de imposto deverá ser criada.

Na figura 4 demonstra-se a classe abstrata *Imposto* que modela os métodos necessários para utilizar as subclasses concretas *IPI* e *ICMS*. Nesta classe estão definidos os métodos que são necessários quando se desejar implementar algum novo imposto designado pelo governo. No diagrama da figura 3 pode-se observar que há um método *createImposto* da classe abstrata *ImpostoFactory* que é responsável pela instanciação de objetos de subclasses de *Imposto*. Cada nova classe que herdar de Imposto e desejar incluir-se na *ImpostoFactory* deve sobrescrever o

método *create*. Isto cria uma fábrica de objetos que funciona de forma polimórfica para inclusão e instanciação de novos objetos de *Imposto*.

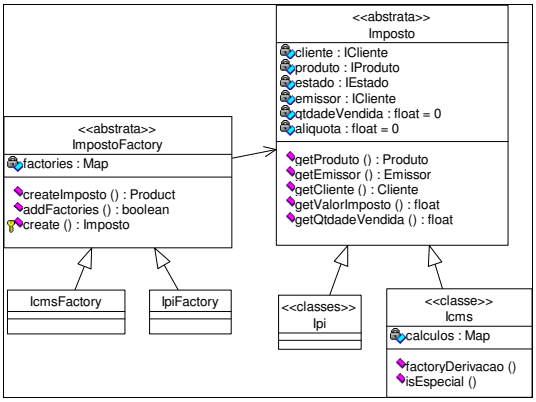


Figura 3: Diagrama da classe abstrata Imposto

4.1.3 Flyweight

O padrão *Flyweight* consiste em suportar grande quantidade de objetos de granularidade fina onde o custo de armazenamento é alto devido à grande quantidade de objetos que podem ser agrupados Gamma (2000). Às vezes pode-se reduzir bastante o número de diferentes classes que se necessita instanciar se puder reorganizar quais instâncias são fundamentalmente de mesma forma exceto por alguns poucos parâmetros. Na tabela 4 demonstra-se item relevante ao padrão *Flyweight* para sua formação e aplicação.

Itens	Descrição
Nome	*Flyweight* – estrutural de objetos.
Motivação	Quando se tem uma grande quantidade de objetos de fina granularidade podendo-se agrupá-los em famílias de objetos conforme as afinidades encontradas.
Conseqüências	a) redução de número de instâncias; b) economia de armazenamento e espaço de compartilhamento de objetos; c) o tempo de execução aumenta porque o estado de cada objeto deve ser calculado.
Implementação	a) remoção de estados extrínsecos; b) a gerencia dos objetos compartilhados.

Tabela 4: Descrição dos itens relevantes do padrão Flyweight. Fonte: Gamma (2000)

Na figura 4 demonstra-se o diagrama de classes da estrutura do padrão *flyweight* sendo modelado no *framework* de cálculo de imposto. Observa-se que a modelagem abriga dois outros padrões que é um *Factory Method* na classe *ICMS* e um *Strategy* com relação à implementação do cálculo de imposto.

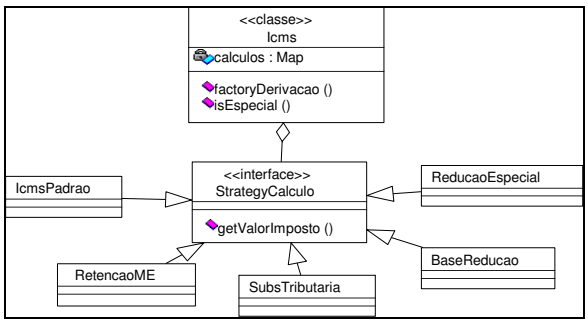


Figura 4: Diagrama de classe demonstrando o padrão *Flyweight*

4.1.4 Strategy

O padrão *Strategy* consiste na definição de uma família de algoritmos, encapsula cada um e faz com que eles possam ser permutáveis. Isto permite que o algoritmo varie independentemente do cliente que o utilizar Gamma (2000). Algumas características deste padrão são exploradas na tabela 5.

Itens	Descrição
Nome	Strategy – comportamental de objetos.
Motivação	Quando se necessita de um algoritmo que trata de modos diferentes os dados submetidos a ele.
Conseqüências	a) famílias de algoritmos relacionados; b) uma alternativa ao uso de subclasses; c) *Strategies* eliminam comandos condicionais; d) uma escolha de implementações; e) os clientes devem estar cientes dos diferentes *Strategies*; f) aumento do número de objetos.
Implementação	a) estratégias de implementação passadas como parâmetro; b) torna os objetos *Strategies* opcionais;

Tabela 5: Descrição dos itens relevante ao padrão Strategy. Fonte: Gamma (2000)

No *framework* de cálculo de imposto verifica-se a aplicação deste padrão quando da definição da classe abstrata *Imposto* que tem um método chamado *getValorImposto()*. Este método é implementado diferentemente nas classes concretas que implementam a classe *Imposto* consistindo em um exemplo de aplicação do padrão *Strategy* (figura 5).

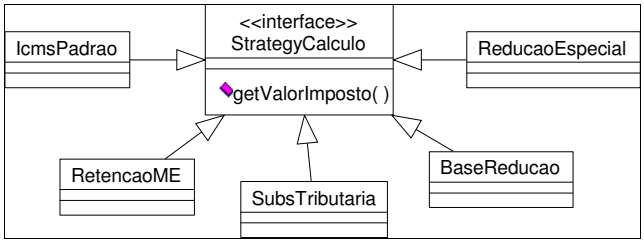


Figura 5: Classe abstrata imposto demonstrando a aplicação do padrão *Strategy*

O diagrama de classes completo pode ser visto na figura 6 onde são demonstradas todas as classes que compõem o *framework* para realização de cálculos de impostos. A partir desde diagrama tem-se a possibilidade de estendê-lo, completando-o com os demais cálculos de impostos que são inerentes a cada negócio.

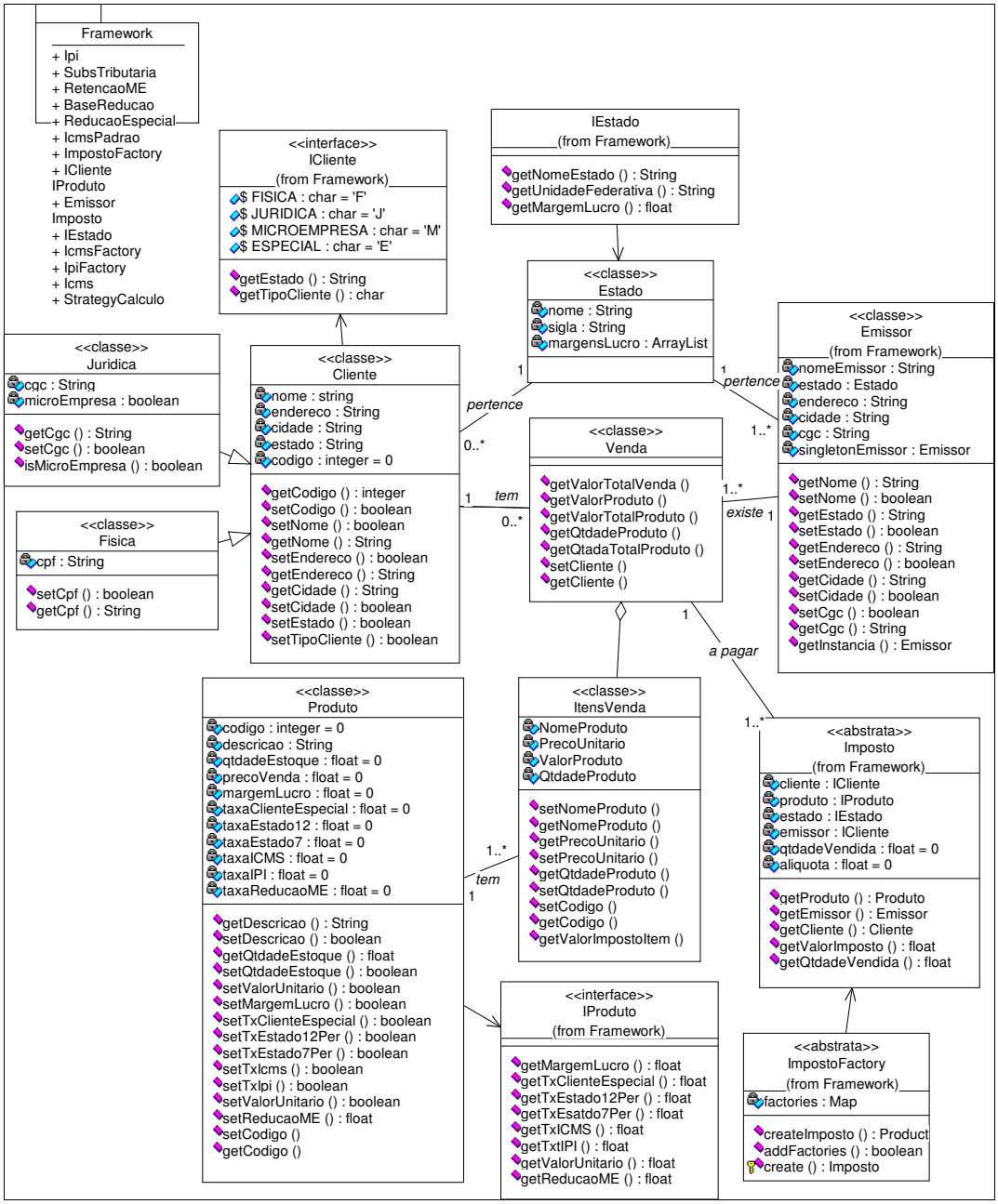


Figura 6: Diagrama de classes do *framework*

O diagrama de seqüência da figura 7 mostra a seqüência de desencadeamento das chamadas dos métodos para criar um novo objeto de imposto. Neste exemplo está sendo criado um objeto *Icms* que é criado dentro da classe *ImpostoFactory*.

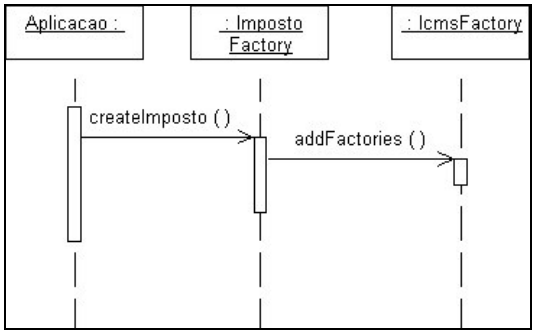


Figura 7: Diagrama de seqüência da criação de um objeto imposto(*createImposto*)

No diagrama de seqüência demonstrado na figura 8 verifica-se o desencadeamento das chamadas dos métodos que calculam o valor de um imposto. O diagrama mostra o *factoryDerivacao* que cria todos os impostos que são derivados da classe *Icms* e chama o método *getValorImposto* implementado nas classes que implementam a estratégia de cálculo de cada derivação do ICMS.

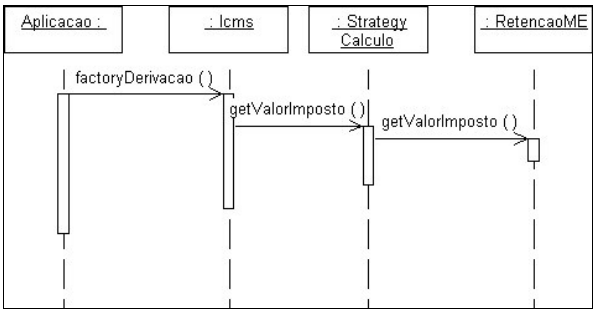


Figura 8: Diagrama de seqüência do cálculo de um imposto(*getValorImposto*)

Para demonstrar a utilização do *framework* foi construído um protótipo de um sistema de automação comercial. A figura 9 mostra a tela de atendimento de um cliente e a aplicação dos impostos incidentes durante o processo de venda segundo as leis tributárias vigentes. Maiores informações podem ser obtidas em Grott (2003).

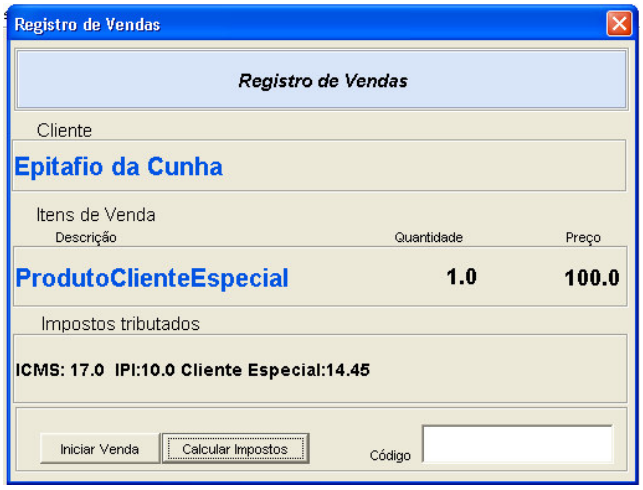


Figura 9: Tela de atendimento ao cliente na compra de mercadorias.

5 Conclusões

A utilização de padrões de projeto dentro da modelagem e codificação cria micro unidades que podem ser alteradas com pouca ou nenhuma interferência no software. Os padrões selecionados, *Singleton, Factory Method, Flyweight e Strategy* criam um alto nível de abstração da maneira como será utilizada a implementação do padrão.

Com a modelagem e implementação de cálculos de impostos foi possível utilizar padrões de projetos, que solucionaram desde o problema da não necessidade de criação de várias instâncias de um mesmo objeto, que é único para todo o sistema; até a criação de objetos tendo somente uma classe responsável pela instanciação, tornando-se uma fábrica de objetos; passando pela utilização de um padrão que dita uma estratégia a ser adotada quando há algoritmos muito semelhantes para cálculos de diferentes impostos.

A implementação dos padrões resultou na formação de um *framework* facilitando as tarefas do desenvolvedor e sendo extensível quando houver necessidade de acrescentar novas classes ou implementar novos padrões. Para o desenvolvedor de uma aplicação, o funcionamento dos cálculos será totalmente abstraído pelo conhecimento agregado ao *framework*, fazendo com que o desenvolvedor tenha mais tempo nas tarefas inerentes a aplicação. Outra vantagem apresentada é o baixo acoplamento aliado à facilidade de utilização, pois o *framework* necessita da implementação de três interfaces *IProduto*, *ICliente* e *IEstado*.

Referências

ALUR, Deepk; CUPRI, John; MALKS, Dan. **As melhores práticas e estratégias de design.** Rio de Janeiro: Campus, 2002.

BOOCH, Grady. **Object-oriented analysis and design with applications.** Santa Clara: Addison-Wesley Publishing Company, 1994.

BUSCHMANN, Frank; MARTIN, Robert C; RIEHLE, Dirk, et al. **Pattern languages of program design 3**. Reading : Addison-Wesley, 1998.

COAD, Peter; MAYFIELD, Mark. **Projeto de sistemas em Java**: construindo aplicativos e melhores *applets*. São Paulo: Makron Books, 1998.

GAMMA, Erich et al. **Padrões de projeto:** soluções reutilizáveis de software orientado a objetos. Porto Alegre: Bookman, 2000.

GROTT, Marcio Carlos. **Reutilização de soluções com *patterns* e *frameworks* na camada de negócio.** 2003, 122f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

PREE, Wolfgang. **Design patterns for object-oriented software development.** Nova York: Addison-Wesley, 1995.

PRESSMAN, Roger S. **Engenharia de software.** Tradução Jose Carlos Barbosa dos Santos, revisão técnica Jose Carlos Maldonado, Paulo César Masieiro e Rosely Sanches. São Paulo: Makron Books, 1995.

SUN MICROSYSTEM. **Java blueprints guidelines, patterns, and code for end-to-end Java applications**, [S.l],[200?]. Disponível em: <http://java.sun.com/blueprints/*patterns*/index.html>. Acesso em: 10 nov. 2002.

TestCen: Ferramenta de Suporte ao Planejamento de Teste Funcional de Software a partir de Diagramas de Caso de Uso

Juliano Bianchini (FURB)
fjuliano@inf.furb.br

Everaldo Artur Grahl (FURB)
egrahl@furb.br

Universidade Regional de Blumenau – FURB
Departamento de Sistemas e Computação
Grupo de Qualidade de Software
Rua Braz Wanka, 238 – Vila Nova
89035-160 – Blumenau / SC

Resumo. Atualmente existem muitas pesquisas sobre a utilização da UML para a geração de casos de testes funcionais. Este trabalho analisou algumas alternativas existentes na literatura sobre o uso de diagrama de casos de uso, um dos diagramas mais usados na UML, e a partir deste estudo foi criada uma ferramenta de suporte ao planejamento de testes funcionais. Inicialmente foi modificada a ferramenta CASE ArgoUML para incluir extensões apropriadas a casos de testes. Posteriormente foi desenvolvida uma ferramenta para permitir a leitura destas extensões e documentar os testes seguindo orientações previstas no padrão IEEE 829 que trata da documentação de testes de software.

Palavras-chave: Teste de software, Teste funcional, Casos de teste, Casos de uso, IEEE 829.

1 Introdução

O papel do teste é encontrar erros na implementação de um software e validar os requisitos implementados. Isso significa que o teste tem um papel importantíssimo na qualidade do produto de software. Porém, em muitos casos, a preparação e o desenvolvimento dos casos de teste são feitos sem muito planejamento. Além disso, freqüentemente os testes são selecionados de forma aleatória ou *ad-hoc* (momentânea) e os casos de teste são desenvolvidos de uma forma não estruturada e não sistemática. Esta é uma realidade encontrada no desenvolvimento de softwares comerciais, onde há recursos limitados e tempo escasso para o teste.

Apesar de existirem várias técnicas e estratégias aplicadas a teste de software, a prática ainda está distante da teoria (RYSER; GLINZ, 2003, p. 1-2). Alguns problemas encontrados incluem:

a) falta de planejamento dos tempos e custos: na prática, o planejamento dos testes é feito tardiamente no processo, quando ele está quase no final. Desta forma há pressão para manter menor custo e tempo e somado ao fato de que o projeto pode estar atrasado, a preparação e execução do teste são feitas superficialmente, comprometendo a garantia da qualidade do produto de software;

b) falta de documentação: os testes não são devidamente preparados, não há um plano de teste e os testes não são documentados;

c) o teste é a última etapa do processo de desenvolvimento: o desenvolvimento dos casos de teste é feito apenas quando o software está quase pronto. Porém, a fase de testes deveria começar imediatamente após a especificação ter sido desenvolvida.

Desta forma, muitos erros, omissões e inconsistências podem ser encontradas nas fases de análise e projeto. É mais barato corrigir erros nestas fases iniciais do que após a implementação ter sido feita;

d) casos de teste não são gerados de forma sistemática: os testes são escolhidos de forma aleatória e os casos de teste gerados com base no conhecimento do testador e sem procedimento definido.

A Unified Modeling Language (UML) está sendo muito utilizada atualmente para modelar sistemas. A sua utilização para o desenvolvimento dos casos de teste tem sido estudada e evoluída durante os últimos anos. Um diagrama que tem sido estudado para teste é o diagrama de caso de uso, que é uma forma de capturar as funcionalidades e comportamentos de um sistema na perspectiva do usuário. O caso de uso é usado para ajudar na elicitação e documentação dos requisitos dos usuários. Desta forma, o caso de uso é uma fonte de informações que ajuda não só o desenvolvimento do software, mas também, a atividade de teste de software. Entre as propostas de utilização de casos de uso para o teste funcional encontra-se a apresentada por Heumann (2004) e o teste de caso de uso estendido, proposto por Binder (2000, p. 722-731), que são apresentadas neste artigo.

O padrão IEEE 829 tem por objetivo descrever os documentos necessários para apoiar a atividade de teste de software. Os documentos deste padrão descrevem o planejamento, especificação e a geração de relatórios de testes (IEEE, 1998). Este padrão será usado neste artigo como base para documentação dos casos de teste criados utilizando as abordagens mencionadas anteriormente.

Dentro deste contexto o objetivo deste artigo é apresentar a ferramenta TestCen que permite apoiar o planejamento de testes funcionais de software criados a partir dos Diagramas de Casos de Uso utilizando o padrão IEEE 829.

O artigo está organizado da seguinte forma: Na seção 2 faz-se uma descrição da UML e mais especificamente os Diagramas de Casos de Uso no contexto de testes de software. Na seção 3 é apresentada a ferramenta construída. Na seção 4 é descrito um estudo de caso ilustrando o uso da ferramenta. Na última seção 5 são apresentadas as principais conclusões do artigo.

2 UML e Teste de Software

A UML é um mecanismo muito eficiente e prático, que traz muitos benefícios às atividades de análise e projeto de software. Utilizá-la na atividade de testes poderia trazer benefícios ao desenvolvimento de software. Um sistema especificado através de casos de uso provê muitas das informações necessárias para se testar um sistema, pois ele descreve os requisitos funcionais de um software, isto é, pode-se testar e validar os requisitos de software com base nos casos de uso.

Esta seção apresenta os padrões propostos por Binder (2000, p.722-731) e Heumann (2004), para utilização de casos de uso no teste funcional de software, pois estes se apresentaram mais completos e abrangentes para diagramas de caso de uso, além de serem mais aplicáveis ao contexto de testes funcionais. Estes padrões foram utilizados neste artigo com o intuito de simplificar e orientar quem estiver elaborando os casos de teste.

2.1 Geração de casos de teste a partir dos casos de uso

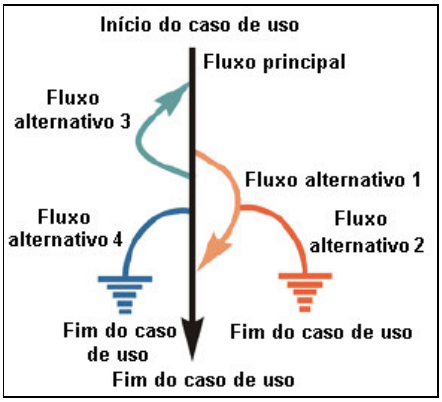
Heumann (2004) apresenta uma metodologia para criar casos de teste a partir dos casos de uso executando três passos:

a) para cada caso de uso, gerar uma lista de cenários[1] de casos de uso;

b) para cada cenário, identificar, ao menos, um caso de teste e as condições que o farão ser executado; e

c) para cada caso de teste, identificar os dados que serão utilizados para o teste.

No primeiro passo, deve-se ler a descrição textual do caso de uso e identificar o fluxo principal e os fluxos alternativos. Por exemplo, um caso de uso para descrever o uso de um sistema para fazer validação de usuários terá um fluxo principal que é: o usuário digita seu nome e senha corretamente e o sistema vai para a tela principal. Um fluxo alternativo seria: o usuário digita seu nome corretamente e uma senha inválida e, então, o sistema emite um aviso e permanece na tela de validação. A figura 1 exemplifica o fluxo principal e os fluxos alternativos para um caso de uso..



Fonte: Heumann (2004)

Figura 1: Fluxo principal e os fluxos alternativos para um caso de uso

No segundo passo, os casos de teste para cada cenário devem ser identificados. Para tanto deve-se analisar os cenários, revisar a descrição do caso de uso e então criar o caso de teste. Criar uma tabela que represente valores de entrada e saída pode ser muito útil neste ponto, ajudando na organização e montagem dos casos de teste.

No terceiro e último passo, deve-se revisar e validar os casos de teste para assegurar a meticulosidade e identificar casos de teste redundante ou faltantes. Depois de aprovados, os dados de teste devem ser identificados. Sem os dados de teste, os casos de teste não podem ser executados. Heumann (2004) afirma que com a metodologia apresentada, desenvolvedores podem simplificar o processo de teste, incrementar a eficiência e ajudar na certeza de se ter uma cobertura completa dos testes.

2.2 Teste de caso de uso estendido

Binder (2000, p. 722-731) apresenta um padrão chamado de Teste de Caso de Uso Estendido (TCUE), que tem por objetivo desenvolver testes de sistema pela modelagem dos requisitos essenciais em forma de casos de uso estendidos. Este padrão procura encontrar falhas como:

a) características de interação (comportamentos da interface com o usuário) indesejáveis;

---

1 Cenário de caso de uso: lista ordenada de interações entre parceiros, usualmente entre um sistema e uma lista de atores externos ao sistema. Pode caracterizar uma seqüência de passos para interação ou uma lista de possíveis interações (RYSER e GLINZ, 2003).

b) saídas incorretas;

c) erro de execução levando o programa a ser terminado;

d) funções (requisitos funcionais) omitidas;

e) funções extras (apareceram sem ser requisitadas), etc.

Primeiramente, os casos de uso estendidos devem ser desenvolvidos, sendo que estes devem conter as seguintes informações:

a) um inventário completo das variáveis operacionais;

b) uma especificação completa do domínio de cada variável operacional;

c) os relacionamentos operacionais para cada caso de uso;

d) a freqüência relativa para cada caso de uso (opcional).

Para cada caso de uso estendido deve-se executar os seguintes passos:

a) identificar as variáveis operacionais: as variáveis operacionais são todas as variáveis que são explicitamente parte da interface que suporta o caso de uso. Entre elas estão entradas e saídas do sistema, condições do ambiente que resultam em diferentes comportamentos dos atores e estados do sistema;

b) identificar os domínios das variáveis operacionais: os domínios são desenvolvidos definindo quais são os valores válidos e inválidos para uma variável;

c) desenvolver os relacionamentos operacionais: nesta etapa é modelado o relacionamento entre as variáveis operacionais que determinam diferentes respostas do sistema. Esta modelagem pode ser feita através de uma tabela de decisão. Quando todas as condições de uma linha da tabela de decisões forem verdadeiras, uma ação esperada é produzida. Cada linha da tabela de decisão é chamada de variante. A tabela deve ser modelada de forma que cada linha seja exclusiva, isto é, para determinada condição do sistema, apenas uma linha da tabela de decisão represente esta condição;

d) desenvolver casos de teste: cada variante da tabela é verdadeira e falsa pelo menos uma vez. Este passo requer dois casos de teste para cada variante: um que represente a variante falsa e outro que represente a variante verdadeira. Os resultados esperados para cada caso de teste são tipicamente desenvolvidos por inspeção, isto é, uma pessoa observa os valores de entrada dos casos de teste e desenvolve os resultados esperados. Preferencialmente, esta pessoa deve ter grande conhecimento do negócio, principalmente do requisito em teste, e das formas como o sistema em teste deve ser usado.

O desenvolvimento dos casos de uso estendidos deve ser feito assim que os casos de uso foram desenvolvidos e validados. Já os casos de teste devem ser desenvolvidos assim que os casos de uso estendidos foram desenvolvidos e validados e o sistema em teste já tenha passado pelo teste de integração que demonstre que os componentes necessários para suportar o caso de uso estejam operacionais.

Além disso, é possível rastrear os casos de teste para cada caso de uso conforme exemplo mostrado na tabela 1:

Tabela 1: Matriz de rastreabilidade entre casos de uso e casos de teste

	Caso de teste	Caso de teste 2	...	Caso de teste
Caso de uso 1	✓			✓
Caso de uso 2		✓		
...				
Caso de uso	✓			✓

Fonte: Binder (2000, p. 729)

Entre as vantagens de utilizar este padrão estão:

a) a utilização dos casos de uso está consolidada nos processos de análise e projeto, isso facilita o desenvolvimento dos casos de teste;

b) casos de uso refletem o ponto de vista do usuário (cliente), sendo que o foco dele está voltado para as funcionalidades que estão implementadas ou não e é isso que determinará se o sistema em teste atende ou não suas necessidades;

c) casos de uso estendidos provêem uma forma sistemática de desenvolvimento das informações necessárias para o projeto de teste. A informação deverá ser desenvolvida para testar qualquer caso de uso;

d) se o sistema em teste for desenvolvido a partir de casos de uso ambíguos, inconsistentes ou incompletos, estes logo serão apontados pelos testes.

Por outro lado, entre as desvantagens pode-se apontar:

a) casos de uso não são usados para especificar, entre outras, performance e tolerâncias a falhas;

b) a UML define relacionamentos do tipo *extends* e *includes* para os casos de uso. Esta metodologia não está preparada para suportar as dependências entre casos de uso.

3 Desenvolvimento da Ferramenta

Com o intuito de suportar as metodologias de teste com base no caso de uso e documentar a etapa de teste de software (seguindo o padrão IEEE 829), foi desenvolvida a ferramenta TestCen que possibilita a documentação do projeto de teste, seus casos de teste e os procedimentos para execução de cada caso de teste. Além disso, cada caso de teste pode ser executado manualmente, seguindo os casos de teste especificados.

Para facilitar e incentivar o planejamento dos casos de teste logo no início do processo de desenvolvimento foi criada uma extensão para a ferramenta CASE ArgoUML (TOLKE, 2004) que possibilita especificar casos de teste para cada caso de uso criado. Os dados de teste criados no ArgoUML podem ser importados na ferramenta TestCen. Optou-se por utilizar a ferramenta ArgoUML por ser software livre e ser muito adotada no meio acadêmico.

As alterações feitas na ferramenta CASE ArgoUML possibilitam especificar os casos de teste que podem ser aplicados para testar e validar os casos de uso. A figura 2 mostra um exemplo de especificação de caso de uso e os casos de teste aplicáveis ao mesmo.

Através do botão criado na barra de ferramentas do diagrama de casos de uso (ver figura 3), é possível criar estereótipos para documentar casos de teste e associar os mesmos ao caso de uso correspondente. A cada estereótipo de teste pode-se especificar o ambiente necessário para execução, procedimentos especiais e os passos (procedimentos) para execução de cada caso de teste. As figuras 4 e 5 mostram as guias para cadastro dos dados dos casos de teste.

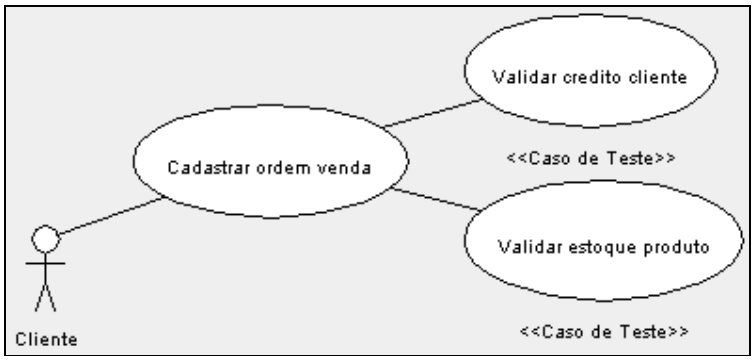


Figura 2: Exemplo de caso de uso e casos de teste aplicáveis ao mesmo

Figura 3: Novo botão na barra de ferramentas

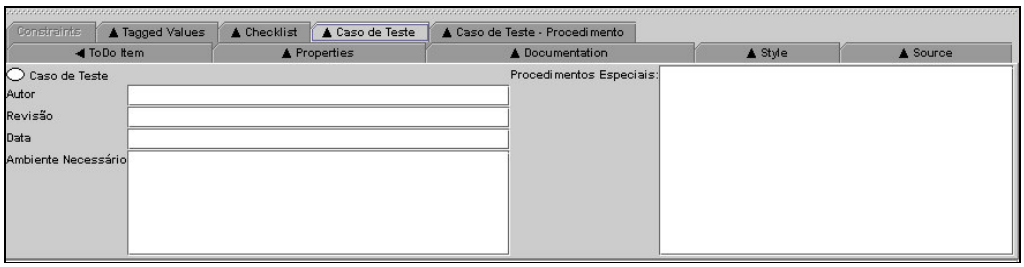


Figura 4: Guia *Caso de Teste*

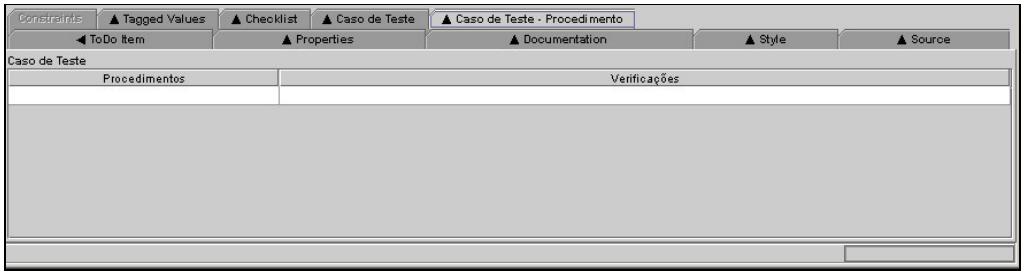


Figura 5: Guia *Caso de Teste – Procedimento*

Os dados especificados nos casos de teste podem ser exportados, para um arquivo XMI, através do menu *Tools* opção *Export as XMI*. Os dados de teste do arquivo podem ser importados

para a ferramenta TestCen utilizando a opção *Importar XMI (ArgoUML)* do menu *Casos de Teste*. Maiores detalhes podem ser obtidos no trabalho de Bianchini (2004).

A seguir é apresentada uma breve descrição de cada caso de uso existente. Os atores envolvidos com a ferramenta são: o analista de testes responsável pela especificação dos casos de teste (o que testar e como testar) e o testador responsável pela parte operacional dos testes (execução). O analista de testes deve ter domínio do negócio do sistema em desenvolvimento.

a) realizar leitura dos diagramas: este caso de uso tem como pré-condição a criação de diagrama de casos de uso e suas extensões de teste na ferramenta CASE ArgoUML. A partir disto o analista de teste pode exportar os dados de teste para um arquivo de interface (arquivo formato XMI) e importá-los na ferramentas TestCen. Os atributos da especificação do caso de teste devem ser baseados no padrão IEEE 829 para especificação do caso de teste e especificação do procedimento de teste;

b) criar projeto e caso de teste: o analista de testes cria um projeto de teste e os casos de teste que compõe o projeto. Cada caso de teste pode estar associado a um caso de uso. Os atributos devem ser baseados no padrão IEEE 829 para especificação do projeto de teste, especificação do caso de teste e especificação do procedimento de teste;

c) executar casos de teste: a partir dos casos de teste planejados, o testador pode escolher um caso, de cada vez, para ser executado. Para cada execução do caso é criado um histórico, com os dados utilizados nos testes e os resultados obtidos, para posterior consulta;

d) gerar relatório de erros encontrados: o testador seleciona um caso de uso e poderá gerar um relatório dos erros encontrados. Os atributos do relatório devem ser baseados no padrão IEEE 829 para relatórios de incidentes;

e) gerar gráfico com casos de teste por caso de uso: o testador poderá selecionar e gerar um gráfico do tipo barra com o número de casos de teste por caso de uso;

f) gerar gráficos de cobertura dos testes: o testador poderá selecionar e gerar um gráfico na tela do tipo pizza com o número de casos de teste executados e não executados em relação ao total de casos de teste criados;

g) gerar resumo da execução do projeto de teste: o testador poderá gerar um relatório com sumário do projeto de teste. Os atributos do relatório devem ser baseados no padrão IEEE 829 para relatórios de resumo dos testes.

4 Estudo de caso

Como estudo de caso, será usado um sistema fictício de controle de uma biblioteca. Neste sistema pode-se cadastrar o acervo, alunos, bibliotecários, bem como controlar os empréstimos e devoluções de livros. Entre os casos de uso obtidos, tem-se o caso de uso empréstimo de livros, conforme mostrado na figura 6. Para este caso de uso, o aluno informa os livros a serem emprestados. A bibliotecária informa para o sistema o código de cada livro e o sistema registra a data e hora do empréstimo, o código do livro e qual a data de devolução do mesmo. Existem alguns detalhes relevantes que devem ser tratados pelo sistema: 1) o aluno tenta emprestar um livro, porém existem livros emprestados cuja data de entrega já venceu. O sistema não deve permitir o empréstimo; 2) o aluno tenta emprestar um livro, mas existem multas não pagas de atraso na entrega de livros. O sistema não deve permitir o empréstimo; 3) o aluno tenta emprestar

um livro, mas o número máximo de empréstimos já excedeu. O sistema não deve permitir o empréstimo.

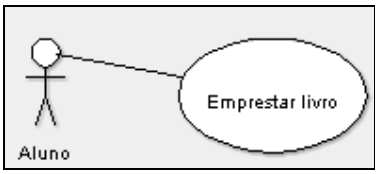


Figura 6: Caso de uso do empréstimo de livros

O desenvolvimento dos casos de teste para o caso de uso consiste em seguir as metodologias de Binder (2000) e Heumann (2004) e, então, documentar os mesmos utilizando as extensões feitas no ArgoUML e/ou na ferramenta TestCen. Seguindo o primeiro passo apresentado por Heumann (2004), foi encontrado, lendo-se a descrição do caso de uso, o fluxo principal e os fluxos alternativos do caso de uso. Além disso, as pré-condições e pós-condições foram identificadas. A tabela 2 mostra os cenários encontrados para o caso de uso em questão.

Tabela 2: Descrição do caso de uso de empréstimo de livros

Caso de uso emprestar livro	
Ator primário	Aluno
Ator secundário	Bibliotecário
Pré-condições	O aluno deve estar cadastrado no sistema de acadêmicos e ter um cadastro e senha na biblioteca.
Fluxo Principal	O aluno informa os livros a serem emprestados. A bibliotecária informa para o sistema o código de cada livro e o sistema registra a data e hora do empréstimo, o código do livro e q ual a data de devolução do mesmo.
Fluxo Alternativo 1	O aluno tenta emprestar um livro, porém existem livros emprestados cuja data de entrega já venceu. O sistema não deve permitir o empréstimo.
Fluxo Alternativo 2	O aluno tenta emprestar um livro, mas existem multas não pagas de atraso na entrega de livros. O sistema não deve permitir o empréstimo.
Fluxo Alternativo 3	O aluno tenta emprestar um livro, mas o número máximo de empréstimos já excedeu. O sistema não deve permitir o empréstimo.
Pós-condições	Para o fluxo principal, o sistema deverá ter registrado com sucesso o empréstimo do livro. Para os fluxos alternativos o sistema deverá emitir mensagens de erro e/ou alerta.

O próximo passo é desenvolver as especificações de teste de cada um dos cenários, seguindo as metodologias propostas por Binder (2000) e Heumann (2004). Para este estudo será demonstrado o desenvolvimento do fluxo principal.

Lendo a descrição do fluxo principal, foram identificadas as variáveis operacionais para este cenário: o código do livro, o código do aluno, a data e hora de empréstimo e a data de devolução. Para auxiliar na criação dos casos de teste, foi criada uma tabela (temporária) contendo as variáveis operacionais e seus valores para teste, conforme mostra a tabela 3.

Tabela 3: Variáveis operacionais e valores para teste

Código do aluno	**Código do livro**	**Data e hora do empréstimo**	**Data da devolução**
001	005	Data e hora do momento do empréstimo	Data do momento de empréstimo + 7 dias

Por último, foram desenvolvidos os procedimentos para execução dos testes. Estes foram desenvolvidos com base nas descrições dos casos de uso e nas variáveis operacionais identificadas. A tabela 4 ilustra a descrição dos procedimentos de teste.

Tabela 4: Procedimentos para execução do teste para fluxo principal

Procedimentos	Verificações
Na tela de validação do aluno, digitar como código do aluno **001** e senha **abc123** (já previamente cadastrados).	O sistema deverá ir para a tela de empréstimo de livros.
Informar como código do livro a ser emprestado **005**.	O sistema deverá trazer as informações do livro: nome autor, descrição, editora e n° da edição.
Confirmar a empréstimo do livro para o aluno **001**.	O sistema deverá mostrar uma mensagem de confirmação do empréstimo, seguido de uma lista de livros emprestados pelo aluno, onde para o livro 005 a data e hora do empréstimo deve ser a data e hora do momento de empréstimo e a data de devolução deve ser a data do momento de empréstimo + 7 dias.

4.1 Documentação dos casos de teste desenvolvidos

O cadastro dos casos de teste pode ser feito na extensão da ferramenta CASE ArgoUML e exportado para a ferramenta TestCen ou pode ser cadastrado diretamente na ferramenta TestCen. Este cadastro feito na ferramenta CASE ArgoUML é útil para o analista de sistemas quando este está especificando os casos de uso. Neste momento, pode ser cadastrado um roteiro de como os testes devem ser feitos. No momento em que existir uma versão funcional do software, as especificações criadas no ArgoUML podem ser exportadas para a ferramenta TestCen e, então, detalhadas a nível de interação com o software em teste.

A figura 7 mostra a especificação dos casos de teste para o caso de uso deste estudo de caso. Nesta figura, cada um dos cenários é especificado, no diagrama de casos de uso, como um caso de teste. Para cada caso de teste são cadastrados os dados. As figuras 8 e 9 mostram os cadastros feitos para o caso de teste desenvolvido neste estudo de caso.

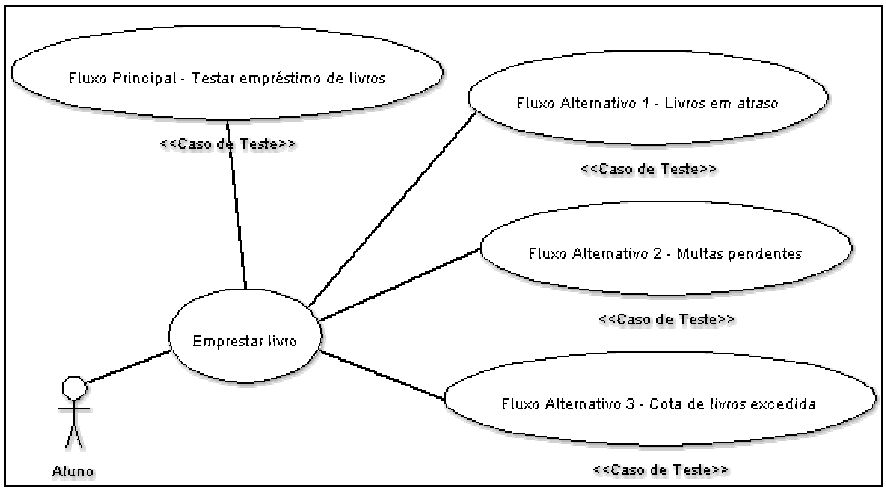


Figura 7: Casos de teste para o caso de uso do empréstimo de livros

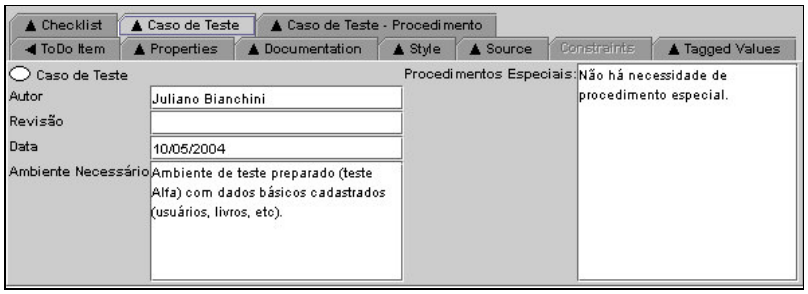


Figura 8: Cadastro do ambiente necessário e dos procedimentos especiais para o caso de teste

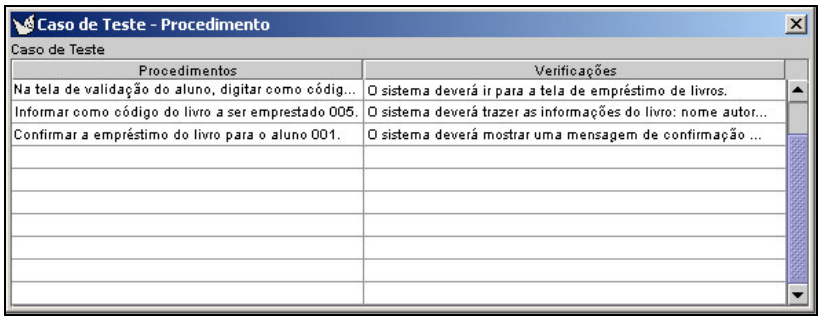


Figura 9: Cadastros dos procedimentos para execução do caso de teste

Como mencionado anteriormente, os cadastros feitos no ArgoUML, podem ser exportados para um arquivo XMI e, a partir deste, importado para a ferramenta TestCen (ver figura 10). Uma vez importados, estes podem ser alterados e executados. Cada execução pode gerar um histórico, que será útil para gerar relatórios de erros por caso de teste e um resumo do projeto de teste, além dos gráficos de caso de teste por caso de uso e do gráfico de casos de teste executados e não executados em relação ao total de casos de teste criados. Estas informações são úteis para saber o estado do projeto de teste e, ainda, ajudam na rastreabilidade dos casos de teste em relação aos casos de uso.

Dentre os documentos especificados pelo padrão IEEE 829 apenas a especificação do plano de teste e o relatório de transição de item de teste não foram implementados. O primeiro não foi coberto por ser um documento de planejamento global dos testes, onde são descritas as atividades, ferramentas e estratégias que serão utilizadas na fase de teste de um software. Este documento é criado juntamente com planejamento de um projeto de software e neste ponto não se tem uma visão detalhada dos requisitos. O segundo não foi coberto por ser um documento de comunicação entre a área de desenvolvimento e a área de testes.

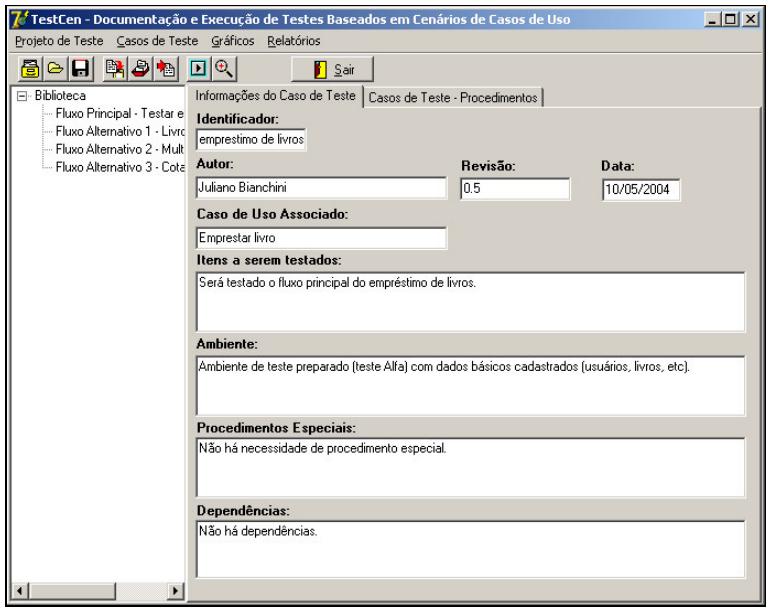


Figura 10: Dados de teste importados do ArgoUML

5 Conclusões

A utilização das ferramentas facilita e organiza todo o processo de teste. Além disso, contribui para a re-execução dos testes, isto é, no momento em que o software é alterado todos os testes executados anteriormente podem ser executados da mesma forma que foram executados nas vezes anteriores.

A utilização dos casos de uso no teste é facilitada a medida que a especificação de requisitos e a especificação dos casos de uso é feita de forma detalhada. Isto é, a elicitação de requisitos e a análise estão intimamente ligadas ao teste e validação de requisitos.

É importante salientar que o trabalho apresentado apenas supre parte do que a atividade de teste deve contemplar. Isso significa que o teste não se limita apenas a testar e validar os requisitos funcionais. Outros testes, como teste de performance, de carga, de unidade e integração, devem ser executados durante a atividade de testes e estes também devem ser feitos de forma sistemática e organizada. A ferramenta também pode ser usada em disciplinas de engenharia de software visando facilitar o entendimento de conceitos sobre testes de software a partir da UML.

Referências

BIANCHINI, Juliano. **Ferramenta de suporte ao planejamento de teste funcional de software a partir de diagramas de casos de uso.** 2004. 75 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

BINDER, Robert V. **Testing object-oriented systems:** models, patterns and tools. Addison-Wesley, 2000.

HEUMANN, Jim. **Generating test cases from use cases**. Disponível em: <http://therationaledge.com/content/jun_01/m_cases_jh.html>. Acesso em: 20 jan. 2004.

INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS. **IEEE Std 829-1998:** IEEE standard for software test documentation. Nova York, 1998.

RYSER, Johannes; GLINZ, Martin. **A practical approach to validating and testing software systems using scenarios**. Disponível em: <http://www.ifi.unizh.ch/groups/req/ftp/papers/QWE99_ScenarioBasedTesting.pdf>. Acesso em: 01 set. 2003.

TOLKE, Linus. **ArgoUML Project**. Disponível em: <http://argouml.tigris.org>. Acesso em: 15 abr. 2004.

SAMOA – Uma Ferramenta para Detecção de Padrões de Projetos em Diagramas UML, na WEB

Edemberg Rocha da Silva (CEFET-PB / UNED - Cajazeiras)
edemberg@cefetpb.edu.br

Ulrich Schiel (UFCG)
ulrich@dsc.ufcg.edu.br

Resumo. O uso de padrões de projeto é considerado uma das mais valiosas técnicas para produzir projetos software com qualidade. Uma forma de melhorar o uso de padrões de projeto é identificar suas realizações e inferir um conhecimento para melhorá-las. Esta tarefa de encontrar todas as realizações de padrões num projeto caracteriza-se por ser tediosa para o engenheiro de software. O presente artigo propõe um sistema assistente para programadores e arquitetos de software para auxilia-los nessa tarefa, chamado SAMOA (**S**istema de **A**poio a **M**odelagem Orientada a **O**bjetos de **A**plicações).

Palavras chaves: padrões de projeto, detecção automática de padrões de projeto, geração de código, geração de críticas, UML**.**

1 Introdução

O projeto de sistemas de software complexos tem-se mostrado uma tarefa difícil, e ferramentas de suporte são necessárias para produzir projetos de qualidade. Para auxiliar esta tarefa, uma idéia importante tem sido utilizar soluções típicas já encontradas para problemas padrão. Estas soluções padrão são conhecidas como *padrões de projeto* (*design patterns*). Estes *padrões* têm sido amplamente aceitos pela comunidade *orientada a objetos* - em particular desde a publicação do livro de GAMMA (GAMMA,1995) - e são considerados um dos mais valiosos mecanismos adotado pelos projetistas de sistemas complexos. Além disso, padrões têm sido reconhecidos como um importante meio de representar as abstrações que um projetista pode usar para favorecer uma melhor compreensão, qualidade e reutilização de um projeto de software complexo.

Uma técnica das mais simples, porém muito poderosa para aprimorar um projeto, é o uso de padrões quando possível, e seguir regras para realizá-los. A aplicação desta técnica para um projeto existente é tediosa, pois requer encontrar todas as realizações desses padrões no projeto. Isto força analisar os diagramas elaborados no projeto para identificar essas possíveis realizações e, em seguida, aplicar regras para aprimorá-las. A automação desta tarefa é um elemento desejável para melhorar o trabalho do arquiteto de software. Este artigo mostra uma visão geral de um assistente automatizado para programadores e arquitetos de softwares, chamado de *SAMOA (**S**istema de **A**poio a **M**odelagem Orientada a **O**bjetos de **A**plicações)*, que se confronta com o problema de automatizar o refatoramento de projetos existentes, que fazem uso de padrões de projeto, e propõe críticas para a melhoria do seu emprego (ROBBINS, 1998).

O restante do artigo está organizado da seguinte forma: na seção 2 serão discutidos assistentes automatizados de produção de software. Na seção 3 será introduzido um meta-modelo adotado para representar padrões de projeto e a seção 4 introduz a arquitetura do SAMOA. Na seção 5 será discutido o framework para detecção de padrões de projeto adotado pelo SAMOA. Na

seção 6, um estudo de caso será exibido. Na seção 7, trabalhos relacionados estarão destacados e concluiremos comentando a importância da ferramenta proposta na seção 8.

2 Assistentes Automatizados

Assistentes automatizados são sistemas de suporte ao engenheiro de software para produzir um software de melhor qualidade. Uma categoria desses assistentes é classificada como *sistemas de críticas* que provêem críticas sobre artefatos existentes para melhorar a sua utilização, baseando-se na analise dos artefatos em desenvolvimento e sugerindo melhorias baseadas em regras predefinidas.

Padrões de projeto podem ser usados para descrever um sistema de software complexo em termos de uma abstração em mais alto nível do que classes, objetos e mensagens, descrevendo situações típicas de componentes de soluções para sistemas de software. Contudo, pouquíssimos assistentes automatizados empregam padrões como abstrações básicas para projeto e engenharia reversa, embora esses padrões derivem de uma concreta experiência. Num projeto, um engenheiro de software pode encontrar uma solução de projeto similar a um padrão sem notar a semelhança. Também, apenas a experiência pode encaminhar o engenheiro ao uso de padrões, e regras aonde utilizar um padrão particular não são, geralmente, triviais. Essas características sugerem que os engenheiros possam se beneficiar de um assistente automatizado capaz de criticar seus projetos em relação ao uso de padrões de projeto.

O SAMOA é um sistema crítico para trabalhar numa interação direta com o arquiteto de software e propor críticas a padrões específicos - subconjunto dos padrões identificados pela GoF – *Gang of Four* (GAMMA,1995). Esta tarefa é realizada através da detecção da realização de padrões, num projeto em construção descrito em UML (BOOCH, 2000), e apresentar um conjunto de críticas visando sua melhoria. O SAMOA poderá receber dois tipos de entradas: a primeira é um diagrama de classes em UML exportado no formato XMI; a segunda, em um processo de engenharia reversa, arquivos gerados a partir de códigos fontes em Java (JAVA). Sobre ambas entradas, será realizado um trabalho de detecção de realização de padrões de projeto. Se as informações necessárias para detectar um padrão podem ser obtidas de um diagrama, chamaremos este padrão de *"detectável"*. Alguns padrões estão fora dessa categoria, pois não estão completamente definidos em termos de classes, objetos e interações, e chamaremos estes de padrões *"não detectáveis"*. Isto é o caso, por exemplo, de padrões como *Façade* ou o *Interpreter* (GAMMA,1995). Estes são caracterizados pelo seu papel num projeto inteiro e não pelas suas estruturas, portanto sua detecção automática requer uma compreensão mais aprofundada do modelo e, atualmente, nenhum sistema capaz de desempenhar esta atividade está disponível. Nesse trabalho nos deteremos aos padrões detectáveis.

Quando a realização de um padrão é detectada, o SAMOA verifica regras a serem aplicadas a determinados padrões, para selecionar um conjunto de criticas visando a melhoria dessas realizações. Essas críticas são direcionadas para melhoria do uso de padrões de projeto, sugerindo:

- nome para classes, atributos e operações: por exemplo, o nome de um *factory method* no padrão *Factory Method* deverá ter o sufixo *Factory*;
- escopo para operações: por exemplo, métodos *hook* no padrão *Template Method* devem ser declarados *protected*;
- operações que são prováveis de serem perigosas para reuso, tais como prover um acesso direto ao *subject* de um objeto *proxy* no padrão *Proxy*;
- técnicas que podem ser usadas para resolver problemas de projetos: por exemplo, o padrão *Iterator* pode ser usado para acessar componentes de um objeto *composite* no padrão *Composite*.

As regras, para realizar críticas sobre as realizações de padrões encontrados, são especificadas em uma linguagem lógica orientada a objetos e formão o conjunto de conhecimento base do SAMOA. A ajuda que este sistema poderá prover ao engenheiro não está limitada ao propósito de regras para estabelecer criticas aos padrões empregados. A detecção automática de padrões, num modelo ou classes *Java*, pode representar uma valiosa ferramenta para verificar a coerência entre o projeto concreto e as intenções do engenheiro. O usuário poderá fornecer qualquer conjunto de classes Java ou diagrama ao SAMOA, para este inferir conhecimento em relação a realização de padrões. A discrepância entre o resultado deste processo e as intenções do engenheiro pode ter as seguintes conseqüências:

- se o engenheiro quiser empregar um padrão mas este não pôde ser detectado, provavelmente o projeto contém algum erro;
- a detecção de um padrão onde o engenheiro não planejou usá-lo, induz a uma melhor compreensão e documentação do projeto.

O sistema também auxilia na instanciação de padrões do GoF, assim como a geração de código, em Java, dos mesmos.

3 Visão Geral do Meta-Modelo

Técnicas baseadas em meta-modelagem consistem em definir um conjunto de meta-entidades para descrever as entidades do modelo. A descrição de um design pattern é obtida pela composição dessas meta-entidades. Esta composição segue regras semânticas, fixadas pelos relacionamentos entre meta-entidades. Deste ponto de vista, meta-modelagem é um meio para formalizar padrões.

Um meta-modelo padrão não captura o que é um padrão no geral, mas como ele é usado em um ou vários casos específicos (AMIOT, 2001), por exemplo, aplicação, validação, representação estrutural etc.. Um meta-modelo padrão nunca “produz” padrões, mas sim modelos de padrões.

O meta-modelo incorpora um conjunto de entidades e regras de interação entre eles. Todas as entidades necessárias para descrever a estrutura e comportamento dos padrões de projeto introduzidos por (GAMMA, 1995) são representadas. A figura 1 mostra o meta-modelo baseado em fragmentos elaborado por (MEIJERS,1997) e adotado nesta pesquisa devido ao fato da facilidade de geração de código, que é um dos focos do SAMOA. Este meta-modelo de padrão é uma instancia da classe *Pattern*. Ela consiste de uma coleção de entidades (instancias de PEntity), representando a noção de participantes como definidas por (GAMMA, 1995). Cada entidade contém uma coleção de elementos (instancias de *PElement*), representando os diferentes relacionamentos entre entidades. Se necessário, novas entidades ou elementos podem ser adicionados pela especialização das classes *PEntity* ou *PElement*.

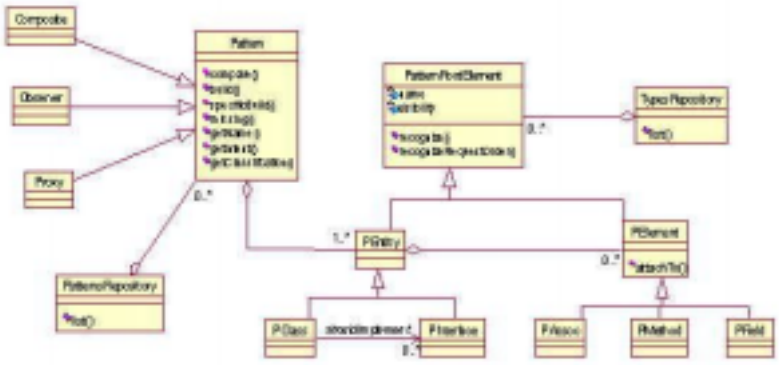


Figura 1: diagrama simplificado de classes do meta-modelo (MEIJERS, 1997)

O meta-modelo define a semântica dos padrões. Um padrão é composto de uma ou mais classes ou interfaces, instancias de *PClass* e *PInterface* (subclasses de *PEntity*). Uma instância de *PEntity* contém métodos e atributos, instâncias de *PMethod* e *PField*. Uma associação (classe *PAssoc*) diz respeito a uma *PEntity* que referencia outra *PEntity*. Por exemplo, uma associação estabelecida entre uma classe *B* e uma classe *A* é definida usando duas instancias da classe *PClass*, *A* e *B*, e uma instância da classe *PAssoc* entre *A* e *B*.

4 Arquitetura do Sistema

O projeto do SAMOA é composto por quatro módulos:

- módulo de entrada – recebe um diagrama de classes UML corrente, editado numa ferramenta CASE, e exportado no formato XMI. Com isso, o SAMOA fica independente de qualquer ferramenta, e o uso de XMI irá prover uma maior interoperabilidade entre o SAMOA e qualquer editor de diagramas UML. Uma vez recuperado o modelo, este será mapeado para uma representação padrão baseada no modelo proposto por (MEIJERS, 1997) que foi implementado por (PATTERNSBOX,2000) (ver seção 5). Alternativamente, em um processo de engenharia reversa, o SAMOA poderá receber como entrada arquivos fonte em Java, também com o intuito de mapea-los para uma representação padrão;
- detecção da realização de padrões de projeto - uma vez que as entradas foram mapeadas para uma representação padrão, inicia-se um processo de detecção de padrões de projeto. Desde que os padrões encontram-se armazenados num repositório, seguindo o modelo proposto por (MEIJERS, 1997), desencadeia-se um processo de fragmentação das entradas do sistema e um processo de introspecção é iniciado. Existe hoje um framework chamado *PatternsBox* (PATTERNSBOX,2000), que implementou todo o meta-modelo de MEIJERS e realiza a detecção de padrões de projeto. Esse framework esta conectado ao SAMOA para auxiliar a realização deste módulo;
- geração de críticas sobre a aplicação dos padrões – uma vez que os padrões foram detectados, inicia-se um processo de inferência de conhecimento sobre tais padrões, visando estabelecer críticas sobre a má utilização dos padrões (ver seção 2) e provendo sugestões para sua melhoria. Um conjunto de críticas, associadas aos padrões de projeto, já estará armazenado numa base de conhecimento lógica, tendo suas regras especificadas na linguagem lógica, afim

de utilizar dos seus recursos para representar informações orientadas a objetos que esta dispõe (KIFER, 1995). Quando um padrão for detectado, este será repassado para a base de conhecimento; esta listará, como saída, algumas críticas ao usuário, sobre o mau uso do respectivo padrão;

- instanciação e geração de código – o SAMOA não só funcionará como um processo de detecção e geração de críticas sobre empregos de padrões de projeto. Os padrões poderão ser instanciados, desde que estejam definidos no repositório de padrões e em seguida ter seu código fonte, em Java, gerado.

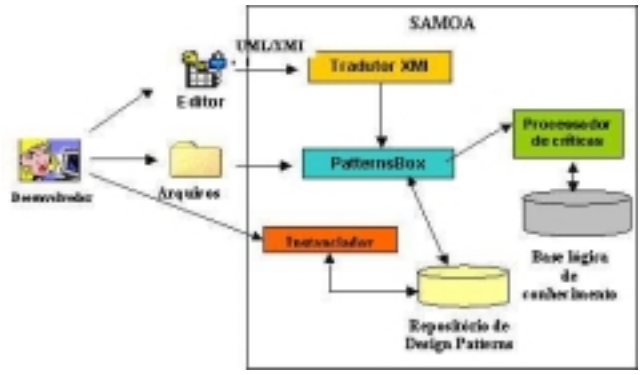


Figura 2: arquitetura geral do SAMOA

Mostramos, na Figura 2 a arquitetura geral do SAMOA. Componentes do sistema:

- Tradutor XMI: extrai as informações contidas no formato XMI, mapeando-os para classes Java e em seguida as repassa para o PatternsBox.
- PatternsBox ver seção 5.
- Processador de críticas: os padrões de projeto detectados pelo PatternsBox são passados ao Processador de críticas. Em seguida, este módulo comunica-se com a base lógica no intuito de inferir críticas aos padrões encontrados. Tal processador enviará consultas em uma linguagem lógica para a base de conhecimento.
- Base lógica de dados: contém um conjunto de possíveis críticas relacionadas a uma má realização de um determinado padrão.
- Repositório de padrões de projeto: contém a definição dos padrões de projeto detectáveis.
- Instanciador: permite que o usuário instancie um padrão de projeto qualquer, desde que este esteja definido no repositório de padrões de projeto, gerando seu código fonte em Java.

O SAMOA foi projetado para funcionar na WEB, podendo dar suporte a ferramentas CASE WEB. Esta última pode construir seus diagramas UML e em seguida exportar para XMI e repassa-lo ao SAMOA. Em resposta, o SAMOA retorna um arquivo XML contendo os padrões detectados.

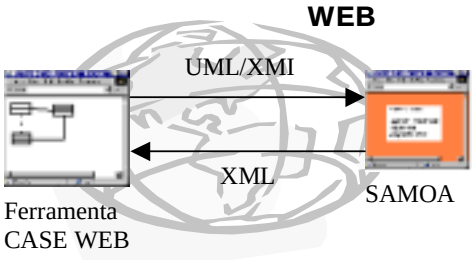


Figura 3: interoperação entre o SAMOA e uma ferramenta CASE WEB

Um protótipo do SAMOA foi desenvolvido e testado em uma rede local. Diagramas UML editados nas ferramentas CASE, Rational® Rose 2000 e Poseidon® 2003, foram utilizados para testar a interoperação delas com o SAMOA.

5 PatternsBox

Desenvolvido por Hervé Albin-Almiot da École des Mines de Nantes, França um framework que implementou o meta-modelo proposto por Meijers (MEIJERS, 1997), visando a detecção de padrões de projeto (PATTERNSBOX,2000).

Seu sistema de detecção é projetado para analisar arquivos fonte Java.

A detecção é baseada principalmente na informação estrutural de um padrão de projeto (seu template). Em seguida utiliza um repositório que possui todos os constituintes de um padrão (elementos e entidades). Esse repositório, instância da classe *TypesRepository* (figura 1), contém todos os *PEntities* e *PElements* definidos por Gamma et.al. (GAMMA,1995).

A detecção é decomposta em dois passos:

(a) a classe *PatternIntrospector*, encarregada de realizar a detecção, submete todos os elementos sintáticos (classes, interfaces, métodos...) encontrados no código fonte do usuário e os armazena no *TypesRepository*.

O *PatternIntrospector* constrói um modelo concreto que representa o código do usuário contendo apenas constituintes definidos no meta-modelo. Finalmente, ele solicita cada modelo abstrato encontrado no *PatternsRepository* para determinar quais padrões foram detectados.

(b) Cada modelo abstrato é examinado e a determinação de quais entidades (instancias de *PEntity*) do modelo submetido podem ser associadas a diferentes papéis. Devem ser consideradas heranças e associações, e o modelo deve conter todas as entidades descritas no modelo abstrato.

6 Problema Exemplo

Para elucidar um dos funcionamentos do SAMOA, realizaremos um estudo de caso que trata da detecção da utilização do padrão *Composite*, num diagrama de classe UML. Seja o seguinte diagrama de classe, editado numa ferramenta CASE (o Poseidon® 2003):

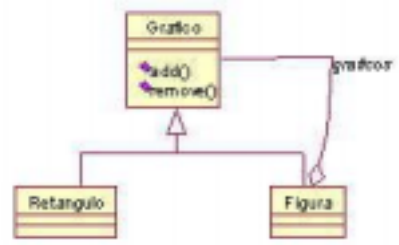


Figura 4: realização do padrão Composite

Agora suponha que o projetista deseja saber se ele esta fazendo uso correto do padrão Composite. Ele exportará o modelo no formato XMI para entrada no SAMOA. Este realizará a detecção de padrões de projeto no modelo fornecido e efetuará um conjunto de criticas, caso haja, sobre ele. O sistema, então, irá fornecer a seguinte saída:

Padrões detectados		Criticas listadas
Composite		
Gráfico	Component	1. Não colocar a referencia para o *filho* em Gráfico. 2. Considerar inserir o método getGraficos em Figura. 3. Considerar o uso do padrão Iterator para acessar os objetos "Gráfico" em Figura. 4. Em Gráfico deverão ser definidas tantas operações para Retângulo e Figura quanto possível.
Retângulo	Leaf	
Figura	Composite	

Figura 5: algumas criticas selecionadas sobre o modelo da fig. 4

A saída fornecida e ilustrada na figura 5, é a mesma tanto para as entradas no formato UML/XMI (figura 6) quanto para os arquivos os fontes *Java*.

```
<?xml version = '1.0' encoding = 'UTF-8' ?>
<XMI xmi.version = '1.2' xmlns:UML = 'org.omg.xmi.namespace.UML' timestamp = 'Wed
Nov 12 22:15:42 GMT-03:00 2003'>
    ....
   <UML:Class xmi.id = 'lsm:161b9b2:f8d2a07169:-7ff9' name = 'Grafico' visibility = 'public'
isSpecification = 'false' isRoot = 'false' isAbstract = 'true' isActive = 'false'>
        <UML:Classifier.feature>
          <UML:Operation xmi.id = 'lsm:161b9b2:f8d2a07169:-7ff8' name = 'add'
            visibility = 'public' isSpecification = 'false' ownerScope = 'instance'
            isQuery = 'false' concurrency = 'sequential' isRoot = 'false' isLeaf = 'false'
            isAbstract = 'true'>
     ....
       </UML:Class>
       <UML:Class xmi.id = 'lsm:161b9b2:f8d2a07169:-7ff0' name = 'Retangulo' visibility =
'public'  isSpecification = 'false' isRoot = 'false' isLeaf = 'false' isAbstract = 'false'
        isActive = 'false'>
         ...
       </UML:Class>
    ...
  </XMI>
```

Figura 6: simplificado XMI gerado pelo Poseidon® 2003 a partir do diagrama da fig. 4

Note que, no caso de um processo de reengenharia, fica difícil para o engenheiro inferir conhecimento de um conjunto de classes em *fontes Java*. Ele não tem uma visão estrutural, mas sim um conjunto de caracteres Unicode armazenados em arquivos. Então, se ele desejar detectar a existência de padrões neles (a fim de obter um maior nível de conhecimento sobre os mesmos), será uma tarefa árdua e, principalmente, se o engenheiro quiser saber se existem algumas possíveis melhorias na utilização desses padrões. Neste caso, o SAMOA torna-se bastante útil para efeitos de documentação dos fontes existentes.

7 Trabalhos Relacionados

Pouquíssimos trabalhos têm sido realizados no campo da detecção automática de padrões, no tocante a diagramas de classe. A maioria deles trabalham com processos de reengenharia, buscando detectar padrões em códigos fonte.

Keller et.al. (KELLER, 1999) descrevem uma analise estática para descobrir padrões de sistemas escritos em C++. O sistema PAT - Program Analysis Tool - (PRECHELT, 1998) detecta padrões estruturais pela extração de informações de projeto nos cabeçalhos de arquivos C++ e os armazena em fatos Prolog. Os padrões são armazenados como regras e a pesquisa é feita pela execução de consultas Prolog.

Já no trabalho de Brown (BROWN, 1996) a detecção de padrões está restrita a sistemas escritos em Smalltalk. Um meta-modelo para representar padrões de projeto num modelo OMT foi proposto por Meijers (MEIJERS, 1997). Já Albin-Almiot (PATTERNSBOX,2000) realizou sua implementação, em Java, deste modelo.

8 Conclusão

O uso de padrões de projeto é considerado fundamental para a produção de projetos de qualidade. Uma das técnicas para melhorar um projeto existente é pesquisar por todas as realizações de padrões, para a aplicação de regras para melhorá-las. Tal técnica visa encontrar todas as realizações de padrões de projeto empregadas num projeto, sendo esta tarefa tediosa para o engenheiro. Esta pesquisa trata de uma ferramenta de assistência automatizada, para programadores e arquitetos de software e que introduz o SAMOA para uma orientação no uso correto de padrões de projeto. Esse sistema procura prover críticas sobre modelos UML, especificamente diagramas de classe ou arquivos fontes *Java*, sobre a realização dos padrões encontrados.

O coração do SAMOA é o módulo que automaticamente detecta a realização de padrões. O processo de realização de críticas inicia-se por verificar um conjunto de regras para testar se essa realização pode ser melhorada. Atualmente, o SAMOA esta sendo projetado para reconhecer um subconjunto dos padrões do GoF (GAMMA, 1995).

Referências

AMIOT, H. A. and GUÉHÉNEUC, Y.G. **Meta-modeling Design Patterns: application to pattern detection and code synthesis**. ECOOP'2001.

BOOCH, G., Rumbaugh, J., Jacobson, I. **UML – Guia do Usuário**. Editora Campus, 2000.

BROWN, K. **Design reverse engineering and automated design pattern detection in Smalltalk**. Relatório técnico TR-96-07, University of Illinois at Urbana-Champaign, 1996.

GAMMA, E. , HELM, R., JOHNSON, R. e VLISSIDES, J. **Design Patterns: Elements of Reusable Object-Oriented Software**. Addison-Wesley, 1995.

JAVA. Linguagem de Programação Orientada a Objetos disponível em http://java.sun.com.

KELLER, R. K., SCHAUER, R., ROBITAILLE, S. and PAGE, P. **Pattern-Based Reverse-Engineering of Design Components**, ISCE, pág. 226-235, 1999.

MEIJERS, M. **Tool Support for Object-Oriented Design Patterns**. Tese de Doutorado, Universidade de Utrecht, 1997.

PATTERNSBOX. Framework disponível em http:www.emn.fr/albin.

PRECHELT, L. and KRÄMER, C. **Functionality versus Practicality: Employing Existing Tools for Recovering Strutural Design Pattern**, J.UCS, 1998.

ROBBINS, J. E. **Design Critiquing Systems**. Relatório técnico UCI-98-41, University of California, 1998.

Rastreabilidade de requisitos através da *web*

Fernando dos Santos (FURB)
fds@inf.furb.br

Karly Schubert Vargas (FURB)
karly@inf.furb.br

Christian Rogério Câmara de Abreu (FURB)
crca@inf.furb.br

Resumo. O presente artigo apresenta uma ferramenta para o gerenciamento de requisitos via web, cuja ênfase está no uso de matrizes de rastreabilidade como forma de implementar ligações entre requisitos de software. A através destas ligações é possível apresentar ao analista as conseqüências de uma alteração ou exclusão de requisitos no projeto.

Palavras-chave: Gerenciamento de Requisitos, Rastreabilidade.

1 Introdução

"O software de computadores tornou-se uma força motora. É o motor que dirige a tomada de decisão nos negócios. Serve de base à moderna investigação científica e às soluções de problemas de engenharia. É um fator chave que diferencia os produtos e serviços modernos. Está embutido em sistemas de todas as naturezas: de transportes, médicos, de telecomunicações, militares, de processos industriais, de produtos de escritório,... a lista é quase sem fim" (Pressman, 2002, p.3). A tendência é um crescimento ainda maior, pois a busca de melhorias no desenvolvimento de produtos e serviços vem exigindo um grau cada vez maior de automação. Devido a isto o desenvolvimento de software tem se tornado mais complexo por atingir áreas que demandam de um conhecimento mais aprofundado.

Com este aumento da complexidade dos projetos de software, tornou-se imprescindível um completo acompanhamento e planejamento de todas as etapas do projeto, como forma de garantir que o software atenda as necessidades levantadas.

Dentro das diversas áreas da engenharia de software, destaca-se a área de engenharia de requisitos que Sommerville (2003) define como o processo de descobrir, analisar, documentar e verificar as funções e restrições de um sistema. Esta etapa é de fundamental importância, pois faz o levantamento dos objetivos do projeto, descrevendo quais as funções que o software deverá possuir, bem como as restrições de cada função. Caso ocorra algum problema nesta etapa, falhas irão ocorrer durante o desenvolvimento do software.

Mas existe uma dificuldade para este acompanhamento quando as equipes de desenvolvimento e os analistas possuem uma distância geográfica que os separa. Pois uma alteração tem que ser vista por todos os envolvidos no momento em que ocorre para que seja mantida a consistência das informações. Para ajudar nisso, surge à internet como forma de interligar todas as pessoas envolvidas no projeto do software.

Este artigo tem por objetivo apresentar uma ferramenta para o gerenciamento de requisitos com o diferencial de que sua utilização é feita via web, possibilitando que equipes alocadas em locais diferentes possam ter sempre a sua disposição, informações atualizadas. Além disto, a ferramenta oferece flexibilidade para a confecção das matrizes de rastreabilidade, e permite a visualização do impacto das alterações de determinados requisitos sobre os requisitos do projeto.

2 Requisitos de Software

2.1 Requisitos

Segundo Peters (2001) um requisito de software é uma descrição dos principais recursos de um produto de software, seu fluxo de informações, comportamento e atributos. Em suma, um requisito de software fornece uma estrutura básica para o desenvolvimento do produto de software.

Os requisitos mostram as características que o cliente deseja no produto final,. Definem os critérios de aceitação de um produto, e são captados na fase de elicitação do projeto, onde são levantadas todas as necessidades do sistema. Seu levantamento deve ser feito de forma bastante cuidadosa, consultando a todas as pessoas que estarão envolvidas no uso do produto final. Com isso é possível evitar vários problemas posteriores ao desenvolvimento do software.

Porém, mesmo que seu levantamento seja feito de forma cuidadosa, um problema comum é a instabilidade dos requisitos, uma vez que os mesmos sofrem alterações no decorrer do desenvolvimento. Aqui entra a importância de haver uma ferramenta que permita uma manipulação destes requisitos, podendo haver um controle de quais são os requisitos atuais, quem é responsável pelo requisito e quais os seus atributos.

Os requisitos podem ser agrupados em três principais tipos, sendo que sua nomenclatura varia de acordo com o autor (Peters, 2001):

a) Funcional (ações principais): Descreve as atividades do sistema, o que ele deve fazer.
b) Comportamental (atividades de controle): Descreve a hierarquia das funções e atividades do sistema, faz o controle do sistema.
c) Não-Comportamental (atributos): Descreve as regras de funcionamento e condições necessárias para a execução de algumas funções. Trata também da garantia da qualidade.

Independente de sua classificação, um requisito possui atributos que definem as características do requisito. Os atributos especificam a prioridade, o status do requisito e seu autor, entre outras informações.

Mas não basta somente possuir a descrição dos requisitos, é preciso criar ligações entre eles. Estas ligações permitem a análise do grau de influência que uma alteração pode causar no restante do projeto. Aqui entra a rastreabilidade dos requisitos.

2.2 Rastreabilidade

“Existem muitas relações entre requisitos e outros requisitos entre os requisitos e o projeto do sistema. Há também elos entre os requisitos e as razões básicas da proposição desses requisitos”. (Sommerville, 2003, p. 120). Uma parte crítica do gerenciamento de alterações é a avaliação do impacto da mudança no restante do sistema. Se a mudança é proposta enquanto os requisitos estão sendo desenvolvidos, deve ser identificado como a alteração afeta outros requisitos. Se a alteração é proposta enquanto o sistema está em implementação, o impacto de alteração envolve verificar como a alteração afeta os requisitos, o *design* do sistema e sua implementação. Se a alteração é proposta depois que o sistema foi colocado em operação, deve haver também uma verificação adicional a fim de identificar como todos os *stakeholders* do sistema podem ser afetados pela alteração.

Marquioni (2004) define alguns tipos de rastreabilidade, mostrados na tabela 1.

Tipo de rastreabilidade	Descrição
Requisitos – Fontes	Link do requisito às pessoas ou documentos que especificaram o requisito.
Requisitos – Razão	Link do requisito com uma descrição de "porque" foi especificado. Pode ser uma destilação de informações de várias fontes.
Requisitos – Requisitos	Link do requisito com outros requisitos que sejam, de alguma maneira, dependentes dele. Deve ser um link de "mão dupla" ("depende" e "é dependente de").
Requisitos – Arquitetura	Link do requisito com os subsistemas onde estes requisitos estão implementados. Particularmente importante se os subsistemas estiverem sendo desenvolvidos por subcontratados diferentes.
Requisitos – *Design*	Link do requisito com hardware ou componentes de software específicos no sistema que são usados para implementar o requisito.
Requisitos – Interface	Link do requisito com as interfaces de sistemas externos que serão usados na provisão dos requisitos.

Tabela 1: Tipos de rastreabilidade

Para facilitar este rastreamento de requisitos, freqüentemente são utilizadas tabelas de rastreabilidade, que criam associações entre os requisitos. Uma forma de visualização gráfica desta rastreabilidade é uma matriz de rastreabilidade, que mostra a ligação entre os requisitos, onde a linha é dependente da coluna e a coluna depende da linha. Está matriz demonstra de que forma um requisito influencia em outro, possibilitando uma análise do impacto de uma alteração do requisito.

Na ferramenta desenvolvida, foi implementado a rastreabilidade do tipo Requisito-Requisito, e utilizada como forma de visualização a matriz de rastreabilidade, com a utilização de flechas indicando a direção da dependência.

A tabela 2 apresenta um exemplo de uma matriz de rastreabilidade retirada do artigo de Marquioni (2004).

	R1	R2	R3	R4	R5	R6
R1			**X**	**X**		
R2					**X**	**X**
R3				**X**	**X**	
R4		**X**				
R5						**X**
R6						

Tabela 2: Matriz de rastreabilidade

No exemplo, R1 é dependente de R3 e R4; R2 é dependente de R5 e R6, etc. Se for proposta uma alteração no requisito R4, a leitura da coluna R4 aponta que R1 e R3 dependem de R4. Deve ser avaliado com isso o impacto em R1 e R3 com relação à alteração proposta a R4.

O volume de dados que serão controlados pela matriz é significativo, por isso este tipo de controle não é recomendado para um projeto muito grande, nestes casos deve optar-se por outra forma de visualização, como uma lista de rastreabilidade (Marquioni, 2004).

3 Desenvolvimento da Ferramenta

A ferramenta desenvolvida, denominada Requisite Online, propõe-se a auxiliar analistas de sistema no gerenciamento de requisitos de projetos de software.

Para isto, a ferramenta permite o cadastramento de projetos de software. Em cada projeto, devem ser cadastrados os requisitos do mesmo. Estes requisitos são agrupados de acordo com seus tipos (por exemplo: requisitos funcionais, requisitos não funcionais, ...), tipos estes que também são cadastrados pelo analista. Também é possível montar diversas matrizes de rastreabilidade para um mesmo projeto, sendo estas matrizes definidas em função dos tipos de requisitos. A ferramenta também disponibiliza ao analista um relatório com as informações cadastradas no projeto (requisitos e matrizes).

Na figura 1 é apresentado o diagrama de casos de uso com a representação das funcionalidades do sistema, e na seqüência, um descritivo das mesmas.

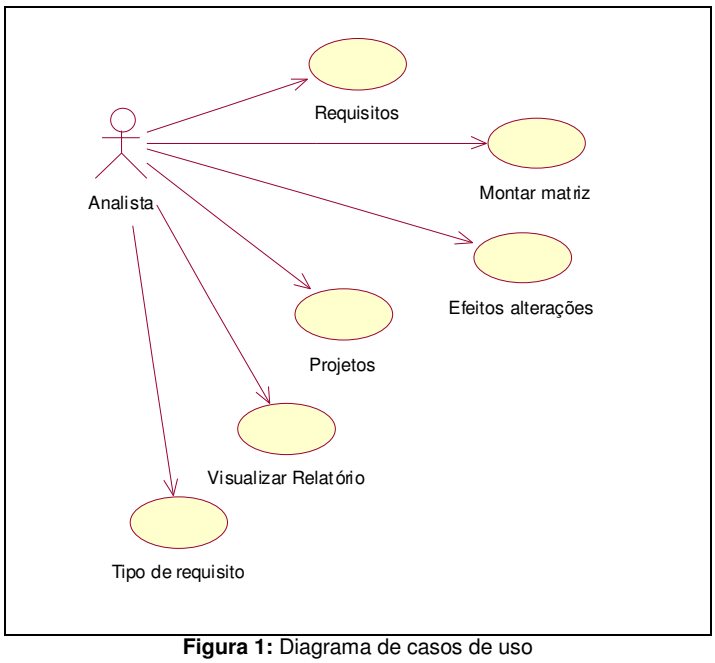


Figura 1: Diagrama de casos de uso

Projetos: Este é um caso de uso secundário e trata do cadastro de um novo projeto.

Tipo de requisito: Este é um caso de uso secundário e trata do cadastro dos tipos de requisitos (funcionais, não funcionais, regras de negócio, etc.). O cadastro de tipos de requisitos pode ser feito de duas formas: a primeira é o cadastro de um tipo geral, que será utilizado em todos os projetos; a segunda forma é o cadastro de um tipo de requisito específico para um projeto.

Requisitos: Este é um caso de uso secundário, e só ocorre após o caso de uso Projetos. Trata do cadastro dos requisitos do projeto de software e seus atributos.

Montar matriz: Para este caso acontecer é necessário que tenham ocorrido os seguintes casos de uso: Projetos, Requisitos, Tipo de requisito. Trata da montagem da matriz de rastreabilidade,

feita com base nos requisitos cadastrados. O analista define quais tipos de requisitos irão compor a rastreabilidade, em seguida define a dependência entre os requisitos. A matriz pode ser visualizada após sua confecção. O analista pode montar quantas matrizes achar necessário.

Efeitos alterações: Este caso só acontece após a ocorrência do caso de uso Montar matriz. Com a matriz de rastreabilidade montada, o analista pode simular alterações nos requisitos e verificar suas conseqüências.

Visualizar Relatório: Este caso só ocorre após o caso de uso Projetos. Neste caso o analista pode visualizar e imprimir as informações sobre o projeto, sendo elas: nome, requisitos e matriz(es) de rastreabilidade do projeto.

3.1 Ferramentas Utilizadas

Para o desenvolvimento da ferramenta, foi utilizada a linguagem de programação PHP em conjunto com JavaScript. Para persistência dos dados, utilizou-se o sistema gerenciador de banco de dados MySQL.

3.1.1 PHP

Soares (2001) afirma que PHP é uma combinação de linguagem de programação e servidor de aplicações. Você pode programar em PHP como em qualquer outra linguagem, definindo variáveis, criando funções, realizando loops, enfim tudo o que é necessário no mundo da programação. Mas o que realmente difere PHP das outras linguagens de programação é a sua capacidade de interagir com o mundo WEB, transformando páginas estáticas em verdadeiras fontes de informação.

As características que foram observadas para a escolha da linguagem de programação são apresentadas por Soares (2001), dentre as quais cita-se que o PHP roda no servidor, é portável, possui código nativo para muitos bancos de dados, entre eles o MySQL e pode ser embutido no HTML.

3.1.2 MySQL

Conforme Soares (2001), o MySQL é um gerenciador de banco de dados mais utilizado no mundo Linux, senão for o mais utilizado, pois é uma ferramenta muito poderosa, segura e fácil de utilizar. Além disto, o MySQL é gratuito.[...] Uma das vantagens do MySQL é a sua disponibilização em várias plataformas.

Os fatores determinantes na escolha do MySQL para persistência dos dados foram a sua gratuidade e facilidade de uso. A gratuidade garante que nenhum custo precise ser criado ou mantido para utilização do sistema, e a facilidade de uso permitiu que o desenvolvimento se desse de forma consistente e rápida.

3.1.3 JavaScript

De acordo com Feather et al (1997), JavaScript é uma linguagem de script (ou roteiro) orientada a objetos, usada para desenvolver aplicações clientes e para Intranets/Internet. Recebe comandos de uma página HTML e, em resposta, pode executar uma ação diferente.

No desenvolvimento desta ferramenta, JavaScript foi utilizado para validações em campos de entrada de dados e também para exibição de diálogos de confirmação, quando necessário.

3.2 Implementação

Na figura 2, temos o diagrama entidade-relacionamento das entidades desenvolvidas para a persistência dos dados e implementação da ferramenta.

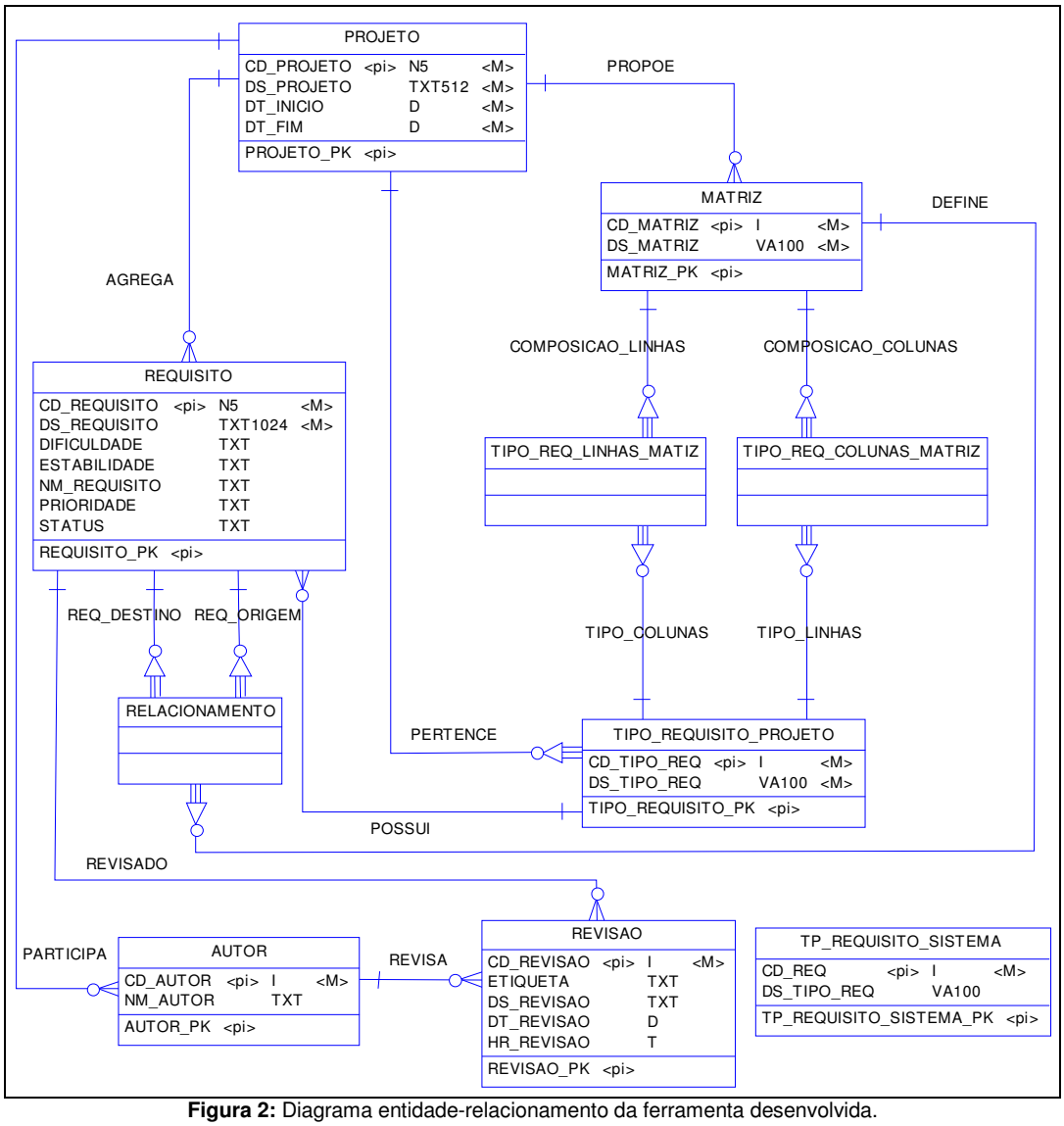


Figura 2: Diagrama entidade-relacionamento da ferramenta desenvolvida.

Para cada projeto cadastrado, é armazenado na entidade PROJETO, seu nome, sua data de inicio, e sua data de conclusão.

Um projeto agrega vários requisitos, e estes são mantidos na entidade REQUISITO. Cada requisito é composto dos seguintes atributos: nome, autor (do requisito e revisões), prioridade (nível de importância do requisito), status (estado em que o requisito se apresenta), dificuldade (dificuldade no desenvolvimento do requisito), estabilidade (estabilidade que apresenta o requisito em relação ao projeto), descrição e revisões (alterações efetuadas no requisito, se houverem).

Cada requisito cadastrado é agrupado em um determinado tipo de requisito. Um projeto pode apresentar vários tipos de requisitos, e estes são armazenados na entidade TIPO_REQUISITO_PROJETO. O analista tem ainda a possibilidade de cadastrar no sistema os tipos de requisitos que o mesmo considera "padrão" para todo novo projeto (criando tipos de requisitos globais). Estes tipos são salvos na entidade TP_REQUISITO_SISTEMA, e a cada novo projeto criado o sistema realiza a inserção automática destes tipos de requisitos no novo projeto.

Para um projeto, o analista pode propor/montar diversas matrizes de rastreabilidade. Para cada nova matriz é criada uma entrada na entidade MATRIZ. Estas matrizes são montadas em função dos tipos de requisitos que são selecionados para suas linhas e para suas colunas (por exemplo, requisitos funcionais *versus* requisitos não funcionais). Estes tipos, são armazenados nas entidades TIPO_REQ_LINHAS_MATRIZ e TIPO_REQ_COLUNAS_MATRIZ, respectivamente. A implementação de matrizes de rastreabilidade desta maneira, além de permitir grande flexibilidade ao analista possibilitando a construção de diversas matrizes de rastreabilidade, elimina a necessidade de que cada novo requisito seja adicionado manualmente na(s) matriz(es), pois a mesma relaciona tipos de requisitos, e não simplesmente requisitos.

4 O Requisite Online

A ferramenta desenvolvida é acessada através de um browser, sendo necessário que o usuário esteja conectado a internet durante o uso da ferramenta. O usuário, depois de conectado a ferramenta, pode navegar pelo browser e executar todas as funções do sistema. Após realizar todas as suas tarefas, o usuário pode desconectar-se da ferramenta apenas fechando o browser. O Requisite Online está disponível na internet, no seguinte endereço: http://campeche.inf.furb.br/phps/crca/requisitos/conteudo.

4.1 Começando seu uso

A figura 3 mostra a tela inicial, que é vista pelo usuário assim que o mesmo abre a ferramenta.

Figura 3: Tela inicial

Nesta tela, o analista cria um novo projeto e define tipos de requisitos globais para os projetos. Os tipos de requisitos globais serão inseridos automaticamente em todo novo projeto. A

grande vantagem do uso destes tipos de requisitos globais, é que isto torna possível padronizar os tipos de requisitos de todos os novos projetos desenvolvidos.

Para iniciar um novo projeto o analista deve selecionar “Novo Projeto”, informando um nome e a data de inicialização do mesmo, após este passo o projeto será criado na área de “Projetos Cadastrados”, podendo ser iniciada sua manipulação.

4.2 Principais Funcionalidades

Nas seções a seguir, serão apresentadas as principais funcionalidades da ferramenta e de que forma elas podem ser utilizadas.

4.2.1 Cadastro de requisitos

O primeiro passo de um projeto é o cadastro dos requisitos. Estes requisitos devem representar as necessidades do software que esta sendo gerenciado. Cada requisito possui um tipo específico, tendo o analista a possibilidade de cadastrar os tipos que considerar necessário, não sendo limitado aos tipos apresentados por Peters (2001). A figura 4 Apresenta a tela para o cadastro do requisito. É solicitado um nome para o mesmo, o autor do requisito, a prioridade, o status do desenvolvimento, o grau de dificuldade, a estabilidade, e uma descrição completa do requisito. A tela é a mesma para todo novo requisito, independente de tipo.

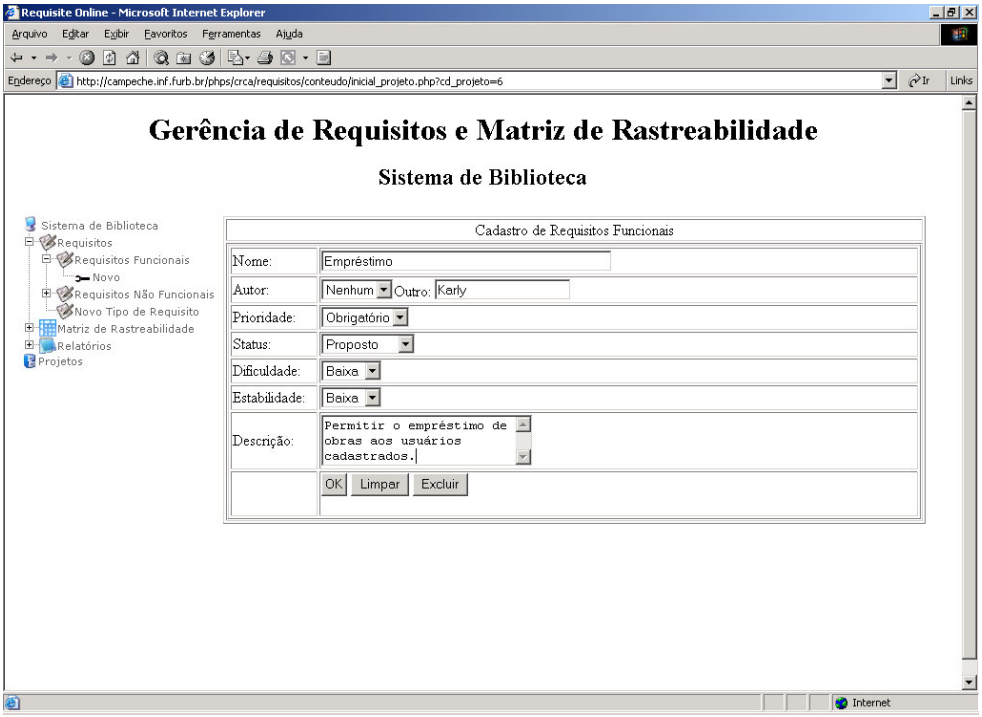


Figura 4: Cadastro de requisito

4.2.2 Matriz de Rastreabilidade

A forma escolhida para gerenciar os requisitos nesta ferramenta foi o uso de matriz de rastreabilidade, onde o analista cria ligações entre os requisitos. O tipo de rastreamento utilizado é

o de requisitos-requisitos, ou seja, vínculo do requisito com outros requisitos que sejam, de alguma forma dependentes dele (Marquioni, 2004).

O sistema gera uma matriz de rastreabilidade simples, que permite registrar as dependências entre os requisitos. Na primeira linha e coluna desta matriz são dispostos os nomes dos requisitos. A dependência entre os requisitos é registrada com um clique na célula de intersecção dos requisitos. Para visualização das dependências, são utilizadas setas que indicam qual(is) o(s) requisito(s) que dependem de um requisito e quais o(s) requisito(s) que são dependentes deste requisito. A leitura da dependência deve ser feita em função da direção da seta. Por exemplo, se a seta parte do requisito A e aponta para o requisito B, lê-se que o requisito B é dependente do requisito A. Devido a visualização da rastreabilidade ser feita com matriz de rastreabilidade, deve-se somente gerenciar projetos com um número pequeno de requisitos, recomendável no máximo 250. (Marquioni, 2004).

A figura 5A mostra o processo para a criação de uma matriz. É necessário um nome e quais tipos de requisitos serão rastreados. Os requisitos dos tipos escolhidos são automaticamente incluídos na matriz de rastreabilidade.

A figura 5B mostra a matriz que o analista pode criar, com o relacionamento entre os requisitos. Depois de montada, é possível realizar simulações com a matriz para verificar o efeito de alteração de requisitos.

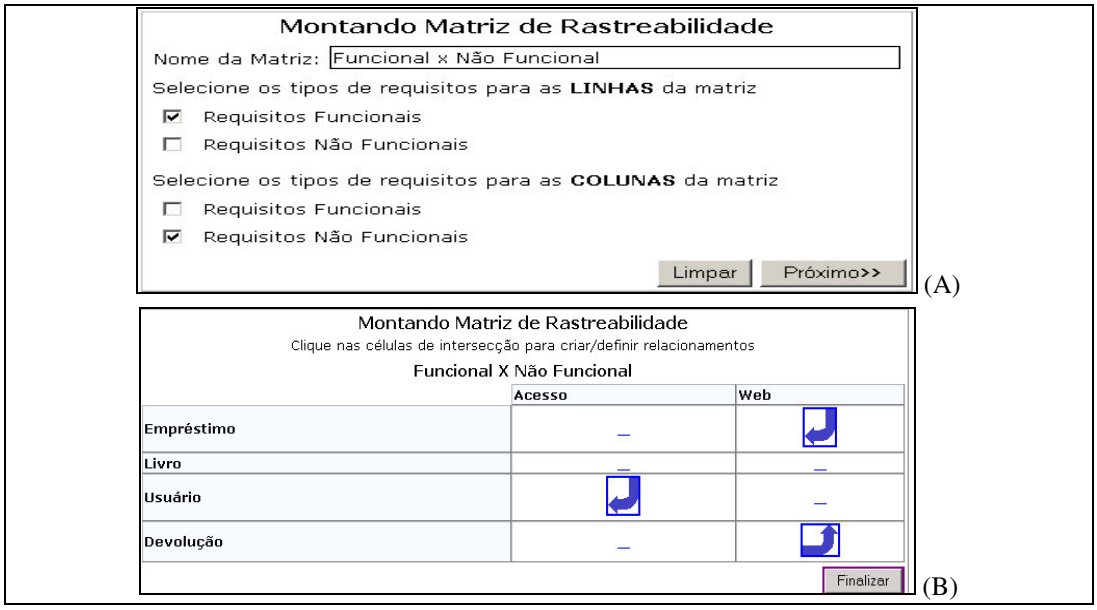


Figura 5: Montando Matriz de rastreabilidade

A figura 6 mostra a utilização prática da matriz de rastreabilidade. Trata-se da possibilidade de verificar os efeitos de alterações em requisito(s). Para isto, o analista seleciona, a partir da lista de requisitos que compõem uma matriz, quais destes requisitos serão alterados (figura 6A). Em seguida, o sistema apresenta a visualização, na forma de matriz, dos impactos (em outros requisitos) que a alteração irá causar. Estes impactos são indicados com cores, onde a coluna/linha do requisito alterado é colorida de amarelo, e a coluna/linha do requisito dependente do alterado é colorida de vermelho. A célula de intersecção entre os requisitos é colorida de laranja (figura 6B).

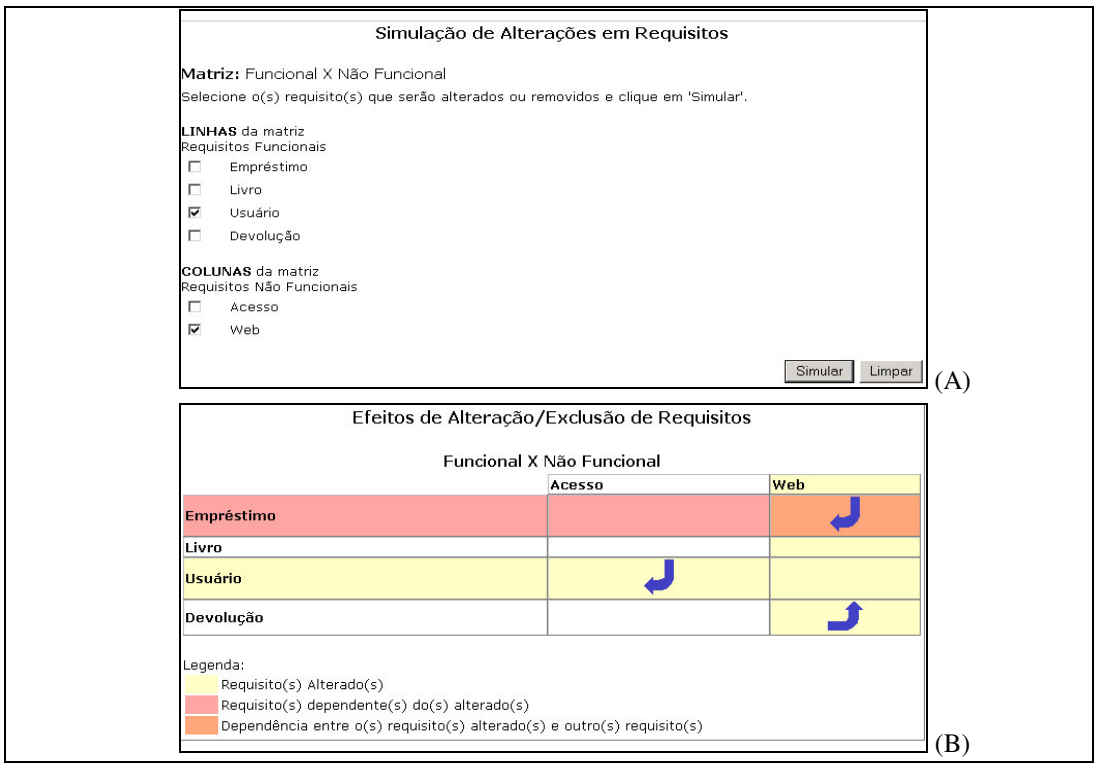


Figura 6: Simulação de alteração de requisito

5 Conclusões

A linguagem PHP além de ser de fácil manipulação mostrou-se bastante eficiente e eficaz no que diz respeito ao tempo de resposta nas requisições ao banco de dados.

O gerenciamento de requisitos poderá ser realizado remotamente, ou seja, o analista não depende de sua estação de trabalho para executar as atividades que necessita, não havendo portanto, limitações geográficas para uma equipe de desenvolvimento. Além da vantagem de ser independente de plataforma.

A ferramenta desenvolvida possui algumas vantagens em relação a outras ferramentas disponíveis no mercado. Uma delas é o fato de sua utilização ser via *web*, proporcionando uma maior agilidade no gerenciamento dos requisitos. Outra vantagem é o fato de que é possível construir diversas matrizes de rastreabilidade, e suas linhas e colunas não ficam restritas a um único tipo de requisito, podendo assim, rastrear diferentes tipos de requisitos em uma única matriz, o que não é permitido na ferramenta Rational RequisitePro. Com a construção de diversas matrizes de rastreabilidade, surge a possibilidade de gerenciar um número maior de requisitos, superando o limite de 250 requisitos por projeto.

O uso de uma ferramenta para o gerenciamento dos requisitos auxilia o analista na organização e controle dos requisitos. Porém é fundamental lembrar a importância da etapa de elicitação dos requisitos pois, em nada adianta uma boa ferramenta de gerenciamento se os requisitos não forem captados de forma correta, refletindo todas as funções que o sistema deve realizar.

Referências

FEATHER, Stephen; CASSADY-DORION, Luke. **JavaScript em exemplos**. São Paulo : Makron Books, 1997. xxv, 357 p.

MARQUIONI, Eduardo. **Os processos típicos da engenharia de requisitos – parte 5**, São Paulo, 2004. Disponível em <http://www.choose.com.br/infochoose/artigos/45art03.htm>. Acessado em 8 de julho de 2004.

PETERS, James F; PEDRYCZ, Witold. **Engenharia de software** : teoria e prática. Rio de Janeiro: Campus, 2001. xvii, 602 p.

PRESSMAN, Roger S. **Engenharia de software**. 5.ed. São Paulo : McGraw-Hill, 2002. xxvii, 843 p.

SOARES, Walace. **MySQL** : conceitos e aplicações. São Paulo : Érica, 2001. 294 p.

SOMMERVILLE, Ian. **Engenharia de software**. 6.ed. São Paulo : Addison Wesley, 2003. xiv, 592 p.

Ferramenta de apoio ao ensino de algoritmos.

Rafael de Santiago (UNIVALI)
santiago@inf.univali.br

Rudimar Luís Scaranto Dazzi (UNIVALI)
rudimar@inf.univali.br

Resumo. Este trabalho pretende apresentar o Construtor e Interpretador de Algoritmos para Programação (CIFluxProg) e os testes efetuados com ele em sala de aula. O CIFluxProg é uma ferramenta de apoio para as disciplinas iniciais da área de programação, como por exemplo, as disciplinas de algoritmos. Desenvolveu-se esta ferramenta que permite aos alunos implementarem e testarem suas soluções lógicas de programação tanto em Portugol como em Fluxograma. Contando com o auxílio do teste de mesa e com a interpretação da solução, inclusos na ferramenta, a verificação de integridade das soluções pode ser verificada. Esta ferramenta flexibiliza o processo de treinamento dos alunos, uma vez que estes podem verificar o funcionamento das suas soluções na prática, visualizando em detalhes os passos e os resultados da sua solução.

Palavras-chaves: Ensino em Ciência da Computação, Algoritmos, Fluxogramas.

1 Introdução

A aprendizagem de lógica de programação é muito importante para todas as carreiras ligadas à informática. Programação é, sem dúvida, a disciplina mais importante para a formação daqueles que terão no desenvolvimento de softwares o produto final do seu trabalho. Uma vez que a aprendizagem de programação ocorre praticamente, durante todo o curso, o baixo índice de assimilação dos estudantes nas disciplinas cujos requisitos exigem o conhecimento de programação tem sido um grande problema enfrentado em muitas instituições. Rocha (1991) afirma que estamos tendo um fracasso no ensino de programação e Gomes (2000) fala do insucesso generalizado verificado na aprendizagem de programação.

Diversos sistemas para implementação de animações de algoritmos e de estruturas de dados (STUBBS, WEBRE 1988; SZWARCFITER, MARKENSON 1994) foram produzidos desde o trabalho pioneiro de Brown (1987, 1988), como por exemplo, (AMORIM, REZENDE 1993; BROWN 1991; STASKO 1990). Vários destes sistemas exploram muito bem a potencialidade do uso de visualizações gráficas das operações realizadas nas estruturas de dados como ferramenta de ensino. Algumas ferramentas mais recentes podem ser verificadas em Cares (2002), Medeiros (2001) e Mendes e Gomes (2000).

O sistema CIFluxProg é uma ferramenta que foi construída com intuito de auxiliar os estudantes de computação no aprendizado da lógica de programação de computadores. Não são muitas as opções de ferramentas disponíveis neste contexto e nem sempre possuem recursos visuais e sintaxe equivalente a utilizada em sala de aula para a construção das soluções computacionais, sejam com a utilização de Fluxograma ou de Português Estruturado (Portugol). Isto torna difícil e desagradável o processo de aprendizagem que deveria ser na medida do possível fácil e prazeroso (MEDEIROS e DAZZI 2002).

A principal meta desse sistema é disponibilizar aos aprendizes de lógica de programação uma maneira fácil e intuitiva de testar os conceitos ensinados em sala de aula nas disciplinas que envolvem lógica de programação. Esta ferramenta em seus dois módulos, permite a construção e teste de algoritmos confeccionados como Fluxogramas ou Portugol. Com isso, atende tanto aos alunos que possuem perfil mais visual quanto no textual.

Esta ferramenta, com o que foi construído até o momento, não tem o objetivo de ser uma ferramenta de ensino propriamente, mas um software de apoio ao professor, permitindo ao mesmo efetuar a prática do ensino de algoritmos em computador, com os recursos utilizados em sala. Este deve ser utilizado inicialmente com supervisão do professor, mas pode ser disponibilizado para os alunos confeccionarem e testarem suas soluções fora do horário de aula.

A utilização de uma ferramenta computacional para os alunos confeccionarem seus algoritmos, permitindo aos mesmos testarem suas soluções visualizando o resultado gerado por elas, é antes de tudo, um grande motivador do processo de ensino aprendizagem (DAZZI, MIRANDA, SOUZA 2000; SANTIAGO e DAZZI 2003).

Mendes e Gomes (2000) em seu sistema denominado SICAS, nos diz que este permite a construção de resoluções para problemas, bem como a sua simulação, observação e análise. Caso a solução não seja adequada, o aluno poderá detectar e corrigir os erros cometidos, de forma a encontrar uma solução satisfatória. Este processo de detecção e correção de erros é fundamental para o desenvolvimento nos aprendizes de programação. A construção da resolução do problema é realizada no SICAS, através de fluxogramas. Esta decisão foi tomada no sentido de privilegiar o uso de representações gráficas (fluxogramas), em detrimento de especificações verbais (pseudocódigo), não apenas pelos estudos analisados, mas por uma forte convicção de que essa forma de representação é mais apelativa (prendendo mais a atenção do aluno), tira mais partido do potencial do sistema visual humano para facilitar a compreensão, é mais clara e está menos sujeita a erros, mantendo uma atividade mais organizada e estruturada.

Esta proposta avança um pouco mais nestes contexto, disponibilizando tanto o ambiente visual com fluxogramas, como o textual, com portugol, permitindo com isso que os alunos possam escolher a seu critério qual das opções utilizar. Também permite desta forma, que os professores possam optar pela forma que mais lhe seja favorável, tornando assim o processo mais flexível.

2 A Ferramenta

O CIFluxProg é a composição de dois ambientes de desenvolvimento, um para a confecção de Fluxogramas e outro para a confecção de algoritmos em Portugol, junto com um compilador e um interpretador. Isso permite que sejam criadas e testadas tanto soluções em fluxogramas quanto em português estruturado, em um ambiente visual. Com a possibilidade de execução das soluções geradas com a visualização do resultado ou dos erros cometidos, é mais simples e agradável para os alunos interagirem e entenderem os processos de confecção de soluções de problemas computacionais.

A ferramenta foi desenvolvida na linguagem C++, contando com um interpretado de código desenvolvido na mesma linguagem, mas com apoio da ferramenta Lex & Yacc.

2.1 Interpretador

Para que haja execução de código na ferramenta CIFluxProg, um interpretador foi desenvolvido. Analisando a solução do usuário em busca de erros léxicos e sintáticos e se possível executando a mesma.

Para a execução de soluções em fluxogramas, foi construído um algoritmo que monta um código na memória um código compatível com o interpretador. O interpretador, por sua vez, analisa o código montado e o executa.

A gramática que o interpretador utiliza pode ser visualizada na Tabela 1. Esta gramática privilegia um conjunto reduzido de instruções, para permitir que os alunos coloquem em prática

seus algoritmos desde os primeiros testes até o ponto considerado necessário antes de apresentar uma linguagem de programação, por parte dos professores das disciplinas. Sendo assim foram disponibilizadas as estruturas básicas de desvio e repetição, além dos elementos necessários para estruturar as soluções como um programa.

Tabela 1: Gramática de alguns elementos na linguagem.

Portugol	**Descrição**
Inicio	Identifica o início do portugol
Fim	Identifica o fim do portugol
{	Inicio de bloco
}	Fim de bloco
se( )	Desvio Condicional
senao	Negação do “se ( )”
enquanto( )	Laço condicional
para __ ate __ passo __	Laço condicional com repetição incremental
inteiro	Tipo de dado numérico inteiro
real	Tipo de dado numérico real
logico	Tipo lógico de dados
cadeia	Tipo de dado de cadeia de caracteres
verdadeiro	Valor verdadeiro do tipo de dado lógico
falso	Valor falso do tipo de dado lógico
leia( )	Instrução para entrada de dados
escreva( )	Instrução para a saída de dados

2.2 Módulo de Portugol

O módulo de portugol foi desenvolvido para usuários com perfil verbal, pois estes possuem maior facilidade para exercitar sua lógica de programação em ferramentas que apresentam suas soluções de forma mais textual.

O módulo descrito (Figura 1) apresenta uma caixa de texto para a edição de códigos, uma barra de estruturas (para usuário inserirem algum texto padrão como IF..ELSE), e uma barra de menus. A ferramenta também possibilita salvar, abrir e imprimir fluxogramas.

Como principal atrativo do módulo, pode ser destacada a opção de execução do código escrito. Essa execução acontece com uma solicitação ao interpretador de código (desenvolvido especificamente para a ferramenta CIFluxProg).

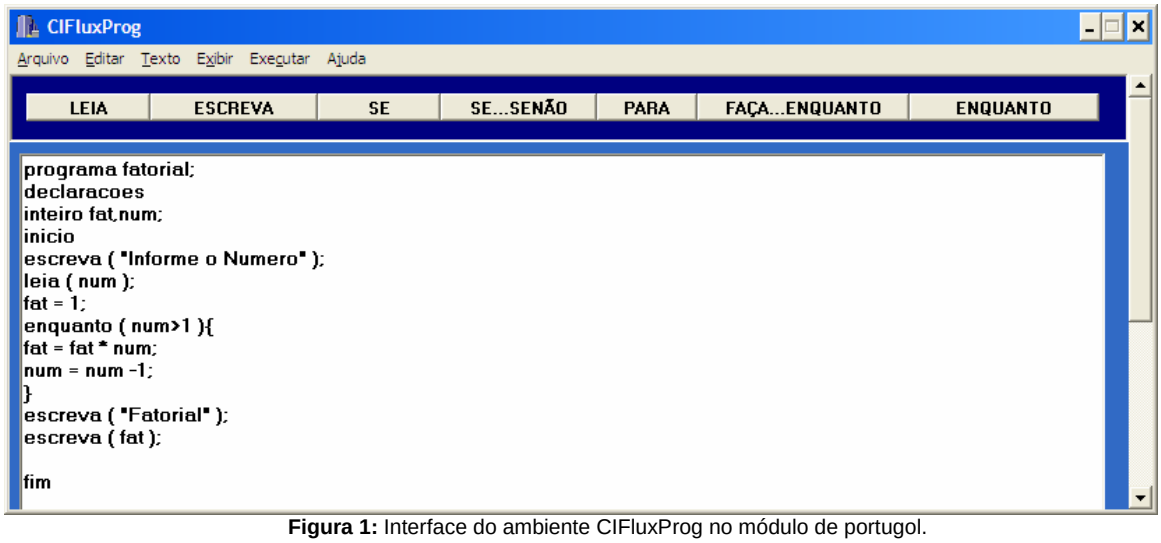


Figura 1: Interface do ambiente CIFluxProg no módulo de portugol.

A Figura 2 demonstra um algoritmo de fatorial executando no módulo de portugol. Nesta figura, pode-se observar ao lado direito teste de mesa da solução, contendo a cada linha as modificações ocorridas nas variáveis declaradas. Ao centro pode-se visualizar que o resultado da solução foi 120, sendo que no teste de mesa constata-se que o valor inserido para se chegar ao fatorial foi 5.

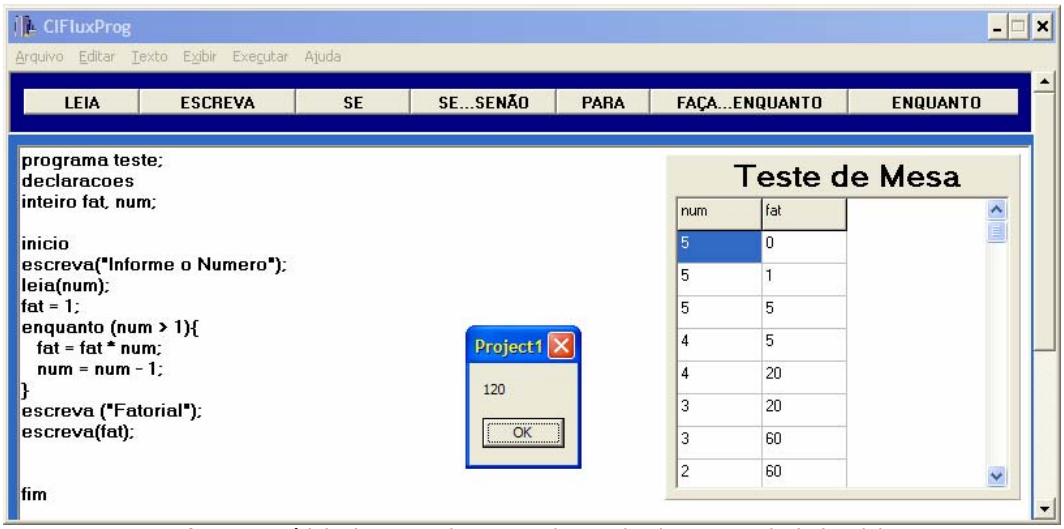


Figura 2: Módulo de portugol executando um algoritmo que calcula fatorial.

2.3 Módulo de Fluxograma

O módulo de fluxograma (Figura 3) foi o ponto mais importante para o projeto da ferramenta CIFluxProg, pois foi a necessidade inicial, visto que este recurso (Fluxograma) passou a ser utilizado na disciplina de algoritmos do curso de ciência da computação e não se tinha acesso a qualquer ferramenta que desse suporte a confecção e testes dessas soluções.

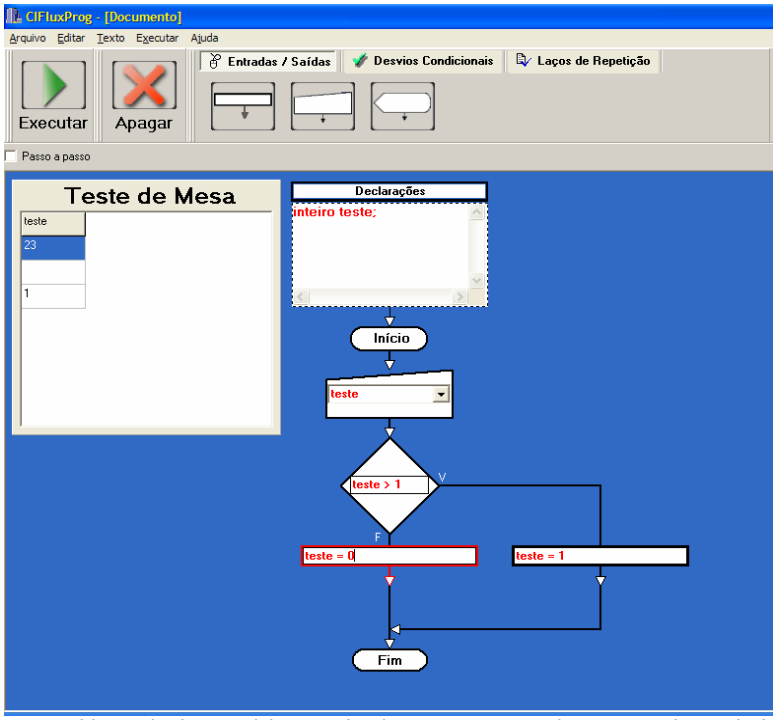


Figura 3: Ambiente de desenvolvimento de Fluxogramas, com destaque na barra de ferramentas.

Como detalhe importante deste módulo é o suporte a aninhamento de símbolos. O aninhamento acontece quando se tem dentro de um símbolo dos tipos Laço de Repetição ou Desvio Condicional um outro símbolo também destes mesmos tipos. É imprescindível que o sistema suporte esse tipo de encadeamento, pois este tipo de estrutura é bastante utilizado na resolução de problemas computacionais. A ferramenta similar que fora anteriormente desenvolvida não tinha esse tipo de recurso o que praticamente inviabilizava seu uso.

Todos os símbolos disponibilizados na barra de ferramentas possuem campos editáveis pelo usuário. É nesses campos que o usuário deve inserir os nomes e valores de variáveis, condições lógicas, etc.

Quando um símbolo é inserido seu desenho é criado na tela e automaticamente o cursor do teclado é direcionado para a área de edição do símbolo criado, permitindo assim uma fácil manipulação das informações necessárias para a futura interpretação e execução do fluxograma.

O sistema também disponibiliza um botão localizado no canto superior esquerdo da tela com o título "executar". Sua função é repassar a solução desenvolvida para o interpretador que retornará como resposta a execução do algoritmo.

O usuário dispõe também de recursos para guardar em seu computador o material que esta sendo desenvolvido no sistema. As opções disponíveis na ferramenta, que são as opções tipicamente encontradas nos softwares em geral como salvar e abrir arquivos. Com isso não há a necessidade de que o usuário recrie todo um fluxograma a cada vez que for usar o sistema.

Todas as soluções implementadas em quaisquer dos módulos pode ser aberto no outro, sem perda ou problema de qualquer natureza. O interpretador utilizado para executar o fluxograma é o mesmo que o utilizado para o Portugol. Isso permite total compatibilidade entre as soluções, deixando o usuário totalmente livre para escolher qual das opções lhe agrada mais. Podendo ainda verificar a solução em ambas as formas de implementação (fluxograma ou portugol, uma vez que o código gerado em ambas as ferramentas é totalmente compatível), e se desejar executá-la verificando que o resultado é o mesmo.

Na Figura 4 pode-se constatar que o mesmo algoritmo submetido aos dois módulos (algoritmo de fatorial), obteve os mesmos resultados tanto na execução, quanto na interpretação.

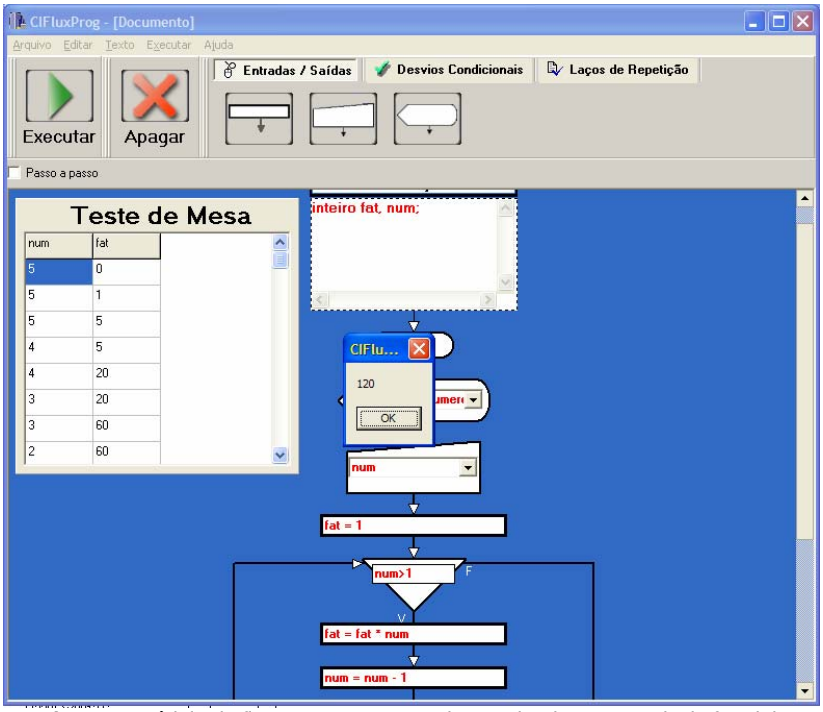


Figura 4: Módulo de fluxograma executando um algoritmo que calcula fatorial.

3 Conclusão

Como pode ser observado neste artigo, a ferramenta CIFluxProg[1], possui várias características que a credenciam como uma boa opção para a utilização em aulas práticas de algoritmos. Esta agrega duas opções de ensino, tanto com a utilização da clássica ferramenta denominada Fluxograma, como do Portugol, permitindo com isso atender os alunos com maior

[1] A ferramenta em sua versão experimental está disponível no endereço: http://www.cttmar.univali.br/~gia/projetos_pesquisa.htm

dificuldade no desenvolvimento da lógica necessária para a solução computacional dos problemas, sejam eles com perfil tendendo mais ao tipo de raciocínio lógico-matemático e visual-espacial, como os que possuem perfil tendendo mais para o verbal.

Essa ferramenta se mostrou bastante agradável para os usuários, que se empolgaram bastante ao utilizá-la, nos testes preliminares efetuados com uma das turmas de algoritmos do primeiro período do curso de ciência da computação do CTTMar-UNIVALI. Esse fato parece demonstrar que o interesse pelo aprendizado está diretamente relacionado a motivação dos alunos para com os recursos utilizados. O fato destes alunos deixarem de utilizar apenas papel e o lápis para utilizar também o computador, tanto para escrever suas soluções, como para testar e visualizar os resultados, parece ter gerado esse efeito motivador. Outro ponto que agradou foi a possibilidade de visualizar e testar a mesma solução tanto em portugol como em fluxograma. Esse objetivo pode se considerar alcançado, mesmo com a ferramenta em fase final de acabamento.

Acredita-se que esta ferramenta possibilite melhorias no processo de ensino aprendizagem das disciplinas que a utilizarem, minimizando com isso um dos grandes problemas existentes nestas disciplinas, que é o alto índice de reprovação. Só o fato observado da disposição dos alunos em fazer exercícios ter aumentado já começa a validar esse processo de melhoria, uma vez que um dos grandes problemas observados para o baixo rendimento dos alunos em sala é a falta de exercícios efetuados fora do horário de aula.

Referências

AMORIM, R. V.; REZENDE, P. J. Compreensão de Algoritmos através de Ambientes Dedicados a Animação. In: **SEMISH**, 10., 1993.

BROWN, M. H. Zeus: A System for Algorithm Animation and Multi-View Editing. **Procedings...** IEEE Workshop on Visual Languages, 1991.

BROWN, M. H. **Algorithm Animation.** The MIT Press, 1987.

BROWN, M. H. **Exploring Algorithms Using Balsa-II.** Computer, maio 1988. p. 14-36.

CARES, P. L. L. **Ambiente para teste de mesa utilizando fluxograma.** Trabalho de Conclusão (Graduação)–Faculdade de Ciência da Computação, Universidade do Vale do Itajaí, Itajaí, 2002.

DAZZI, R. L. S.; MIRANDA, E. M.; SOUZA, E. M. S. SAEL: Sistema de apoio ao ensino de lógica de programação. In: **WORKSHOP DE INFORMÁTICA APLICADA À EDUCAÇÃO**, Araraquara, 2000.

GOMES, A. J. **Ambiente de suporte à aprendizagem de conceitos básicos de programação**, Dissertação (Mestrado)–Universidade de Coimbra, 2000.

MEDEIROS, C. L. **Aplicação web para realizar teste de mesa em algoritmos.** Trabalho de Conclusão (Graduação)–Faculdade de Ciência da Computação, Universidade do Vale do Itajaí, Itajaí, 2001.

MEDEIROS, C. L.; DAZZI, R. L. S. Aprendendo algoritmos com auxílio da WEB, In: **CONGRESSO BRASILEIRO DE COMPUTAÇÃO**, 2., 2002, Itajaí. **Anais...** Itajaí: UNIVALI – CTTMar, 2002.

MENDES, A. J. N.; GOMES, A. J. Suporte a aprendizagem de programação com o ambiente SICAS. In: **CONGRESSO IBERO AMERICANO DE INFORMÁTICA EDUCATIVA-RIBIE**, 5., 2000, Viña del Mar-Chile. **Anais...** Viña del Mar-Chile: Universidad de Chile, 2000.

PRICE, A. M. A.; TOSCANI, S. S. **Implementação de linguagens de programação: Compiladores.** Porto Alegre: Sagra Luzzatto, 2. ed., 2001.

ROCHA, H. V. **Representações Computacionais Auxiliares ao Entendimento de Conceitos de Programação**, Unicamp, 1991.

SANTIAGO, R.; DAZZI, R. L. S. Ferramentas que auxiliam o desenvolvimento da lógica de programação. In: **SEMINCO - SEMINÁRIO DE COMPUTAÇÃO**, 12., 2003. **Anais...** Blumenau: FURB, 2003. p.113-120.

SOUZA, E. M. S.; GRANDI, G.; SOUZA, O. R. M.; DAZZI, R. L. S. Reavaliando o ensino de algoritmos. In: **SIMPÓSIO CATARINENSE DE COMPUTAÇÃO**, 1., 2000. **Anais...** Itajaí: UNIVALI, 2000. vol. 3, p. 69-79.

STASKO, J. T. **Tango: A Framework and System for Algorithm Animation.** Computer, setembro 1990. p. 14-36.

STUBBS, D. F.; WEBRE, N. W. **Data Structures with Abstract Data Types and Pascal**, Pacific Grove, Brooks/Cole, 2 ed., 1988.

SZWARCFITER, J.; MARKENSON, L. **Estruturas de Dados e seus Algoritmos**, LTC, 1994.

Uma Estratégia para Aplicar Mineração de Dados no Acompanhamento do Aprendizado na EaD

Claudivan Cruz Lopes (CEFET-PB/UNED-Cajazeiras)
claudivan@cefetpb.edu.br

Ulrich Schiel (UFCG)
ulrich@dsc.ufcg.edu.br

Resumo. Uma dos grandes problemas da Educação a Distância está na dificuldade de acompanhar o aprendizado dos alunos à distância. Esta dificuldade se justifica por diversos motivos, dentre eles, a falta de contato presencial entre professores e alunos. Sendo assim, faz-se necessário o desenvolvimento de modelos que representem o status de aprendizagem dos alunos de um curso à distância. Seguindo esta linha de pesquisa, este artigo propõe uma estratégia para o acompanhamento do aprendizado na Educação à Distância baseada nas práticas de acompanhamento do ensino presencial, acrescida da tática de análise de dados, onde fatores do acompanhamento podem ser relacionados para se verificar a aprendizagem de forma mais elaborada através da geração de um novo conhecimento descoberto com a utilização de ferramentas de Mineração de Dados.

Palavras-chave: Educação a Distância, Acompanhamento do Aprendizado, Mineração de Dados

1 Introdução

Segundo Neto (1998), a Educação a Distância (EaD) é uma forma de organização de ensino-aprendizagem, na qual alunos estudam, quer em grupo, quer individualmente em seus lares, locais de trabalho ou outros lugares, com materiais auto-instrutivos distribuídos por meios de comunicação, garantindo a possibilidade de comunicação com docentes, monitores ou outros alunos.

Tomando como base esta definição, podemos constatar as seguintes características da EaD: **(1)** separação física entre professores e alunos; **(2)** auto-aprendizagem do aluno; **(3)** utilização de meios de comunicação para unir professores e alunos e para transmissão dos materiais didáticos e **(4)** atendimento aos alunos geograficamente dispersos.

Notoriamente, dadas estas características, podemos observar que a implementação da EaD exige que alguns desafios sejam enfrentados, dentre eles: **(1)** uma mudança cultural é exigida para que a EaD atinja os seus objetivos educacionais uma vez que o aluno não está presente no momento da aula. Portanto, manter a atenção e o compromisso do aluno torna-se uma tarefa difícil; **(2)** para a elaboração de um curso pode ser necessária uma equipe multidisciplinar: um professor fica responsável pelo planejamento e elaboração dos conteúdos, um programador é responsável por programas os sistemas e um pedagogo se encarrega de supervisionar o andamento do curso; e **(3)** conforme Masetto (2000), outro grande desafio a ser enfrentado é o acompanhamento do aprendizado dos alunos devido a falta de contato presencial, onde a ausência da percepção do professor quanto ao estado de compreensão de seus alunos pode levar ao insucesso de um curso a distância. Assim, quando se pensa em acompanhamento a aprendizagem na EaD, percebe-se claramente que um longo caminho pode ser construído, e um dos grandes problemas é articular as inúmeras vantagens criadas pelas tecnologias e criar modelos de acompanhamento do aprendizado que efetivamente contenham alguma possibilidade de mudança qualitativa.

Por outro lado, o uso do computador em ambientes de EaD torna possível a captura de algumas características do aprendiz à distância, de forma a analisá-las de maneiras análoga ao

comportamento de um aluno de um curso presencial. Por exemplo, o grau de interesse e a participação do aluno podem ser refletidos pela freqüência com que o mesmo acessa o curso, enquanto que o comportamento social pode ser delineado pelo rastreamento de suas interações com o ambiente de EaD através de mecanismos de comunicação como *chat* ou *e-mail*. Tais observações consideram que a análise do histórico das ações do aluno pode revelar a sua forma de estudo e seu aproveitamento. A princípio, essa análise acarreta dois problemas: **(1)** identificar quais são os fatores que devem ser observados; e **(2)** como mensurá-los.

Considerando esta problemática, o presente artigo propõe uma estratégia para o acompanhamento do aprendizado no contexto de um ambiente de suporte a EaD. Tal estratégia contempla a modelagem de um conjunto de dados sobre os alunos de um curso à distância, de forma que este possa ser devidamente utilizado para a descoberta de conhecimento através de técnicas de Mineração de Dados, onde padrões comportamentais ou características interessantes a cerca do processo de ensino-aprendizagem possam ser encontradas.

O uso de Mineração de Dados é justificado segundo vários motivos: **(1)** à medida que um curso à distância é concebido para atender ao público remoto, o contingente de alunos pode crescer consideravelmente, e, portanto, a tarefa de acompanhar o aprendiz pode-se tornar árdua; **(2)** normalmente, os cursos de EaD mantêm seus dados em bancos de dados e a natureza histórica destes dados pode ser útil para análises prospectivas; **(3)** conforme Bernhardt (2001), as decisões baseadas em dados históricos ajudam os educadores a enxergarem quem são seus alunos e quais qualidades e dificuldades eles compartilham; **(4)** Johnson (2000) destaca que a implantação de um programa de coleta e análise de dados pode levar a melhorias na educação como nenhuma outra inovação o fez; **(5)** pouco se tem feito em termos de sistemas de suporte a decisão em EaD. A maioria dos ambientes existentes não oferece recursos sofisticados para apoiar decisões.

Para apresentar esta proposta, organizamos as seguintes seções subseqüentes: a seção 2 realiza uma revisão sobre o acompanhamento do aprendizado na modalidade de educação presencial e a distância; a seção 3 descreve a estratégia de acompanhamento; A seção 4 mostra uma aplicação da estratégia descrita; e por fim, a seção 5 apresenta as conclusões.

2 Acompanhamento do Aprendizado

No ensino presencial, o professor tem à sua disposição uma série de instrumentos de acompanhamento que numa ocasião ou noutra podem ser aplicados, onde os seus resultados podem prestar um grande serviço para a tomada de decisão. Para Linderman (1986), tais decisões podem representar uma **operação preditiva**, ou seja, com base no desempenho presente e passado, deve-se formar um juízo sobre o possível sucesso ou fracasso de um estudante em várias atividades que ele empreenderá futuramente; ou uma **operação classificatória**, onde o professor classifica os alunos com base na consecução de certos objetivos escolares.

Adicionalmente, uma tendência no acompanhamento da aprendizagem é o processo ***Data-driven Decision Making*** (**D^3M**), onde vários fatores são correlacionados para se verificar o aprendizado de maneira mais elaborada. O processo de D^3M admite o uso de uma base de dados baseado no aluno, agregando informações sobre a sua vida escolar. Esta base de dados identifica quem cada estudante é (**dados demográficos** como idade, sexo, situação financeira, nível de escolaridade, região de procedência, etc; e **dados comportamentais**, como o número de reprovações, a situação no curso, etc) e o que eles sabem (**dados sobre avaliações**). Assim, o professor pode obter informações individuais, sumariar os resultados dos estudantes (agregação) e reorganizar as informações para entender resultados sob a óptica de diferentes grupos de estudantes (desagregação e análise das informações fazendo cruzamentos de dados).

Por sua vez, não existe uma prática pedagógica de acompanhamento da aprendizagem voltada especificamente para a Educação a Distância. No entanto, Vieira *apud* Neder (2002), Souza (1999) e outros destacam que o acompanhamento do aprendizado em EaD pode se sustentar em princípios análogos ao da educação presencial, porém recomendam que o mesmo seja complementado com um acompanhamento especial. Neste sentido, alguns trabalhos foram propostos. Por similaridade, os dividimos em duas categorias: **acompanhamento informal** e **acompanhamento alternativo**.

No **acompanhamento informal** são capturadas algumas características do aprendiz com o intuito de analisá-las de maneira análoga ao comportamento de um aluno de um curso presencial, onde a linguagem corporal, o grau de interesse, a participação e a sociabilidade podem ser vistas sob a forma da interação do aluno com o ambiente, provendo meios suficientes para avaliá-lo e fornecendo *feedback* necessário sobre o acompanhamento do mesmo. Santos (1999), Menezes *et al.* (1999), Machado e Becker (2002), Alves *et al.* (2002), Souto *et al* (2001) e Silva (2002) são exemplos de pesquisadores que propuseram ambientes de EaD com suporte ao acompanhamento informal.

No **acompanhamento alternativo** as realizações feitas pelos alunos ao longo de um curso são criteriosamente avaliadas segundo uma série de itens subjetivos possibilitando *feedback* aos professores quanto às qualidades e dificuldades dos mesmos; além de que, os próprios alunos podem refletir e acompanhar seu próprio desempenho e verificar a aquisição de conhecimentos, habilidades e atitudes comprovadas através dos resultados de todos os seus trabalhos. Taminoto (1998), Nascimento (2002), Otsuka *et al.* (2002) são exemplos de pesquisadores que desenvolveram trabalhos nesta área.

Analisando os trabalhos citados como exemplos de pesquisas relacionadas aos acompanhamentos informal e alternativo, podemos constatar que não existe um conjunto padrão de informações que se julgue relevante ao acompanhamento do aprendizado na EaD, ou seja, as informações manipuladas em cada ambiente variam de um para outro.

Adicionalmente, nenhum deles mostrou-se vinculado ao processo D^3M. Este processo pode auxiliar o professor nas suas atividades de acompanhamento e diagnóstico de problemas de aprendizagem não somente relacionadas aos resultados das avaliações, mas também abrangendo outros fatores como dados demográficos e comportamentais, proporcionando análises mais sofisticadas.

3 Uma Estratégia para Aplicar Mineração de Dados no Acompanhamento do Aprendizado na EaD

Uma estratégia para o acompanhamento do aprendizado na EaD baseado em técnicas de Mineração de Dados deve considerar uma série de requisitos fundamentais para que sua utilização seja aplicada. A seguir, são descritos os requisitos básicos levantados pelos autores.

Requisito R1. O acompanhamento do aprendizado deverá ser apoiado com a utilização de uma ferramenta de Mineração de Dados interativa com o minerador, de forma que o mesmo possa: **(1)** selecionar a porção dos dados que ele deseja correlacionar; e **(2)** visualizar os padrões de conhecimento extraídos;

Requisito R2. A estratégia deve dispor dados sobre o acompanhamento informal (as interações do aluno com o ambiente de EaD e com os materiais didáticos disponibilizados pelo professor), provendo dados para análises de características como graus de interesse, de participação e de sociabilidade;

Requisito R3. A estratégia deve dispor dados sobre o acompanhamento alternativo (os itens de julgamento das características subjetivas dos alunos). Dessa forma, o professor terá condições

de analisar outras habilidades que não apenas aquelas relacionadas aos aspectos técnicos, mas também pessoais e de condicionamento afetivo;

Requisito R4. A estratégia deverá dispor de dados sobre as avaliações dos alunos e dados pessoais, demográficos e comportamentais sobre o próprio aluno. A união destas informações juntamente com os aspectos informais (Requisito R2) e alternativos (Requisito R3) do acompanhamento proverá um background completo das realizações, habilidades e características dos alunos. E esta coleção de dados, em grandes quantidades, poderá gerar padrões e tendências interessantes que favoreçam o acompanhamento do aprendizado;

Requisito R5. Todos os dados do acompanhamento informal, alternativo e os dados sobre avaliações, demográficos e comportamentais poderão ser consultados sem nenhuma restrição de projeção e junção entre eles.

3.1 Descrição da Estratégia

O objetivo da nossa estratégia é dispor meios para o desenvolvimento de um Sistema de Apoio à Decisão (SAD) que contempla uma coleção de informações de interesse sobre os estudantes de um curso a distância, de maneira que esta coleção seja utilizada para análises prospectivas a cerca do acompanhamento da aprendizagem dos mesmos.

A estratégia se aplica na situação onde existe um ambiente de EaD que não dá suporte à análise de dados através do uso de Mineração de Dados. Para tal, a mesma foi dividida em duas fases: a **fase de planejamento** e a **fase de análise de dados**. A seguir, descreveremos cada uma delas.

3.1.1 Fase de Planejamento

Na fase de planejamento, devemos definir e modelar os dados de interesse e definir o tipo de mineração que deve ser realizada. Sob o ponto de vista da definição e modelagem dos dados de interesse, devemos acompanhar o seguinte roteiro:

1º - Definir padrões de consultas para os grupos de usuários. Padrões de consultas são as possibilidades de consultas que os usuários poderão efetuar ao utilizar o SAD. Por exemplo, professores poderão consultar dados sobre suas disciplinas e turmas. Por sua vez, coordenadores somente consultarão os seus cursos;

2º - Definir um conjunto de visões para atender aos padrões de consultas. Uma visão pode ser vista como uma relação contendo algum dado que contemple ou o requisito R2 (dado do acompanhamento informal) ou o R3 (dado do acompanhamento alternativo) ou o R4 (dado demográfico ou comportamental ou sobre avaliação); porém uma visão nunca deve contemplar mais de um requisito. O motivo para esta restrição é simples: como os requisitos R2, R3 e R4 contemplam dados de acompanhamento distintos, o usuário tem a possibilidade de correlacioná-los na forma que desejar. Se uma visão contemplar, por exemplo, o requisito R2 e o R3, o usuário terá que correlacionar dados de acompanhamento informal e alternativo respectivamente. E isto pode não ser desejado ou pode não fazer sentido.

Destacamos dois mecanismos para a implementação do conjunto de visões:

Abordagem virtual, com a utilização de *Views* dos SGBDs. Com ele, é possível criar tabelas virtuais acessíveis apenas por usuários autorizados. Para cada padrão de consulta, pode-se criar um conjunto de *Views*;

Abordagem materializada, com a utilização de um sistema de Data Warehousing. Tais sistemas contêm ferramentas de extração, limpeza, agregação e consolidação dos dados de interesse num Data Warehouse de acompanhamento.

Logicamente, como estamos tratando do acompanhamento da aprendizagem, estamos definindo a granularidade mínima de análise como sendo as realizações, características e atitudes pertencentes ao aluno. Além do mais, a identificação do aluno deve ser camuflada como forma de segurança e ética.

Ainda sob o ponto de vista de definição de informações de interesse, podem ocorrer os seguintes casos:

Caso 1: Quando o requisito R2 não é satisfeito, isto indica que no sistema de EaD não foi previsto o acompanhamento informal. Para remediar tal situação, pode-se adotar as seguintes posições:

1º - Utilize os *logs* dos servidores *web* para conhecer os acessos aos materiais didáticos disponíveis pelo professor através de sua url. No *log* do servidor é possível descobrir informações como usuário, *url*, data de acesso e duração do acesso. Assim, é possível capturar dados sobre as interações do aluno com os recursos didáticos;

2º - Crie meios de capturar as interações do aluno com o ambiente de EaD, como participações em *chat*, listas de discussões ou correio eletrônico.

Caso 2: Quando o requisito R3 não é satisfeito, o acompanhamento alternativo não foi previsto. Ou seja, no sistema de EaD, a única forma de avaliação prevista é baseada em notas. Para contornar tal situação, é necessária uma mudança estrutural na instituição de EaD. Quanto se resolve adotar a avaliação alternativa, todos os envolvidos no processo educacional devem estar dispostos a acreditar nas vantagens que este método de avaliação pode trazer. Resolvidos tais conflitos, a instituição deve pré-definir os artefatos que poderão ser produzidos pelos alunos, bem como os itens de julgamento que avaliarão tais artefatos. Padronizados os artefatos e os itens de avaliação, o passo seguinte é disponibilizá-los para o sistema de apoio a decisão.

Um mecanismo para implementação de tal tarefa é com a utilização de tabelas de dados. Com tabelas, os portfolios dos alunos contendo todas as suas realizações - artefatos, bem como a avaliação dos mesmos, poderiam estar armazenadas apropriadamente. Assim, pode-se utilizar tais tabelas na criação de *views* ou cubos para os padrões de consultas.

Caso 3: Quando o requisito R4 não é satisfeito, quer dizer que os dados sobre avaliações, comportamentais, pessoais e demográficos sobre os alunos não foram disponibilizados. Para remediar tal situação, deve-se adotar as seguintes estratégias:

1º - Considerando que não existam dados demográficos e comportamentais:

o Se existe um cadastro de alunos, incremente o cadastro com os dados demográficas e comportamentais de interesse, caso o cadastro não possua tais dados;

o Se não existe um cadastro de alunos, crie um cadastro de alunos com dados demográficos e comportamentais de interesse;

o Disponibilize os dados demográficos utilizando o mecanismo de visões citado anteriormente, de acordo com o padrão de consulta do usuário.

2º - Considerando que não existem dados sobre avaliações, a estratégia é criar meios de armazenar as tarefas que expressam o desempenho do aluno em termos de atividades propostas pelo professor. Cada atividade proposta pelo professor pode ter um resultado. Por exemplo, a participação ou não em uma reunião, a nota de um trabalho, etc. Considerando isso, esse tipo de informação depende das atividades escolhidas pelo professor e da forma que ele escolheu para validá-las, ou seja, o critério para decidir se elas foram cumpridas ou não pelos alunos.

Como resultado desta fase de planejamento, temos um **MAD - Modelo Analítico de Dados**, refletindo um completo background escolar dos estudantes. Assim, um conjunto de dados demográficos, comportamentais, informais, alternativos e dados sobre avaliação estarão

disponíveis num único ambiente. Adicionalmente, com a formalização de um MAD, o usuário terá a possibilidade de navegar e selecionar os dados de interesse que estão disponíveis para uma consulta em questão. Sendo assim, o usuário usará o MAD para escolher os atributos que serão utilizados para análises prospectivas.

Fechando a fase de planejamento, o último tópico a ser analisado é a disponibilidade de tarefas de mineração a serem utilizadas pelos usuários. Para esta questão, o interessante é se fazer um estudo das necessidades dos usuários e das potencialidades que cada técnica de Mineração de Dados pode lhes trazer, avaliando assim, se uma determinada tarefa pode ser utilizada ou não. Por exemplo, a mineração por agrupamento poderia ser utilizada para agrupar um conjunto de alunos segundo alguma característica; a mineração por associação poderia ser utilizada para verificar associações entre características distintas (informal versus alternativo; alternativa versus demográfica; etc) dos estudantes; a tarefa de classificação poderia ser usada para dispor perfis de alunos segundo algumas habilidades; e assim sucessivamente.

No que diz respeito aos mecanismos de implementação, existe hoje uma gama de softwares de mineração de dados. Todos eles proporcionam uma série de algoritmos já implementados e diversas ferramentas de visualização de resultados. Dentre elas, podemos destacar: DBMiner, IBM Inteligent Miner e o *framework* WEKA.

3.1.2 Fase de Análise de Dados

Propomos nesta fase uma abordagem semi-automática para extração de conhecimento baseado em técnicas de mineração de dados com o intuito de gerar regras, padrões, tendências e/ou associações relativos às características do estudante armazenadas historicamente no banco de dados e disponíveis através do MAD. Tais padrões podem revelar um conhecimento útil no que diz respeito ao acompanhamento do aprendizado, apoiando os educadores no processo de tomada de decisão e gerando novas oportunidades estratégicas ao providenciar as seguintes capacidades: **(1)** Identificação das causas de problemas, não apenas dos sintomas; **(2)** Possibilidade do cruzamento de informações demográficas, comportamentais, informais, alternativas e sobre avaliação propiciando uma análise mais elaborada que o tradicional; **(3)** Troca de pressentimentos e hipóteses por fatos; **(4)** Avaliação de necessidades e alocação de recursos para satisfazê-las; **(5)** Verificação do cumprimento das metas de ensino; **(5)** Identificação de perfis de potenciais alunos desistentes; **(6)** Determinação de atividades mais eficazes no curso; **(7)** Questões que tradicionalmente requeriam análises extensivas podem ser respondidas diretamente através dos dados; etc.

Achamos conveniente dividir esta fase de Análise de Dados em três etapas: **Etapa de Coleta de Dados**, **Etapa de Mineração de Dados** e **Etapa de Análise e Incorporação dos Resultados**.

Na *Etapa de Coleta de Dados*, o usuário utilizará as informações previstas num MAD e formulará uma consulta com os dados de seu interesse de acordo com algum padrão de consulta disponível para o mesmo.

Na *Etapa de Mineração de Dados*, as tarefas de mineração previstas na fase de planejamento devem ser colocadas à disposição do usuário. Além do mais, o usuário, depois de realizar uma consulta num MAD a fim de selecionar os dados de interesse, tem a possibilidade de escolher que padrão de conhecimento deseja saber e que tipos de regras devem ser descobertas.

Na *Etapa de Análise e Incorporação dos Resultados*, o usuário tem a possibilidade de visualizar o conhecimento minerado e decidir se tal conhecimento é útil ou não para o seu trabalho de acompanhamento.

4 MIDAS-POETA: Um SAD para o Ambiente Portfolio-Tutor

Nesta seção apresentaremos uma aplicação da estratégia descrita anteriormente, resultando num Sistema de Apoio à Decisão denominado MIDAS-POETA. Tal sistema, desenvolvido pelos autores, foi concebido como um módulo de análise de dados para um ambiente de suporte a EaD chamado Portfolio-Tutor; este último desenvolvido por Nascimento (2002). A integração e a utilização destes ambientes é feita de maneira complementar no processo de ensino/aprendizagem na EaD: enquanto o Portfolio-Tutor (um sistema tutor inteligente) dá suporte ao ensino propriamente dito, o MIDAS-POETA permite o acompanhamento do aprendizado.

Descreveremos resumidamente o Portfolio-Tutor e, a seguir, apresentaremos o MIDAS-POETA.

4.1 O Sistema Portfolio-Tutor

O Portfolio-Tutor tem como objetivo proporcionar um sistema tutor acoplado a um portfolio eletrônico. Ele apresenta duas camadas: a **camada portfolio** e a **camada tutor**. A camada portfolio tem como objetivo auxiliar o professor no acompanhamento alternativo dos alunos. Para tal, o sistema faz uso de um portfolio eletrônico (POETA) no qual são depositados os artefatos gerados pelo aluno e as respectivas avaliações. Os julgamentos são compostos por itens subjetivos, como capacidade de síntese, capacidade crítica, criatividade e outros; e também de itens somativos, onde uma nota é atribuída às atividades do aluno. Já a camada tutor tem como objetivo auxiliar o aluno numa seção de ensino e classificá-lo em níveis de aprendizagem segundo o desenvolvimento de suas habilidades ao longo do tempo.

A base de dados do Portfolio-Tutor apresenta as seguintes partes:

- Usuários: armazenam as informações sobre os alunos, professores, as interações do aluno com os outros alunos através de trocas de mensagens de e-mail's e os históricos de interações dos alunos aos materiais didáticos;

- Estrutura Curricular: armazena informações sobre disciplinas, cursos e matrícula em disciplinas;

- POETA: armazenam os artefatos produzidos pelos alunos e os resultados dos itens de julgamento subjetivos atribuídos pelo professor. Para compor o portfolio do aluno, cada atividade refere um artefato, que é avaliado segundo os itens de avaliação padronizados pela escola. Para um artefato, porém, podem existir diversas atividades associadas. O professor julga as atividades atribuindo notas aos itens de avaliação subjetivos. E para cada atividade realizada pelo aluno, são entregues os documentos que comprovam a sua realização;

- Tutor: representa as ações realizadas pelo tutor para produzir uma seção de ensino virtual.

Apesar de ser um sistema que promove a EaD de forma transparente e efetiva, o Portfolio-Tutor apresenta a limitação de não proporcionar um módulo que dê suporte as decisões a cerca do processo de aprendizagem. Além disso, com a utilização do portfolio, uma grande quantidade de dados estará disponível. E estes dados podem revelar informações úteis (padrões, tendências, associações, etc) que favoreçam o acompanhamento do aprendizado dos alunos. Diante desta limitação, foi desenvolvido o sistema MIDAS-POETA.

4.2 O Sistema MIDAS-POETA

A Figura 1 abaixo ilustra a arquitetura de alto nível do MIDAS-POETA. Conforme observado, o sistema faz uso da base de dados do Portfolio-Tutor para a criação do MAD como um conjunto de visões sobre este banco de dados. Assim, o usuário utilizará o MAD e selecionará os atributos que lhe interessarem para realizar uma consulta.

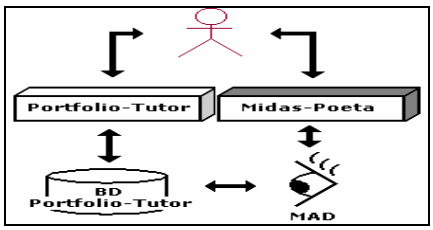


Figura 1: Arquitetura de Alto Nível do MIDAS-POETA

A utilização do MIDAS-POETA pode ser realizada a qualquer momento do processo de ensino/aprendizagem. Porém, é de valia notar algumas situações interessantes:

- O professor pode utilizar o sistema no final de um semestre letivo ou no início do semestre. Assim, ele pode averiguar o aprendizado passado e presente e realizar predições sobre o aprendizado futuro;
- O professor pode utilizar o sistema quando existir um baixo desempenho acadêmico dos alunos. Desta forma, ele pode detectar problemas no ensino e corrigi-los em tempo hábil, sem que a(s) sua(s) turma(s) seja(m) prejudicada(s);
- O professor pode utilizar o sistema ao final de um período letivo realizando uma consulta já explorada em períodos anteriores. Assim, ele pode verificar se houve alguma mudança entre o padrão de conhecimento passado e o padrão de conhecimento gerado no presente. Esta situação é importância, já que à medida que novos dados vão sendo inseridos na base de dados, novos padrões de conhecimento podem ser descobertos, tornando o antigo padrão ultrapassado ou não mais útil;
- O diretor pode usar o sistema no final de um semestre letivo para averiguar problemas de ensino em uma ou mais disciplinas. Com isso, o diretor pode, em reuniões programadas, expor tais problemas aos professores de forma que eles possam tentar solucioná-los;
- O diretor pode utilizar o sistema quando detectar um alto índice de trancamentos ou desistência. Assim, ele pode identificar o motivo de tais abandonos.

4.2.1 Aplicando a Fase de Planejamento da Estratégia no MIDAS-POETA

Os usuários do sistema MIDAS-POETA terão dois perfis: o perfil professor e o perfil diretor. No caso do perfil professor, o mesmo terá permissão para acessar os dados de acompanhamento de todas as suas disciplinas. Para o perfil diretor, o mesmo terá acesso aos dados de acompanhamento de toda a instituição.

Ainda para o perfil professor, são adotados os seguintes padrões de consulta: **(1)** Consultar uma disciplina (o professor terá acesso aos dados de acompanhamento das turmas referentes a alguma disciplina que o mesmo leciona); e **(2)** Consultar todas as suas disciplinas (o professor terá acesso aos dados de acompanhamento de todas as turmas referentes a todas as disciplinas que o mesmo leciona). Já o perfil diretor terá disponíveis os seguintes padrões de consulta: **(1)** Consultar uma disciplina (o diretor terá acesso aos dados de acompanhamento de todas as turmas referentes a uma disciplina em particular, não importando quem são os professores responsáveis); **(2)** Consultar um curso (o diretor terá acesso aos dados de acompanhamento de todas as turmas de um determinado curso); e **(3)** Consultar a instituição (o diretor terá acesso aos dados de acompanhamento de todas as turmas da instituição, não importando de qual curso e disciplina a turma faz parte).

Seguindo a fase de planejamento, definimos as informações e a modelagem do Modelo Analítico de Dados – MAD. Para definir as visões e atributos de interesse que compõem o MAD,

devemos ter em mente que o sistema deve dispor de dados sobre o acompanhamento informal, dados sobre o acompanhamento alternativo, dados demográficos, comportamentais e dados sobre avaliação. Sendo assim, esclareceremos apenas as informações do banco de dados do Portfolio-Tutor que estão disponíveis para atender e estes requisitos.

Para contemplarmos os aspectos informais do acompanhamento, necessitamos de dados sobre as interações do aluno com outros alunos e sobre as interações do aluno com os materiais didáticos disponíveis. A base de dados Portfolio-Tutor prevê o armazenamento das interações do aluno com seus colegas através de e-mails e também prevê a interação do aluno com os recursos didáticos.

Para contemplarmos os aspectos alternativos do acompanhamento, precisamos de dados sobre os itens de julgamento da avaliação autêntica. Neste sentido, o sistema Portfolio-Tutor prevê os dados necessários ao acompanhamento alternativo armazenando os resultados dos itens de avaliação para cada artefato produzido pelo aluno.

Finalizando a concepção do MAD, falta a definição dos dados demográficos, comportamentais e dados sobre avaliação do aluno. Com relação aos dados demográficos e comportamentais, o Portfolio-Tutor possui um cadastro de alunos que contém atributos como idade, sexo, região de procedência e a situação do aluno no curso (desistente, trancado, cursando). Quanto aos dados sobre avaliação, o Portfolio-Tutor armazena as atividades realizadas pelos alunos e os desempenhos em cada atividade.

Como o MAD reflete as informações de interesse relacionadas a um aluno, o modelamos, para o sistema MIDAS-POETA, conforme a Figura 2.

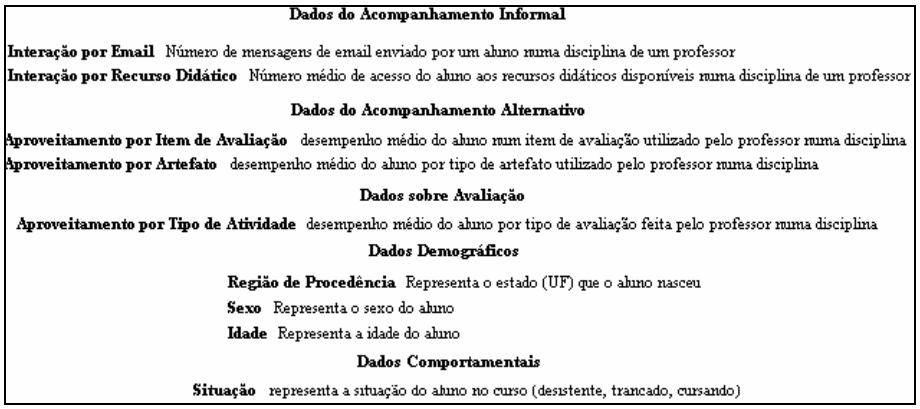


Figura 2: Modelo Analítico de Dados do MIDAS-POETA

No tocante às tarefas de mineração disponíveis no sistema, o MIDAS-POETA disponibiliza as tarefas de associação, classificação e agrupamento. A razão desta escolha foi influenciada pelas idéias de Linderman (1986), onde o mesmo atribuiu, como resultado da tomada de decisão na educação presencial, operações preditivas e classificatórias. Assim, consideramos a sua teoria como base também para sistemas de apoio a decisão baseada na modalidade de EaD.

4.2.2 Projeto e Implementação do MIDAS-POETA

Na implementação do MAD, utilizamos um mecanismo de visão tal que, para cada visão do MAD, foi criada uma visão no SGBD SQL Server 7.0. A Figura 3 ilustra as visões e os seus respectivos atributos de interesse, que estão sublinhados.

Interação por E-mail	(idAluno, idDisciplina, idProfessor, numMensagens)
Interação por Material Didático	(idAluno, idDisciplina, idProfessor, idMaterial, numAcessos)
Aproveitamento por Item de Avaliação	(idAluno, idDisciplina, idProfessor, idItem, desempenho)
Aproveitamento por Artefato	(idAluno, idDisciplina, idProfessor, idAretfato, desempenho)
Aproveitamento por Atividade	(idAluno, idDisciplina, idProfessor, idAtividade, desempenho)
Região de Procedência	(idAluno, uf)
Sexo	(idAluno, sexo)
Idade	(idAluno, idade)
Situação	(idAluno, situação)

Figura 3. Conjunto de Visões para o MAD

Para atender a fase de análise de dados da estratégia de acompanhamento e o requisito de mineração de dados interativa, utilizamos uma interface para coleta de dados de interesse - etapa de coleta de dados, uma interface para seleção da tarefa de mineração - etapa de mineração de dados, e uma interface para apresentação dos resultados - etapa de visualização e incorporação dos resultados. Com tais interfaces, o usuário tem a possibilidade de selecionar os dados para uma consulta, a tarefa de mineração e visualizar os resultados. Caso o padrão produzido não tenha sido satisfatório, ele poderá voltar no processo e recomeçá-lo.

A etapa de coleta de dados produz comandos SQL para realizar junções entre as visões selecionadas pelo usuário, gerando um *resultset* preparado para a mineração. Tal resultado deve conter apenas os atributos de interesse das visões selecionadas, podendo ser algum dos atributos sublinhados de acordo com a Figura 3.

Na implementação das tarefas de mineração de dados, foi utilizado o *framework* WEKA desenvolvido por Witten & Frank (2000). Este framework provê as implementações dos algoritmos de classificação, associação e agrupamento utilizados no sistema.

4.2.3 Um Cenário de Execução do MIDAS-POETA

A fim de ilustrar uma das potencialidades do MIDAS-POETA, mostraremos um cenário de execução do sistema com uma interação de um professor. Para tal, a base de dados foi povoada com dados fictícios da disciplina Economia Geral de um curso fictício de pós-graduação em Macroeconomia à distância.

Com o objetivo de correlacionar o aproveitamento do item de avaliação "*criatividade*" com o número de acessos aos materiais didáticos tipo "*conceito*" e a situação do aluno no curso, são selecionados os atributos (e valores) das seguintes visões: **(1)** atributo "***desempenho***" do item de avaliação "***criatividade***" da visão *Aproveitamento por Item de Avaliação*; **(2)** atributo "***numAcessos***" do material didático tipo "***conceito***" da visão *Interação por Material Didático*; e **(3)** atributo "***situação***" da visão *Situação*. A Figura 4 abaixo ilustra parte do *resultset* desta seleção.

DesempenhoCriativiadade	NumAcessosConceito	Situação
Baixo	Freqüente	Desistente
Fraco	Pouco Freqüente	Cursando
Ótimo	Regular	Desistente
Ótimo	Nenhum	Desistente
Excelente	Pouco Freqüente	Trancado
.....	.....	.....

Figura 4. Parte do Resultado da Consulta

Ao ser escolhida a tarefa de associação, são geradas regras de associação conforme abaixo:

- 80% dos alunos com desempenho Baixo no item de avaliação "criatividade" tem acesso Pouco Freqüente aos materiais didáticos "conceito";

- 80% dos alunos Desistentes apresentam desempenho Excelente no item de avaliação "criatividade"

- 80% dos alunos que Trancam , apresentam desempenho Regular no item de avaliação criatividade e tem acesso Freqüente aos materiais didáticos "conceito"

Obviamente, considerando que os dados utilizados no processo de mineração são fictícios, não cabe aqui uma discussão sobre a validade e o significado práticos do padrão de conhecimento produzido.

5 Conclusão

Acompanhar o aprendizado em EaD não é uma tarefa trivial e envolve vários fatores complicadores, por exemplo, a falta de contato face-a-face entre professores e alunos. Dada esta dificuldade, diversas estratégias e modelos computacionais foram desenvolvidos a fim de auxiliar os professores nesta tarefa.

Propõe-se neste artigo uma estratégia de desenvolvimento de um Sistema de Apoio a Decisão para o acompanhamento do aprendizado em EaD, na qual, a partir de uma base de dados que contempla diversas variáveis de acompanhamento, são utilizadas técnicas de Mineração de Dados para extrair padrões úteis e que favoreçam o processo de ensino/aprendizagem.

São pontos fortes do trabalho apresentado:

- descrição de uma estratégia para o acompanhamento do aprendizado em EaD, que permite a coleta de um conjunto de dados sobre o aluno (MAD) e possibilita a análise desses dados;
- incorporação de um recurso inovador – Mineração de Dados - para apoiar a tomada de decisões relacionadas ao acompanhamento do aprendizado;
- desenvolvimento de um ambiente de descoberta de conhecimento aplicado à educação;
- definição de um conjunto-referência de informações (o MAD do sistema MIDAS-POETA) para o acompanhamento do aprendizado em ambientes de EAD;
- definição de um mecanismo de visões que implementa o conjunto o conjunto-referência citado.

Referências Bibliográficas

ALVES, R. M., ERRICO, L., MESQUITA, R. C. **Safes: Um servidor de Avaliações Online para Ensino via Web**. II Seminário Nacional de Tecnologia para EAD. Universidade Federal de Uberlândia. Uberlândia, 2002.

BERNHARDT, V. L., **Intersections**. National Staff Development Council. Journal of Staff Development. Vol. 21. Nº 1. EUA. 2001.

JOHNSON, J. H., **Data-driven School Improvement**. ERIC Digest. ERIC Clearinghouse on Educational Management Eugene. Nº 109. EUA. 2000.

LINDERMAN, R. H., **Medidas Educacionais**. Editora Globo. 1ª Edição. Rio Grande do Sul, 1986.

MACHADO, L. S., BECKER, K., **O Uso da Mineração de Dados na Web Aplicado a um Ambiente de Ensino a Distância**. In: I Workshop de Teses e Dissertações em Banco de Dados. XIX Simpósio Brasileiro de Banco de Dados. Gramado. 2002.

MASETTO, M. T. **Mediação Pedagógica e o Uso da Tecnologia**. In: Moran, J. M., Masetto, M. T., Behrens, M. A. Novas Tecnologias e Mediação Pedagógica. Editora Papirus, Campinas, 2000.

MENEZES, R. A., FUKS, H., GARCIA, A. B. **Utilizando Agentes no Suporte à Avaliação Informal no Ambiente de Instrução Baseada na Web – AulaNet**. In: X Simpósio Brasileiro de Informática na Educação. Fortaleza. 1999.

NASCIMENTO, D. M. C., **Um Sistema Tutor Acoplado a um Portfolio Eletrônico no Contexto da Educação a Distância – PortfolioTutor**. Dissertação de Mestrado, Universidade Federal de Campina Grande, 2002.

NETO, F. J. S. L., **EAD: Regulamentação, Condições de Êxito e Perspectivas**, 1998. Disponível em www.intelecto.net/ead_textos/lobo1.htm. Acesso em Out/2002.

OTSUKA, J. L., LACHI, R. L., FERREIRA, T. B., ROCHA, H. V., **Suporte a Avaliação Formativa no Ambiente de Educação a Distância TelEduc**. In: VI Congresso Iberoamericano de Informática Educativa. Espanha. 2002.

SANTOS, N. **Web-based Education: How to Assess Students Performance?** In: Site'99. Society for Information Technology and Teacher Education International Conference. USA. 1999.

SILVA, D. R., **Um Ambiente para Descoberta de Conhecimento com Suporte de Data Warehousing e sua Aplicação para Acompanhamento do Aluno em Educação a Distância.** Dissertação de Mestrado. UFSCAR. 2002.

SOUTO, M. A., BICA, F., WARPECHOWSKI, M., VICARI, R. M., OLIVEIRA, J. P. M., ZANELLA, R., SONNTAG, A. A., **Ferramentas de Suporte a Monitoração do Aluno em um Ambiente Inteligente de Ensino na Web**. In: XII Simpósio Brasileiro de Informática na Educação. Vitória, 2001.

SOUZA, D. S., **Desafios da Gestão de Sistemas de EAD**. In: X Simpósio Brasileiro de Informática na Educação. Curitiba, 1999.

TAMINOTO, S. L., **Towards an Ontology for Alternative Assessment in Education**. Metting of IEEE Learning Technology Standards Committee, Pittsburgh, USA, 1998

VIEIRA, M., **A Avaliação na Educação a Distância: Um Estudo sobre o Curso de Contemplação para Licenciatura em Biologia, Física, Matemática e Química**. II Seminário Nacional de Tecnologia para EAD. Universidade Federal de Uberlândia. Uberlândia, 2002.

WITTEN, I. H., FRANK, E., **Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations**. Morgan Kaufmann Publishers. San Francisco, EUA, 2000.

Ferramenta Avaliativa Pedagógica para Cursos a Distância Baseada em Testes Adaptativos Informatizados e Teoria de Resposta ao Item

Fabrícia Damando Santos (UFG)
fdamando@ufgvirtual.ufg.br

Lucília Ribeiro (UFG)
lucilia@inf.ufg.br

Weber Martins (UFG/UCG)
weber@eee.ufg.br

Leonardo Guerra de Rezende Guedes (UFG/UCG)
lguedes@eee.ufg.br

Resumo: A comunicação tem contribuído para as transformações nas diversas áreas do conhecimento. A utilização da Web em ambientes educacionais expandiu-se muito e a utilização de softwares para esses novos ambientes educacionais está em crescente estudo e aprimoramento. Neste artigo, apresenta-se uma ferramenta computacional para realização de avaliação de alunos em cursos a distância, utilizando como fundamentação a teoria de resposta ao item e os testes adaptativos informatizados. O objetivo final é apresentar ao aluno questões de seu nível individual de habilidade. Experimentos foram conduzidos para avaliar o sistema proposto em duas versões. A velocidade de busca do nível de habilidade do aluno define as duas versões. Resultados promissores foram obtidos, mostrando a adequação de cada uma das versões.

Palavras Chaves: Educação a Distância, Avaliação, Teoria de Resposta ao Item, Testes Adaptativos Informatizados, informática na Educação.

1 Introdução

A avaliação é uma questão muito importante no ensino. Cada vez mais, os professores estão buscando novas práticas avaliativas, passando a contestar a avaliação como significado social e político. Evitar que a avaliação assuma uma função comparativa e classificatória é busca constante. Para cursos a distância, essa preocupação é importante. O desenvolvimento de softwares aplicados na avaliação de alunos, que forneçam informações de acompanhamento do aluno para o professor, objetiva o aprimoramento do ensino através da Internet.

A proposta apresentada é uma ferramenta computacional destinada ao professor que receberá informações sobre as habilidades dos alunos, os níveis de dificuldade das questões, acompanhamento do aluno no teste e as notas ponderadas equalizadas. Esta ferramenta é implementada a partir da Teoria de Resposta ao Item e das metodologias dos Testes Adaptativos Informatizados. A detecção e classificação de dificuldades e habilidades dos alunos servirão principalmente como instrumento de auto-avaliação para o professor.

2 Teoria de Resposta ao Item

A Teoria de Resposta ao Item é uma reunião de modelos estatísticos usados para fazer predições, estimativas ou inferências sobre as habilidades (ou competências) medidas em um teste. Através dos modelos estatísticos, é possível predizer tais habilidades por meio de correspondências

entre a pontuação obtida por um estudante em uma situação de teste e os itens a ele fornecidos [Hambleton & Swaminathan, 1985; Rudner, 1998].

A Teoria de Resposta ao Item propõe modelos que representam a relação entre a probabilidade de uma resposta certa a um item e a habilidade de um aluno, proporcionando a avaliação individual do aluno, pois, cada estudante responderá itens referentes à sua habilidade, tornando a avaliação personalizada. Os modelos de respostas fornecem diferentes conjuntos de itens em uma avaliação, permitindo comparar o desempenho entre eles e, portanto, auxiliando o professor na tarefa de avaliação. A Teoria de Resposta ao Item possui vários modelos, diferenciados pelas curvas características e forma de pontuação empregada.

Dentre os modelos propostos pela teoria, optamos por utilizar o Modelo Logístico de um Parâmetro, também conhecido como "The Rasch" [Baker, 2001]. A equação que caracteriza a probabilidade do aluno responder corretamente é mostrada abaixo.

$$P_i(\theta) = \frac{1}{1 + \exp^{-1(\theta - b_i)}}$$

onde:

$P_i(\theta)$ é a probabilidade de um determinado aluno com habilidade θ, responder a um item i corretamente;

b_i é o índice de dificuldade do item.

O parâmetro b_i pode ser alterado à medida que os estudantes passam a realizar o teste e a responder corretamente ou incorretamente ao item. Os valores para b_i variam neste modelo de -2.0 (itens fáceis) a +2.0 (itens difíceis).

A partir do modelo estudado, desenvolvemos uma nova proposta para o cálculo da pontuação obtida por um aluno e a sua habilidade.

3 Proposta de Modelos de Pontuação e Habilidade

Propomos, neste trabalho, uma nova abordagem de pontuação e habilidade do aluno. Nestas abordagens, utilizamos os conceitos de Esperança Matemática e Mediana.

3.1 Pontuação

Frequentemente, a apresentação de notas para a instituição é necessária. Nestes momentos, a pontuação é realizada a partir da medida de habilidade do aluno. No caso, propomos calcular a Esperança Matemática de acerto de uma questão de nível de dificuldade b_i sendo $P_i(\theta)$ a probabilidade de acerto da questão por um aluno com nível de habilidade θ.

$$E(x) = b_i . P_i(\theta)$$

A nota π do aluno é definida como a razão entre o desempenho obtido pelo desempenho esperado conforme seu nível de habilidade. Matematicamente, é expressa pela equação abaixo.

$$\pi = \frac{\sum_{i=1}^{n} R_i . E_i(x)}{\sum_{i=}^{n} E_i(x)} = \frac{\sum_{i=1}^{n} R_i . b_i . P_i(\theta)}{\sum_{i-1}^{n} b_i . P_i(\theta)}$$

Onde R_i representa a resposta do item i (0 ou 1). O acerto corresponde ao valor unitário enquanto o erro é registrado pelo valor nulo.

Perceba que, em termos matemáticos, o denominador representa o desempenho esperado do aluno com seu nível de habilidade. O quociente, a nota, especifica qual razão deste desempenho efetivamente ocorreu, ou seja, quanto o aluno conseguiu concretizar do que seria esperado dele. Deve-se ressaltar que o desempenho esperado pressupõe níveis contínuos de acerto e, não, apenas certo e errado (como é o caso). De qualquer modo, trata-se de uma medida individualizada, pois considera o nível de habilidade específico de cada aluno.

3.2 Habilidade

Para se estimar a habilidade inicial de um aluno, podemos utilizar um valor constante ("default"), e, a partir desse valor, ajustar a habilidade do aluno, obtida através das respostas dos itens do teste.

A habilidade do aluno é resultante da probabilidade de acerto de uma questão de determinado nível de dificuldade b_i. Um aluno com habilidade θ terá 50% de probabilidade de acerto de questões com nível de dificuldade $b_i = \theta$.

Assim, devemos encontrar qual nível de dificuldade de questão b_i o aluno terá 50% de chances de acerto, que será realizada pela busca da mediana das questões aplicadas. Em outras palavras, o nível de habilidade procurado divide o conjunto de questões (ordenado por nível de dificuldade) em duas metades.

Desenvolvemos uma nova fórmula para cálculo da habilidade dos alunos. Para a Teoria de Resposta ao Item, a habilidade varia de -2.0 a +2.0. Como no nosso caso o índice de dificuldade de um item pode variar de 0 a 10, ajustamos a fórmula da habilidade para:

$$b_i = (b_{i'} / \ 2{,}5) - 2$$

onde:

b_i é o índice de dificuldades dos itens;

$b_{i'}$ é o nosso valor do nível de dificuldade (0 a 10).

Para calcularmos a habilidade θ, trabalhamos com o valor mediano dos itens aplicados no teste, levando-nos a um valor central.

$$\theta = [(med_{niveis} / 2{,}5) - 2]$$

Portanto, aplicando o valor da habilidade no Modelo The Rash, encontra-se a probabilidade de um aluno responder corretamente um item dado a sua habilidade.

4 Modelos Desenvolvidos

Foram desenvolvidos dois modelos avaliativos que podem ser usados em cursos a distância. Cada modelo possui as suas peculiaridades e junto a esses modelos desenvolvemos uma Ferramenta Computacional para apoio pedagógico que auxilia o professor a identificar as habilidades dos alunos, realiza o acompanhamento dos alunos no curso e das questões relacionadas ao teste.

Para o desenvolvimento dos modelos avaliativos, utilizamos um banco de itens (questões) com vários níveis de dificuldade, variando de 0 (mais fáceis) a 10 (mais difíceis). As questões são

cadastradas pelo próprio professor, composta por um código, descrição da pergunta, cinco alternativas, o nível de dificuldade inicial da questão e o gabarito (resposta correta).

4.1 Administração de Itens no Teste

Todos os alunos iniciam o teste com uma questão de mesmo nível de dificuldade. No caso, iniciamos um teste com uma pergunta de nível de dificuldade 5 (dificuldade intermediária). Tal habilidade irá se ajustar no decorrer do teste.

O aluno é submetido ao teste, onde responde questões que revelam seu conhecimento, retratado pela sua habilidade. No ***Modelo Avaliativo I (conservador)***, quando o aluno acerta um item, o próximo item a ser administrado tem nível de dificuldade acrescido de uma unidade. Se errar, o nível é diminuído em uma unidade. Obviamente, há de se respeitar os limites superior (10) e inferior (0). Um exemplo da estratégia de seleção de itens do teste aplicado com o Modelo Avaliativo I é mostrado abaixo (ver Figura 1).

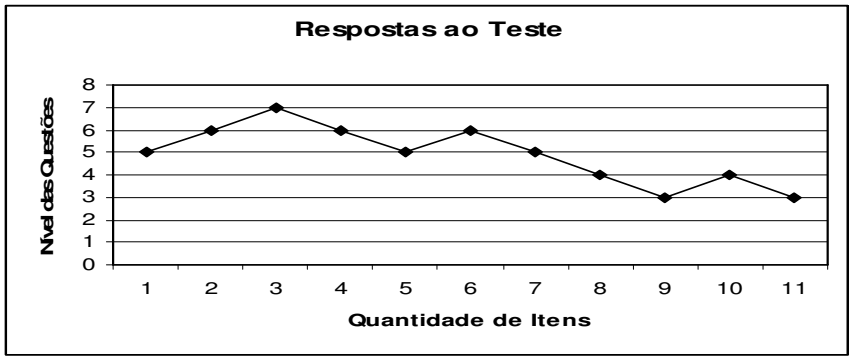


Figura 1: Registro de resposta do aluno no Modelo Avaliativo I.

Quando dois alunos fazem o teste simultaneamente, provavelmente não respondem as mesmas questões, tendo em vista que são escolhidas de acordo com a habilidade de cada aluno.

O ***Modelo Avaliativo II (adaptação rápida)*** possui a mesma dinâmica do modelo anterior. A diferença está na forma com a qual o nível da próxima questão é definido. O segundo modelo busca com mais rapidez sintonizar-se no nível do aluno. Ao contrário do incremento/decremento unitário constante, o segundo modelo adota estratégias diferentes para o acerto e erro. Quando ocorre acerto, o próximo nível é definido pela média do nível corrente e o limite superior (10) (ver equação abaixo).

$$n_{novo} = \frac{n_{atual} + n_{máx}}{2}$$

Na presença de erro, temos duas situações. Quando se trata de erro isolado (“primeiro erro”), usamos a média entre o nível corrente e o nível do item anterior (ver Equação 4a). Quando se trata de erro não isolado (tendo ocorrido erro no item anterior), a média entre o nível atual e o limite inferior (0) é empregada (Equação 4b). Na necessidade de converter valores fracionários para inteiros (3,5, por exemplo), tendo em vista que os níveis são números inteiros, leva-se para o nível imediatamente superior (4, neste mesmo exemplo).

$$n_{novo} = \frac{n_{atual} + n_{anterior}}{2} \quad (4a) \qquad n_{novo} = \frac{n_{atual} + n_{mín}}{2} \quad (4b)$$

A Figura 2 mostra uma das execuções do segundo modelo. Convém registrar que se tratando das mesmas questões utilizadas no primeiro modelo (ou melhor, extraídas do mesmo banco de questões). A comparação entre as figuras 1 e 2 esclarece a maior velocidade na busca do nível de habilidade do aluno apresentada no segundo modelo, onde saltos maiores são empregados.

Figura 2: Registro de resposta do aluno no Modelo Avaliativo II.

4.2 Variação do Nível de Dificuldade da Questão

À medida que as questões são respondidas, os níveis de dificuldade variam, caracterizando os modelos desenvolvidos como auto-adaptativos. Quanto mais alunos acertarem uma determinada questão, ela é considerada mais fácil e conseqüentemente seu nível de dificuldade diminui. Por outro lado, quanto mais alunos responderem incorretamente uma questão, a questão é considerada mais difícil, elevando seu nível. O novo nível da questão é calculado através da seguinte equação:

$$nível = \left\{10 - \left[\left(\frac{num\_acertos}{num\_acessos}\right) * 10\right]\right\}$$

5 Ferramenta Computacional

Programamos para aos Modelos Avaliativos I e II, uma ferramenta computacional para apoio pedagógico que, através dos testes realizados com os alunos, auxilia o professor a identificar as habilidades dos alunos, fazer acompanhamento dos mesmos e o acompanhamento dos níveis de dificuldade das questões. A ferramenta foi desenvolvida em linguagem ASP® (Active Server Pages), interface HTML, sendo a validação dos campos realizada através de JavaScript e do banco de dados Access®.

A ferramenta oferece a edição de questões, a correção dos testes aplicados aos alunos e posterior divulgação dos resultados, podendo ser utilizado pelo administrador do sistema, pelo professor e pelo aluno.

5.1 Manipulação das Questões

O professor é responsável pelo cadastro, alteração e exclusão de questões. O aluno é cadastrado no sistema pelo administrador, recebe uma senha para utilizar a ferramenta. Quando o aluno entra no sistema para realizar o teste, escolhe entre o Questionário I (Modelo Avaliativo I) ou o Questionário II (Modelo Avaliativo II). Após tal escolha, as questões são administradas, estabelecendo os modelos de busca descritos anteriormente.

No momento da realização do teste, é mostrado um formulário ao aluno com o número da questão, o nome do aluno, o enunciado e as opções de respostas, além do botão "confirmar resposta". Após o aluno escolher sua resposta, o próprio sistema faz a correção e armazena as informações no banco de dados. Com os dados armazenados, tanto para professor quanto para o aluno, o relatório torna-se de fácil manipulação.

5.2 Apresentação dos Resultados (Relatório)

Ao final do teste, um relatório instantâneo é mostrado ao aluno, onde as questões que foram respondidas incorretamente são listadas. Neste momento, o aluno pode direcionar o estudo para as questões e assuntos de maior dificuldade, mostrando que o relatório concretiza o apoio à aprendizagem.

Para o professor, os resultados se mostram mais robustos. O professor visualiza o desempenho de cada aluno ou da turma, em forma de nota, listagem das questões respondidas pelos alunos e as "notas" sugeridas. O relatório mostra ainda a mediana, a habilidade do aluno e média de todas as notas dos alunos no teste. A ferramenta foi desenvolvida para oferecer informações tanto para o professor quanto para o aluno, simplificar o trabalho de correção de questões, fornecendo apoio ao professor.

A forma de medir a habilidade do aluno contribui para identificar suas dificuldades e facilidades, fazendo com que, dentro de seus limites, qualquer aluno consiga realizar o teste, que será ajustado ao seu perfil.

6 Análise dos resultados

Para validação da ferramenta desenvolvida, desenvolvemos um teste em Inglês para os modelos avaliativos I e II. Cadastramos 186 questões com nível de dificuldade variando de zero a dez. Para cada modelo, tivemos dez alunos realizando o teste. Optamos por um único grupo de questões, pois as comparações entre os modelos tornam-se mais precisas.

Como mostrado na Figura 3, observa-se que 60% dos alunos obtiveram notas mais altas no Modelo Avaliativo II (Questionário II). O desempenho quanto às habilidades (ver Figura 4) correspondeu às definições anteriores. Para evitar efeitos de ordem de apresentação, alternou-se a aplicação dos questionários de aluno para aluno.

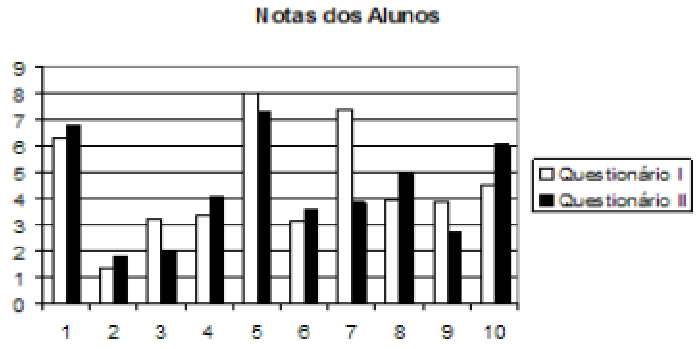


Figura 3 - Descrição das notas obtidas pelos alunos nos dois questionários.

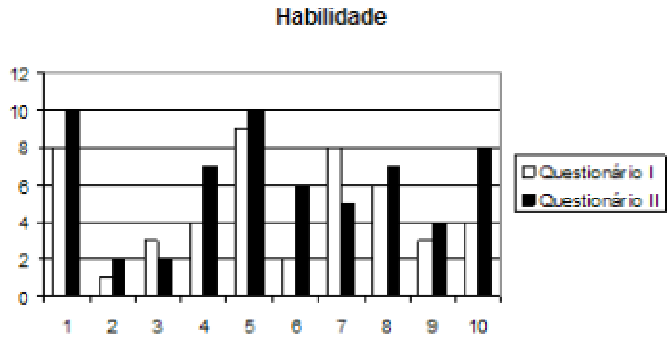


Figura 4 – Habilidade dos Alunos

Podemos afirmar que os alunos estão aptos a responderem questões com níveis de dificuldade próxima ao de suas habilidades. Com a habilidade do aluno é possível desenvolver testes individualizados para cada um, auxiliando o processo didático-pedagógico.

O tempo médio gasto pelos alunos para responder cada questionário foi de aproximadamente 11,5 minutos. A maioria das questões respondidas apresentou nível de dificuldade 5 (seguidas pelas de nível 4) e as menos respondidas foram as de nível de dificuldade 9 (seguidas pelas de nível zero).

Um fator interessante observado foi o quesito “facilidade do modelo”, ou seja, a manifestação de qual dos modelos foi percebido como o mais fácil (Questionário I ou Questionário II). Todos os alunos concordaram em que o questionário mais fácil era aquele em que ele realizava primeiro. Inferimos, portanto, a fadiga como fator associado ao quesito “dificuldade do modelo”.

Antes dos alunos responderem aos questionários, foi perguntado a eles qual o conhecimento admitido por eles em Inglês. Cerca de 50% dos alunos admitiram ter conhecimento médio, 40% básico e 10% avançado. Ao final do teste, perguntamos qual a opinião sobre a interface da ferramenta. Todos responderam ser de fácil manuseio.

6.1 Curvas

Para melhor visualização e análise dos dados obtidos, selecionamos alguns alunos e suas respectivas curvas dos níveis de dificuldade calculados.

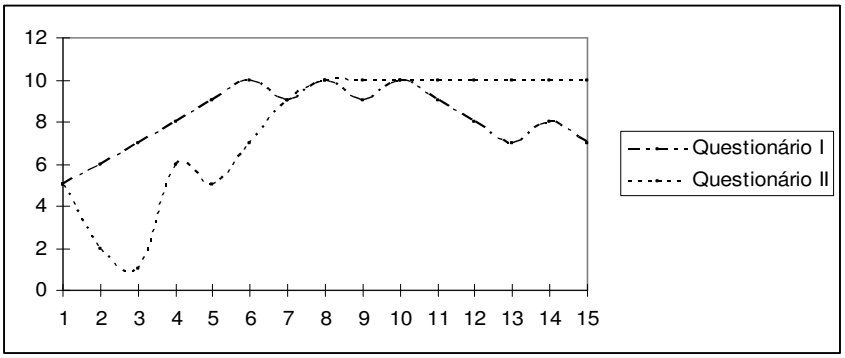


Gráfico 1- Curvas das habilidades para o Aluno X

Nota-se que, no caso do aluno X (ver Gráfico 1), o Modelo Avaliativo I (Questionário I) apresenta aumentos gradativos dos níveis de dificuldade das questões até o aluno alcançar o nível mais alto. Somente depois encontramos variações alternadas para os níveis e posterior queda gradativa. Para o Modelo Avaliativo II (Questionário II), há variabilidade maior nos níveis inicialmente e logo depois ocorre estabilização.

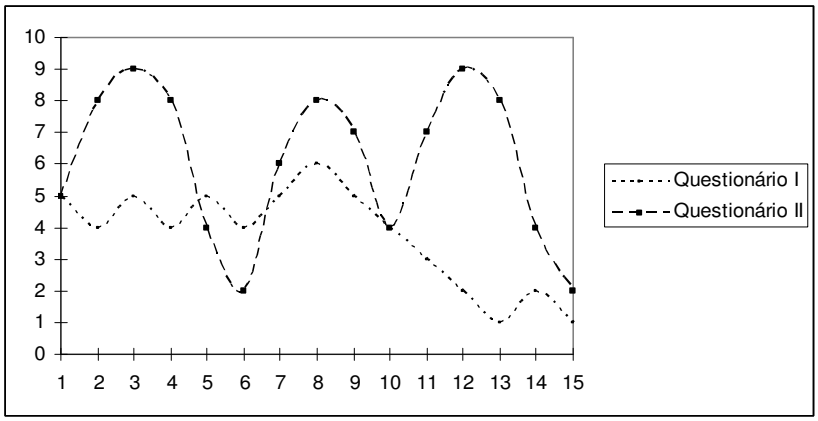


Gráfico 2- Curvas das habilidades para o Aluno Y

No caso do aluno Y (ver Gráfico 2), fica claro que houve maior variabilidade no Modelo Avaliativo II (Questionário II), porém, sempre que houve a apresentação de questões com níveis de dificuldade muito altos, o aluno as respondeu incorretamente e, em seguida, foi administrado uma questão de nível de dificuldade menor. O Modelo Avaliativo I (Questionário I) apresentou-se

mais estável no início, porém seguidos de erros consecutivos. Cabe salientar que o Questionário II foi respondido primeiro para este aluno.

De modelo geral, observamos que o Modelo Avaliativo I não submeteu os alunos a uma grande variabilidade de níveis de dificuldade das questões, apresentando-se mais estável e desafiando o aluno em seu nível de habilidade. Já para o modelo avaliativo II, houve uma maior variabilidade de nível de dificuldade, potencializando o acerto.

No Modelo Avaliativo I, os níveis de dificuldade variam menos, e pode ser bem aplicado para alunos onde as suas habilidades são intermediárias (nem muito alta e nem muito baixa), pois o nível de uma questão varia gradativamente. Porém, nos casos com alunos de habilidades altas, a mudança de nível é menos rápida, o que leva ao aluno responder questões nas quais ele já possui conhecimento, sendo consideradas fáceis.

Para o Modelo Avaliativo II, chega-se mais rápido à habilidade do aluno nos casos em que o aluno tem conhecimento avançado ou básico. Se o aluno possui habilidade alta, o modelo II administrará questões de maior nível de dificuldade de forma mais rápida, pois sempre trabalha com a média dos níveis das questões (atual e máxima). Caso o aluno possua uma habilidade baixa, o modelo II também direcionará de forma mais rápida a questões de nível menor.

Quanto aos níveis de dificuldade das questões, eles variaram no decorrer do teste, tornando o banco de itens mais calibrado, ou seja, questões com níveis de dificuldade "reais" para um aluno ou turma. Especificamente, 44% das questões aumentaram o nível de dificuldade e 42% diminuíram o seu nível de dificuldade, o que realmente demonstra que estão se ajustando ao longo do tempo.

6.2 Cálculo da Correlação

Realizamos o cálculo da correlação entre as habilidades e notas obtidas pelos alunos nos Questionário I e II. Conseguimos o valor de 0,94 para o Questionário I e de 0,97 para o Questionário II, o que demonstra matematicamente resultados satisfatórios. A planilha abaixo pode esclarecer melhor os resultados encontrados.

Aluno	Habilidade Modelo I Questionário I	Nota Modelo I	Habilidade Modelo II Questionário II	Nota Modelo II
A	8	6,29	10	6,81
B	1	1,36	2	1,76
C	3	3,22	2	1,96
D	4	3,38	7	4,11
E	9	8,03	10	7,26
F	2	3,16	6	3,6
G	8	7,38	5	3,86
H	6	3,97	7	5
I	3	3,85	4	2,72
J	4	4,49	8	6,09

7 Conclusão

A ferramenta desenvolvida oferece apoio ao professor, mostrando resultados relevantes para decisões de o quê ensinar. Pode ser utilizada para acompanhamento do aluno no curso, auxiliando

o professor a diagnosticar o desempenho do aluno e da turma, assim como sua concepção sobre dificuldade das questões presentes no banco de itens.

Quanto aos dois modelos propostos, observou-se que o modelo com menos variações (Modelo I) pode ser bem aplicado a todos os alunos e principalmente àqueles com nível de habilidade média, pois vai testando o aluno nível a nível. Para os casos de alunos avançados, o Modelo II se apresenta mais favorável, pois faz com que o aluno responda questões com níveis de dificuldade mais compatíveis com a sua habilidade, o que pode favorecê-lo no sentido de manter seu interesse.

Referências

Baker, F. B. **The Basis of Item Response Theory. Second Edition**. ERIC Clearinghouse on Assessment and Evaluation, 2001.

Hambleton, R. . and Swaminathan, H. **Response Theory: Principles an Application**. Kluwer Publishing, 1985.

Rudner, M. L. **An On-Line, interative, computer adaptive testing mini-tutorial**, 1998. Disponível em:< http://www.ericae.net/scripts/cat/catdemo.html>. Acesso: Mai, 2002.

Simulação para Ensino de Conceitos da Orientação a Objetos

Mariane Fogaça Galhardo (FACENS)
marianefg@yahoo.com.br

Luciana Aparecida Martinez Zaina (FACENS)
luciana@facens.br

Resumo. Existem conteúdos que apresentam graus de dificuldades de assimilação diferentes. Na área de programação de computadores, como por exemplo, no paradigma orientado a objetos, a falta de prática pode trazer dificuldades durante o processo de aprendizagem da mesma. Utilizar uma ferramenta como auxílio ao ensino nesta área pode trazer grandes contribuições.

Este artigo tem como objetivo apresentar uma ferramenta que simula os conceitos da Orientação a Objetos permitindo a interação do aluno durante o processo de aprendizagem.

Palavras-chave: Simulação, Tecnologia para Ensino, Programação Orientada a Objetos.

1. Introdução

A velocidade com que as mudanças tecnológicas vêm acontecendo atualmente e a necessidade de se estar cada vez mais especializado no mercado de trabalho, vêm mudando e aumentando a importância do uso da tecnologia na educação já há algum tempo, principalmente no ensino de linguagem de programação. A tecnologia auxilia na capacidade de transpor os conceitos estudados para uma linguagem de programação de forma prática.

Além disso, a necessidade de receber estímulos durante o processo de aprendizagem é evidente nos alunos de um modo geral. Estas necessidades podem surgir por diversos motivos como a falta de tempo, dificuldades na compreensão de assuntos complexos, entre outros. Por isso, muitos docentes têm utilizado recursos tecnológicos para tornar suas aulas mais motivadoras e interativas, proporcionando maiores vantagens educacionais para o professor, o aluno e também para a instituição de ensino de um modo geral.

Entretanto, não basta só a existência do recurso tecnológico para que o processo de ensino-aprendizagem ocorra de maneira satisfatória. É necessária a participação do aluno e do docente durante este processo para que o mesmo apresente resultados gratificantes.

Dentre os vários recursos tecnológicos utilizados, o computador tem o seu lugar reservado, pois possibilita, por exemplo, simulações de situações próximas da realidade.

A simulação é muito utilizada em várias áreas de ensino, não só na informática. Durante este trabalho a simulação será apresentada como ferramenta de apoio para alunos iniciantes em linguagem de programação, mais especificamente, alunos iniciantes na área de Programação Orientada a Objetos.

A disponibilização da ferramenta será em uma plataforma que permitirá sua utilização através da Web, possibilitará ao aluno visualizar como se constrói uma classe usando técnicas da orientação a objetos, sem a necessidade de um conhecimento prévio de nenhuma linguagem de programação deste paradigma.

Através de algumas interações, o aluno informará os dados solicitados pelo simulador, o sistema transportará estes dados traduzindo-os para um formato de uma linguagem orientada a objetos: a linguagem Java. Assim, o aluno poderá visualizar como o código é gerado no formato

de uma linguagem específica através da tradução das informações que ele determinou de maneira abstrata.

2. A utilização da Tecnologia no Ensino

Apenas a existência do recurso tecnológico para o que processo de ensino-aprendizagem ocorra de maneira satisfatória não é o suficiente, pois também é necessária a participação do aluno e do docente durante este processo para que o mesmo apresente resultados gratificantes, segundo Guimarães (2003).

Quando alunos testam dados criados por eles mesmos, recebem resultados diferentes e podem com isso descobrir novos comportamentos, gerar erros ou apenas confirmar o que foi exposto durante aulas teóricas.

Os alunos desenvolvem habilidades cognitivas. Mas o mais interessante de tudo isso é que através desses dados aleatórios de cada aluno, todos aprendem mais, inclusive o professor; e a interação da classe como um todo aumenta, segundo Bridgeman (2000).

Assim, o importante não é apenas o uso da tecnologia, mas sim a aplicação sistemática e ordenada da mesma de forma a obter um resultado bem sucedido.

3. Ensino da Linguagem de Programação

O ensino de linguagens de programação às vezes apresenta conceitos difíceis de se entender, quando vistos em sala de aula, por serem muito abstratos.

Alunos iniciantes nesta área apresentam dificuldades durante o aprendizado de uma linguagem de programação. O domínio da lógica de programação se torna necessário para facilitar a assimilação do abstrato.

O fluxograma é uma ferramenta muito utilizada para auxiliar na compreensão da lógica de programação, pois utiliza-se de objetos gráficos para representar comandos.

No entanto, os fluxogramas não possibilitam que a lógica seja executada em um computador, necessitando com isso que o aluno imagine a execução do programa, segundo Silveira-Esmin (2003). Além disto, um fluxograma fica distante da real construção de um código.

Uma ferramenta de apoio possibilita ao aluno visualizar conceitos abstratos de forma prática e transparente e, conseqüentemente, torna o ensino mais proveitoso.

4. Simulação: Um apoio importante para o ensino

Silveira-Esmin (2003) mostra que determinadas disciplinas apresentam conteúdo de difícil assimilação, por serem constituídas de assuntos abstratos, complexos ou até mesmo cansativos.

Utilizar a simulação como apoio para tornar a aprendizagem mais dinâmica e atrativa pode facilitar a visualização de conceitos complexos além de permitir o entendimento de conceitos abstratos através de exemplos concretos, segundo Schimiguel-Araújo-Amaral (2003).

Schimiguel-Araújo-Amaral mostra que simulações podem ser feitas utilizando-se vídeos, animações e imagens em movimento. Esse recurso tecnológico pode ser utilizado tanto em aulas presenciais como a distância, ou em sistemas híbridos (mesclando aulas presenciais e a distância).

5. Ferramenta de Apoio

A partir das dificuldades encontradas na transição do ensino da programação imperativa para o ensino orientado a objetos decidiu-se desenvolver uma ferramenta que apoiasse os alunos

iniciantes neste novo paradigma. A ferramenta tem como foco principal o apoio aos conceitos estudados durante as aulas presenciais de um curso de Engenharia da Computação.

A ferramenta é constituída por páginas HTML que explicam e exemplificam os conceitos fundamentais da orientação a objetos e um simulador para transformação dos conceitos abstratos em concreto.

5.1. Definição dos Conceitos Fundamentais

A ferramenta tem como função explicar teoricamente os conceitos da Programação Orientada a Objetos, exemplificando de maneira simples e de fácil entendimento estes mesmos conceitos ao longo das páginas HTML, conforme pode ser observado na Figura 1. Estes conceitos dão suporte para o aluno interagir com a ferramenta, resolvendo os exercícios propostos e construindo seus próprios códigos.

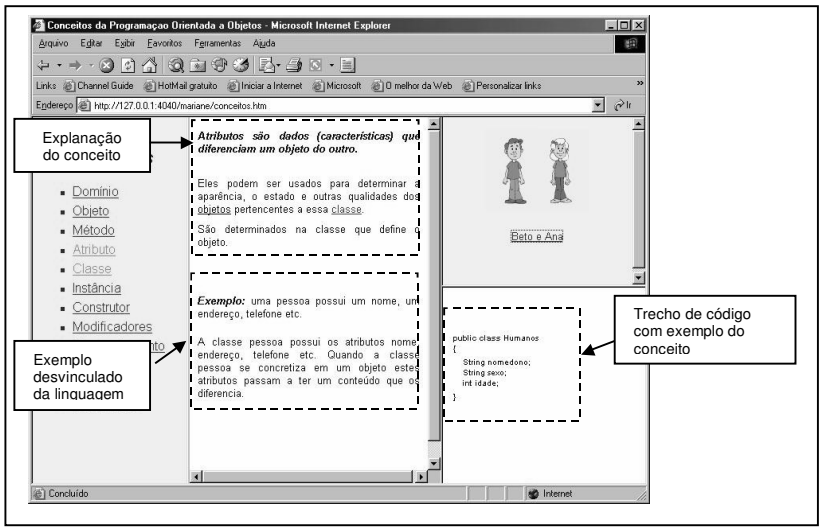


Figura 1: Apresenta o conceito de atributo e seus respectivos exemplos

O protótipo apresenta uma lista de conceitos básicos sobre orientação a objetos que darão suporte ao aluno para interagir com a ferramenta.

5.2. Resolução de Exercícios

O simulador receberá um enunciado do professor, que poderá ser salvo de forma a poder utilizá-lo mais vezes (Figura 2). O aluno, por sua vez irá buscar o enunciado proposto e salvo pelo professor (Figura 3).

É importante salientar que não houve a preocupação na distinção entre o usuário professor e aluno, através de mecanismos de controle de autenticação de senhas, tendo visto que este não era o objetivo principal do projeto. Este controle será implementado na nova fase da ferramenta.

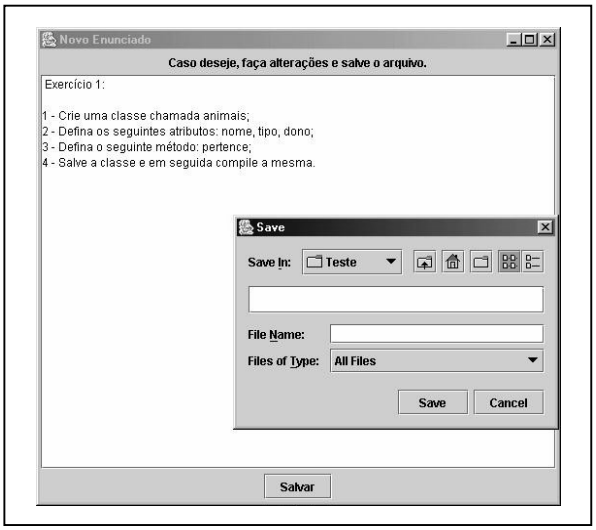


Figura 2: Cadastro dos enunciados de exercícios propostos pelo professor

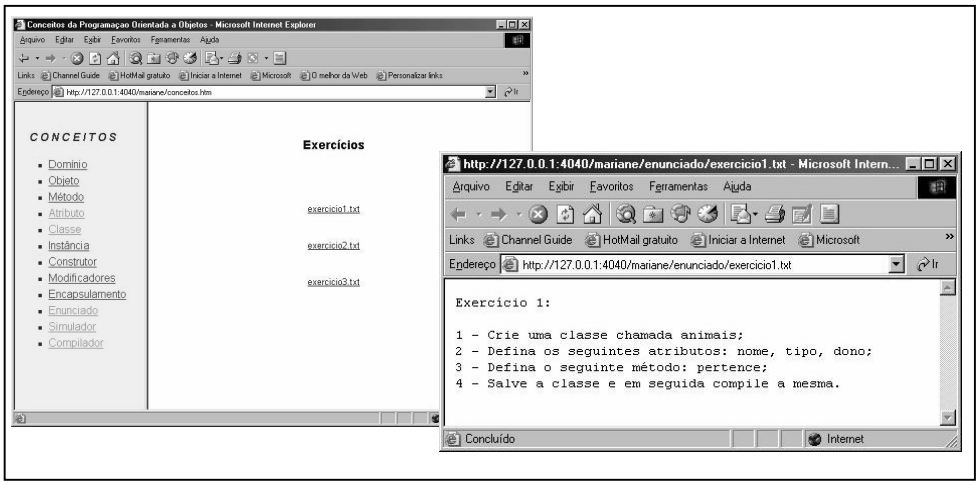


Figura 3: Enunciados dos exercícios disponíveis aos alunos

Após verificar o enunciado através do link “Enunciado” o aluno deve acessar o link “Simulador” para iniciar o processo de resolução.

Durante o processo de resolução do exercício o aluno fornecerá, gradativamente, todos os dados necessários para construção do código-solução, que partem das características gerais da classe, passam pela definição dos atributos e métodos e terminam com a definição dos valores iniciais dos atributos para serem utilizados pelo construtor da classe. O percurso percorrido pela ferramenta é apresentado na Figura 4.

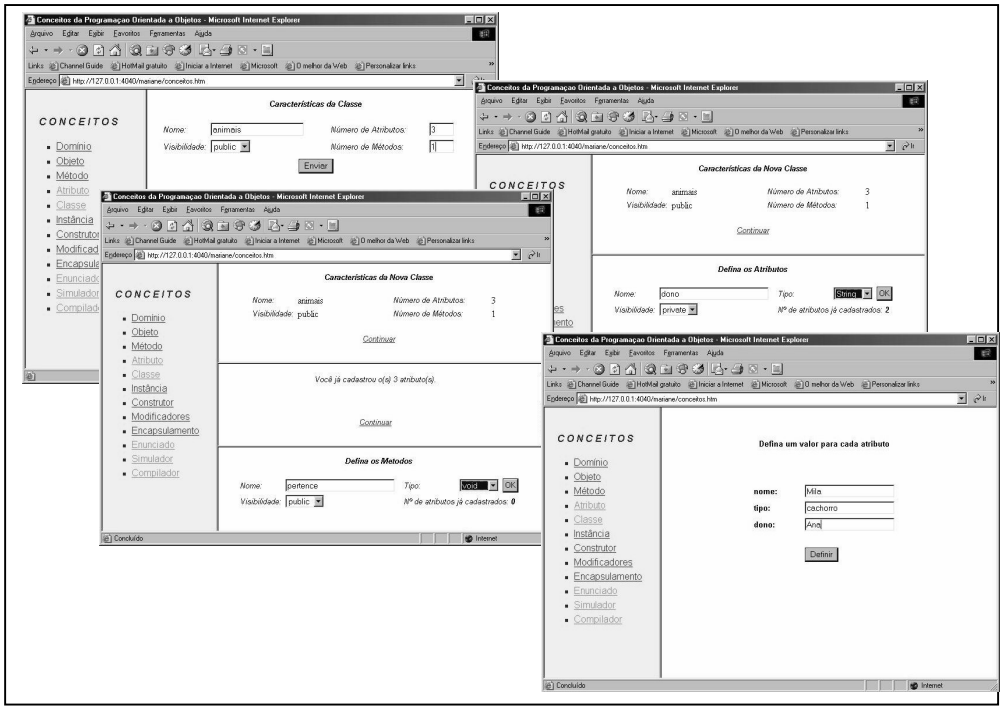


Figura 4: Construção da solução do exercício passo a passo

Quando o aluno terminar de fornecer todos os dados solicitados para a resolução do exercício, a ferramenta irá gerar para o aluno o código correspondente aos dados que ele informou em Java.

Esse código poderá ser visualizado e, se o aluno quiser poderá alterar o mesmo. A classe gerada pode ser salva no diretório desejado (Figura 5).

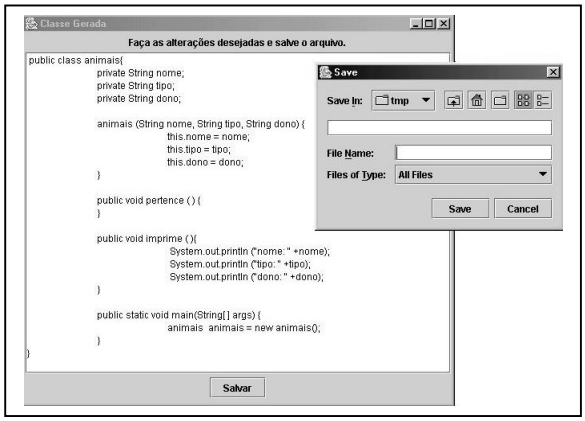


Figura 5: Código gerado pela ferramenta que pode ser armazenado pelo aluno em qualquer pasta

Após ter construído a solução para o exercício o aluno terá a possibilidade de analisar o código gerado e comparar o mesmo com as definições e exemplos dos conceitos da orientação a objetos disponibilizados na ferramenta. Desta forma, ele pode realizar um paralelo do abstrato (conceitos) para o concreto (código gerado).

O resultado obtido é de grande valia ao aluno, pois mostra que a partir de dados abstratos por ele informados, um código numa linguagem orientada a objetos foi gerado. Ressaltando dessa forma a importância do estudante ter bem sedimentado os conceitos da programação orientada a objetos antes de construir seus códigos. O entendimento das conceituações passa a ser fundamental para a geração do resultado final.

Além disto, fazer a transcrição de abstrações para a geração concreta de um código é muito valorosa para que o aluno entenda o processo de construção que deve ser utilizado para a resolução de problemas.

5.3. Compilando os Códigos Gerados pela Ferramenta

Caso o aluno tenha feito alguma modificação na classe, possíveis erros poderão ser detectados através da compilação do código gerado (Figura 6). Recurso este também disponibilizado pela ferramenta, que irá apresentar ao final da compilação um relatório de erros ou uma mensagem de sucesso da compilação.

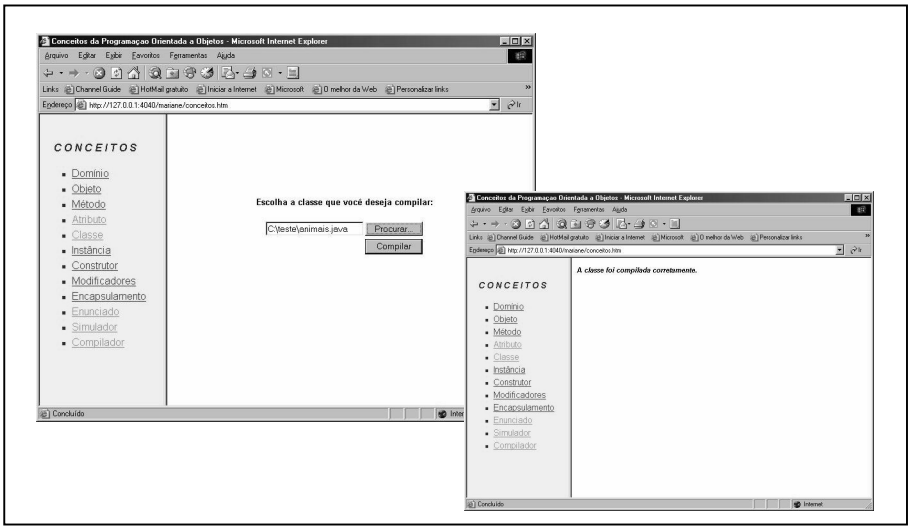


Figura 6: Compilação do código gerado pela ferramenta

6. Conclusões

O aluno que está iniciando em curso de programação normalmente apresenta maior afinidade com o paradigma imperativo. Isto porque o mesmo se aproxima mais dos processos lógicos utilizados na resolução de problemas.

Desta forma, é importante apresentar aos alunos, diferentes alternativas que auxiliem a aprendizagem de um novo paradigma de programação.

O uso da tecnologia no ensino abre espaço para um novo cenário que enriquece o processo de ensino-aprendizagem. Nesse novo cenário professor e aluno devem interagir, participando dinamicamente do processo, construindo juntos o caminho da aprendizagem.

O aluno passa a não ser um objeto estático como antigamente e o professor necessita abordar os conceitos de uma forma mais dinâmica.

Através de um simulador é possível se verificar a mudança de comportamento de ambos os lados, pois esta ferramenta proporciona um relacionamento mais interativo entre aluno e professor. Além de estimular o aluno a aprender, auxilia o professor no processo de ensino promovendo um crescimento pessoal para ambos.

O protótipo apresentado neste trabalho prioriza que os esforços dos alunos iniciantes na programação orientada a objetos sejam concentrados nos conceitos fundamentais durante o estudo. Por isso, a ferramenta disponibiliza os conceitos iniciais e necessários ao aluno para desenvolver códigos modelados em uma linguagem orientada a objetos.

Uma das vantagens da ferramenta é o aluno não precisar aprender a sintaxe de uma linguagem em específico para criar uma classe, será necessário apenas o conhecimento dos conceitos básicos da programação orientada a objetos para que ele possa preencher as características solicitadas e ter como retorno sua classe criada em uma determinada linguagem. Tal aplicação permite que o aluno visualize aquilo que foi estudado, fortalecendo assim os alicerces que irão sustentar sua aprendizagem no assunto.

Referências Bibliográficas

BRIDGEMAN, S.; GOODRICH, M.T.; KOBOUROV, S.G.; TAMASSIA, R.; **PILOT: An Interactive Tool for Learning and Grading:** SIGCSE Bulletin Conference Proceedings – The 31° SIGCSE Technical Symposium on Computer Science Edutcation. 2000. 5f.

CORTELAZZO, I.B.C.; **Redes de Comunicação e Educação: Mudanças no Paradigma:** Revista Brasileira de Aprendizagem Aberta e a Distância.

COUTINHO, L.M.; **Educação à Distância: Algumas Considerações:** Revista Brasileira de Aprendizagem Aberta e a Distância. 2003.

GUIMARÃES, A.S.; **Novo Ecossistema Cognitivo: Pensamentos sobre Tecnologia de Informação e Comunicação e a metamorfose do Aprender:** Revista Brasileira de Aprendizagem Aberta e a Distância. 2003.

KORHONEN, A.; MALMI, L.; **Algorithm Simulation with Automatic Assessment:** SIGCSE Bulletin – The 5º Annual SISCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education. 2000. 4f.

MATTA, A.E.R.; **Comunidades em redes de computadores: abordagem para a Educação a Distância – EAD acessível a todos:** Revista Brasileira de Aprendizagem Aberta e a Distância. 2003.

OLIVEIRA, I.C.A.; ARAÚJO, C.F.; LÉDON, M.F.P.; **Objetos de Aprendizagem para o Ensino de Programação em Engenharia e Computação:** 3rd International Conference on Engineering and Computer Education. São Paulo, 2003. 5f.

PIERSON, W.C.; RODGER, S.H.; **WEB-based Animation of Data Structures Using JAWAA:** 2001. 5f.

RODRIGUEZ, M.I.; **Para tratar EAD com o devido Respeito:** Revista Brasileira de Aprendizagem Aberta e a Distância. 2003.

ROMISZOWSKI, A.; **E-learning: a diferença entre teoria e realidade:** Revista Brasileira de Aprendizagem Aberta e a Distância. 2003.

SANTOS, A.I.; **A Eficácia dos Fóruns de Discussão na Aprendizagem On-Line:** Revista Brasileira de Aprendizagem Aberta e a Distância. 2003.

SCHIMIGUEL, J.; ARAÚJO, C.F.; AMARAL, L.H.; **Desenvolvimento de Simulações para o Aprendizado em cursos na Web:** 3rd International Conference on Engineering and Computer Education. São Paulo, 2003. 5f.

SILVEIRA, I.J.; ESMIN, A.A.A. **AVA – Um ambiente Visual para a construção de algoritmos:** 3rd International Conference on Engineering and Computer Education. São Paulo, 2003. 6f.

STERN, L.; SONDERGAARD, H.; NAISH, L.; **A Strategy for Managing Content Complexity in Algorithm Animation:** SIGCSE Bulletin – The 4º Annual SISCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education. 1999. 4f.

TORI, R.; **O Virtual que marca Presença:** Revista Brasileira de Aprendizagem Aberta e a Distância. 2003.

Agentes Móveis Inteligentes no Suporte à Ubiqüidade dos Serviços de Telecomunicações

Juliana Oliveira de Carvalho (FIMES-GO)
juliana@fimes.edu.br

Luis Fernando Faina (UFU-MG)
faina@facom.ufu.br

Resumo. Como resultado da convergência dos mercados de telecomunicações, computação e entretenimento, a portabilidade e adaptabilidade dos serviços decorrentes da mobilidade pessoal têm sido objeto de pesquisa de vários organismos. Neste contexto, VHE (*Virtual Home Environment*) trata da portabilidade de serviços adaptáveis através dos limites das redes. Neste artigo, discutimos não só aspectos da liberdade de mobilidade em larga escala, mas também propomos uma arquitetura baseada nos paradigmas de objetos distribuídos e agentes móveis inteligentes no suporte ao VHE. De modo a avaliar a arquitetura proposta, implementamos um protótipo onde os aspectos de mobilidade e de adaptação do serviço são analisados.

Palavras-chave: Agentes Móveis Inteligente, CORBA, JESS, Telecomunicações

1 Introdução

Nos últimos anos, o mundo tem sido palco de grandes mudanças, tais como: a revolução na comunicação e nos computadores, a ubiqüidade da Internet, o uso cada vez maior da computação móvel e o aumento na banda de rede para valores onde serviços multimídia efetivos são agora viáveis. Essas mudanças nos conduzem a um futuro onde a comunicação estará em todo lugar, cada um de nós poderá estar conectado a infra-estrutura de telecomunicações global onde quer que estejamos e pequenos dispositivos farão parte da infra-estrutura de comunicação/computação e a informação será digital e acessível através da rede[3].

Neste contexto, além da desregulamentação do setor, o crescimento do comércio internacional de equipamentos e serviços tem contribuído substancialmente para o seu crescimento e gerado alguns ganhos econômicos, dentre eles serviços contemplando principalmente comunicação multimídia e interatividade.

Em paralelo, a convergência das telecomunicações com as redes de computadores está criando uma nova sinergia, muito evidente no crescimento da Internet, que continua a dobrar em tamanho a cada ano. Ela constituirá o núcleo dos futuros serviços, isto significa, que poderemos estarmos constantemente on-line: mensagens de e-mail com arquivos em anexos serão instantaneamente recebidas no terminais móveis, mediante a um simples botão estaremos ligados à rede de nossa empresa.

Estamos interessados em um ambiente de comunicação integrado que permita o acesso a um dado conjunto de serviços a partir de diferentes localizações e dispositivos, salvaguardando naturalmente a habilidade de cada dispositivo em tratar a informação em trânsito (p. ex., vídeo, áudio, *mail* e mensagem de *fax*)[4]. Nesse contexto, usuários serão providos de acesso a serviços adaptados às suas necessidades por meio de algum dispositivo. Assumimos também que os *browsers* estão se tornando a interface padrão para aplicações tais como acesso de informações pessoais, comércio eletrônico, vídeo-conferência, gerenciamento de rede, dentre outras. Usuários farão uso dessa funcionalidade também em dispositivos portáteis como telefone celular e/ou variações.

Este artigo está organizado na seguinte forma: na Seção 2 caracterizamos os principais requisitos para mobilidade de pessoal e de serviço. A Seção 3 descreve o suporte à mobilidade pessoal e de serviços utilizando Agentes Móveis Inteligentes. Na Seção 4, mostramos a dinâmica do modelo através de Redes de Petri. Na Seção 5, discutimos alguns aspectos de implementação. Por fim, a Seção 6, apresenta as contribuições e conclusões do trabalho.

2 Suporte à Mobilidade Pessoal e de Serviço

Nos dias correntes, a comunicação sem fio é o segmento com o mais rápido crescimento na indústria das telecomunicações. A demanda por serviços básicos de voz e dados estendeu os limites da tecnologia *wireless* para o mercado cada vez maior de serviços ubíqüos. Satisfazer esta demanda requer o desenvolvimento de redes *wireless* que não somente suportam serviços multimídia integrados, mas também os forneça de maneira ininterrupta, direta e com qualidade de serviço aos terminais móveis.

A próxima geração de sistemas *wireless*, combinando as funcionalidades das redes *wireless* e fixas, irá empregar uma rede de provedores de serviço e equipamentos interoperáveis para garantir serviços multimídia em qualquer parte do globo. Um importante desafio frente a esta nova geração de sistemas será o gerenciamento da mobilidade.

Um sistema *wireless* eficiente deverá ser capaz de localizar terminais móveis a qualquer tempo para fornecer seus serviços, assim como manter as conexões à medida que os terminais se movam de uma área para outra. Deste modo, para prover uma comunicação *wireless* global, será necessário o desenvolvimento de uma arquitetura que garanta mobilidade de terminal, mobilidade pessoal e portabilidade global dos serviços.

A mobilidade de terminal permite a mudança de localização do terminal para pontos onde a conexão (*wireline* ou *wireless*) possa ser mantida ou restabelecida. Quando da mudança de localização, o terminal pode ter o serviço em curso interrompido ou não, dependendo da manutenção ou não da conexão. A identificação de um terminal é feita através da atribuição de um número ao mesmo. Para possibilitar a localização do terminal, uma infra-estrutura é oferecida pela rede a qual o terminal está associado. Diferentes redes provêem diferentes mecanismos para tratar a mobilidade de terminal.

Já a mobilidade pessoal permite aos usuários o acesso a seus serviços independente do seu terminal ou ponto de acesso, um conceito conhecido como *global roaming*. Este conceito estende-se para outras redes tais como Internet e Telecom Fixa. Para que esta integração possa acontecer, uma visão mais realista da evolução de cada uma das redes deve ser considerada pelas demais. Hoje em dia, usuários são identificados por números atribuídos a seus terminais. Para a próxima geração, ao usuário será atribuído um número pessoal universal (*universal personal telecommunication number*) que o distinguirá independente do terminal a partir do qual tenha estabelecido acesso a rede.

Finalmente, a portabilidade dos serviços proporciona aos usuários, que estão fora do alcance de sua rede *home*, o acesso a serviços personalizados dentro dos limites estabelecidos pelas funcionalidades da rede visitada. Nesse contexto, VHE (*Virtual Home Environment* [14]) é definido como um conceito de sistema para a portabilidade de serviços adaptáveis através dos limites das redes e entre terminais. O conceito de VHE é tal que usuários UMTS (*Universal Mobile Telecommunications System*) terão à sua disposição o mesmo conjunto de serviços personalizados.

A liberdade da mobilidade global nesta escala irá requerer a coordenação de uma grande variedade de provedores de serviço, garantia de compatibilidade entre as redes *backbone* (Telecom Fixa, Telecom Móvel e Internet), e obtenção de acordos entre operadoras de redes. Enquanto acordos entre operadoras de redes são governados por contratos comerciais, a mobilidade global necessitará de acordos envolvendo países, regiões, provedores de serviço e aumento do espectro de freqüência disponível. Nesse último aspecto, a ITU (*International Telecommunication Union*) já está promovendo o uso eficiente do espectro de freqüência para facilitar o desenvolvimento de sistemas de comunicação pessoal e global como parte do processo de definição dos sistemas móveis de terceira geração (*Third Generation Mobile Systems*).

3 Agentes Móveis no Suporte à Mobilidade

O uso da tecnologia de agentes no nosso modelo, por um lado realça a tecnologia de objetos distribuídos e por outro, proporciona autonomia e flexibilidade nos processos de personalização e

disponibilização dos serviços quando da necessidade de suporte à mobilidade pessoal e de serviço. Neste modelo, os agentes podem se mover para um outro provedor para avaliar se a nova infra-estrutura oferece suporte a adaptação de serviços de outros domínios (agentes com certo grau de inteligência). Não obstante, nossa proposta também se utiliza da tecnologia de objetos distribuídos como forma de interoperar com sistemas legados de subscrição, acesso e serviço presentes nos provedores.

3.1 Modelo Proposto para Suporte à Mobilidade

Para o suporte à mobilidade pessoal e de serviços sobre redes heterogêneas fixas e móveis, algumas considerações devem ser estabelecidas no sentido de garantir a interoperabilidade entre serviços nas redes Internet, Telecom Fixa e Móvel. Dentre elas, destacamos: a) a Rede Internet constitui a fonte de serviços para usuários; b) a Rede Internet funciona como elemento integrador das várias arquiteturas de rede envolvidas. Neste contexto, onde o mesmo conjunto de serviços está disponível de forma independente da rede em uso, muito provavelmente a qualidade de serviço será variável e dependente do suporte das redes nas quais o serviço é disponibilizado[2].

Assumimos, também, que a existência de um ambiente de suporte ao processamento distribuído (DPE) e uma camada genérica de serviço sobre redes heterogêneas (fixas e móveis) parecem ser promissoras no suporte à distribuição, interoperabilidade, e tempo de desenvolvimento de serviços dado a grande diversidade presente entre as redes.

Na Fig. 1, apresentamos os principais componentes envolvidos no provimento de serviços, considerando o suporte à mobilidade pessoal e de serviço sobre redes heterogêneas. Estes componentes são: provedores de serviço (*retailers*), redes *home* e visitada, usuário e dispositivo no qual o serviço será oferecido. O suporte à mobilidade de terminal, como discutido anteriormente, se dá pela arquitetura de rede em questão, por exemplo, telefones celulares na Rede Telecom Móvel e *notebooks* em uma Rede WLAN. Ainda na Fig. 1, o modelo proposto estabelece um Ponto de Referência entre Provedores (PR-Prvd), além de coexistir com sistemas legados de subscrição, acesso e serviço, o que de certo modo, contribui para preservar a infra-estrutura já existente de suporte ao oferecimento de serviços presente nos provedores de serviço.

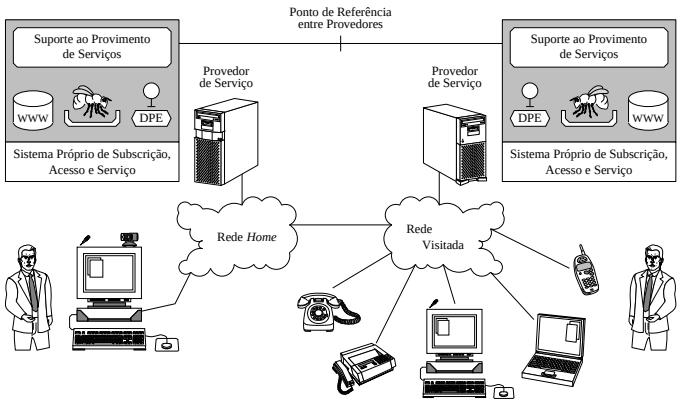


Figura 1: Modelo Proposto para Suporte à Mobilidade Pessoal e de Serviço.

Na nossa proposta, um provedor de serviço pode desempenhar o papel de Provedor Contratado (*Home Retailer*) ou de Provedor Visitado (*Foreign Retailer*) como ilustrado na Fig. 2. Provedor Contratado é aquele com o qual o usuário contratou serviços e assim dispõe de uma *profile* e de um contrato que reflete os direitos concedidos ao usuário pelo subscritor. Já no Provedor Visitado, o usuário não tem contrato de uso de serviços, mas pode utilizá-lo como ponto de contato inicial para

uso de serviços de algum domínio *home*, desde que em conformidade com o ponto de referência entre provedores de serviço.

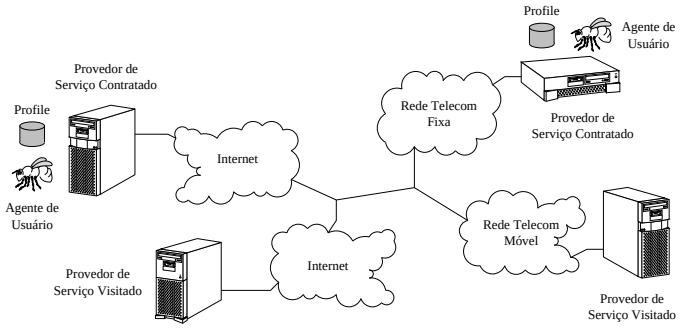


Figura 2: Domínio Contratado de um Usuário.

Um usuário pode ter contrato com vários provedores e assim terá associado uma *profile* de usuário e de serviços com cada um deles. No domínio visitado, o provedor visitado proporciona um ponto de contato inicial para um usuário através de seu nome@domínio, que para ser válido na federação, deve ser autenticado pelo provedor contratado. Serviços de algum domínio contratado poderiam então ser disponibilizados no novo domínio, considerando naturalmente a nova infra-estrutura (p. ex., recursos de *hardware* e *software* no provedor e terminal). Neste caso, os provedores além de estarem em conformidade com o ponto de referência entre provedores, acordam a contabilidade dos serviços oferecidos através de federação na disponibilização dos mesmos.

Nossa proposta está alinhada com o modelo descrito em [12] contemplando um *framework* de agentes inteligentes para o suporte à seleção de serviços e provedores de serviços. Assume-se que quando da escolha do serviço no provedor visitado, o mesmo seja capaz de oferecê-lo. Cabe ao agente de usuário salvaguardar a *profile* do usuário e informações acerca dos serviços por ele contratado. Com a mesma abordagem o Projeto ACTS (*Advanced Communication Technologies and Services*) CLIMATE e MONTAGE[6].

3.2 Descrição dos Componentes do PR-Prvd

Nesta seção, apresentamos uma descrição geral dos componentes do ponto de referência entre provedores de serviço e suas interações com os componentes particulares de subscrição, acesso e serviço do provedor, onde estes últimos não fazem parte do PR-Prvd . A Fig. 3 apresenta os componentes do ponto de referência e suas interações — 3 componentes fixos e 2 móveis (agentes móveis inteligentes).

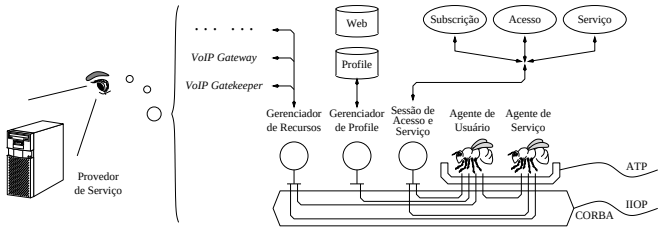


Figura 3: Arquitetura do Provedor de Serviço para Suporte à Mobilidade.

3.2.1 Componente Agente do Usuário

É um componente de serviço que representa o usuário na rede de serviços em cada provedor contratado, e é acessível a partir de um endereço de rede fornecido por um serviço de diretório ou serviço de nomes. O agente de usuário age como um ponto único de contato na adaptação dos serviços de um domínio contratado.

O agente de usuário negocia o modelo e aspectos suportados pela sessão de acesso e serviço no provedor, além de gerenciar as preferências do usuário no acesso, uso e adaptação dos serviços. Suas operações são independentes do serviço. Também permite o uso de serviços a partir de diferentes terminais incluindo o suporte do registro do usuário nesses terminais para receber convites. O agente de usuário pode participar de uma ou mais sessões de acesso e/ou serviço simultaneamente, mas esta replicação é limitada pelo número de sessões simultâneas previamente contratado.

3.2.2 Componente Agente de Serviço

É um componente de serviço que modela uma variedade de aplicações de tal forma que possibilita a adaptação do serviço ao novo domínio, considerando a *profile* do usuário na configuração desses serviços. Isto pode ser conseguido seja pela personalização ou reconfiguração de serviços já existentes.

O agente de serviço traz flexibilidade, permitindo a distribuição de *software* e/ou provisão de serviço sob demanda configurável, portanto, não só o arranjo da rede de acesso mas o dispositivo final do usuário devem ser considerados. Através do suporte apropriado, o agente de serviço pode interagir com outras aplicações ou com o usuário.

3.2.3 Componente Gerenciador de *Profile*

É um componente de serviço a partir do qual se obtém as preferências do usuário sobre o acesso e uso de serviços.Também podem ser obtidas informações acerca do contrato de serviços com relação ao uso de infra-estrutura específica de *hardware* e *software*.

Para a adaptação do serviço ao novo domínio e às capacidades do terminal, mesmo que o usuário não disponha de serviços contratados neste provedor, o acesso e uso de serviços se dá após o provedor contratado autenticá-lo através de solicitação encaminhada pelo provedor visitado conforme o PR-Prvd entre provedores de serviços.

3.2.4 Componente Gerenciador de Recursos

É um componente de serviço a partir do qual se obtém informações da infra-estrutura específica de *hardware* e/ou *software* no suporte da mobilidade de serviços. Também é a partir deste componente que o agente de serviço reserva o recurso, assim como gerencia seu ciclo de vida e sua operação.

Deste modo, este componente em conjunto com o agente de serviço adaptam o serviço às novas condições de arranjo de rede e capacidades do terminal. O uso da infra-estrutura requerida no suporte à mobilidade de serviço é dependente de permissões de uso de recursos obtidas em um contrato *default* no provedor visitado.

3.2.5 Componente Sessão de Acesso e Serviço

É um componente de serviço que permite a troca de informações entre os componentes agente de usuário e de serviço com o usuário. Esta troca de informações ocorre através das sessões de acesso e/ou serviço do provedor com o qual o usuário estabeleceu conexão. Este componente exporta através de sua interface funcionalidades que permitem convidar um participante ou ser convidado a participar de uma sessão em curso.

4 Dinâmica do Modelo através de Redes de Petri

O comportamento dinâmico dos componentes do PR-Prvd durante as diversas fases de acesso e uso de um serviço são complementares à descrição dos componentes. As interações serão apresentadas por meio de cenários e baseadas nas ações tomadas pelo usuário no acesso, estabelecimento e uso dos serviços. A evolução dos cenários é apresentada por diagramas de eventos e a especificação formal pela Rede de Petri.

A Rede de Petri é uma ferramenta gráfica e matemática que se adapta bem a um grande número de aplicações nas quais as noções de eventos e de evoluções simultâneas são importantes[1]. Com a Rede de Petri e com o auxílio dos diagramas de troca de mensagens foi possível compreender com detalhes a dinâmica dos vários cenários dentro dos quais um usuário possa se encontrar. Através da Rede de Petri foi possível avaliar o modelo proposto no que se refere a ocorrência de *deadlock* e *loops*, freqüentemente encontrados em ambientes distribuídos.

Para cada cenário, condições prévias e um estado posterior são esperados. Condições prévias definem as condições que devem estar estabelecidas para que os eventos de um cenário sejam gatilhados. Já as condições posteriores são caracterizadas pelo estado de vários componentes do cenário após eventos terem ocorrido.

4.1 *Login* em um Provedor Visitado

Este cenário permite que o usuário estabeleça uma sessão de acesso em um provedor visitado, ou seja, o usuário não tem contrato de uso de serviços neste domínio. Ao iniciar processo de *login*, os passos descritos em seqüência ou os eventos apresentados no diagrama de eventos de *login* da Fig. 4.

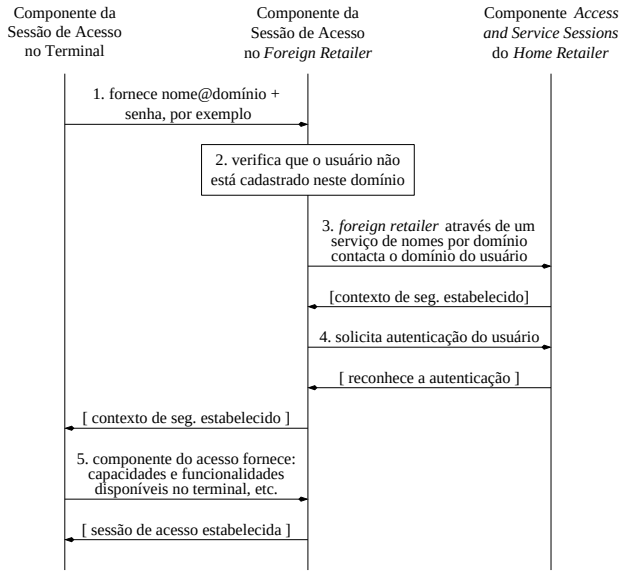


Figura 4: *Login* em um Provedor Visitado.

1. o usuário utilizando-se de algum suporte no terminal, fornece informações para que as mesmas possam ser endereçadas;
2. de posse das informações do usuário, o provedor verifica que o mesmo não possui serviços contratados no domínio, e assim assume a condição de provedor visitado;
3. busca através do serviço de nomes o domínio fornecido e informções de contexto de segurança para que dados do usuário possam ser submetidos à autenticação;

4. após submissão dos dados para autenticação e uma vez obtida a autenticação, o endereço de rede do agente do usuário é fornecida ao provedor visitado;
5. algum componente da sessão de acesso do domínio do usuário fornece ao provedor informações sobre o domínio do usuário.

A Fig 5 ilustra a Rede de Petri para o cenário anteriormente visto. Como pode ser observado, a dinâmica inicia somente quando o componente de sessão de acesso no terminal é instanciado. Ao fornecer dados na sessão de *login* a transição adjacente na seqüência do fluxo de marcação é sensibilizada (t_2), por conseguinte ao caminhar ocorre transição de estado. Neste estado (Solicitar Contexto), o componente encapsula os dados do usuário para encaminhá-los ao provedor visitado (t_3). Nesta transição, duas marcas partem da mesma nas direções objeto local e remoto.

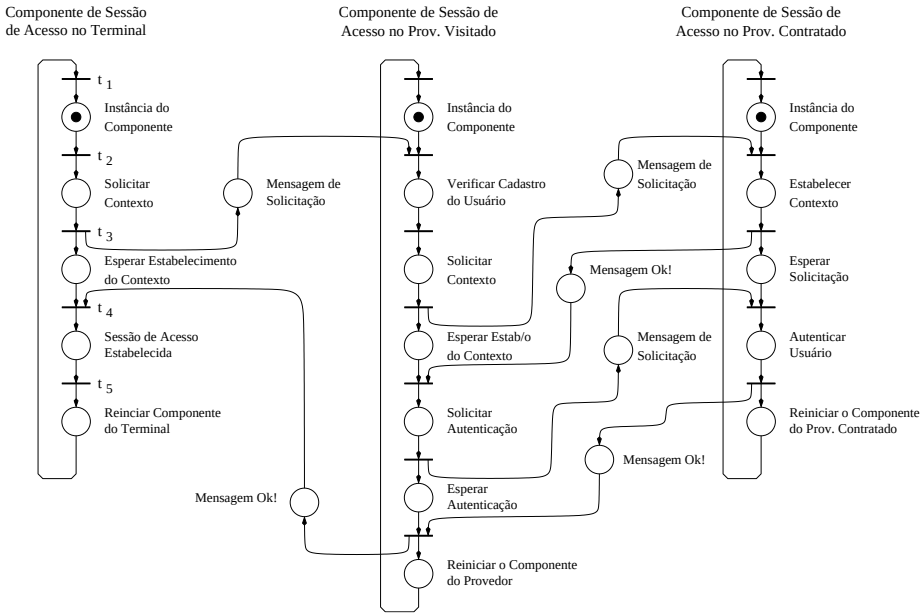


Figura 5: Rede de Petri para o *Login* em um Provedor Visitado.

Percebe-se que para o objeto remoto (componente no provedor visitado) não há dinâmica sem a marcação vinda do terminal do usuário. O mesmo raciocínio pode ser aplicado entre o componentes do provedor visitado e do provedor contratado. Na verdade, a dependência se dá nas duas vias, ou seja, mensagens são bloqueantes. Voltando ao componente no domínio do usuário, a marcação prepara a transição t_4 para ser sensibilizada, ou seja, só será sensibilizada quando do retorno da invocação do provedor visitado. Por fim, a transição t_5 leva ao estado de uma nova sessão de acesso.

4.2 Selecionar Serviço no Provedor Visitado

Para selecionar um serviço no provedor visitado; assumimos que o usuário já estabeleceu uma sessão de acesso com o provedor. Caso o usuário tenha listado os serviços opcionais disponíveis, o *User Agent* já possui estas informações e assim sendo, selecionar um serviço implica em primeiro lugar solicitar a lista de serviços que podem ser adaptados. Os passos que descrevem este processo são apresentados a seguir, enquanto que o diagrama de eventos correspondente pode ser vistos na Fig. 6.

1. usuário solicita serviço opcional, de algum provedor contratado, que possam ser adaptados às capacidades do terminal e do provedor ao qual está conectado;

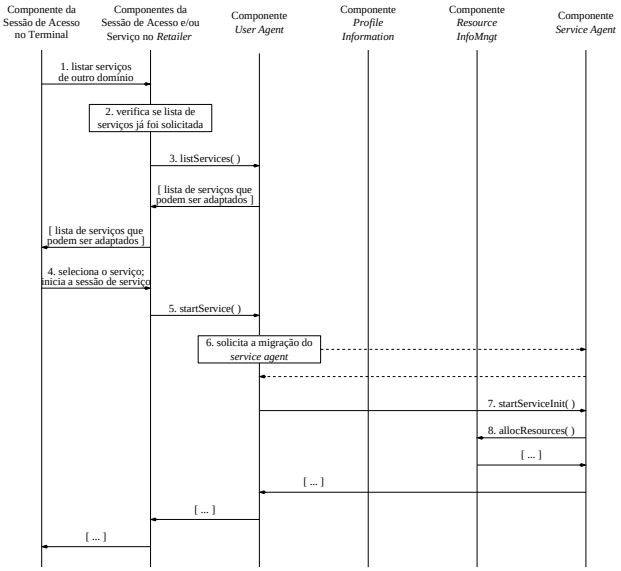


Figura 6: Seleção de Serviço no Provedor Visitado

2. componente da sessão de acesso do provedor verifica se o usuário já solicitou a lista de serviços opcionais disponíveis; em caso negativo, a lista de serviços é solicitada invocando a operação discoverServices() no mesmo objeto;

3. componente da sessão de acesso e serviço do provedor obtém a lista de serviços adaptáveis ao novo domínio que por intermédio de outros componentes do provedor pode então ser disponibilizada no terminal do usuário;

4. usuário seleciona serviço, iniciando uma sessão específica junto ao provedor;

5. componente da sessão de acesso e serviço do provedor solicita ao *user agent* o serviço;

6. *user agent* solicita a migração do *service agent* para em conjunto com o provedor visitado adaptar o serviço considerando as capacidades do terminal;

7. após receber informações de que o *service agent* migrou e já se encontra instalado na agência, o *user agent* solicita início de um serviço;

8. o componente *service agent* requisita a alocação dos recursos necessários e, em sendo confirmada a requisição, o serviço pode então ser gerenciado por alguma interface homem-máquina disponível no terminal para o usuário.

A Fig 7 ilustra a Rede de Petri para o cenário anteriormente visto. Como pode ser observado, a dinâmica inicia somente quando o componente de sessão de acesso no terminal é instanciado. A transição adjacente na seqüência do fluxo de marcação é sensibilizada (t_2), por conseguinte ocorre transição de estado. Neste estado (Solicitar Lista de Serviço de outro domínio), o componente encapsula os dados do provedor contratado para encaminhá-los ao provedor visitado (t_3). Nesta transição, duas marcas partem da mesma nas direções objeto local e remoto.

Percebe-se que para o objeto remoto (componente no provedor visitado) não há dinâmica sem a marcação vinda do terminal do usuário. O mesmo raciocínio pode ser aplicado entre o componentes do provedor visitado, do *user agent*, do gerenciador de recursos e do *service agent*. A construção da matriz de marcações acessíveis nos mostra a ausência de *deadlocks* e *loops* nesta rede.

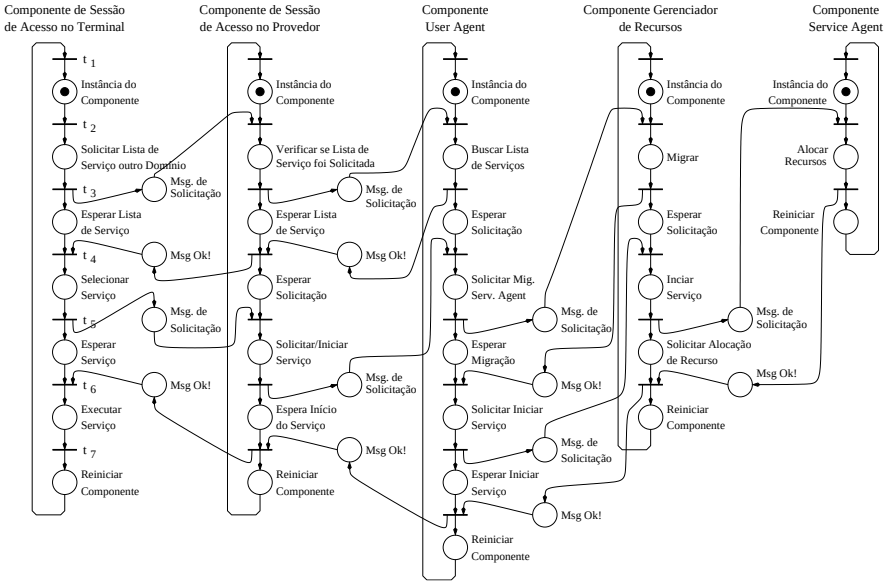


Figura 7: Rede de Petri para Seleção de um Serviço no Provedor Visitados.

5 Aspectos da Implementação

Neste trabalho propomos o uso de agentes móveis inteligentes representando o usuário e atuando na configuração de seus serviços, sendo entidades de *software* autônomas, assíncronas, dotadas de habilidades e objetivos próprios (Fig. 8.a.). Adicionalmente, nossos agentes contemplam dois predicados, mobilidade e inteligência.

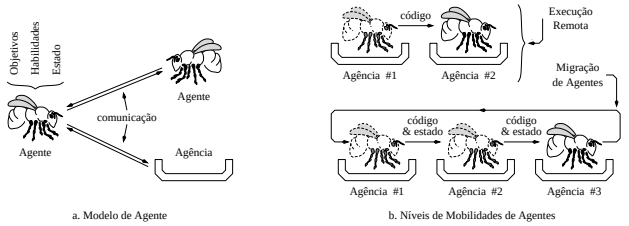


Figura 8: Modelo de Agente

Geralmente, dois níveis de mobilidade de agente podem ser identificados: execução remota e migração do agente, como apresentado na Fig. 8.b. Na execução remota, um agente (código e dados) é transferido para um sistema remoto onde é ativado e executado integralmente e para interagir com recursos localizados no destino. Já na migração do agente, o agente pode se mover de um nó para outro durante sua execução a fim de acompanhar progressivamente o cumprimento de sua tarefa. Uma agência que suporta a migração do agente suspende sua execução, despacha para um outro nó na rede, e retoma a execução a partir do ponto onde foi interrompido.

Como já apresentado, os componentes da arquitetura implementada (Fig. 3) foram especificados e estão em conformidade com a especificação CORBA (*Common Object Request Broker*) [8]. O qual define uma linguagem neutra, IDL (*Interface Definition Language*), que permite descrever a especificação de um objeto CORBA através de suas interfaces. As interfaces descritas em IDL podem ser implementadas em qualquer linguagem para a qual o mapeamento IDL esteja disponível. Deste modo, aplicações CORBA escritas em diferentes linguagens podem interoperar.

Como infra-estrutura CORBA, utilizamos o ORB (*Object Request Broker*) disponível no ambiente de desenvolvimento Java™ 2[7] (Java™ IDL). Suportando a especificação CORBA/IIOP[8] e possuindo mapeamento IDL/Java[9], Java™ IDL implementa um ORB para linguagem Java. Para usar Java IDL faz-se necessário a instalação da Plataforma Java™ 2 JDK 1.4 ou superior que já inclui o compilador `idltj`, além das classes `org.omg.CORBA`. Toda esta infra-estrutura precisa estar instalada no ambiente do Web *browser* (p. ex., com auxílio de um Plug-in).

Para suportar os agentes, utilizamos o conceito de Agência da especificação MAF (Mobile Agent Facility) [10]. A Agência é um plataforma que pode criar, executar, transferir e finalizar um agente. Na verdade, MAF é o resultado dos esforços de padronização da *ObjectSpace* [13] e da OMG (*Object Management Group*) para dentre outros aspectos garantir a interoperabilidade entre as diferentes plataformas, quando em conformidade com a especificação[15].

Para a infra-estrutura de agentes móveis, utilizamos uma implementação da especificação MAF, ou seja, contempla objetos que exportam as interfaces `MAFFinder` e `MAFAgentSystem`. A interface `MAFFinder` é um serviço de nomes que define operações para registrar, remover o registro e localizar agentes, lugares e agências (*agent systems*). Enquanto que a interface `MAFAgentSystem` define métodos e objetos que suportam tarefas de gerenciamento de agentes tais como: procurar o nome de um sistema agente e receber um agente. Na implementação, agentes são objetos CORBA.

Além da mobilidade, nossos agentes necessitam de conhecimento (inteligência) de modo que possam avaliar a infra-estrutura do provedor e terminal a partir dos quais o usuário solicita um serviço. Para incorporar tal propriedade aos nossos agentes, utilizamos a plataforma JESS (*Java Expert System Shell*)[5]. O JESS suporta o desenvolvimento de sistemas inteligentes baseados em regras de conhecimento do tipo *se-então*, possibilitando a junção com código Java. A versão utilizada foi JESS 6.0 que é compatível com a versão Java JDK 1.4.

Na Fig. 9 descrevemos um cenário onde o usuário se move para outro domínio (mobilidade pessoal), gerando a necessidade de adaptação de serviço (mobilidade de serviço). Conforme descrito na Seção 4, o usuário estabelece uma sessão de acesso com o provedor visitado, solicita a lista de serviços possíveis de serem oferecidos e seleciona a adaptação de um serviço no novo domínio.

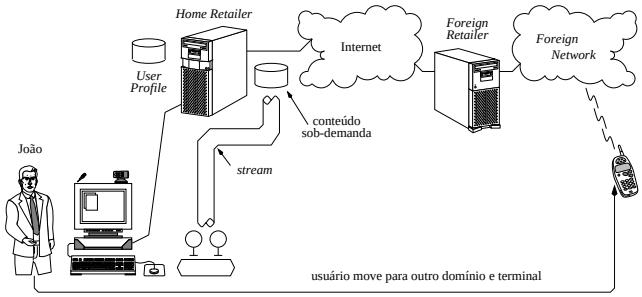


Figura 9: Suporte à Mobilidade no Serviço de Conteúdo Sob Demanda.

Neste ponto, o representante do usuário na rede é conectado pelo provedor visitado, assim, o *user agent* migra para o novo domínio. Nosso agente é serializado como uma seqüência de bytes e transmitida através da rede e desserializada no novo domínio. De fato, Java já oferece esta funcionalidade, mas não contempla a manutenção do estado do agente, ou seja, ele deve ser reinstanciado do início. Como o que estamos propondo e implementando é a migração, faz-se necessário manter o estado do agente. Conseguimos isto, levando a base de conhecimento e a base de dados do mesmo para o novo domínio. Logo, ao ser reinstanciado, o agente não só dá continuidade ao que estava fazendo como também irá inferir um novo estado, dependente agora das novas condições em que se encontra e da requisição encaminhada pelo usuário.

No provedor visitado o *user agent* coleta informações a respeito do usuário, do terminal e

dos recursos que o novo domínio disponibiliza. No atual estágio da implementação estes dados são fornecidos pelos objetos do PR-Prvd. Já dispomos dessa implementação, ainda não integramos como o serviço de tele-conferência. A aplicação de tele-conferência utilizada já atende as novas necessidades de disponibilização e gerência de serviços na Internet[11]. Descrevemos a seguir parte da base de conhecimento utilizada pelo agente para inferir quais serviços podem ser adaptados e em que condições. A regra apresentada avalia quais serviços podem ser adaptados.

```
(defrule servicos_adaptaveis
  ?serv <- (servico (nome ?n) (recursos ?r_topo $?r_resto))
  ?term <- (terminal (recursos $?list_1&: (neq $?list_1 nil)))
  ?rtlr <- (retailer (hdwr_sftw $?list_2&: (neq $?list_2 nil)))
  ?adpt <- (serv_adp (nome ?serv1&: (eq ?serv1 ?n)) (recursos $?list_3))
  =>
  (bind ?length_1 (length$ ?list_1))
  (bind ?i 1)
  (bind ?length_2 (length$ ?list_2))
  (bind ?j 1)
  (while (or (<= ?i ?length_1) (<= ?j ?length_2)))
    (bind ?topo_i (nth$ ?i ?list_1))
    (bind ?topo_j (nth$ ?j ?list_2))
    (if (or (eq  ?topo_i ?r_topo) (eq ?topo_j ?r_topo))
      then (bind ?i (+ ?length_1 ?i))
           (bind ?j (+ ?length_2 ?j))
           (modify ?adpt (recursos ?r_topo $?list_3))
      else (bind ?i (+ 1 ?i))
           (bind ?j (+ 1 ?j)) ))
  modify ?serv (recursos $?r_resto) ))
```

Esta base de conhecimento avalia se o terminal e o provedor visitado podem em conjunto oferecer serviços oriundos do provedor contratado. Depois do *user agent* avaliar quais serviços são adaptáveis à nova infra-estrutura, a lista de tais serviços é disponibilizada para o usuário. Ao selecionar um serviço, o *user agent* solicita a migração do *service agent*. Nosso *user agent* escrito em linguagem de *script* somou 5 Kbytes.

6 Conclusão

Este trabalho propõe uma arquitetura de suporte à mobilidade pessoal e de serviço em serviços de telecomunicações e de informação, bem como o desenvolvimento de um protótipo para avaliar a adequação deste modelo. Este modelo está baseado na junção dos paradigmas de objetos distribuídos e de agentes móveis inteligentes. Cada paradigma oferece vantagens em diferentes aspectos, e assim em conjunto podem prover o máximo de flexibilidade no suporte à mobilidade de aplicações distribuídas configuráveis como os serviços emergentes de telecomunicações.

Dentre os aspectos relevantes desta proposta, destacamos:

- proposição de uma arquitetura que coexiste com sistemas legados de subscrição, acesso e serviço no provedor, preservando a infra-estrutura já existente;
- o agente de serviço traz flexibilidade ao permitir a distribuição do *software* e/ou provisão do serviço sob demanda configurável;
- descentralização dos serviços de controle e gerenciamento de *software*, trazendo o controle, gerenciamento e capacidade de adaptação tão próximo quanto possível dos seus recursos através do agente de usuário;
- provedores podem ou não permitir o uso de recursos de *hardware*/*software* dependendo das capacidades do terminal.

No que se refere a implementação do *user agent* usando a plataforma JESS, cabe como ressalva a necessidade de coordenação dos nomes de serviços, de recursos contemplados nos terminais bem como de recursos de infra-estrutura oferecida pelos provedores, pois em muito facilita a geração da base de conhecimento. Por fim, nosso agente mantém o estado quando da sua reinstanciação remota, ou seja, o agente de fato sofre uma migração. Tal funcionalidade se deve ao uso do JESS.

7 Agradecimentos

Esta pesquisa é suportada pelo CNPq através de uma bolsa de mestrado concedida à aluna Juliana O. Carvalho e teve a contribuição do Prof. Eleri Cardozo (DCA-FEEC-Unicamp) e de Eliane G. Guimarães (ITI - Campinas) na implementação da especificação MAF utilizada neste artigo.

Referências

[1] CARDOSO, J., AND VALETTE, R. *"Redes de Petri"*. Editora da UFSC, Florianópolis, 1997.

[2] CHAKRAVORTY, R., KAR, S., AND FARJAMI, P. End-to-end internet quality of service (qos): An overview of issues, architectures and frameworks. In *3rd International Conference on Information Technology - ICIT* (Bhuwaneshwar, India, December 2000).

[3] CLARK, D., PASQUALE, J., AND ET AL. "Strategic Direction in Networks and Telecommunications". *ACM Computing Surveys 28*, 4 (Dezembro 1996).

[4] FAINA, L. F. *"Uma Arquitetura para Suporte à Ubiqüidade dos Serviços de Telecomunicações Baseada na Arquitetura TINA e em Agentes Móveis"*. Tese de Doutorado, Faculdade de Engenharia Elétrica e de Computação, UNICAMP, Campinas, Brasil, Dezembro 2000.

[5] FRIEDMAN-HILL, E. J. *"Jess, The Expert System Shell to the Java Platform"*. Sandia National Laboratories, 2001. URL at `http://herzberg.ca.sandia.gov/jess`.

[6] JORMAKKA, H., VALTARI, K., D. PREVENDOUROU, A. K., AND RAATIKAINEN, K. "Agent-based TINA Access Session Supporting Retailer Selection in Personal Mobility Context". In *TINA'99 Conference* (Oahu, Hawaii, USA, Apr. 1999), pp. 68–76.

[7] MICROSYSTEMS, S. *"Java™ 2 Software Development Kit"*. Sun Microsystems, 2000. `http://www.java.sun.com/products/`.

[8] OMG. "Common Object Request Broker: Architecture and Specification – Version 2.3.1". Document formal/99-10-07, Object Management Group, Oct. 1999. `http://www.omg.org`.

[9] OMG. "Java Language Mapping to OMG IDL". Document formal/99-07-59, Object Management Group, July 1999. URL at `http://www.omg.org`.

[10] OMG. " Mobile Agent Facility Specification". Document, Object Management Group, Jan. 2000. URL at `http://www.omg.org`.

[11] PINTO, R., FAINA, L. F., MAFFEIS, A., GUIMARÃES, E., MIGLINSK, C., AND CARDOZO, E. Uma arquitetura para disponibilização e gerência de serviços na internet. In *XX Simpósio Brasileiro de Redes de Computadores - SBRC* (Búzios, RJ, Maio-Junho 2002), vol. I.

[12] PREVEDOUROU, D., ZYGOURAKIS, K., EFREMIDIS, S., STAMOULIS, G., KALOPSIKAKIS, D., KYRIKOGLOU, A., SIRIS, V., ANAGNOSTOU, M., TZIFA, L., LOUTA, T., DEMESTICHAS, P., LIOSSIS, N., KIND, A., VALTARI, K., JORMAKKA, H., AND JUSSILA, T. "Use of Agent Technology in Service and Retailer Selection in a Personal Mobility Context". *IEEE Computer Networks* (1999).

[13] SPACE, O. "Voyager Core Technology". Technical Overview Version 1.0, Object Space, Inc., Dezembro 1997. `http://www.objectspace.com`.

[14] UMTS. "Service Aspects: Virtual Home Environment". UMTS Forum Report 22.70 Version 3.0, Universal Mobile Telecommunications System, 1999.

[15] WONG, J., HELMER, G., NAGANATHAN, V., AND POLAVARAPU, S. "SAMART Mobile Agent Facility". *The Journal of Systems and Software 56* (Setembro 2001).

Problemas de Segurança em Sistemas Operacionais: Uma Questão em Aberto Há Quase 30 Anos

Mauro Marcelo Mattos
mattos@furb.br

Resumo. O presente trabalho apresenta algumas reflexões acerca da origem dos problemas de segurança em sistemas operacionais. O trabalho defende a tese de que o tripé conceitual: virtualização do operador, virtualização do hardware e o conceito de programa constituem a origem do problema. O trabalho apresenta uma proposta de solução indicando que as pesquisas na área de sistemas operacionais baseados em conhecimento apresentam-se como possibilidades reais a serem consideradas na busca das soluções para tais problemas.

Palavras-chave: Segurança, Sistemas Operacionais Baseados em Conhecimento

1 Introdução

O impacto e a evolução das tecnologias de informação nas últimas décadas têm apresentado sucessivos desafios, tanto aos indivíduos em relação à sociedade, como às empresas em relação ao mercado globalizado. Senra (1999) complementa que: "A informação sempre foi importante, promovendo, em diferentes tempos e espaços, expressivas aberturas de mundo. Entretanto, no limiar do terceiro milênio, em que o marcante movimento de globalização agita, para o bem e/ou para o mal, o nosso viver, sua importância se apresenta ainda mais intensa e potente. De fato, em um mundo que concorre, a informação que engendra conhecimento se faz o recurso básico da produção, levando a uma intensa renovação organizacional".

Neste contexto, a dependência e demanda crescentes da sociedade, em relação às Tecnologias de Informação e Comunicação, têm ressaltado uma série de problemas relacionados ao processo de desenvolvimento de software – um dos componentes fundamentais na concepção das TICs – quais sejam: alto custo, alta complexidade, dificuldade de manutenção e uma disparidade entre as necessidades dos usuários e o produto desenvolvido. Segundo Cordeiro (2000) apud Bona (2002), "*empresas de software em todo o mundo empregam perto de 7 milhões de técnicos e geram anualmente uma receita de mais de 600 bilhões de dólares, com taxa de crescimento de mais de 25% nos últimos três anos*". Contudo, segundo Semeghini (2001) apud Bona (2002) "*no Brasil, cerca de 87% das empresas de desenvolvimento de software são de pequeno e médio portes. Isto implica que existem limitações de recursos a serem aplicados em treinamento e consultorias voltadas à implantação de qualidade de software e de um processo que garanta resultados adequados* ". Segundo Tkach e Puttick (1994), "*A medida em que a capacidade computacional e as facilidades de acesso a estes recursos computacionais têm aumentado com os avanços tecnológicos, a complexidade dos sistemas a serem desenvolvidos tem aumentado da mesma forma. Mais e mais usuários são atendidos pelos sistemas computacionais e cientes destas facilidades disponíveis, estes usuários estão demandando cada vez mais novas facilidades*". Além da dificuldade em atender a estas novas demandas, o *backlog* de manutenção também é um aspecto crítico a ser considerado. Várias das técnicas para análise, projeto e documentação de sistemas utilizadas para implementar sistemas que atendam àquelas demandas, não são flexíveis o suficiente para viabilizar um rápido atendimento das mesmas. Esta situação é comumente referida na literatura como: a crise do software. (Tkach e Puttick,1994).

A referida crise tem várias facetas e a afirmação a seguir caracteriza bem a complexidade do problema: "*A metáfora do computador transparente descreve um dos principais objetivos da engenharia de software contemporânea – ramo da engenharia da informação que se preocupa com o desenvolvimento dos programas complexos (software) necessários para converterem um montículo*

inerte de dispositivos (hardware), num instrumento poderoso tão fácil de ser usado como são o lápis e o papel. Quem já esperou um dia ou mais por um programa de computação convencional que nem sequer chegou a ser processado apenas porque uma vírgula fora mal colocada, poderá testemunhar que ainda não chegou o dia em que a transparência computacional instantânea estará a serviço de todos." (Oettinger,1966).

O presente trabalho apresenta algumas reflexões acerca da origem dos problemas de segurança em sistemas operacionais. O texto está organizado da seguinte forma: a seção 2 procura caracterizar a questão de que sistema operacional também é um projeto de software e que precisa ser analisado sob esta ótica. A seção 3 caracteriza a tese do tripé conceitual. A seção 4 procura caracterizar objetivamente a origem dos problemas de segurança. A seção 5 caracteriza a importância do emprego de técnicas de IA e Robótica na solução do problema. A seguir é apresentada a definição e uma breve caracterização do modelo de sistema operacional baseado em conhecimento desenvolvido pelo autor. Finalmente são apresentadas as conclusões do trabalho.

2 Projeto de sistema operacional como projeto de software

O problema destacado por Oettinger infelizmente ainda hoje, 38 anos depois, é uma realidade nos ambientes de desenvolvimento de software – um simples ponto, ou vírgula, ou ponto e vírgula, indevidamente posicionado ou ausente, pode ser o fator determinante na produção de um software que funciona corretamente ou não. E são fatores como este que fazem com que a crise do software ainda hoje continue presente nas organizações. Corroborando esta afirmação, é apresentado em Rational (2001) apud Ferreira (2002) um estudo realizado em 1995 pelo Standish Group, abrangendo mais de 352 empresas e envolvendo mais de 8.000 projetos de software, o qual apresentou os seguintes resultados: (i) 31% dos projetos de software são cancelados antes de serem concluídos; (ii) 53% dos projetos ultrapassam os custos estimados e,(iii) em pequenas empresas, apenas 16% dos projetos de software são concluídos dentro do tempo e do orçamento previstos. Nas grandes empresas, este número cai para 9%.

A partir destas considerações cabe destacar: **um sistema operacional é um produto de software** – portanto, está sujeito aos mesmos problemas. Um exemplo disto é a afirmação recente de Steve Ballmer (CEO da Microsoft) (WATER,2003):"*O sistema operacional Windows 2003 Server é seguro por definição (secure by design) e nós investimos* ***U$200 milhões*** *neste projeto. Ele é seguro por definição com 60% menos superfície de ataque em comparação ao NT4 service pack 3*". A afirmação de Nagle (1994) também caracteriza estes aspectos: "*Muitos projetos recentes tais como Windows NT, Mach 3.0, Chorus, System V e Sprite são projetados para minimizar o custo do porte para outras plataformas suportando múltiplas APIs. Contudo, as facilidades adicionais destes projetos possuem um custo associado: são mais lentos que os sistemas tradicionais*".

Desde os primórdios da computação, a comunidade de Sistemas Operacionais estuda e implementa estratégias e soluções em software, conjugadas com soluções em hardware, que visam permitir a utilização do hardware da forma mais adequada e eficiente possível. Caracterizada como uma área que desenvolve soluções de baixo nível, devido à íntima relação com as características e funcionalidades do hardware sobre o qual estas soluções são construídas, experimentou um período de evidência até o final da década de 80. Precursora da área de Engenharia de Software, a comunidade de Sistemas Operacionais experimentou todo tipo de dificuldades no que tange ao desenvolvimento de suas soluções de software, visto que, para viabilizá-las, teve que propor e testar, na prática, várias técnicas de gerência de projetos, especificação de software, construção de compiladores, depuradores, otimizadores de código e até mesmo linguagens de programação. Restrita a grupos que se foram especializando dentro das instalações fabricantes de hardware, os cientistas de software começaram a entrar em evidência a partir do momento em que a IBM decidiu comercializar tanto hardware como software. Segundo Deitel (1990), este é o marco que estabeleceu o início da área

de Engenharia de Software. Isto porque problemas que eram antes restritos aos fabricantes e às poucas instalações que possuíam computadores extrapolaram com o surgimento de empresas denominadas de *software houses*. Seguindo este processo evolutivo, a área de Engenharia de Software começou a identificar segmentos que necessitavam aprofundamento de pesquisa e vários cientistas que anteriormente trabalhavam com Sistemas Operacionais, paulatinamente começaram a enquadrar suas atividades de pesquisa e desenvolvimento em segmentos da área de Engenharia de Software.

O aspecto filosófico que está por trás da evolução da computação em geral é o de liberar os programadores de detalhes desnecessários. A idéia tem sido de que, quanto maiores os níveis de abstração permitidos, mais complexos são os problemas que eles podem resolver. Isto porque havia no passado uma realidade impondo sérias restrições na forma de utilização das máquinas – custo de hardware e software muito elevados e requisitos de qualidade, confiabilidade e performance incomparavelmente inferiores aos de hoje. Segundo Moeller (1991, alguns dos maiores avanços na história da computação convencional têm sido motivados por esta filosofia, conforme se pode observar nos seguintes exemplos: (*i*) **Linguagens de montagem:** para evitar que um programa fosse codificado em zeros e uns, foram desenvolvidos os montadores e, a partir daí, os programadores passaram a expressar suas idéias em termos de instruções mnemônicas de mais alto nível; (*ii*) **Linguagens de alto nível:** para evitar preocupações com detalhes de baixo nível, foram desenvolvidas as linguagens de programação contendo estruturas lógicas mais abstratas. Isto permitiu que os programadores pudessem expressar suas intenções sem ter que se preocupar com um tipo em particular de hardware e/ou conjunto de instruções; (*iii*)**Alocação dinâmica de memória:** para liberar os programadores do planejamento antecipado de suas necessidades de memória, foram desenvolvidas estratégias de alocação dinâmica e disponibilizadas em linguagens de alto nível (ex: Java) e,(*iv*) **Orientação a objetos e aspectos**: para auxiliar o programador a organizar e proteger suas estruturas desenvolveu-se o conceito de objetos, o qual permite manipulações de mais alto nível sobre as mesmas. A técnica de programação orientada a aspectos (AOP) visa solucionar estes problemas através do principio da separação de interesses (*concerns*), no qual são separados o código de um componente do código de um aspecto. Observa-se que este processo evolutivo tem ocorrido em função de um usuário especializado conhecido como *"o programador"*. Este vem recebendo todas as atenções por décadas, enquanto outro tipo de usuário, conhecido como usuário final, não vem recebendo a atenção devida. O próprio caminho evolutivo de hardware facilitou o surgimento de desenvolvedores de hardware específicos, os quais importaram desenvolvedores para implementar o que começou a ser denominado de *firmware* (e/ou a camada de *device drivers*) específicos para aquele hardware. À medida que este processo tornou-se mais e mais rápido, em função da explosão do uso de tecnologia de computação na sociedade, mais e mais o conceito de projetistas de sistemas operacionais tornou-se nebuloso.

Segundo Cordeiro (2000) apud Bona (2002), inicialmente acumulados na parcela de hardware, os custos de sistemas computacionais deslocaram-se para a parcela de software. Este fenômeno ocorreu em função da estabilidade alcançada pelos projetos de hardware, bem como pelo surgimento de exigências crescentes em complexidade para os sistemas de software. Também contribuíram problemas emergentes nos processos de desenvolvimento e manutenção de software. Esta migração pode ser constatada a partir da verificação de que a grande maioria dos eventos científicos e das publicações especializadas da área de Sistemas Operacionais, ou acabaram ou foram incorporados como tópicos em eventos ou publicações mais abrangentes.

3 O tripé conceitual

De acordo com Brachman (2002), "[…]*O tamanho dos sistemas de software está aumentando num espiral virtualmente descontrolado – hoje é comum encontrarem-se aplicações que ocupam 50Mbytes e que possuem 5 milhões de linhas de código – e nós sabemos que este crescimento em*

complexidade inevitavelmente conduz a sérias e novas vulnerabilidades. Sob a perspectiva de segurança, maior complexidade significa mais possibilidades para um intruso encontrar caminhos para entrar; sob a perspectiva de robustez, mais elementos significam mais caminhos que podem conduzir a erros; sob a perspectiva de manutenção, mais código significa que os custos de manter as coisas rodando crescem continuamente; e, sob a perspectiva de treinamento, significa que mais pessoas precisam gastar mais tempo aprendendo como usar sistemas baseados em computador. E eu ainda não toquei em algo que nos afeta todos os dias – a chamada usabilidade dos nossos sistemas.Tudo isto nos conduz a uma conclusão muito clara: nós necessitamos de alguma coisa dramaticamente diferente. Nós não podemos meramente aumentar a capacidade e velocidade dos nossos computadores. Nós não podemos somente fazer uma engenharia de software melhor. Nós precisamos mudar nossa perspectiva em sistemas computacionais e mudar a trajetória atual[…]"

O cenário apresentado anteriormente deve-se, em parte, à enorme demanda por novas soluções que a sociedade impõe ao mercado e que, portanto, fazem com que as empresas tenham que produzir soluções em software cada vez mais rápido. Este processo conduz às questões de reaproveitamento de esforço (código, conhecimento, recursos financeiros, etc) para "não ser necessário reinventar a roda". Na área da pesquisa também existe muita pressão por prazos e resultados, o que induz a adoção de uma base estável como forma de ganhar tempo e concentrar esforços no alvo dos projetos. Contudo, há um outro aspecto a ser considerado e que está relacionado à **virtualização do operador** de computador. À medida que ocorrem a evolução tecnológica e a miniaturização dos computadores, várias funções vêm sendo incorporadas ao sistema na forma de comandos e interpretadores de comandos. A figura do operador desapareceu no contexto de computadores *desktops* (*notebooks* e *palmtops*) e surgiu a figura do usuário.

Hoje quem usa um computador tem que saber como operá-lo, ou seja, como preparar os dados, como recuperar arquivos, como configurar programas, onde colocar os arquivos, em que formato armazenar, quando realizar uma cópia de segurança (*backup*), etc. Ou seja, o processo de virtualização do operador transformou o sistema operacional num facilitador de uso (uma espécie de *proxy server*) e o usuário num operador especializado. Esta necessidade de aprender a operar é, por si só, um fator de exclusão social, na medida em que pessoas necessitam ser treinadas a operar "botões lógicos" para serem aproveitadas pelo mercado de trabalho. E como a área de computação evolui muito rapidamente, esta necessidade de (re) treinamento torna-se algo tão corriqueiro que chega a ser óbvia no modelo atual. Este caráter de virtualização do operador pode ser caracterizado a partir da seguinte afirmação de De Souza (1993): "*Na Engenharia Semiótica, a interface é vista como uma mensagem unidirecional enviada dos projetistas para os usuários. Segundo esta abordagem, a mensagem é um artefato de metacomunicação, já que não apenas os projetistas se comunicam com os usuários, mas a própria interface troca mensagens com os últimos. Neste contexto, a interface é um conjunto de signos que são representações que os projetistas usam para comunicar aos usuários como manipular o sistema para atingir seus objetivos*".

Um outro conceito que influencia a forma de concepção das soluções computacionais hoje é decorrente da introdução do conceito de multiprogramação. Isto porque, até o surgimento deste conceito (como forma de otimização do uso de recursos), os programas eram enfileirados ou agrupados em lotes para serem executados. Cada programa assumia o controle da máquina durante sua execução e somente ao término de um programa é que o próximo da fila iniciava a sua operação. A **virtualização do hardware,** introduzida pelo conceito de multiprogramação, acabou com o enfileiramento de programas através da divisão do tempo de processador em pequenas fatias denominadas *time-slices*. A decorrência imediata disto foi o surgimento dos problemas de proteção (código e dados) que estão presentes até hoje.

E finalmente, mas não menos importante, é o **conceito de programa**. Um programa é uma abstração muito mais complexa do que aquela considerada nos estudos de Engenharia de Software. Lá

são considerados os aspectos funcionais e não funcionais de uma solução computacional para um problema qualquer. No entanto, a questão é mais complexa. Na realidade, um programa representa a intromissão virtual de uma pessoa (ou grupo de pessoas) dentro da máquina do usuário. Um programa implementa decisões e ações que foram concebidas em contextos diversos, por pessoas com as mais diversas intenções e competências. A Figura 1 caracteriza esta situação demonstrando que a dificuldade de interação é inerente ao próprio modelo.

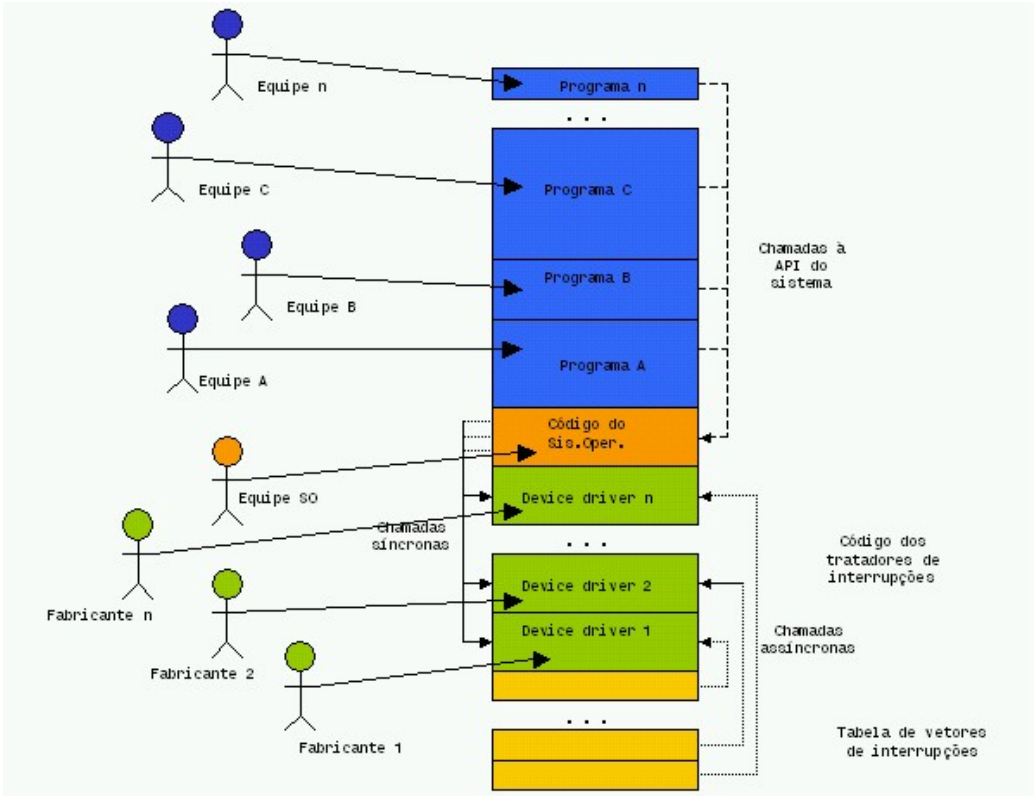


Figura 1 - Intromissão virtual de pessoas na máquina do usuário

Em verdade estão representados os grupos que desenvolvem o nível geralmente denominado de: software básico. A seguir aparecem os desenvolvedores do sistema operacional propriamente dito. Acima destes estão os desenvolvedores de software aplicativo. Cada equipe com seus particulares objetivos e requisitos de performance.

O surgimento do conceito de agentes representa uma forma inequívoca desta interpretação. O agente comporta-se como se fosse a incorporação virtual do programador ou da equipe que tem o conhecimento sobre o funcionamento de um programa ou área de aplicação. Ou seja, incorpora o fato de que, na realidade, quem tem o controle da situação é esta entidade externa. Embora seja uma solução interessante, esta metáfora é construída sobre uma base insegura (multiprogramação) onde não são considerados os aspectos de "convivência mútua" entre os "representantes dos vários desenvolvedores externos"..Este aspecto, em particular, colabora de forma importante para agravar as questões de segurança. Isto porque cada programa, individualmente, não sabe o que estão fazendo ou pretendem fazer os demais programas que, em determinado momento, concorrem pelas fatias de tempo do processador. E o SO também não tem esta informação. Assim, o risco é iminente a cada fatia de tempo.

Portanto, o tripé conceitual: **virtualização do operador**, **virtualização do hardware** e **conceito de programa**, caracterizam os fundamentos básicos do atual modelo de construção de sistema operacional, em particular, e de soluções em software, em geral. E, conforme será mostrado neste trabalho, problemas identificados já há quase 3 décadas persistem através das várias gerações de tecnologias desenvolvidas sobre este tripé conceitual.

A preocupação com qualidade em projetos de sistemas operacionais não é recente. Nicol em 1987 afirmava que: "*A história recente· destaca o esforço dos projetistas de sistemas em aspectos quantitativos (processadores mais rápidos, maior capacidade de memória, maior eficiência, tempo de resposta mais rápido, etc). Em função disto, menos atenção tem sido dada à qualidade do serviço prestado ao usuário. Como conseqüência, um maior esforço têm sido dedicado a interfaces gráficas de comunicação, facilitando a interação com o usuário*".

Fazendo-se uma analogia entre a situação acima descrita e aquela associada à filosofia de favorecer o programador em detrimento do usuário, pode-se afirmar que, do ponto de vista do usuário final, a atual tecnologia de sistemas operacionais pode ser classificada no mesmo nível evolutivo daquele caracterizado pelo advento das linguagens de alto nível. O usuário já pode usar comandos mais abstratos para operar a máquina, através de ícones e interfaces gráficas, mas ainda tem que se preocupar com o nome de pastas e arquivos e configurações técnicas como *url,smtp,dns*,etc., as quais muitas vezes fogem do seu escopo de conhecimento e o impedem de ser produtivo em seu ambiente de trabalho. Para exemplificar esta situação, basta consultar a tela principal de configuração do *browser* Internet Explorer da Microsoft - o qual é amplamente utilizado atualmente. São sete pastas (algumas contendo sub-pastas) contendo opções de configuração que nem sempre o usuário sabe exatamente qual a sua finalidade.

Segundo Ahmed (1998), a indústria de microprocessadores está em meio a uma revolução graças a uma série de recursos tais como: reutilização de projetos, melhores práticas em engenharia de lógica digital e eficiência de fabricação. A indústria de software também está vivenciando um processo evolutivo que, apesar de não tão intenso quanto o de hardware, caracteriza-se por: mudanças nos modelos de construção de software, de disponibilidade de recursos de hardware e software de apoio (ferramentas *CASE*), de interfaces de comunicação homem-máquina e de distribuição de recursos de hardware e software (através da facilidade cada vez maior de interligação de equipamentos).

Não obstante este avanço há uma área que tem apresentado pouca evolução perante o quadro que foi apresentado anteriormente – a área de sistemas operacionais. E esta evolução tem ocorrido no sentido de encontrar-se uma solução de compromisso entre a demanda de necessidades cada vez maiores dos usuários e, em contrapartida, a evolução muito rápida do hardware.

É inegável que apesar dos ganhos em produtividade o modelo atual tem apresentado deficiências clássicas. Um dos fatores críticos neste processo permanece sendo o software e, mais particularmente, o sistema operacional. Ele é o responsável imediato por uma grande parcela de contribuição para que o computador ainda seja inacessível a grande parte da sociedade, na medida em que ainda é complicado, instável e varia muito de equipamento para equipamento.

A possibilidade de acesso a informações sob demanda em qualquer local caracteriza uma tendência de pesquisa multidisciplinar cada vez maior, principalmente a partir da disseminação de redes sem fio (*wireless*) e tecnologias de interconexão cada vez mais disponíveis (*hardware* e protocolos). A proposta de espaços inteligentes abre uma nova dimensão, em termos computacionais, comparada àquela aberta com a introdução da tecnologia de redes. Deixou-se de ter equipamentos independentes e passou-se a contemplar o compartilhamento de recursos. Agora se quebra o modelo de interligação de computadores e abre-se uma nova dimensão, permitindo a integração ao ambiente computacional de outros equipamentos que hoje requerem configuração, coordenação e tratamento diferenciados. Neste contexto, os pesquisadores do laboratório Pablo (PABLO,2001) da Universidade

de Illinois, afirmam que a confluência de tecnologias de redes móveis, monitores *head-mounted,* vídeo digital de alta definição e ambientes virtuais imersivos, acenam para uma nova era. É possível prever que, nesse novo cenário, uma poderosa base tecnológica poderá aumentar a capacidade intelectual humana. No futuro, o ambiente de informações deverá apresentar, entre outras características, a mobilidade, a disponibilidade e sensibilidade ao contexto em que operam.

Há várias implicações tanto tecnológicas como sociais nesta visão de ambiente computacional, visto que, de um lado, a proposta envolve alterações em toda a infra-estrutura que conhecemos hoje e, portanto, requer grandes investimentos para alimentá-la; de outro, isola ainda mais sociedades que não dispõem dos recursos financeiros necessários à aquisição e assimilação da tecnologia. Wang e Garlan (2000) afirmam que a infra-estrutura computacional de hoje não suporta o modelo de computação de forma adequada, tendo em vista que os computadores interagem com os usuários em termos de abstrações de baixo nível: aplicações e dispositivos individuais. O grande desafio consiste, portanto, em adequar os requisitos identificados nos chamados espaços inteligentes, teletrabalho e comércio eletrônico, com aqueles necessários à geração de uma tecnologia que, ao mesmo tempo, dê suporte ao projeto tecnológico e consiga tornar as tecnologias computacionais mais eficientes, amigáveis e adaptáveis, ou seja, possibilitem a reconfiguração do sistema operacional em função de alterações no ambiente, de forma a minimizar a necessidade de intervenção do usuário. Porém, grande parte do esforço de pesquisa atualmente desenvolvido, adota a abordagem de promover pequenas alterações no modelo atual de sistemas computacionais, partindo do princípio de que este modelo é aparentemente a melhor solução.

Os argumentos acima apresentados, bem como os que são apresentados no trabalho de Mattos(2003), permitem afirmar que o modelo sobre o qual o desenvolvimento de aplicações está baseado necessita de uma revisão conceitual importante. Isto porque, apesar do enorme avanço tecnológico, problemas identificados especificamente na área de sistemas operacionais há 28 anos atrás, permanecem presentes. Saliente-se que, no passado, muitos problemas eram decorrentes da falta de qualidade do software, da rotatividade de pessoal, do elevado número de programadores envolvidos e da pouca prática no desenvolvimento de software e hardware (Deitel,1990). Hoje, embora contemos com uma área de Engenharia de Software com vasta experiência acumulada, com métodos de gerência de projetos amplamente conhecidos, altos níveis de abstração suportados por ambientes de desenvolvimento e linguagens de programação, os problemas continuam recorrentes. O enfoque que o mercado selecionou, não necessariamente fundamentando-se em critérios científicos, mas em leis de mercado, baseia-se em propostas originadas na década de 70 (**Unix**) e 80 (**Windows 2000** – baseado no *kernel Mach*). Embora haja alguma independência entre projetos de pesquisa e as soluções comerciais, a base conceitual continua praticamente a mesma: usuário, programa, multitarefa.

Em função dos argumentos arrolados e considerando as questões apresentadas em Bomberger (1992) e Colusa (1995), é possível afirmar que somente será possível minimizar os problemas relacionados à aplicação de tecnologias de informação nas áreas de computação ubíqua, comércio eletrônico e teletrabalho, a partir de um novo modelo de sistemas operacionais. Se for feita uma comparação entre as projeções futuristas e a situação dos dias de hoje, não é difícil verificar a necessidade de profundas mudanças nos conceitos atuais, sobre os quais está baseada a indústria de software. Devem ser considerados, a partir dessa nova ótica, os problemas ainda não totalmente solucionados no contexto de computadores pessoais (*desktop*, *palmtop*, *pda*, etc), tais como: composição de componentes, reconfiguração automática, adaptação dinâmica, gerência de recursos e energia, intercomunicação transparente e metáforas de acesso às informações.

Não obstante os argumentos apresentados, a chamada da IEEE (Cole,2003) reconhecendo que a base sobre a qual as TICs (tecnologias de informação e comunicação) estão sendo concebidas necessita de reformulação, e conclamando a comunidade acadêmica e comercial a fazer parte de uma

força-tarefa encarregada de propor a base para a construção de sistemas operacionais seguros, reforça a importância de uma nova abordagem na construção de sistemas operacionais. Assim sendo, uma nova concepção de sistema operacional, baseado em conhecimento, constitui-se não só como elemento de estoque de informação, mas também como fator de geração de novas demandas.

4 O paradigma de construção de soluções em software

Neste ponto do trabalho inserimos a Figura 2a que representa o contexto onde os projetistas de sistemas operacionais trabalham e, em última instância, a visão que o sistema operacional tem do contexto onde está operando. Os projetistas de sistemas operacionais consideram basicamente as características do hardware sobre o qual o sistema operacional será construído, tais como: (i) Velocidade de processador; (ii) Formato de instruções; (iii) Freqüência de relógio; (iv) Características de barramento; (v) Características dos controladores de periféricos, etc.

Por outro lado, inserimos a Figura 2b que representa a visão sobre a qual está baseada toda a tecnologia de computação. Esta visão pode ser resumida da seguinte forma: cada aplicação em particular é concebida para ser executada em uma máquina: (i) Abstrata exportada pelo conjunto hardware físico e sistema operacional; (ii) Contendo dispositivos lógicos sempre disponíveis e sem limite de tamanho; (iii) Com um processador 100% do tempo disponível para a aplicação; (iv) Tendo como meio de comunicação de ocorrências físicas ou lógicas a abstração de mensagens; (v)Com uma capacidade de memória infinita; (vi) E que, através de meta-canais de comunicação (RPC, Mensagens, *Jini*, CORBA, RMI), pode comunicar-se com outras aplicações (locais ou remotas) e com elas trocar informações de alto nível. Sobre esta máquina abstrata é que são desenvolvidas e projetadas soluções de Computação Ubíqua, Comércio Eletrônico, Ensino à Distância, Teletrabalho e as demais tecnologias de informação conhecidas. Os métodos de concepção e ferramentas de desenvolvimento de software também utilizam esta máquina abstrata como premissa básica. As soluções para os problemas apresentados anteriormente também partem desta visão.

O aspecto mais importante a ser ressaltado é que a visão que os hackers geralmente utilizam para a construção de vírus de computador e a partir da qual realizam seus ataques, ignora as abstrações promovidas pelo uso de tecnologias como Orientação a Objetos/Aspectos, Reflexão Computacional ou qualquer outra tecnologia empregada e segue rigorosamente a visão apresentada à esquerda (Figura 2a). As soluções de segurança geralmente utilizam o sistema de arquivos para armazenamento das senhas, embora criptografadas. Ou então envolvem listas de controle de acesso ou permissões de acesso a um sistema de arquivos em particular. Ocorre que a abstração de sistema de arquivos é muito alta. Ela existe desde há muito tempo para facilitar o processo de desenvolvimento de aplicações e não para armazenar dados importantes. A melhor solução de controle de acesso pode ser facilmente quebrada a partir de alguma das seguintes estratégias: (i) Acesso ao dispositivo físico através da instalação de mais de um sistema operacional na mesma máquina; ou, (ii) A instalação de um disco rígido (ilicitamente obtido) como dispositivo secundário em uma outra máquina de mesma arquitetura, disponibiliza o acesso às informações lá armazenadas, à revelia de senhas, listas de controle de acesso e toda a parafernália de segurança projetada a partir das abstrações apresentadas à direita da linha pontilhada.

Além disso, o enfoque principal concentra-se no sentido de instrumentalizar o sistema operacional a partir do uso de técnicas de extração de informações apresentadas em Skjellum(2000). Observa-se claramente que as abordagens, via de regra, localizam-se na fase 2 do ciclo evolutivo dos sistemas operacional, apresentada em Mattos(2003). Ou seja, baseiam-se na construção de bibliotecas especializadas que executam sobre o sistema, as quais, numa versão seguinte, poderão incorporar o escopo do sistema ou não (dependendo de fatores estratégicos geralmente orientados pelo mercado). Para isto lança-se mão das seguintes técnicas: Inteligência Artificial, Agentes, Análise de perfil baseada em histórico e Visualização de informação entre outras.

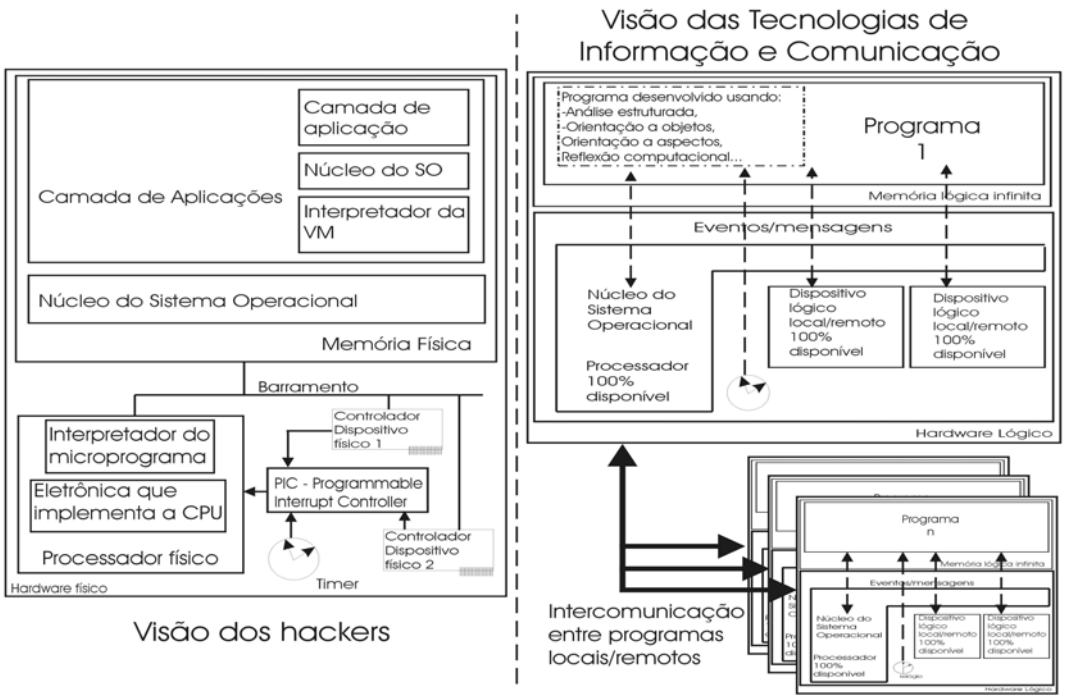


Figura 2 – (a)Modelo de hardware – (b) modelo de software

5 Contribuições da área de IA e Robótica

Segundo Nicol et al. (1989),"*Os sistemas operacionais devem ser o fator determinante nesta reformulação, já que é a partir das abstrações fornecidas pelos mesmos que são construídos os programas de aplicação*". Por outro lado, pesquisas na área de robótica, adotando concepções e abstrações completamente diferentes, já há muito tempo tem demonstrado projetos bem sucedidos que envolvem aspectos como autonomia e comportamento inteligente, os quais os projetos de sistemas operacionais estão longe de atingir. Estes aspectos, por si sós, caracterizam a necessidade de uma revisão dos conceitos da área de sistemas operacionais, mas este fato fica ainda mais evidenciado a partir da análise das demandas projetadas e da forma como são utilizadas tecnologias de informação para fazer-lhes frente conforme descrito em Mattos(2003).

Neste sentido, tomando-se como referência o trabalho de Mattos (2003) e comparando-se: (i) as características apresentadas e os requisitos dos projetos de sistemas operacionais identificados naquele trabalho, (ii) os requisitos de computação ubíqua projetados como a próxima grande área de pesquisas a ser atacada; (iii) a forma como tradicionalmente as demandas não previstas pelos projetistas de sistemas operacionais são corrigidas e, (iv) os requisitos dos projetos de robótica, percebe-se claramente que a maioria destas questões permanece em aberto ainda hoje. É interessante observar que a categoria de projetos de robótica transcendeu há muito tempo ao modelo de "fazer-pela-máquina", ou seja, transcendeu ao modelo de programação pré-estabelecida da seqüência de ações a serem executadas. Isto foi possível a partir da adoção de quatro conceitos importantes: (i) O conceito de mundo; (ii) O conceito de incorporação (*embodiment*); (iii) O conceito de planos em substituição ao conceito de programa; e (iv) O conceito de aprendizagem. Estas questões dominaram o centro das discussões na primeira fase dos projetos de robótica e de Inteligência Artificial Simbólica. Enquanto isso, os projetos de sistemas operacionais tradicionais mantêm-se fiéis ao

modelo de Entrada-Processamento-Saída e ao conceito de multiprogramação (que virtualiza o *hardware*).

6 SOBC: uma proposta de solução

Conforme apresentado em Mattos(2003), a idéia de sistema operacional baseado em conhecimento não é recente. No entanto, invariavelmente as propostas foram concebidas a partir do modelo atual, sem atacar diretamente a questão do tripé conceitual que o caracteriza. Em conseqüência, as abordagens acabaram desviando-se dos objetivos iniciais e derivando os esforços no sentido de aplicar técnicas da Inteligência Artificial sobre algum tipo de núcleo que suporta multitarefa contribuindo, sem dúvida, para facilitar a utilização da máquina, mas efetivamente distanciando-se do que deveria ser um Sistema Operacional Baseado em Conhecimento.

Piaget (1977) apud Costa (1993) afirma que: "*nenhum trabalho científico pode avançar metodicamente sem que seu objeto de estudo esteja definido, o que se impõe, sem dúvida, para delimitar o domínio de que ele se ocupa*". Assim, como os trabalhos anteriores em sistemas operacionais baseados em conhecimento não se apoiaram em uma definição adequada, em verdade seu domínio de interesse, até aqui, não ficou perfeitamente delimitado. Por conseguinte, o emprego de alguma ou várias técnicas de inteligência artificial na área de sistemas operacionais, segundo o paradigma atual, passou a ser erroneamente caracterizada como um sistema operacional baseado em conhecimento. Definir o que é um sistema operacional baseado em conhecimento, portanto, é um passo preliminar necessário para esclarecer o objeto do estudo em questão. Assim sendo, e tendo esta concepção bem clara, faz-se imprescindível caracterizar o que se entende por um sistema operacional baseado em conhecimento.

Um Sistema Operacional Baseado em Conhecimento é: **Um sistema de software incorporado, localizado, adaptável e autônomo, baseado em conhecimento, que possui identidade e conduta inteligente quando executado.**

Um aspecto de fundamental importância no modelo proposto em Mattos (2003) é que, ao contrário dos tradicionais modelos de construção de núcleos de SO (monolítico, camadas, ...) representados na Figura 3a, , onde fica clara a tendência de permitir às aplicações estender a funcionalidade do sistema a partir de *plug-ins* instaláveis pelo usuário, no modelo descrito em Mattos (2003), as aplicações afetam o mundo e sentem, instantaneamente, o efeito de outras entidades que nele atuam (Figura 3b). Ou seja, ao invés de deixar as aplicações penduradas ao redor do núcleo, o modelo proposto as embute dentro do modelo de mundo proposto. Assim não se faz necessário um mecanismo de mensagens na forma tradicional de sinalização via troca de mensagens (mensagens *send*/*receive*). A diferença entre o proposto e os modelos reflexivos é que nestes a aplicação não fica sabendo que houve um *trap* (reificação) para o meta-nível, enquanto no modelo proposto há uma constatação imediata e explícita deste fato, já que o código a ser executado é disparado em função dos parâmetros: estado do mundo/prazo. Sob a ótica do desenvolvimento de aplicações, o modelo proposto conduz a uma evolução na forma como ocorre o processo de desenvolvimento de soluções em software. Isto porque se passa de um modelo de ferramentas case gráficas, onde o projetista manipula ícones, para um modelo de ferramenta case interativa, onde o conhecimento fornecido é validado e assimilado pelo sistema. O que ocorre no modelo atual pode ser caracterizado através da seguinte analogia: ao não ensinarmos o sistema operacional a resolver os problemas, jamais poderemos dispor de um verdadeiro assistente pessoal no rigor da expressão. Isto porque os programas de aplicação continuam sendo construídos segundo a concepção de quem os desenvolveu (programou).

A questão da aprendizagem e adaptação, quando eventualmente existe, é construída sob a ótica de quem (o SO) jamais terá condição de prever os efeitos de todas as possíveis implicações que as

generalização e abstrações disponibilizadas no programa terão no contexto do usuário. Além disso, os sistemas de aprendizagem caracterizam-se também como uma casca (*Shell*) que, para o sistema operacional, continua sendo como qualquer outro programa em execução. Assim sendo, o que cada programa individualmente "aprendeu" fica restrito a ele e este conhecimento não é compartilhado pelas demais aplicações e pelo sistema operacional.

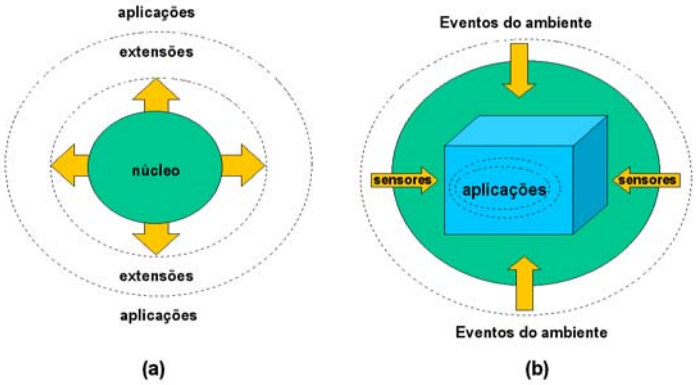


Figura 3 – Comparativo entre modelos exógenos (a) e o modelo proposto endógeno (b)

Para que seja possível atender aos requisitos identificados, uma nova geração de sistemas operacionais terá que verificar e gerenciar todos os aspectos relativos às atividades que estão ocorrendo na máquina em determinado momento. Assim sendo, algumas abstrações, tal como são conhecidas hoje, deverão ser absorvidas, dando origem a uma nova concepção de aplicação e forma de interação. Outra abstração importante que deve desaparecer nesta nova geração é o conceito de programa como é conhecido hoje – um código binário, uma configuração de bits compatível com a arquitetura do processador alvo, que recebe uma fatia de tempo para ser executada e, neste momento, possui o controle completo do mesmo, respeitadas as restrições tradicionais de visibilidade de memória, direitos de acesso, etc.

Se o sistema possui identidade, é ele quem decide quanto de seus recursos emprestar à coletividade (SMA), dependendo do que ele está encarregado de fazer no momento. Máquinas de casa podem ter comportamentos diferentes das máquinas utilizadas em ambientes comerciais e o conhecimento adquirido nestes ambientes será diferente.

Finalmente, é possível afirmar que o modelo de mundo proposto é uma estrutura hiperdimensional que permite às aplicações perceber instantaneamente quaisquer alterações ambientais. Da mesma forma, cada uma das aplicações pode "sentir" o efeito de outras que estão executando simultaneamente. Isto porque qualquer flutuação na disponibilidade de algum recurso, imediatamente altera o estado dos sensores da dimensão lógica. Como toda a aplicação precisa estabelecer o que fazer em cada situação, a reatividade é inerente ao modelo.

7 Conclusão

O aspecto mais relevante deste trabalho é a constatação sobre o fato de que, apesar do enorme esforço desenvolvido pelos pesquisadores da área de sistemas operacionais, problemas identificados por Linde em 1975 e, portanto, há aproximadamente 28 anos atrás, ainda continuam presentes e recorrentes. Isto permite concluir, irrefutavelmente, que as abordagens atuais não são suficientemente adequadas para atender aos requisitos esperados dos sistemas computacionais atuais. O trabalho conclui introduzindo o conceito de um modelo de sistema operacional baseado em conhecimento que procura adotar conceitos sedimentados nas áreas de IA e Robótica no sentido de incorporar conhecimento e comportamento inteligente no âmbito de um sistema operacional.

Referências

AHMED,O. **The Application-specific Operating System**. Computer Design, p. 78, oct. 1998.

BRACHMAN,R.J. **Systems That Know What They're Doing**. IEEE Intelligent Systems. November/December 2002. p.67-71.

BOMBERGER,A.C. *et al*. **The KeyKOS Nanokernel Architecture**. Proceedings of the USENIX Workshop on Micro-Kernels and Other Kernel Architectures, 1992.

BONA,C. **Avaliação de Processos de Soft**ware: Um Estudo de Caso em XP e Iconix. Dissertação (Mestrado) Eng. de Produção e Sistemas, Universidade Federal de Santa Catarina, UFSC, Florianópolis, 2002.

COLE,J. **IEEE begins standard to create baseline for more secure operating systems**. Disponível em: <http://standards.ieee.org/announcements/pr_p2200.html>. Acesso em 20 out. 2003.

COLUSA. **Omniware Technical Overview**. Colusa Software White Paper. Disponível em: <http://www.cs.umbc.edu/ agents/papers/omniware.ps>. Acesso em 15 mar. 1995.

COSTA,A.C.R. **Inteligência de Máquina**: Esboço de uma Abordagem Construtivista. Tese (Doutorado). Pós-Graduação em Ciência da Computação, Universidade Federal do Rio Grande do Sul, UFRGS,Porto Alegre, 1993.

DEITEL,H.M. **An Introduction to Operating Systems**. Addison-Wesley Publishing Company, Reading, Massachusetts, 1990.

DE SOUZA,C.S. **The Semiotic Engineering of User Interface Languages**. International Journal of Man-Machine Studies, v.39,p.753-773. 1993.

FERREIRA,M.P. **Desenvolvimento de Software Alinhado aos Objetivos Estratégicos do Negócio**: Proposta de Uma Metodologia. Dissertação (Mestrado em Eng.de Produção) Eng.de Produção e Sistemas, Universidade Federal de Santa Catarina, UFSC,Florianópolis, 2002.

MATTOS,M.M. **Fundamentos conceituais para a construção de sistemas operacionais baseados em conhecimento**. Tese (Doutorado). Pós-Graduação em Eng.de Produção, Universidade Federal de Santa Catarina, Florianópolis, 2003.

MÖLLER,B. **Relations as a program development language**. IFIP TC2 Conference. In B. Möller (ed.): Constructing programs from specifications.North Holland , Pacific Grove, USA, 1991.

NAGLE,D.,et al. **Optimal Allocation of On-chip Memory for Multiple-API Operating Systems**. In Proc.of the 21st Annual International Symposium on Computer Architecture, p. 358-369, apr. 1994.

NICOL, J. **Operating System Design**: Towards a Holistic Approach. Operating System Review. v.21, n.1, 1987.

NICOL, J. *et al*. **Cosmos**: An Architecture For a Distributed Programming Environment. Computer Communications, v.12, n.3, p.147-157, 1989.

PABLO. **Intelligent Information Spaces**: A test bed to Explore and Evaluate Intelligent Devices and Augmented Realities. Disponível em: <http://www-pablo.cs.uiuc.Edu/Project/SmartSpaces/ SmartSpaceOverview2.htm>. Acesso em 3 mai.2001.

SENRA,N.C. **Informação estatística**: política, regulação , coordenação. Revista Ciência da Informação – Políticas e Gestão da Informação. V.28, n.2, p.124-125, maio/ago. 1999.

OETTINGER,A.G. **O Uso do Computador na Ciência**. Scientific American 1966. Computadores e Computação:Textos do Scientific American. Editora Perspectiva. 1977, 332p.

SKJELLUM,A. *et al*. **Systems Administration**. Cluster Computing White Paper Version 2.0. Cap 6. Mark Baker – Editor. University of Portsmouth, UK. Dec. 2000.

TKACH,D.;PUTTICK,R. **Object technology in application development**. IBM International Technical Support Organization, The Benjamin/Cummings Publishing Co.,Inc. 1994. 212p.

WANG, Z.; GARLAN, D. **Task-Driven Computing**. Technical Report, CMU-CS-00-154, School of Computer Science, Carnegie Mellon University, May 2000.

WATER,J.K. **Microsoft Unveils Windows Server 2003**. Application Development Trends, June 2003, p.15.

Uma Biblioteca de Processamento de Imagens para o Controle de Qualidade utilizando Dispositivos Portáteis (PDAs) em Sistemas Industriais de Visão

Mário Lucio Roloff (UFSC)
roloff@das.ufsc.br

Marcelo Pires Adur (UFSC)
mpadur@das.ufsc.br

Resumo. O projeto SISPORT é o resultado de um projeto de pesquisa e desenvolvimento de um produto *turn-key* para atender a demanda de sistemas industriais de visão de baixo custo e alta portabilidade. Baseado na arquitetura de PDAs (*Personal Digital Assistant)* , foram desenvolvidas biblioteca para a aquisição, comunicação e processamento de imagem para efetuar o controle da produção.

Palavras-chave: Integração Software/Hardware. Sistemas Industriais de Visão. Processamento de Imagem. Integração da Manufatura. C++. Engenharia de Software. Monitoração da Produção. Controle de Qualidade.

1 Introdução

Neste artigo destaca-se o desenvolvimento da biblioteca do sistema de visão computacional para o PDA (Personal Digital Assistant) do SISPORT – Sistemas Portáteis de Visão. Nas próximas seções será apresentado o módulo de processamento de imagens do SISPORT, chamado de PxImgProcess, esta biblioteca será discutida e os resultados alcançados serão apresentados.

O controle da qualidade e a obtenção de dados confiáveis nos processos de produção são uma necessidade cada vez maior das empresas em diversos setores. A ocorrência de erros nos processos produtivos pode gerar desde pequenos prejuízos (como custos de retrabalho ou matéria-prima perdida) até prejuízos que acabem por afetar a saúde financeira de companhias, como ações judiciais decorrentes destas falhas. Como exemplo, em indústrias farmacêuticas, pode-se perceber claramente os prejuízos que uma falha de produção pode acarretar, um remédio trocado, uma embalagem com defeito, uma posologia mal escrita, são exemplos de problemas cujas conseqüências podem se tornar muito graves. Os sistemas industriais de visão tem como objetivo a análise dos dados de produção de forma a garantir que produtos defeituosos não prossigam nos processos.

Contudo, apesar de permitirem a economia decorrente da eliminação de produtos defeituosos, os sistemas complexos de visão (àqueles que permitem uma visão geral do processo inteiro, ao invés de focalizarem apenas em pontos específicos) ainda são bastante caros, uma vez que são necessários vários sistemas individuais interligados para garantir a inspeção 100% e índice de erros 0%. A redução no custo destes sistemas individuais possibilitaria a utilização de sistemas de visão em maior escala. Em grande parte das indústrias os custos com retrabalho aumentam exponencialmente de acordo com o ponto em que os defeitos são identificados. Nestes casos justifica-se a utilização de literalmente dezenas de sistemas individuais em chão de fábrica. A portabilidade inerente a este sistema possibilitará que a configuração da rede se adapte plenamente às necessidades de cada empresa.

O uso da combinação da tecnologia PDA (Personal Digital Assistant) com o desenvolvimento de uma biblioteca dedicada ao processamento de imagens para a plataforma dos PDA possibilita a portabilidade e a expansão dos sistemas de visão nas linhas. Outro objetivo que busca-se com esta biblioteca desenvolvida para o projeto SISPORT é uma maior praticidade, praticidade quanto à instalação do sistema na linha e também quanto ao uso, configuração e monitoração do ponto de inspeção.

A Figura 1 apresenta a idéia do uso de sistemas portáteis de visão em vários pontos da linha com um ou mais sistemas gerenciadores, podendo ser um notebook ou um desktop, que via uma rede wireless possui a capacidade de visualização da planta da linha com o status dos sistemas de visão. (As linhas ligando os sistemas portáteis e os computadores são para exemplicação da malha de controle)

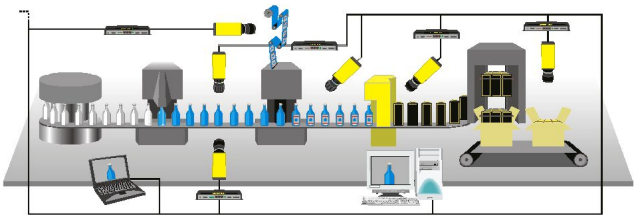


Figura 1: Exemplo de linha com sistemas de visão portáteis

2 Metodologia

Uma metodologia completa constitui-se de uma abordagem organizada para atingir um objetivo, através de passos preestabelecidos. É um roteiro para desenvolvimento estruturado de projetos, sistemas ou softwares, visando a qualidade e produtividade de projetos.

Metodologia não é uma técnica tão somente, pois pode-se utilizar qualquer técnica para o desenvolvimento de projeto, sistema ou software, de acordo com a preferência e competência da equipe multidisciplinar envolvida. Como exemplo de técnicas, podem ser utilizadas a Análise Estruturada, Análise de Pontos por Função, Análise Essencial, Análise Orientada a Objetos entre outras técnicas e suas respectivas ferramentas.

O desenvolvimento de projetos, sistemas ou software pode ser dividido em cinco fases, as quais são desmembradas e subfases e cada uma destas subfases geram pelo menos um produto quando de sua elaboração.

As fases para o desenvolvimento de projetos, sistemas ou software são:

· estudo preliminar ou anteprojeto ou estudo inicial ou primeira visão;

A partir de do estudo inicial, obteve-se uma visão geral de como deveria funcionar o sistema como um todo, o qual é formado por vários módulos e ficou clara a necessidade de uma biblioteca para processamento de imagens.

· análise do sistema ou tecnologia ou reconhecimento do ambiente;

A análise levou à conclusão de que seria necessário o desenvolvimento de um módulo de processamento de imagens que disponibilizasse ao aplicativo as funções a ele atribuídas de maneira simples.

· projeto lógico ou especificação do projeto ou design;

O projeto prevê a disposição dos módulos e como eles devem interagir entre si, bem como quais funções devem ser oferecidas por cada um. Assim sendo foram definidas funções para o módulo de processamento de imagens como:

· *Alocação e deslocação de imagens:* Função designada para "instanciar" ou inicializar uma imagem que será utilizada como parâmetro de entrada para as operações, filtros e ferramentas de processamento da biblioteca.

· *Operações morfológicas:* Funções utilizadas para por em evidência certas características da imagem como contornos, realçar partes claras ou escuras ou reduzir ruídos. Pode-se citar como exemplo as funções: Canny, Erode, Dilate, Open, Close, Gradiente, ente outros.

· *Operações lógicas:* São utilizadas para se realizar funções lógicas entre os pixels de duas imagens, assim como se faz com qualquer sinal digital. Exemplos: AND, OR, NOT, XOR, NAND, entre outros.

· *Filtros e conversão de cores:* Úteis quando se deseja binarizar uma imagem (atribuir a todos os pixels os valores 0 ou 1) ou converter uma imagem colorida para tons de cinza, por exemplo. São funções muito utilizadas. Pode-se citar: Threshold para imagens coloridas ou em tons de cinza, e CvtColor.

· *Ferramentas específicas:* São conjuntos de operações e filtros otimizados para a execução de uma tarefa específica. Aqui encontra-se o grande alvo de pesquisa e trabalho dessa biblioteca, visto que se trata de utilizar as ferramentas mencionadas acima para gerar resultados intermediários os quais devem ser tratados e analisados para a realização da tarefa que retornará não uma imagem, mas sim um resultado ao aplicativo. Este resultado normalmente é da forma aprovado ou reprovado, OK ou NOK. As tarefas designadas a essas ferramentas são Leitura de código bidimensional Data Matrix, leitura de código de barras (Figura 2), leitura de códigos do tipo PharmaCode, OCR (Object Character Verification), OCV (Object Character Recognition), BLOBs.

Figura 2: Exemplo de Código de Barra e Data Matrix

· projeto físico ou execução ou implementação do projeto ou programação;

A implantação do módulo de processamento de imagens se deu através da programação de uma biblioteca dinâmica C/C++. A biblioteca possui uma interface padrão para ser usada como um dos componentes de aplicativos dedicados a aplicações em sistemas de visão.

As ferramentas desenvolvidas foram reunidas em uma biblioteca dinâmica (PxImgProcess) e estão disponíveis para o aplicativo através de chamadas de funções de API (Aplication Program Interface) garantindo com isto a modularidade. Ver Figura 3.

Com isto, a partir de uma imagem e através da utilização de uma dessas ferramentas, ou de um conjunto delas, o software poderá obter resultados de interesse sobre a imagem, podendo

assim tomar decisões de aprovar ou reprovar o objeto em análise ou obter informações relevantes (por exemplo, dimensões, presença ou ausência de componentes).

Para o desenvolvimento da biblioteca de processamento de imagens utilizou-se o ambiente de desenvolvimento Microsoft Embedded Visual C++ 3.0 com a plataforma de desenvolvimento SDK Pocket PC 2002 juntamente com a biblioteca de processamento de imagens OpenCV, programada em linguagem C/C++ a qual teve que ser portada para o sistema operacional do PDA..

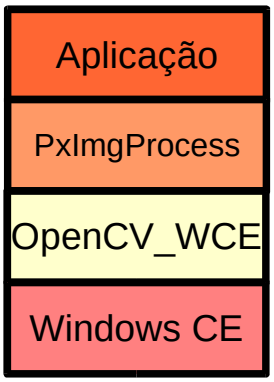


Figura 3: As camadas de software da biblioteca

· projeto de implantação ou projeto de disponibilização e uso;

Sendo a biblioteca de processamento o módulo fundamental do projeto SISPORT a mesma é integrada ao sistema através de sua API. A biblioteca juntamente com os outros módulos, interface, aquisição, comunicação e sinais (entradas/saídas) formam o Sistema de Visão Portátil onde através de um sinal do módulo de sinais, uma imagem é adquirida, processada e o resultado pode ser mostrado a um usuário ou ser utilizado para que o sistema atue sobre a planta. Ver Figura 4.

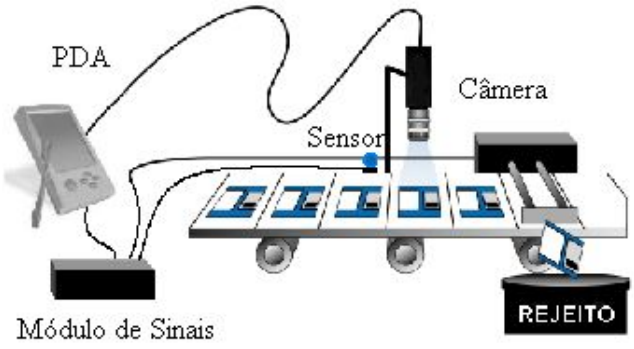


Figura 4: Visualização do Sistema de Visão usando PDAs

Como o SISPORT está sendo desenvolvido num âmbito de transferência de tecnologia, é importante a existência de mais duas fases:

· divulgação dos resultados no meio científico e comercial;

Busca-se aqui a divulgação dos resultados alcançados, através da publicação de artigos, da participação em feiras, congressos, exposições e seminários.

· transferência da tecnologia para a empresa cooperada;

O SISPORT resultou em uma patente de inovação onde os participantes do projeto, UFSC, FINEP e a empresa Pollux dividem os direitos de uso e exclusividade do projeto.

3 Resultados

As figuras seguintes mostram algumas das ferramentas desenvolvidas para a PxImgProcess e os resultados em um determinado objeto. Parâmetros como o tamanho da imagem captuarada e o tempo de captura da imagem irão depender principalmente das características da câmera e da placa de aquisição de imagens utilizados.

Os tempos de processamento alcançados em operações lógicas e filtros dependem principalmente do tamanho da imagem são relativamente maiores quando comparados aos tempos atingidos em um PC, principalmente devido à memória restrita. Porém para a maioria das aplicações, os tempos são satisfatórios.

Os tempos de processamento para aplicações específicas, como a leitura de código DataMatrix, dependem não somente do tamanho da imagem, mas também da qualidade da imagem e do número de informações contidas na marcação.

A ferramenta de BLOB tem como objetivo encontrar padrões na imagem adquirida baseados em características pré-determinadas, como: comprimento, altura, posição, quantidade, além da configuração dos filtros, threshold e do tipo de imagem a ser processada, isto tudo é pré-configurado pelo operador. Ver Figura 5.

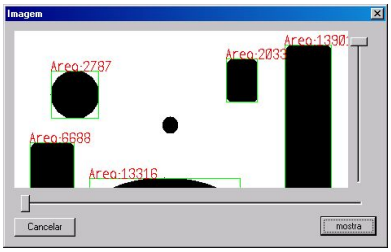


Figura 5: Ferramenta de BLOB

A ferramenta de Data Matrix (DMX) (Figura 6) objetiva a decodificação de uma código bidimensional muito utilizado no setor industrial, em especial no automotivo.

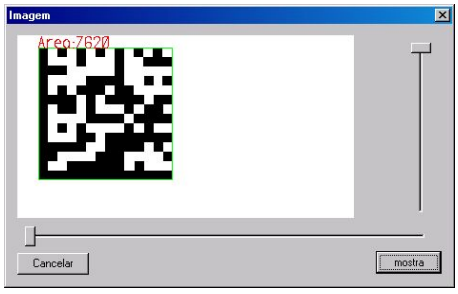


Figura 6: Ferramenta de Data Matrix (DMX)

Figura 7: Tela de configuração das ferramentas da biblioteca

Segue uma lista com os resultados obtidos direta e indiretamente com o desenvolvimento da biblioteca de processamento de imagens para PDAs em sistemas industriais da visão:

Desenvolvimento de biblioteca de aquisição e processamento de imagens para PDAs : objetivo alcançado, o projeto concluiu o desenvolvimento de ferramentas de análise de imagem de maior prioridade para a indústria.

Desenvolvimento de biblioteca especializada de comunicação para PDAs e computadores : desenvolvida e testada, alcançou 100%, inclusive foi divulgada e difundida para outros projetos da comunidade científica.

Desenvolvimento de arquitetura reduzida para sistemas de visão : o sistema foi completado com o PDA, a frame grabber, a câmera industrial, o módulo de sinais. No final do projeto, optou-se por realizar um projeto mecânico para encapsulamento de todo o equipamento para ambientes muito hostis.

Desenvolvimento de sistema de visão portátil e baixo custo : o custo dos equipamentos ficou em torno de 2000,00 US$ ou 6300,00 R$ (3.15 R$/US$) sem considerar o custo final de mercado do sistema, que a empresa não decidiu ainda. Mas o custo final deve ficar em torno de 8000,00 US$, que é o preço de uma câmera inteligente no mercado atualmente (que não oferece todas as vantagens de gerenciamento remoto, integração total com a linha e sistemas de rejeito e iluminação).

Desenvolvimento de biblioteca e hardware para comunicação entre PDAs e componentes eletro-eletrônicos : o módulo de sinais, como foi chamado, foi planejado, desenvolvido e prototipado. Seu desempenho atendeu as exigências tanto no uso com um PC normal como quando integrado com o PDA.

Além disso, os resultados conseguidos com o SISPORT, em especial, com o módulo de processamento de imagens (PxImgProcess) (Figura 7) podem se enquadrar nos aspectos: científico, tecnológico, econômico, social e ambiental.

No aspecto científico conseguiu-se o desenvolvimento de uma biblioteca de processamento de imagens portátil para sistemas baseados na arquitetura PDA. E ainda, contribuiu para o aperfeiçoamento da equipe executora do projeto e do grupo de pesquisas com a difusão dos

conhecimentos. Além disso, possibilitou a publicação de artigos, matérias e reportagens descrevendo a arquitetura criada.

No aspecto tecnológico possibilitou a pesquisa e desenvolvimento da biblioteca de aquisição e processamento de imagens. Além do desenvolvimento de uma biblioteca de comunicação entre os dispositivos via wireless. Com este avanço tecnológico também se conseguiu uma redução nos custos do sistema de visão e sobretudo um aumento incrível na portabilidade e aplicabilidade dos sistemas industriais de visão.

No aspecto econômico destaca-se a redução nos custos para a garantia da qualidade e no retrabalho.

Socialmente consegue-se uma melhoria nas condições do trabalho para os operadores ligados a área de controle de qualidade.

No aspecto ambiental com o módulo de processamento de imagens em conjunto com os outros módulos gera-se uma diminuição da geração de lixo industrial, em especial, nos setores industriais nos quais o retrabalho não pode ser realizado (exemplo, indústria farmacêutica).

4 Conclusões e Perspectivas

O projeto SISPORT, contando com uma boa infra-estrutura e uma equipe de qualidade, alcançou de forma plenamente satisfatória as metas expostas no cronograma físico, além de atingir outros resultados que não estavam previstos no cronograma. Destaca-se o aperfeiçoamento na metodologia de projeto e gerenciamento de atividades da equipe e a familiarização com novas ferramentas que permitem executar as tarefas de gerenciamento e controle do projeto com alta eficiência e qualidade.

Considera-se também excelente a cooperação universidade-empresa realizada no âmbito deste projeto, que pode ser considerada um exemplo na área.

Como perspectiva, destaca-se que o grupo de pesquisa já teve novo projeto aprovado no edital CT-INFO de software livre, o que, pelo menos em parte, se deve a experiência adquirida na execução do projeto SISPORT (Figura 8).

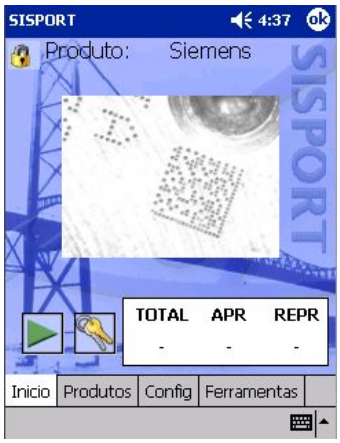


Figura 8: Tela inicial do sistema portátil de visão - SISPORT

De modo geral o módulo de processamento de imagens (PxImgProcess) superou as expectativas iniciais quanto ao desempenho e aplicabilidade ao projeto SISPORT.

A biblioteca OpenCV atendeu aos requisitos e por ter sido escrita em ANSI (padronizada) tornou-se menos complexo portá-la para a plataforma do PDA. Um fato inédito foi a portagem da biblioteca de processamento de imagens OpenCV para a plataforma PDA, compilada para o sistema operacional Pocket PC 2002 da Microsoft.

Outro ponto que merece destaque, são as ferramentas desenvolvidas que em conjunto resolvem a maioria dos problemas de controle de qualidade no setor industrial. Em especial as ferramentas, Blob, que permite a análise e classificação dos objetos com várias variáveis e o Data Matrix, que possibilita o armazenamento de grande quantidade de informações garantindo assim a rastreabilidade dos produtos, evitando, fraudes, roubos.

5 Agradecimentos

Ao FINEP pelo apoio institucional e financeiro na execução do projeto. À comunidade OpenCV cedendo sua biblioteca básica de processamento de imangens para desenvolvimento da versão para PDA. Ao CNPq pelo apoio financeiro na garantia das bolsas de pesquisa do participantes do projeto. À empresa POLLUX pela infra-estrutura fornecida ao projeto e apoio na forma de contra-partida financeira. À UFSC/CTC/DAS pela infra-estrutura e orientação científica fornecida para a realização do projeto SISPORT. A FAPEU pelo apoio institucional na divulgação dos resultados do projeto.

6 Referências

PROSISE, Jeff. Programming Windows with MFC – Second Edition. New York, 2000. Microsoft Press.

SEUL, Michael; O'GORMAN, Lawrence; SAMMON, Michael J.; Pratical Algorithms for Image Analysis – Description, Examples, and Code. UK, Cambridge, 2001. Cambridge University Press.

JAIN, L.C.; JOHNSON, R.P.; TAKEFUJI, Y.; ZADEH, L.A.; Knowledge-Based Intelligent Techniques in Industry. USA, New York, 1999. CRC Press.

SONKA Milan; HLAVAC Vaclav; BOYLE Roger; Image Processing, Analysis, and Machine Vision – Second Edition. USA, Philadelphia 1999. Brooks/Cole Publishing Company.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar; The Unified Modeling Language User Guide. USA, Boston, 1999. Addison-Wesley Longman Inc.

PARKER, J.R.; Algorithms for Image Processing and Computer Vision. USA, New Jersey, 1997. Wiley Computer Publishing.

INTEL Co.; Open source computer vision library - reference manual. USA, New Jersey, 1999-2001. INTEL Press.

INTEL Co.; Integrated performance primitives for intel architecture – reference manual. USA, New Jersey, 2000-2001. INTEL Press.

Uma Arquitetura para o Compartilhamento do Conhecimento em Bibliotecas Digitais

Nikolai Dimitrii Albuquerque (Grupo Stela - UFSC)
nikolai@stela.ufsc.br

Vinícius Medina Kern (Grupo Stela - UFSC)
kern@stela.ufsc.br

Resumo. O compartilhamento de recursos distribuídos, autônomos, heterogêneos e disponibilizados sem a mínima padronização gera a problemática da recuperação de uma elevada quantidade de informações que, na maioria das vezes, não atendem às necessidades dos usuários. Essa realidade é um desafio à comunidade científica que busca a integração, intercâmbio e semântica das informações. Este trabalho apresenta uma arquitetura de integração semântica de bibliotecas digitais, teses e dissertações, trabalhos de conclusão de curso, artigos, periódicos, Currículo Lattes etc. baseada no uso da tecnologia da Web Semântica. As ontologias das fontes de informação são armazenadas em um repositório onde as relações semânticas são estabelecidas, permitindo, assim, a interoperabilidade semântica dos dados. Essa estrutura proporciona um serviço com um maior nível de qualidade, tornando a Web uma ferramenta capaz de tecer uma rede semântica de informações e/ou conhecimentos.

Palavras-chave: Bibliotecas digitais, ontologias, web semântica, interoperabilidade semântica.

1 Introdução

A utilização de recursos na Web ganha cada vez mais importância devido ao fato de ser uma fonte universal de pesquisa, fato este evidenciado principalmente na educação. Essa demanda exige uma infra-estrutura global e um conjunto de padrões para publicação e recuperação de informações, já que a heterogeneidade, a autonomia e a ampla distribuição de dados sem uma padronização tornam o processo de recuperação complexo. Essa complexidade é demonstrada na forma como os usuários da Web efetuam suas pesquisas, com destaque a duas: navegação exaustiva, que dificilmente resulta em sucesso, e buscas por palavras-chave.

Essa realidade torna a Web um desafio para a comunidade científica, que busca sucesso na integração, no intercâmbio e na semântica das informações. Tal integração engloba o gerenciamento dos recursos, envolvendo a avaliação do conteúdo e de seus relacionamentos, e a padronização desses recursos, através da descrição de suas propriedades e da implementação de mecanismos que dêem suporte à descoberta e recuperação dos recursos. Várias iniciativas, como as desenvolvidas pelo *World Wide Web Consortium* (W3C), buscam por intermédio da criação de padrões, arquiteturas de metadados, serviços de inferência e ontologias, entre outros, a melhor forma de tornar as informações também compreensíveis pelas máquinas.

Visando apresentar a arquitetura para o compartilhamento do conhecimento em bibliotecas digitais, o restante deste artigo está estruturado da seguinte forma: na seção Web Semântica será apresentado o caminho para tornar os dados mais compreensíveis às máquinas; na seção Ontologias serão mostradas a representação formal do conhecimento, através dos vocabulários de conceitos, definições e possíveis propriedades, e as relações entre os conceitos e um conjunto de axiomas formais que restringem a interpretação dos conceitos e das relações; na seção Arquitetura de Integração Semântica será apresentada a estrutura de integração, que foi baseada na visão clássica das camadas de informação (BERGAMASCHI, 1999; CALVANESE, 1998); por fim será apresentada a conclusão.

2 Web Semântica

Segundo Daconta (2003), com o advento da Internet, da XML e, mais recentemente da Web semântica o valor até então agregado às aplicações deslocou-se para os dados. Esse fato define essencialmente o que é a Web Semântica. O caminho para tornar os dados mais compreensíveis às máquinas é torná-los mais inteligentes. A Figura 1 demonstra os quatro estágios da evolução dos dados, que se tornam continuamente mais inteligentes. Esses estágios evoluem de dados minimamente inteligentes a dados embutidos de informações semânticas suficientes para que as máquinas possam fazer inferências.

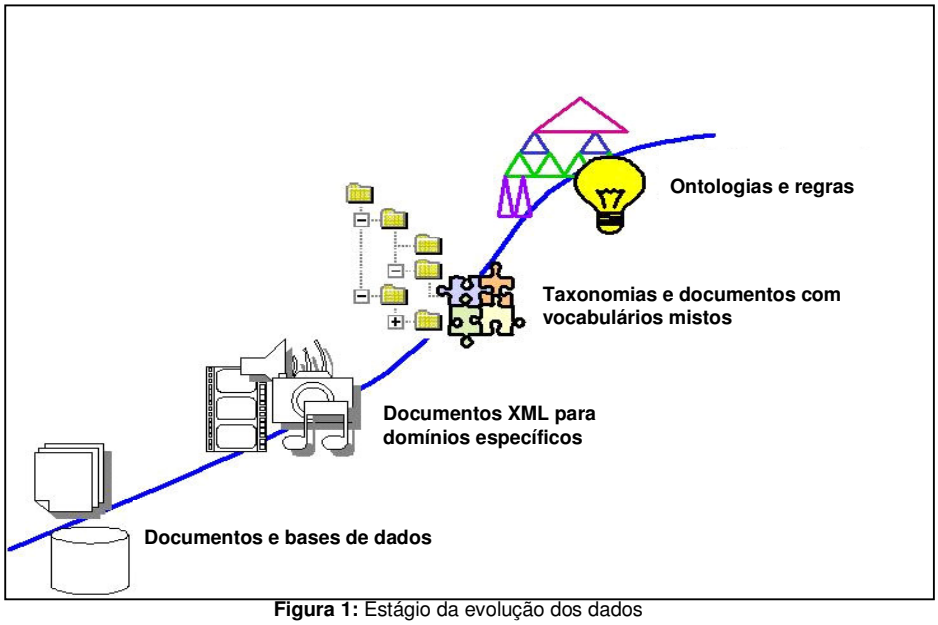


Figura 1: Estágio da evolução dos dados

O detalhamento dos quatros estágios da evolução dos dados é apresentado a seguir.

- **Texto e banco de dados (pré-XML):** o estágio inicial em que a maioria dos dados eram propriedades de uma aplicação. Portanto, a inteligência estava na aplicação e não nos dados.
- **Documentos XML para domínios específicos:** o estágio em que os dados se tornam independentes da aplicação dentro de um domínio específico. Os dados agora são inteligentes o suficiente para se moverem entre aplicações de um domínio. Um exemplo disso poderiam ser os padrões XML das indústrias química, automobilística, aeroespacial etc.
- **Taxionomias e documentos com vocabulários mistos:** neste estágio, os dados podem ser compostos de múltiplos domínios e podem ser classificados dentro de uma taxionomia hierárquica. De fato, a classificação pode ser utilizada para descoberta de dados. Relacionamentos simples entre categorias na taxonomia podem ser usados para relacionar e combinar dados. Portanto, os dados são inteligentes o suficiente para serem facilmente descobertos e sensivelmente combinados com outros dados.
- **Ontologias e regras:** neste estágio, novos dados podem ser inferidos a partir dos existentes seguindo regras lógicas simples. Essencialmente, os dados são suficientemente

inteligentes para serem descritos com relacionamentos concretos, e formalismos sofisticados em que cálculos lógicos podem ser realizados com essa "álgebra semântica". Isso permite a combinação e recombinação de dados até um nível atômico e uma análise mais refinada dos dados. Portanto, neste estágio, os dados não existem como uma bolha, mas sim como parte de um microcosmo sofisticado. Um exemplo dessa sofisticação dos dados é a tradução automática de documentos de um domínio para documentos equivalentes em outros domínios.

Podemos definir dados inteligentes como dados que não dependem de aplicação, classificáveis, que são de fácil composição e que fazem parte de um ecossistema de informação maior (ontologias).

A Web Semântica não é somente para *World Wide Web* (WWW). Ela representa um conjunto de tecnologias necessárias também às atividades desenvolvidas em redes coorporativas, intranets.

Nesse sentido, a Web Semântica irá solucionar vários problemas chaves das atuais arquiteturas de tecnologia da informação como Daconta (2003):

- **Sobrecarga de informação:** na Web há uma quantidade imensa de informações não pertinentes que estão disponíveis e que são fornecidas pelos processos de busca. As ferramentas de busca enfrentam a dificuldade de executar pesquisas entre documentos que não estão diferenciados em termos de assunto, qualidade e relevância. A tecnologia atual não é capaz de diferenciar uma informação comercial de uma educacional, ou informação entre idiomas, culturas e mídia. É necessário haver informações de qualificação da própria informação para ser possível classificá-las e tornar os processos de recuperação de informações mais eficazes.
- **Integração de informações:** a integração de informações na Web é um assunto muito discutido pelos estudiosos da área. A variedade de fontes de informação distintas com diferenças sintáticas, semânticas e estruturais entre elas é muito grande, tornando o compartilhamento, a integração e a resolução de conflitos entre essas informações um problema de difícil solução. Outra questão a ser tratada seria a criação ou remoção de fontes de informação, o que teria de ser realizado com extrema cautela de forma a não causar grandes impactos ao ambiente integrado. Deve-se considerar que as fontes de informação podem ter capacidades computacionais diferentes, podendo variar desde sistemas de banco de dados a arquivos. As informações podem variar de não estruturadas, como imagens e vídeos, a semi-estruturadas, como arquivos de e-mail e páginas Web. A heterogeneidade estrutural e semântica da informação na Web atualmente é imensa, e a maioria das propostas de integração ainda adota soluções com alto índice de centralização, tornando o seu uso na Web inviável. Para tratar esses problemas é necessário considerar questões relevantes como a utilização de metadados e ontologias, visando a busca de uma linguagem única, capaz de estruturar e representar conhecimento e regras.
- **Conteúdo não estruturado:** um dos motivos do grande sucesso da Web é sua liberdade de publicação de informação. Devido a isso, existe uma enorme quantidade de todos os tipos de documentos e recursos disseminados na Web, tais como bancos de dados, artigos, programas, arquivos, etc. Por serem criados de forma autônoma, sem preocupação com regras de estruturação, catalogação e descrições de suas propriedades, essas informações são difíceis de serem abrangidas pelos mecanismos de pesquisa, ocasionando demora e ineficácia na sua localização. Alguns problemas enfrentados pelos mecanismos de busca e recuperação de informações são: demora na localização de informações; informações não localizadas devido às mudanças de URLs; recuperação de um número elevado de informações que, em sua maioria, não atendem às expectativas

dos usuários; e recuperação de informações fora do contexto solicitado pelo usuário, devido a problemas de semântica e ambigüidade. A efetividade dos mecanismos de busca depende principalmente da maneira pela qual as informações foram estruturadas e catalogadas na Web. Os documentos podem ser estruturados e organizados de várias formas diferentes na Web, e as ferramentas de busca têm que utilizar mecanismos de recuperação adequados para cada tipo de organização.

Na proposta de desenvolvimento da Web Semântica (BERNERS, 2003) é sugerida uma arquitetura de três camadas, conforme apresentado na Figura 2:

- **Esquema:** que estrutura os dados e define seu significado;
- **Ontologia:** que define as relações entre os dados;
- **Lógica:** que define mecanismos para fazer inferências sobre os dados.

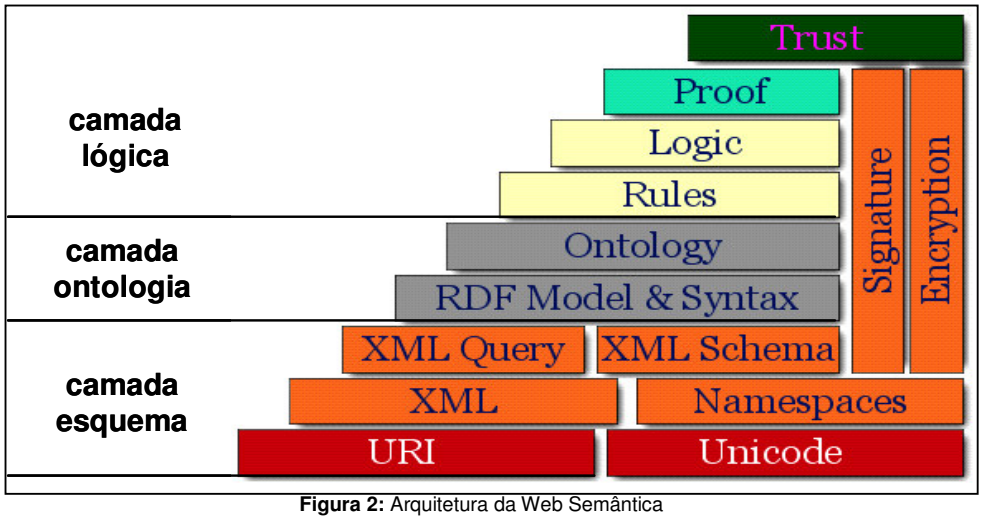


Figura 2: Arquitetura da Web Semântica

A tecnologia da Web Semântica é uma proposta de extensão da Web atual baseada no uso de ontologias para descrever relacionamentos entre objetos, formados com informações semânticas, para automatizar o processamento pelas máquinas.

A Web Semântica não é apenas uma ferramenta para conduzir e auxiliar a execução de tarefas individuais e pesquisas mais eficientes na Web, mas também uma ferramenta para assistir ao desenvolvimento do conhecimento.

3 Ontologias

Segundo Guarino (1997), ontologias são tratadas como um artefato computacional composto de um vocabulário de conceitos, definições e possíveis propriedades, um modelo gráfico mostrando todas as possíveis relações entre os conceitos e um conjunto de axiomas formais que restringem a interpretação dos conceitos e relações, representando de maneira clara e não ambígua o conhecimento do domínio. É importante realçar que, de posse dessa base de conhecimento formalizada como uma teoria lógica, a ontologia não descreve apenas conhecimento imediato, isto é, conhecimento factual que pode ser obtido diretamente a partir da observação do domínio, mas

também conhecimento derivado, ou seja, conhecimento obtido através de inferência sobre o conhecimento imediato disponível.

Segundo Guarino (1997) e Guarino (1998), com base em seu conteúdo as ontologias podem ser classificadas nas categorias que se seguem (Figura 3).

- **Ontologias genéricas:** expressam teorias básicas do mundo, de caráter bastante abstrato, aplicáveis a qualquer domínio, conhecimento de senso comum. Tipicamente, ontologias genéricas definem conceitos tais como coisa, estado, evento, processo, ação, etc., com o intuito de serem especializados na definição de conceitos em uma ontologia de domínio.
- **Ontologias de domínio:** expressam conceituações de domínios particulares, descrevendo o vocabulário relacionado a um domínio genérico, tal como medicina e direito. São construídas para serem utilizadas em um micromundo.
- **Ontologias de tarefas:** expressam conceituações sobre a resolução de problemas, independentemente do domínio em que ocorram, isto é, descrevem o vocabulário relacionado a uma atividade ou tarefa genérica, tal como diagnose ou vendas. O estudo de ontologias de tarefas é a vertente mais recente das pesquisas realizadas sobre ontologias. Sua principal motivação é facilitar a integração dos conhecimentos de tarefa e domínio em uma abordagem mais uniforme e consistente, tendo por base o uso de ontologias. Trabalhos nesta categoria incluem Chandrasekaran & Josephson (1997), Musen (1992).
- **Ontologias de aplicação:** expressam conceitos dependentes do domínio e da tarefa particulares. Esses conceitos freqüentemente correspondem a papéis desempenhados por entidades do domínio quando da realização de uma certa atividade.
- **Ontologias de representação:** segundo Gruber (1995), expressam os compromissos ontológicos embutidos em formalismos de representação de conhecimento.

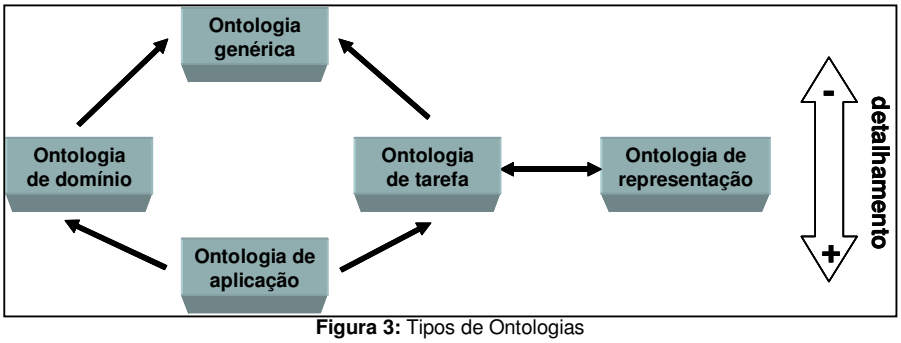


Figura 3: Tipos de Ontologias

Segundo Guarino e Welty (1998), a profundidade ontológica pode ser classificada em quatro níveis:

- **Vocabulário:** em sua forma mais simples, uma ontologia é apenas um vocabulário. Nesse sentido, uma DTD ou um XML-Schema pode definir uma ontologia.
- **Taxonomia:** o significado dos termos é estabelecido pela definição de relacionamentos entre objetos e classes, subclasses e classes-pai. Esses sistemas são denominados taxonomia. Esse tipo de ontologia normalmente é estabelecido por sistemas orientados a objetos. Muitas ontologias existentes são definidas usando-se apenas esses relacionamentos hierárquicos.

- **Sistema relacional:** as ontologias também podem incluir relacionamentos não hierárquicos como nos diagramas de relacionamento de entidades e nos bancos de dados relacionais e, por conseguinte, cada esquema de banco de dados relacional define sua própria ontologia.
- **Teoria axiomática:** além de escrever relacionamentos, as ontologias também podem impor restrições. As restrições são definidas como axiomas. Um axioma é uma afirmação lógica que não pode ser provada a partir de outras afirmações, mas da qual outras afirmações podem ser derivadas.

4 Arquitetura de Integração Semântica

Na abordagem de integração proposta foi utilizada a visão clássica das camadas de informação propostas por Bergamaschi (1999) e Calvanese (1998), como ilustra a Figura 4.

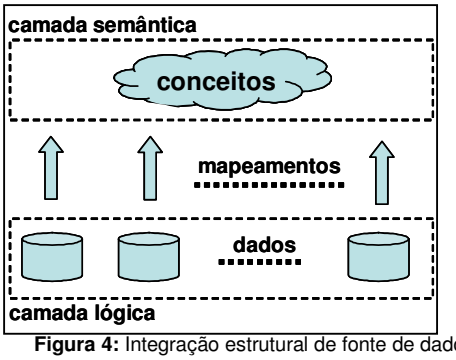


Figura 4: Integração estrutural de fonte de dados

A camada semântica é representada por um modelo conceitual que descreve um domínio particular de interesse, domínio esse que se encontra implícito nas estruturas das fontes a serem integradas. A camada lógica é representada por um modelo lógico que descreve a estrutura das fontes participantes do processo de integração. O mapeamento entre as duas camadas é realizado através do modelo de mapeamento, que contém um conjunto de regras de mapeamento, responsáveis por especificar como elementos do modelo lógico devem ser interpretados no modelo conceitual.

O propósito da arquitetura é prover um sistema de integração de dados baseado em web services e ontologias como o esquema comum para sobrepor a heterogeneidade estrutural das fontes de dados. Esse esquema comum é descrito na linguagem OWL[1] para suportar várias ontologias, relacionadas ou não. A OWL é uma linguagem de marcação semântica para recursos da Web, desenvolvida como uma extensão da linguagem para ontologias DAML+OIL que ocupava o mais alto nível na pilha da rede semântica proposta pela W3C. Segundo Smith (2003), a OWL ultrapassa essa linguagem no que se refere à habilidade para representar na Web conteúdo compreensível por máquinas.

É importante salientar que os dados semi-estruturados, as páginas HTML, os documentos-texto e os arquivos XML não foram contemplados na arquitetura. Caso exista a necessidade de inclusão de tal fonte de dados, será necessário converter os dados para o formato XML, respeitando-se a estrutura sintática de acordo com alguma ontologia disponível no repositório de ontologias. Após a geração do novo formato, os dados deverão ser armazenados em alguma

[1] *Ontology Web Language*

estrutura de SGBD ou então criar algum recurso como wrappers para manipulação dessas fontes de dados. Esse novo recurso, wrapper, desempenharia um papel semelhante aos filtros XML e SQL disponíveis na arquitetura.

O diagrama da arquitetura de integração é apresentado na Figura 5, sendo seus componentes descritos a seguir.

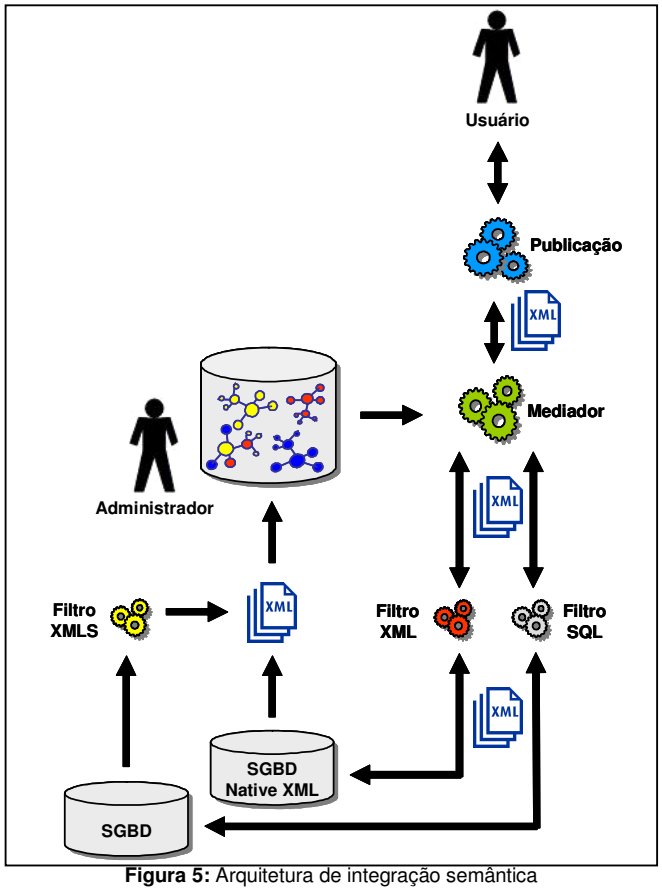


Figura 5: Arquitetura de integração semântica

- **Interface usuário:** responsável pela interação do usuário com a arquitetura. Formada pela interface de publicação, tem a função de automaticamente transformar, converter e publicar os documentos XML nos dispositivos Web. Para isso, integra algumas ferramentas tais como processadores e parsers de algumas linguagens, e separa conteúdo, lógica e estilo para realizar com maior eficiência esse processo. É através desta interface que os usuários submetem a consulta sem ter conhecimento de como as informações estão de fato representadas. Esse processo consiste da identificação das tags de descrição no documento OWL, que univocamente mapeiam um conceito.
- **Interface administrador:** responsável por armazenar informações sobre os conceitos disponibilizados. São tarefas desta interface a criação e a manutenção do esquema conceitual e do catálogo de filtros bem como o suporte à tarefa de mapeamento.

- **Repositório de ontologias:** armazenar os esquemas conceituais que encontram-se expressos diretamente em OWL. Além do esquema conceitual, também é armazenado um catálogo de filtros contendo, para cada filtro, a sua localização, a lista de conceitos e a instância do modelo de mapeamento da fonte de dado a ser integrada.
- **Interface mediador:** camada entre o repositório de ontologias e a interface usuário, sua função é receber as consultas da interface usuário, distribuí-la aos respectivos filtros e, após o processamento integrar os resultados provenientes em formato XML para uma posterior transformação feita pela interface de publicação.
- **Filtro SQL:** responsável por fazer a comunicação com os SGBDs relacionais através de instruções SQL. Essas instruções são geradas com o auxílio da interface mediadora, que, através do acesso ao repositório de ontologias, elabora as consultas.
- **Filtro XML:** responsável por fazer a comunicação com os SGBDs nativos XML por meio de instruções XQuery. A vantagem de se utilizar esse recurso é que a instrução não precisa ser transformada, pois já está no formato XML.
- **Filtro XMLS:** responsável por fazer o mapeamento da estrutura de um modelo relacional para o padrão XML Schema.

A arquitetura de integração semântica de bibliotecas digitais foi aplicada no desenvolvimento do protótipo denominado de Base Científica – BASCIN.

Esse protótipo fez uso da estrutura da Biblioteca Digital de Teses e Dissertações do IBICT[2], que passou a fomentar o desenvolvimento de recursos informacionais brasileiros de interesse para C&T em texto completo de teses e dissertações, e da Biblioteca Digital de Artigos da SciELO[3], que é uma biblioteca eletrônica que abrange uma coleção selecionada de periódicos científicos brasileiros na área da saúde.

Para o desenvolvimento da infra-estrutura foram seguidas as etapas abaixo.

Aquisição dos padrões de metadados das duas bibliotecas digitais:

- MTD-BR em <http://www.ibict.br/schema>
- Artigo DTD-SciELO v3.1 em <http://www.scielo.org/dtd/004_pt.htm>

Mapeamento dos elementos e definições dos dois modelos na linguagem OWL. Para o desenvolvimento do protótipo foi considerado um conjunto mínimo de elementos dos dois padrões, conforme demonstrado na Figura 6.

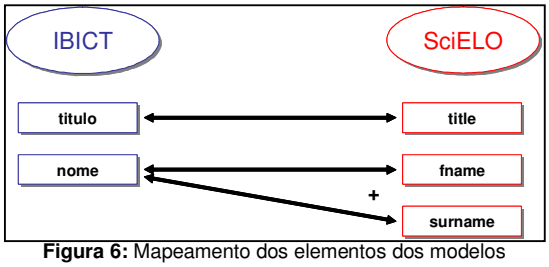


Figura 6: Mapeamento dos elementos dos modelos

[2] http://www.ibitc.br/bdtd

[3] *Scientific Electronic Library Online* – http://www.scielo.org

Para a codificação foi utilizada a ferramenta OilEd 3.5, que trabalha com a linguagem OIL. A própria ferramenta disponibiliza vários conversores, inclusive para linguagem OWL. O resultado da codificação em OWL dos padrões pode ser visualizado nas Figuras 7 e 8.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<owls:Ontology
    xmlns:owls="http://www.w3.org/2003/OWL-XMLSchema"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2003/OWL-XMLSchema http://wonderweb.semanticweb.org/owl/2003/owl1-dl.xsd">
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/scielo.daml#ScieloFname"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/scielo.daml#ScieloTitle"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/scielo.daml#ScieloSurname"/>
</owls:Ontology>
```

Figura 7: Listagem OWL da SciELO

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<owls:Ontology
    xmlns:owls="http://www.w3.org/2003/OWL-XMLSchema"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2003/OWL-XMLSchema http://wonderweb.semanticweb.org/owl/2003/owl1-dl.xsd">
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictTitulo">
        <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictTrabalho"/>
    </owls:Class>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictAutor">
        <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
    </owls:Class>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictTrabalho"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictContribuidor">
        <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
    </owls:Class>
    <owls:ObjectProperty owls:name="file:/D:/ontologies/ibict.daml#e_autor">
        <owls:domain>
            <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
        </owls:domain>
        <owls:domain>
            <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictTrabalho"/>
        </owls:domain>
    </owls:ObjectProperty>
    <owls:ObjectProperty owls:name="file:/D:/ontologies/ibict.daml#ibictCitacao"/>
    <owls:ObjectProperty owls:name="file:/D:/ontologies/ibict.daml#e_autor">
        <owls:domain>
            <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
        </owls:domain>
        <owls:domain>
            <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictTrabalho"/>
        </owls:domain>
    </owls:ObjectProperty>
</owls:Ontology>
```

Figura 8: Listagem OWL do IBICT

A integração semântica é feita relacionando-se os dois modelos em OWL, gerando assim um novo arquivo, como demonstra a Figura 9.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<owls:Ontology
    xmlns:owls="http://www.w3.org/2003/OWL-XMLSchema"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2003/OWL-XMLSchema http://wonderweb.semanticweb.org/owl/2003/owl1-dl.xsd">
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictTitulo">
        <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictTrabalho"/>
    </owls:Class>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/scielo.daml#ScieloFname"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/scielo.daml#ScieloTitle"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictTrabalho"/>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictAutor">
        <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
    </owls:Class>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/ibict.daml#IbictContribuidor">
        <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
    </owls:Class>
    <owls:Class owls:complete="false" owls:name="file:/D:/ontologies/scielo.daml#ScieloSurname"/>
    <owls:ObjectProperty owls:name="file:/D:/ontologies/ibict.daml#e_autor">
        <owls:domain>
            <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictPessoa"/>
        </owls:domain>
        <owls:domain>
            <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictTrabalho"/>
        </owls:domain>
    </owls:ObjectProperty>
    <owls:ObjectProperty owls:name="file:/D:/ontologies/ibict.daml#ibictCitacao"/>
    <owls:EquivalentClasses>
        <owls:Class owls:name="file:/D:/ontologies/ibict.daml#IbictAutor"/>
        <owls:Class owls:name="file:/D:/ontologies/scielo.daml#ScieloSurname"/>
        <owls:Class owls:name="file:/D:/ontologies/scielo.daml#ScieloFname"/>
    </owls:EquivalentClasses>
</owls:Ontology>
```

Figura 9: Listagem OWL da integração dos modelos do IBICT e da SciELO

Após a geração dos modelos em OWL, estes devem ser armazenados em alguma estrutura para futura reutilização. No protótipo foi utilizado o Microsoft SQL Server 2000 como repositório de ontologias.

A partir do momento em que o repositório de ontologias recebe modelos, a interface mediador deve ter acesso ao modelo de integração para que seja executado o processo de orquestração. É esse modelo que permitirá uma visão única do ambiente.

Como no protótipo os modelos foram armazenados no SQL Server, só foi utilizado o filtro SQL pela interface mediador, sendo que esse filtro terá duas solicitações, uma do modelo IBICT e outra da SciELO.

É importante observar que o resultado apresentado ao usuário só será estruturado quando os filtros retornarem as informações no padrão XML. Nesse caso, a interface mediador aguarda por dois retornos para encaminhar o resultado à interface de publicação, que, então, efetua o processamento de transformação e o encaminha ao usuário. Dessa forma, é finalizado o ciclo da arquitetura com relação ao protótipo.

No protótipo, o campo-texto está representando apenas o título do trabalho, mas poderia representar vários campos como resumo, palavras-chave etc. Esse processo de representação é executado no momento do mapeamento dos relacionamentos. A Figura 10 demonstra uma pesquisa de título que contenha a palavra “saúde”.

Figura 10: Aplicação da arquitetura de integração semântica

5 Conclusão

Diversas iniciativas de instituições nacionais e internacionais estão sendo desenvolvidas utilizando ontologias no nível semântico. Tais iniciativas demonstram apenas uma parte do potencial dessas tecnologias, já que os domínios são específicos.

Com a arquitetura proposta, conseguimos integrar diferentes domínios de instituições nacionais, como o Instituto Brasileiro de Informação em Ciência e Tecnologia (IBICT), o Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e a Scientific Electronic Library Online (SciELO). Essa integração foi alcançada com o desenvolvimento (conversão) das ontologias para OWL, permitindo o relacionamento entre as bases de dados das intituições.

Referências

BERGAMASCHI, S., CASTANO, S., VINCINI, M.,**Semantic Integration of Semistructured and Structured Data Sources**, SIGMOD Record, 1999.

BERNERS Lee, T., et al., **Semantic Web Development Proposal**, 2001. Disponível em: <http://www.w3c.org/2001/sw/>. Acesso em 10 ago. 2003.

CALVANESE, D. et al., **Information Integration: Conceptual Modeling and Reasoning Support**, Conference on Cooperative Information Systems, 1998.

CHANDRASEKARAN, B., JOSEPHSON, J. R. **The Ontology of Tasks and Methods**. Stanford University, California, 1997.

DACONTA, M.C., OBRST, L.J., SMITH, K.T., **The Semantic Web**, Wiley Publishing, Indianapolis, 2003.

GRUBER, T. R., **Toward Principles for the Design of Ontologies used for Knowledge Sharing**, Int. J. Human-Computer Studies, v. 43, n. 5/6, 1995.

GUARINO, N., **Understanding, Building and Using Ontologies**, Int. Journal Human-Computer Studies, v. 45, n. 2/3, 1997.

GUARINO, N., **Formal Ontologies and Information Systems**, In: First International Conference, Anais... Trento: IOS Press, Trento, 1998.

GUARINO, N., WELTY, C., **Conceptual Modeling and Ontological Analysis**, LADSEB-CNR, Padova, 1998.

MUSEN, M. A. **Dimensions of knowledge sharing and reuse**. Computers and Biomedical Research, V. 25, P. 436-467, 1992.

SMITH, Michael K; VOLZ, Raphael; MCGUINESS, Debora, **Web Ontology Language (OWL) Guide Version 1.0**, W3C Working Draft, Disponível em: <http://www.w3.org/TR/2002/WD-owl-guide-20021104>. Acesso em 22 de ago. 2003.

COMPARTILHAMENTO DE INFORMAÇÕES ENTRE COMPUTADORES ATRAVÉS DA TECNOLOGIA PEER-TO-PEER (P2P) USANDO A PLATAFORMA JXTA

José Voss Junior (FURB)
juniorvs@inf.furb.br

Francisco Adell Péricas (FURB)
pericas@furb.br

Resumo. Este artigo apresenta o resultado da especificação e da implementação de um protótipo de software para compartilhar arquivos e mensagens entre computadores ligados a uma rede local ou Internet. São apresentados conceitos pertinentes à tecnologia *peer-to-peer* (P2P) e como a mesma se comporta. Para implementação dos recursos disponibilizados pela tecnologia foi utilizada uma plataforma chamada JXTA que, além de especificar um conjunto de protocolos, bibliotecas e serviços para redes *peer-to-peer*, possui uma implementação para linguagem de programação Java.

Palavras-chave: *Peer-to-Peer*; P2P; JXTA; Anúncio de *peers* e conteúdos

1 Introdução

Com o crescimento exponencial do número de computadores ligados à Internet, vem crescendo também a necessidade de redes de computadores mais eficazes, não somente em relação a tecnologias de transmissão, mas também na forma que elas ocorrem. A arquitetura mais utilizada na Internet para comunicação, sem dúvida é a cliente/servidor, onde o cliente pode ser, desde um celular, *pocket*, *desktop* ou qualquer dispositivo que possui processador. Neste caso, todos clientes comunicam-se com um servidor central, o qual é responsável pela comunicação com todos os outros clientes envolvidos. Isso muitas vezes pode provocar uma sobrecarga excessiva no servidor, já que todas mensagens são propagadas pelo mesmo. Uma maneira de diminuir a sobrecarga do servidor é fazer com que cada cliente ligado à rede faça o papel tanto de servidor quanto cliente.

Em novembro de 1996, é liberada a primeira versão do ICQ (segundo definição encontrada (ICQ, 2004) significa *'I Seek You'*) um aplicativo para compartilhamento de mensagens instantâneas, usando uma tecnologia conhecida como *Peer-to-Peer* (P2P), onde cada cliente conectado pode comunicar-se diretamente com outro cliente sem haver necessidade direta de um servidor, pois o cliente também faz este papel. Em 1999, surge o Napster (NAPSTER, 2004), uma aplicação para compartilhamento de arquivos de música entre computadores ligados à Internet, também usando P2P. Desde então, P2P vem ganhando um grande espaço.

Hoje existem diversas aplicações usando redes P2P. Alguns exemplos são o *MSN Messenger* (MICROSOFT CORPORATION, 2004) e *AOL Internet Messenger* (AOL, 2004) para compartilhar mensagens instantâneas e o projeto chamado *Gnutella* (GNUTELLA, 2004) para compartilhar arquivos. Cada uma destas aplicações implementa um conjunto de protocolos proprietários, onde não há compatibilidade entre diversas aplicações do mesmo segmento.

Em abril de 2001, foi proposto pela Sun Microsystems um projeto chamado de JXTA (JXTA, 2004) que tem o objetivo de especificar um conjunto de protocolos para redes P2P.

Este artigo apresenta o resultado de um trabalho que se propôs a estudar a tecnologia P2P e a desenvolver um protótipo de software em redes P2P para compartilhar arquivos entre os

computadores envolvidos na rede, baseado inteiramente no projeto JXTA e usando a linguagem de programação Java. Aspectos detalhados de como é estabelecida comunicação entre *peers* sem a intervenção direta de um servidor, são esclarecidos no decorrer do artigo.

2 Internet

Desde o surgimento da Internet em meados da década de 70, até os dias atuais, o número de computadores conectados à mesma cresce de forma exponencial. O fato é que a Internet se tornou uma rede gigantesca, ligando computadores espalhados por todo o mundo.

O surgimento de novas tecnologias tanto de software quanto de hardware influenciam diretamente na performance da comunicação entre os computadores, Também há influência direta dos meios de comunicação, cuja comunicação muitas vezes é realizado de forma precária.

Quando se fala em Internet logo já se pensa na arquitetura cliente/servidor, que atualmente é a mais utilizada. Nessa arquitetura cada cliente, que pode ser desde um celular, *pocket*, *desktop* ou qualquer dispositivo que possua processador está ligado direta ou indiretamente a um servidor, responsável por manter o cliente conectado. Essa é uma forma de interação onde todas as mensagens trocadas pelos clientes conectadas à rede passam obrigatoriamente pelo gerenciamento do servidor.

Uma das vantagens dessa arquitetura, é que os clientes não necessitam de tanto recurso computacional, pois estão ligadas a um servidor responsável pelo processamento das requisições. O cliente de uma arquitetura cliente/servidor tem um papel passivo, capaz de utilizar um serviço oferecido pelo servidor, mas incapaz de oferecer serviços a outros clientes.

Cada cliente conectado a um servidor forma uma extremidade da Internet, ou seja, não fornecem qualquer serviço a outros clientes. Por essa razão a maioria dos serviços está centralizada em um servidor com IP fixo, facilitando a execução por parte do cliente.

Há uma nova arquitetura conhecida como *Peer-to-Peer* (P2P), que propõe outra forma de interação, onde cada cliente conectado a rede pode comunicar-se diretamente com outro cliente sem haver a intervenção direta do servidor, pois o cliente faz os dois papéis.

2.1 Arquitetura *Peer-to-Peer* e Cliente/Servidor

Segundo Silva (2003), a principal diferença entre arquiteturas P2P e cliente/servidor é o conceito de entidades, onde nessa última existe uma entidade que faz o papel de servidor e outras entidades que fazem o papel de clientes. Já na arquitetura P2P as entidades podem atuar ao mesmo tempo como servidores e como clientes.

Segundo Silva (2003):

> "Uma arquitetura de rede distribuída pode ser chamada P2P se os participantes compartilharem parte de seus próprios recursos de hardware. Esses recursos são necessários para prover os serviços e conteúdo oferecidos pela rede (ex. compartilhamento de arquivos ou espaços de trabalho para colaboração mútua). Os serviços e recursos são acessíveis por todos os pares sem necessidade de passar por nenhuma entidade intermediária. Os participantes dessa rede são tanto provedores de recursos como demandadores desses mesmos recursos".

Segundo Wilson (2004, p. 5), *peer-to-peer* é a chave para realização potencial de processamento intensivo, pois cada *peer* da rede possui mecanismo individual para prover serviço para cada um dos outros *peers*. Ao contrário da arquitetura cliente/servidor, redes P2P não dependem única e exclusivamente de um servidor centralizado para prover acesso a serviços.

P2P tem a capacidade de servir recursos com alta disponibilidade por um menor custo, enquanto maximiza o uso dos recursos de todos os *peers* conectados à rede. Considerando que soluções cliente/servidor necessitam de equipamentos de rede e instalações diferenciadas para conseguir soluções robustas, P2P pode oferecer um nível de robustez semelhante, distribuindo as necessidades por toda rede.

Infelizmente, P2P oferece algumas desvantagens devido à natureza redundante da estrutura da rede que lhe faz necessária. A forma distribuída de canais dessa rede resulta em pedidos de serviços de natureza não-determinística. Por exemplo, clientes requisitam um mesmo arquivo da rede P2P e podem conectar-se em máquinas completamente diferentes por rotas de comunicação diferentes e com resultados diferentes. Pedidos enviados podem não ter resultados imediatos e em muitos casos pode não resultar em qualquer resposta.

Porém P2P pode superar todas essas limitações. Embora recursos possam muitas vezes desaparecer na rede, P2P pode implementar funcionalidades para replicar os recursos mais acessados em outros *peers* da rede, promovendo assim acesso redundante a este recurso. Quanto maior o número de *peers* que possui recursos replicados, menor será a probabilidade de um peer receber um pedido sem resposta. Em resumo, a mesma estrutura que pode causar problemas pode ser a solução para resolvê-los.

2.2 Conceitos *Peer-to-Peer*

Será necessária uma introdução a terminologias e conceitos comuns a uma rede P2P, bem como esclarecer alguns aspectos inerentes a esse tipo de arquitetura.

P2P é a resposta estratégica para as seguintes perguntas: como fazer a conexão entre diversos dispositivos ligados a uma rede e como fazer com que compartilhem informações, recursos e serviços entre eles. Segundo Wilson (2004, p. 15), estas simples perguntas somente poderão ser respondidas com o conhecimento de outras questões importantes:

a) como reconhecer um outro dispositivo presente na rede;

b) como organizar os dados compartilhados;

c) como publicar suas informações para os demais dispositivos;

d) como identificar unicamente um dispositivo;

e) como compartilhar dados na rede P2P.

Todas redes P2P possuem no mínimo os elementos necessários fundamentais para responder todas as perguntas acima. Infelizmente a maioria destes elementos é de difícil implementação, resultando em um código de programa inflexível.

2.2.1 *Peer*

Peer é um nodo na arquitetura P2P, é uma unidade fundamental de processamento de qualquer solução P2P. Pode ser um simples computador, um dispositivo móvel tal como um celular ou uma aplicação distribuída em vários computadores.

Segundo definição de Wilson (2004, p. 16), *peer* é qualquer entidade capaz de executar qualquer trabalho útil e comunicar os resultados desse trabalho para outras entidades da rede P2P, podendo ser direta ou indiretamente. A definição de trabalho útil depende do tipo do *peer*. Existem três possibilidades de tipos de *peer*, sendo que cada uma tem responsabilidade específica. Os tipos são:

a) *peer* simples: designado unicamente para servir o usuário, permite prover e consumir serviços providos por outros usuários da rede;

b) *peer* de encontro: provê o encontro de *peers* de um determinado ponto de rede, encontra outros peers e seus respectivos serviços, resolve questões de descoberta de serviços e propaga mensagens entre *peers* da rede;

c) *peer relay*: possui mecanismo para comunicação com outros *peers* separados por *firewalls* de rede ou um equipamento NAT (tradução de endereço de rede).

2.2.2 Grupo de *Peers*

São grupos de *peers* formados para servir interesses comuns. Podem prover serviços para outros membros do grupo e limitar o acesso a *peers* de outras redes. Segundo Wilson (2004, p. 18), grupo de *peers* é dividido de acordo com os seguintes itens:

a) tipo de aplicação;

b) segurança dos *peers*;

c) informação de status de membros do grupo.

2.2.3 Rede de Transporte

Para troca de dados, *peers* devem possuir uma série de mecanismos para transmitir os dados pela rede. Essa camada se chama rede de transporte, e é responsável por todos aspectos de transmissão dos dados, incluindo a segmentação de pacotes e a adição de cabeçalhos apropriados para controle de destino do pacote. A rede de transporte pode ser, transporte em baixo nível como UDP ou TCP, ou transporte em alto nível como HTTP ou SMTP.

Uma rede de transporte P2P pode ser dividida em três partes:

a) pontos de fim (*endpoint*): uma fonte de destino inicial ou final para qualquer informação transmitida pela rede. Estes pontos de fim correspondem a interfaces usadas para enviar e receber dados da rede;

b) duto (*pipe*): canal virtual, unidirecional e assíncrono para comunicação de dois ou mais pontos de fim (*endpoint*);

c) mensagens: contêm dados para serem transmitidos sobre um duto de um ponto para outro.

Para comunicação usando dutos, o *peer* primeiro necessita achar os pontos finais (*endpoint*), um ponto para ser a fonte da mensagem e outro para cada destino da mensagem um ponto para ser a fonte da mensagem e outro para cada destino da mensagem, assim se forma um canal virtual entre cada um dos pontos. Quando este canal virtual estiver formado as mensagens podem ser transmitidas de um ponto a outro.

Para enviar dados de um *peer* para outro, deve se utilizar mensagens, pois estas contêm os dados a serem transmitidos, *peers* utilizam output *pipe* para enviar dados e o *peer* que recebe a mensagem utiliza input *pipe* para extrair os dados da mensagem recebida.

Um duto (*pipe*) de comunicação entre *peers* pode ser somente unidirecional, sendo assim para haver comunicação entre dois *peers* deve haver dois dutos de comunicação entre eles. Embora uma comunicação bidirecional faça parte de qualquer rede moderna existente, uma comunicação unidirecional pode ser modelada em qualquer ambiente de rede.

2.2.4 Serviços

Serviços provêem funcionalidades para *peers*, ou seja é a realização de qualquer trabalho útil dentro de uma rede P2P. Este trabalho útil pode ser transferir arquivos, prover informações de status, executar cálculos ou fazer qualquer coisa que seja de proveito para outros *peers*. Os serviços são a motivação da junção de dispositivos dentro de uma rede, sem serviços não haveria rede P2P.

Serviços podem ser divididos em duas categorias:

a) serviços de *peer*: serviço oferecido por um *peer* para outros *peers* da rede;
b) serviços de grupos de *peer*: serviço oferecido por um grupo de *peers*.

2.2.5 Anúncios

A estrutura de representação de uma entidade, serviço, ou recurso deve estar disponível para um *peer* ou para um grupo de *peers* como parte de uma rede P2P. Todo serviço disponível deve ser anunciado para que outros *peers* possam utilizá-lo.

Os anúncios podem ser de *peers*, grupos de *peers*, dutos, pontos de fim (*endpoint*). Também podem ser anunciados todos serviços oferecidos por um *peer* ou grupo de *peers*.

2.2.6 Protocolos

Toda troca de dados utiliza protocolos que padronizam a forma de enviar ou receber qualquer informação. Protocolo é um modo de estruturar a troca de informações entre duas ou mais partes, usando regras previamente estabelecidas e de conhecimento de todas as partes envolvidas.

Em P2P, protocolos são necessários para definir todos os tipos de interações entre *peers*. A organização de informações em anúncios simplifica a representação dos dados, simplificando assim a definição dos protocolos.

Aplicações P2P utilizam um conjunto de protocolos previamente estruturados para comunicação, deve haver um mínimo de funcionalidades especificadas como:

a) procura de *peers* em uma rede;
b) procura de serviços providos por um *peer* ou grupo de *peers*;
c) obtenção de status de um *peer*;
d) invocar serviço de um *peer*;
e) criação, união e desvinculação de um *peer* de um grupo;
f) rotas de mensagens para outros *peers*.

2.3 Comunicação P2P

Um dos problemas fundamentais de uma rede P2P é como ativar a troca de serviços entre dispositivos a ela ligados. Para resolver este problema é necessário responder duas questões importantes:

a) como um dispositivo encontra *peers* e serviços em uma rede P2P;
b) como dispositivos participantes em redes locais participam de redes P2P.

A primeira questão é importante porque, sem o conhecimento da existência de um *peer* ou serviço de rede, não é possível o dispositivo fazer parte de rede. A segunda questão é importante porque muitos dispositivos em uma rede P2P serão separados da rede com equipamentos designados para prevenir ou restringir conexões diretas entre dois dispositivos em diferentes redes internas.

2.4 Comparação entre Soluções P2P

Utilizando alguns conceitos definidos na seção anterior é possível comparar soluções P2P proprietárias.

2.4.1 Napster

O Napster (NAPSTER, 2004) foi à aplicação propulsora da plataforma P2P. Consistia em uma aplicação de rede P2P híbrida, pois havia um servidor centralizado onde eram realizadas todas as funcionalidades de procura de anúncios e de *peers*. Podia ser modelado como um único *peer* de encontro e diversos *peers* simples, todos utilizando TCP como rede de transporte. O *peer* de encontro provê a todos os *peers* simples a capacidade de localizar arquivos de música anunciados em um único arquivo, endereço de IP e porta de comunicação. Todos os *peers* simples utilizam essas informações para se conectarem diretamente com outros *peers*, e assim realizar cópias dos arquivos compartilhados.

Napster não prove uma solução completa para evitar *firewalls*, pois é capaz de atravessar somente um. Cada *peer* atua com uma simples rota, não possui capacidade para habilitar outros *peers* como *peers* de rotas.

2.4.2 Gnutella

Na rede Gnulella (GNUTELLA, 2004) cada *peer* atua como *peer* simples, *peer* de encontro e *peer* de rota, utilizando TCP como rede de transporte e HTTP para transferência de arquivos. As procuras realizadas na rede são propagadas por todos os *peers* conhecidos deste *peer*, e estes propagam para os seus *peers* conhecidos. Anúncio de conteúdo consiste em endereço IP, número da porta e um índice que identifica o arquivo no *peer* hospedeiro. Também são anunciados detalhes do arquivo como nome e tamanho. Como acontece com o Napster, o Gnutella pode atravessar apenas um *firewall*.

2.4.3 Kazaa

É uma aplicação P2P criada recentemente, e, segundo Good (2004), possui algumas inovações, tal como a facilidade para compartilhar arquivos com outros usuários, pois permite que seja feita a cópia de múltiplos arquivos simultaneamente. Permite também que sejam compartilhados diversos tipos de formatos, tal como formatos para músicas, vídeos, documentos, planilhas, fotos entre muitos outros. Permite que seja efetuada a cópia particionada de arquivo, onde um mesmo arquivo é copiado de diversos usuários ao mesmo tempo.

Kazaa possui um tipo de arquitetura puramente P2P, ou seja, não tem necessidade gerenciamento explícito por um servidor central.

3 Plataforma JXTA

Conforme JXTA PROGRAMMER GUIDE (2004), JXTA é uma plataforma aberta, uma solução genérica para plataforma P2P. Especifica um conjunto de protocolos para conexão de qualquer dispositivo de rede, tal como telefone celular, *desktop* e servidores. Os protocolos JXTA são independentes de linguagem de programação e de sistema operacional. Esta plataforma chamada de *Project JXTA* (JXTA, 2004) foi proposta em Abril de 2001, por Bill Joy, cientista chefe da Sun Microsystems. Além de manter os protocolos JXTA, que agora estão na sua versão 2.2, o projeto JXTA é responsável pela implementação de referencia dessa plataforma, tendo também que controlar o código fonte de uma variedade de projetos disponíveis no site do projeto.

Atualmente o projeto JXTA tem uma implementação de referência disponível na linguagem de programação Java, e com implementações na linguagem C, Objective-C, Ruby e Perl 5.0. Neste momento o projeto JXTA é utilizado por uma variedade de projetos nas mais diversas áreas de conhecimento, como gerenciamento de conteúdo, inteligência artificial e segurança de sistemas.

3.1 Arquitetura JXTA

A arquitetura da plataforma JXTA está dividida em três camadas: núcleo JXTA, serviços JXTA e aplicações JXTA. A Figura 1 mostra uma visão dessa arquitetura.

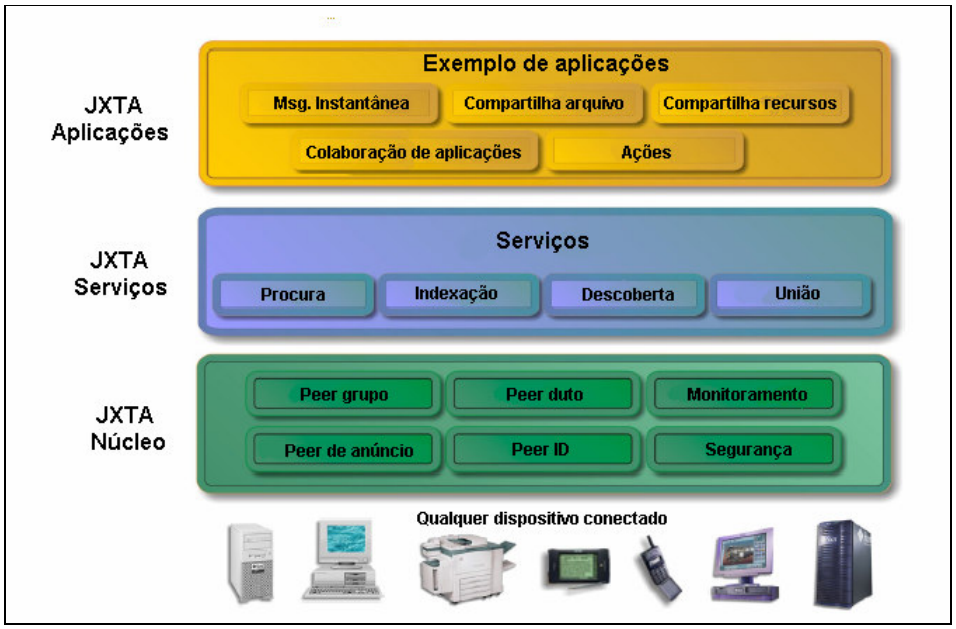


Figura 1: Arquitetura de camadas de JXTA

As camadas são as seguintes:

a) núcleo: essa camada, também conhecida como JXTA core, encapsula funcionalidades básicas comuns a uma rede P2P. Possui mecanismos chaves, onde se destaca a função para algoritmos de encriptação, entrada baseada em senha/usuário e mecanismos para criação de *peers* e grupo de *peers*;

b) serviços: essa camada possui serviços de rede que podem não ser absolutamente necessários para que uma rede P2P possa operar, mas é comum e desejável que a possua. Alguns exemplos de serviços de rede podem ser indexação e procura de arquivos, sistema de armazenamento, compartilhamento de arquivos, serviços de autenticação e serviços de chave pública;

c) aplicações: essa camada possui de fato a integração às aplicações, tal como mensagens instantâneas, compartilhamento de documentos e recursos, gerenciamento de conteúdos e descobertas, aplicações de correio eletrônico, sistemas distribuídos e muitos outros.

O limite entre serviços e aplicações não é rígido, ou seja, uma aplicação para um cliente pode ser visto como um serviço para outro. O projeto JXTA é projetado para ser modular, permitindo ao desenvolvedor poder escolher entre uma coleção de serviços e aplicações tal como é de sua necessidade.

Segundo JXTA PROGRAMMER GUIDE (2004), existem três aspectos chaves na arquitetura JXTA que se destacam entre as demais, conforme descritas:

a) o uso de documentos XML para descrever anúncios de recursos em uma rede P2P;

b) abstração de dutos para os *peers*, sendo que os mesmos não precisam ser ligados explicitamente a uma servidor central de nomes, tal como DNS;

c) um esquema uniforme de endereçamento de *peers* (*peer* ID).

3.2 Conceitos JXTA

Todos os conceitos descritos em tópicos anteriores sobre P2P são equivalentes para toda plataforma JXTA, sendo que os principais são *peers*, grupos de *peers*, dutos e anúncios. A seguir são apresentados alguns dos conceitos pertinentes à plataforma JXTA.

3.2.1 Mensagens

Mensagens são objetos enviados entre *peers* JXTA, ou seja é a unidade básica de troca de dados entre *peers*. Mensagens são enviadas e recebidas por serviços de dutos e por pontos de fim (*endpoint*). Tipicamente aplicações utilizam serviço de dutos para criar, enviar e receber mensagens. Em geral as aplicações não necessitam do uso direto de serviços de pontos de fim (*endpoint*).

Mensagem é essencialmente um conjunto chave/valor, seu conteúdo pode ser de tipos arbitrários e cada plataforma de software descreve como uma mensagem é convertida de uma estrutura de dados nativa (como exemplo objetos da linguagem Java ou C) para uma representação conhecida por toda arquitetura JXTA.

3.2.2 Segurança

Redes P2P dinâmicas como JXTA necessitam um suporte para diferentes níveis de segurança de acesso aos recursos. *Peers* JXTA operam em modelos baseados em papéis, no qual diferentes *peer* se integram na rede e cada um possui níveis de privilégio diferentes, o qual podem executar diferentes tarefas. Cinco são os princípios básicos de segurança necessários: sigilo; autenticação; autorização; integridade dos dados; refutabilidade.

3.2.3 Identificadores (IDs)

Peers, grupos de *peers*, dutos e outros recursos JXTA precisam ser univocamente identificados. O JXTA ID é um identificador único de uma entidade. Algumas entidades que precisam ser unicamente identificadas são: *peers*, grupos de *peers*, dutos e conteúdos. URNs (*Uniform Resource Names*) são utilizados para expressar JXTA IDs, sendo que estes servem como identificadores independentes. Estes identificadores são representados em arquivo texto no formato XML reconhecido por toda plataforma JXTA, como mostrado na Figura 2.

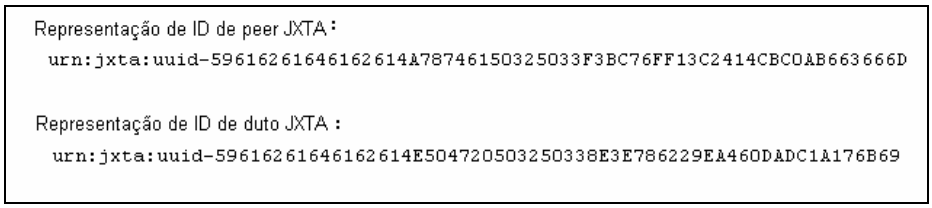


Figura 2: Identificadores da plataforma JXTA

Identificadores são gerados aleatoriamente pela plataforma JXTA. Existe um identificador especial chamado de *NetPeerGroup* ID, que representa o identificador do *Net peer group*, que é o primeiro grupo padrão da plataforma JXTA, sendo que todos os *peer* são criados e inicializados com ele.

4 Desenvolvimento de um Software P2P

Para validar a proposta de implementação de aplicativos P2P através da plataforma JXTA, desenvolveu-se um protótipo de software para compartilhar informações entre diversos computadores ligados entre si, através de rede local ou Internet. O software desempenha um papel tanto de cliente como de servidor baseado no paradigma P2P. Para sua realização foram executados procedimentos de análise, elaboração e implementação do projeto.

O protótipo de software utiliza a plataforma JXTA, implementada com a linguagem de programação Java, para ser multiplataforma, mas foi testado somente no sistema operacional Windows.

4.1 Especificação

Para fazer a especificação deste protótipo foi utilizada uma metodologia orientada a objetos, representada em diagramas da *Unified Modeling Language* (UML).

A Figura 3 representa os principais casos de uso deste protótipo de software.

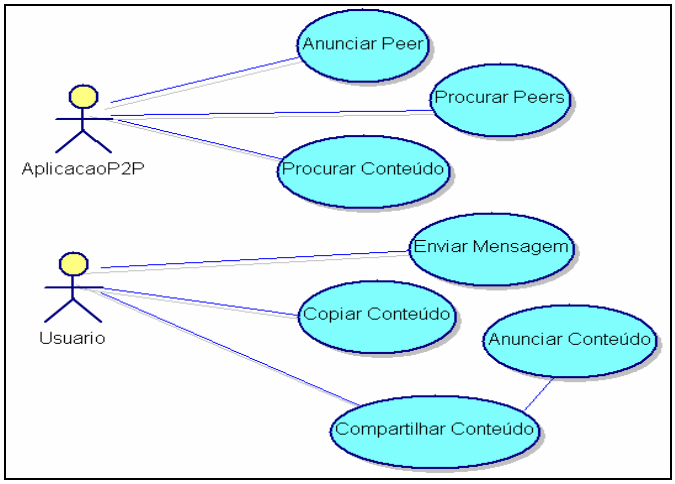


Figura 3: Diagrama de Caso de Uso

Cada um dos casos de uso do protótipo, indicando o nome do caso de uso, o respectivo ator e uma descrição resumida, está apresentado a seguir:

Caso de Uso	Ator	Descrição
Anunciar Peer	AplicacaoP2P	Quando o protótipo é iniciado, o *peer* deve ser anunciado na rede P2P. Isso é necessário para que o mesmo possa ser encontrado por outros *peers* da mesma rede. Este processo ocorre sem interferência do usuário.

Procurar Peers	AplicacaoP2P	A cada 10 segundos o protótipo dispara uma rotina de procura de *peers*. A procura é realizada tanto em rede local quanto na Internet.
Procurar Conteúdo	AplicacaoP2P	O processo de procura de conteúdos ocorre a cada 10 segundos sem interferência do usuário. A procura é realizada sem vínculo direto do *peer* o qual o conteúdo pertence, ou seja, cada anúncio de conteúdo descoberto na procura é independente e tem apenas uma referência para o *peer* a qual este anúncio pertence.
Enviar Mensagem	Usuário	A mensagem é enviada por um usuário do protótipo. Cada usuário pode ser tratado com um *peer* da rede P2P. A mensagem corresponde a um texto simples que é enviada para todos os *peer* da rede.
Copiar Conteúdo	Usuário	Em cada anúncio encontrado pode ser recuperado o conteúdo e copiado para uma pasta local. Para copiar um conteúdo basta recuperá-lo do anúncio.
Compartilhar Conteúdo	Usuário	Existe uma pasta local compartilhada com outros *peers*. Quando se necessita compartilhar algum arquivo este deve ser adicionado nesta pasta. Podem ser compartilhados somente arquivos e não diretórios.
Anunciar Conteúdo	Usuário	A cada 10 segundos é verificada, na área compartilhada, a existência de novos arquivos. Caso seja encontrado, é gerado um anúncio que é publicado na rede P2P.

4.2 Operacionalidade da Implementação

Para utilização desse protótipo é necessário que haja dois ou mais computadores ligados em rede local ou a Internet. Para o funcionamento do protótipo em redes locais não há necessidade de acrescentar nenhuma informação adicional. O mesmo não acontece na execução dos protótipos em computadores ligados a Internet, pois é necessário que seja informado pelo menos o endereço de IP de um *peer* de encontro e um *peer relay*.

A Figura 4 apresenta a tela principal do protótipo, sendo que na mesma possui todos recursos visuais necessários para demonstração de uma rede P2P. A seguir será descrita cada uma das partes do protótipo do software:

a) lista de *peers*: na parte esquerda da tela estão todos os *peers* encontrados na rede P2P. Sendo que o primeiro *peer* é o *peer* local;

b) lista de anúncios de conteúdos: na parte direita da tela estão todos os anúncios de conteúdos encontrados;

c) diretório compartilhado: na parte superior do protótipo está descrito o caminho do diretório compartilhado;

d) informações de *peers* e arquivos: na parte inferior da tela encontra-se uma área contendo diversas informações dos *peers* participantes da rede, e cada um de seus anúncios publicados;

e) envio de mensagens: ainda na parte inferior encontra-se um campo onde são escritas as mensagens que são enviadas para todos os *peers* da rede.

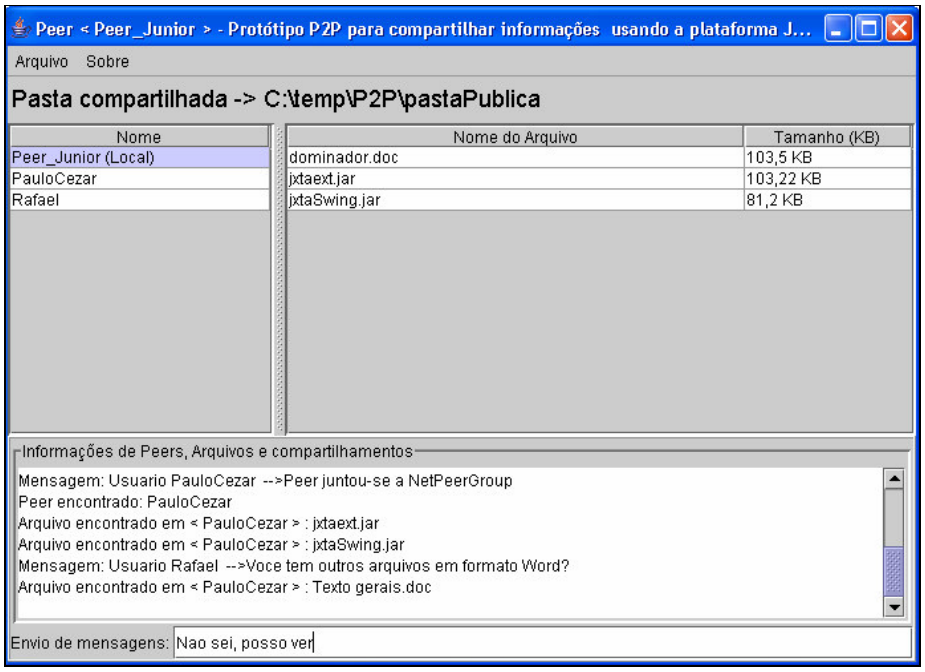


Figura 4: Tela principal do protótipo P2P

Cada operação ou evento que acontece na rede P2P gera uma linha de texto na área informações de *peers*.

Observou-se que no funcionamento do protótipo em redes locais, a procura de anúncios, tanto de *peers* quanto de conteúdos, acontece de forma eficiente. Em uma rede de aproximadamente duzentos computadores, onde cada computador representa um peer, foram selecionados quatro deles para execução do protótipo. Cada *peer* reconheceu em cerca de um minuto todos os outros três peers envolvidos, bem como encontrou todos seus anúncios de conteúdos, que não passou de cinco anúncios por *peer*.

Foi possível também, enviar mensagens para todos os *peers* envolvidos na rede. Quando uma mensagem é enviada todos os *peer* a recebem.

Ao utilizar o protótipo via Internet, notou-se que a comunicação se tornou mais eficiente quando adicionado um número maior de *peers* de encontro para o *peer*. Pois cada *peer* procura anúncios disponíveis em cada um dos *peers* de encontro conhecidos.

5 Conclusões

No decorrer do estudo da arquitetura *Peer-to-Peer* e do desenvolvimento do protótipo de software verificou-se a complexidade de cada componente que faz parte da rede P2P. Teve-se uma idéia geral de como é realizada cada uma das etapas da comunicação, e como de fato os *peers* se comunicam sem haver um servidor central processando todas as requisições.

Ao utilizar a plataforma JXTA para implementação de uma rede P2P, observou-se o quão direto ficou implementar uma comunicação entre computadores sem que haja um servidor, pois a

mesma oferece uma biblioteca que resolve as questões complexas de comunicação P2P. O desenvolvedor sequer precisa conhecer em detalhes todos os conceitos de como é realizada a comunicação entre os *peers*.

Tem-se agora uma noção das possibilidades de futuros desenvolvimentos utilizando-se essa tecnologia ainda pouco explorada, mas que vem ganhando bastante força de mercado. O que pode impulsionar a utilização de softwares P2P é justamente a plataforma JXTA, que apresenta grande facilidade nas implementações P2P.

A vantagem na utilização deste protótipo para compartilhar arquivos e mensagens com outros computadores é que em redes locais não há necessidade de compartilhar pastas na rede, pois tudo que é colocado na chamada pasta pública do protótipo é compartilhado automaticamente com todos os *peers*. Na Internet, todos *peers* da rede podem copiar os arquivos anunciados, sem precisar haver um servidor WEB instalado ou haja disponibilidade em outro servidor qualquer.

Uma limitação do protótipo é que não há compartilhamento de estrutura de pastas e sim somente de arquivos. Outra desvantagem é o excesso de congestionamento que uma rede pode sofrer, pois a cada dez segundos são enviadas requisições, tanto de pesquisa quanto de anúncio (caso houver).

Referências

AOL. **AOL Instant Messenger**. Atlanta, USA, 2004. Disponível em: <http://americaonline.com.br/aim >. Acesso em: 30 maio 2004.

GOOD, Nathaniel S. **Usability and privacy: a study of Kazaa P2P file-sharing**. Palo Alto, USA, [2003?]. Disponível em: <http://www.hpl.hp.com/shl/papers/kazaa/KazaaUsability.pdf>. Acesso em: 30 de mar. 2004.

ICQ INSTANT MESSENGER. **The Community**. Atlanta, USA, 2004. Disponível em: <http://company.icq.com/info/icqstory.html>. Acesso em: 30 maio 2004.

JXTA. **Project JXTA**. San Jose, USA, 2004. Disponível em: <http://www.jxta.org >. Acesso em: 01 jul. 2004.

JXTA PROGRAMMER GUIDE. **Project JXTA v2.0: Java Programmer´s Guide**. San Jose, USA, 2004. Disponível em: < http://www.jxta.org/docs/JxtaProgGuide_v2.pdf >. Acesso em: 01 jul. 2004.

MICROSOFT CORPORATION. **MSN Messenger**. São Paulo, 2004. Disponível em: <http://messenger.msn.com>. Acesso em: 30 maio 2004.

NAPSTER. **Share of music**. San Diego, USA, 2004. Disponível em: <http://www.napster.com >. Acesso em: 20 abr. 2004.

SILVA, William Roger Salabert. **Introdução às redes Peer-to-Peer(P2P)**. Rio de Janeiro, [2003?]. Disponível em: < http://www.gta.ufrj.br/seminarios/semin2003_1/william/index.htm >. Acesso em: 15 mar. 2004.

WILSON, Brendon J. **Project Jxta book**. Vancouver, Canada, [2004?]. Disponível em: <http://www.brendonwilson.com/projects/jxta/pdf/JXTA.pdf>. Acesso em: 27 maio 2004.

PolManXML Protocolo para Distribuição de Políticas de Gerenciamento

Aujor Tadeu Cavalca Andrade (UFSC)
tadeu@lrg.ufsc.br

Carlos Becker Westphall (UFSC)
westphal@lrg.ufsc.br

Resumo. Este artigo tem como objetivo propor a extensão do uso de gerenciamento baseado em políticas de rede. Para tanto, foi criado um protocolo de distribuição de políticas para a comunicação entre o servidor de políticas (PDP) e os dispositivos (PEP), capaz de analisar parâmetros de gerenciamento, reportar condições distintas e aplicar as políticas definidas para o gerenciamento da rede. Neste cenário é definido o XML como meio difusor de políticas entre os participantes do sistema distribuído de gerenciamento. Nossa proposta de distribuição de políticas através do XML, visa otimizar o funcionamento do framework da IETF, possibilitando gerenciamento distribuído, extensibilidade do modelo, flexibilidade no gerenciamento, portabilidade e por fim a capacidade de evolução e crescimento.

Palavras-chave: Gerenciamento por políticas, gerência de redes e PBNM.

1 Introdução

O gerenciamento de redes tem como finalidade conservar o funcionamento da rede como um todo, desde os serviços apresentados aos usuários a transmissão dos dados. Conforme evolução das redes o gerenciamento assumiu papel cada vez mais destacado, onde os administradores de rede aprendem que adicionar largura de banda não é capaz de resolver problemas relacionados ao tráfego e garantir qualidade de serviço às aplicações/serviços não é somente reservar ou diferenciar recursos. Quando há solicitação de um serviço, é essencial conferir os recursos necessários e verificar a disponibilidade para a execução do mesmo. Neste contexto, especificar parâmetros de comportamento para cada elemento, suas características e funcionalidades de forma a ajudar inteligentemente a administração de recurso da rede, por meio de um conjunto de regras que trata dos procedimentos de aplicação e usuários, de modo adaptável a novas condições é a proposta do gerenciamento de redes baseadas em políticas.

O gerenciamento por políticas vem sendo pesquisado pela comunidade científica à medida que seus benefícios vêm sendo constatados. A criação de um grupo de trabalho pela IETF para a pesquisa específica de gerenciamento baseado em políticas [Grupo de Políticas] vem ganhando grande destaque, impulsionando o desenvolvimento e aperfeiçoamento da gerência de redes.

Neste artigo, o objetivo é aperfeiçoar a distribuição de políticas utilizando XML, a fim de identificar sua aplicabilidade e flexibilidade em diversas condições, ampliando sua utilização e verificando sua funcionalidade contribuindo, para pesquisas em gerenciamento de redes. Este artigo apresenta na seção 2 os principais aspectos da arquitetura de gerenciamento de redes baseado em políticas – PBNM; Na seção 3 temos a proposta do protocolo de distribuição de políticas de gerenciamento e seus requisitos; A geração de regras/políticas da proposta na seção 4; Aplicação do protocolo de gerenciamento na seção 5; Na seção 6 conclusões e trabalhos futuros.

2 Arquitetura de Políticas IETF

O gerenciamento baseado em políticas permite formalizar e declarar o comportamento do sistema. Também objetiva orientar, limitar ou aplicar o comportamento definido em entidades autônomas. Em [Sloman et at., 2002] são definidas políticas como "regras que governam as escolhas do comportamento de um sistema", enquanto [Flegkas et al., 2001] descrevem como o "caminho para orientar o comportamento de uma rede ou sistema distribuído através diretrizes informativas de alto nível". As técnicas de políticas diferem de regras de negócios por serem focadas no gerenciamento do sistema e expressas em linguagens entendidas e aplicadas pelo sistema. A gerência de rede baseada em Políticas aplica as políticas ajustadas ao princípio do gerenciamento de redes de computadores.

Os trabalhos iniciais da IETF foram feitos pelo grupo de trabalho de protocolo de locação para recursos - RAP. Eles definiram o *Commom Open Policy Service* (COPS) [RFC 2748, 2000] como protocolo que permite a troca de políticas de informação entre o Ponto de Decisão de Política (PDP) e o Ponto de Aplicação da Política (PEP). Eles definem o PDP como ponto onde as políticas são criadas, enquanto o PEP é o ponto onde as políticas de decisão são realmente aplicadas [RFC 2753, 2000]. COPS é um protocolo adaptável por que permite vários tipos diferentes de clientes e determina a estrutura e armazenamento da informação da política a ser trocada. O gerenciamento de rede baseado em políticas tem motivado amplas pesquisas na última década dentro de um grupo de trabalho IETF (*Internet Engineering Task Force*). O grupo de trabalho do framework de política da IETF criou o COPS, originalmente para ser compatível com a integração de serviços. Hoje em dia é usado como a proposta geral do protocolo de políticas e tem diferentes extensões assim como o *Provisioning Extension* (COPS-PR) [RFC 3084, 2001] usado para suportar configurações de gerenciamento.

Os grupos da IETF e DMTF (*Distribuited Management Task Force*) trabalham juntos para criar diferentes uniformizações relacionadas a padronização de políticas: A IETF é encarregada de definir a arquitetura e a DMTF de especificar como as políticas são armazenadas no repositório de políticas [Boyle, J., et al., 1999].

2.1 Componentes do Framework

Ferramenta de Gerenciamento e Políticas: Fornece a interface para fazer a administração específica da rede e armazenar as políticas no repositório. Também atua conforme monitor de estados e gerência a redes por meio das políticas.

Repositório de Políticas: É um armazém que usa o ponto de decisão para recuperar políticas. As políticas são armazenadas em um alto nível, independentes dos dispositivos da rede e conforme modelo de informação, o modelo indicado pela IETF é *Commom Information Model* – CIM, entretanto repositórios como banco de dados, diretórios ou *web servers* são também amplamente utilizados. O uso de um protocolo de acesso é requerido com a necessidade de se integrar aos outros componentes. A IETF sugere o uso do *Lightweigh Directory Acess Protocol* - LDAP [RFC 2251], entretanto novamente, outras soluções são possíveis como HTTP, SNMP e SSH etc. Na figura 1, temos a arquitetura básica de gerenciamento de políticas e também os protocolos mais utilizados para distribuição, configuração e acesso ao repositório de políticas.

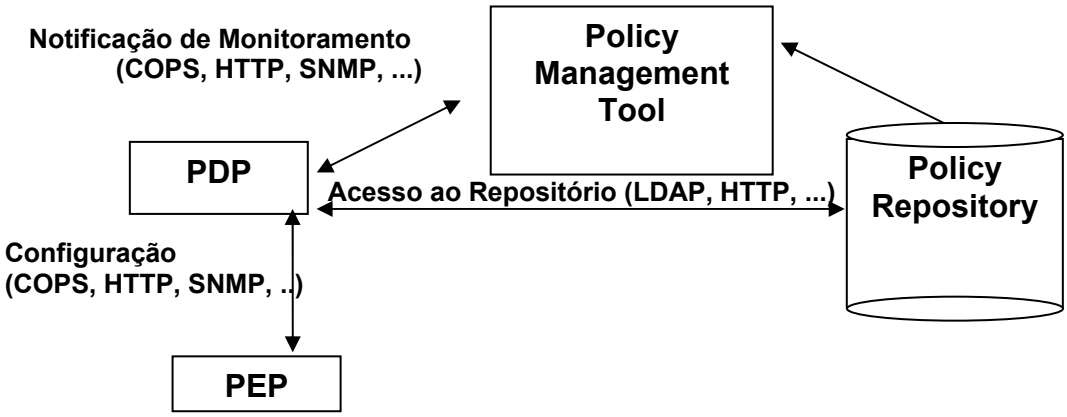


Figura 1: Framework de Políticas da IETF.

Ponto de Decisão da Política - *Policy Decision Points (PDPs)*: São pontos onde todas as decisões que devem ser aplicadas na rede são criadas. PDPs processam a políticas e informações sobre o estado da rede para decidir quais políticas são necessárias aplicar. Estas políticas são enviadas conforme configuração dos dados para PEP ou PEPs correspondentes. A arquitetura considera a possibilidade de ter um componente PDP para um ou vários PEPs e estes ficam com todas regras dos dispositivos.

Ponto de Aplicação da Política – *Policy Enforcement Point* (PEP): São itens envolvidos na execução das políticas que o PDP ordena. Alguns exemplos desses PEPs são roteadores, firewalls, hosts etc. Cada PEP recebe as políticas na forma de configurações específicas de ações, sendo responsáveis pela estabilidade nos dispositivos. O PEP pode também informar ao PDP quando qualquer situação desconhecida for existente. O framework IETF estabelece o COPS [RFC 2753, 2000] para transferência das políticas de decisões para os PEPs e para requisições de PEPs para o servidor de políticas. O modelo é aberto para o uso de outros mecanismos assim como HTTP, FTP ou SNMP. Tendo um grande número de PDPs no interior da rede irá aumenta a dificuldade de gerenciar os PEPs, pois eles iram precisar ser informados de onde localizarem os PDPs para cada tipo de política.

Quanto a Distribuição das políticas, tem basicamente duas formas:

- No modelo *push* a ferramenta de gerenciamento baixa a política do repositório de políticas e envia (uploads) para o PDP. O *upload* para o PDP pode ser executado usando FTP, HTTP, SNMP, dentre outros protocolos [J. Strassner, 2003];
- No modelo *pull* a ferramenta de gerenciamento somente informa o PDP sobre a localização (URL) da política que precisa ser desenvolvida para o componente do PEP. No modelo *pull*, cada PEP é responsável pela comunicação com o PDP e por restabelecer a configuração. O PEP deve também notificar mudanças na comunicação e eventos para o PDP, quando for feita uma nova política de decisão. Também nesse caso, FTP, HTTP, SNMP e protocolos proprietários podem ser usados. A IETF padroniza o uso do LDAP [M. Wahl, 2001].

3 Proposta do Protocolo para Distribuição de Políticas

A proposta para distribuição de políticas através do XML, visa otimizar o funcionamento do framework da IETF, possibilitando gerenciamento distribuído, extensibilidade do modelo, flexibilidade no gerenciamento, portabilidade e por fim a capacidade de evolução e crescimento. A

distribuição de política de gerenciamento através do protocolo COPS padronizado IETF, prevê ao menos um servidor de políticas (PDP) existente em cada domínio administrativo, estabelecendo o transporte através do protocolo TCP e suportando diversos clientes específicos.

As soluções do mercado para política de gerenciamento não seguem as recomendações estabelecidas pela IETF, mas estas têm grande aceitação no mercado. Como exemplo, o PolicyXpert [HP, 2001] da HP e o QPM (QoS Policy Manager) [Cisco, 2000] da Cisco. Porém, todas estas soluções possuem deficiência de integração. O cenário da proposta se concentra entre o ponto de decisão da política - PDP e o ponto de aplicação da política - PEP, onde é criado um protocolo de distribuição de políticas para a comunicação. Este protocolo é inspirado nos fundamentos do COPS, mas implementado em uma linguagem de marcação desenvolvida pela W3C (*World Wide Web Consortium*) o XML (*Extensible Markup Languagem*). Para melhor escolha do protocolo de distribuição de políticas devem ser considerados fatores de extrema importância:

- Todos os PDPs do domínio devem suportar o protocolo adotado para distribuição; e
- Para diferentes domínios a ferramenta de gerenciamento de políticas precisa traduzir o protocolo utilizado.

A idéia de aplicar o XML como meio distribuidor de políticas de gerenciamento surgiu dele proporcionar suporte a integração de grande variedade de aplicações Web, sobrepondo limitações do HTML (*Hiper Text Markup Languagem*) [W3C]. A linguagem de marcação mais popular é o HTML amplamente utilizado por milhares de aplicações de vários níveis em sistemas computacionais, e.g. browsers, software de comunicação e outros. A utilização do XML para difusão das políticas permite maior flexibilidade por ser um padrão W3C. Sua grande aceitação advém de sua portabilidade, HTTP como meio de comunicação e extensibilidade. Para possibilitar a troca de informações de gerenciamento de forma estruturada entre o PEP e o PDP, é necessária a definição do documento XML. O protocolo SOAP provê a definição de um documento XML, permitindo a comunicação entre dispositivos e objetos de qualquer linguagem e em qualquer plataforma [Clements, 2002]. Uma mensagem SOAP é um documento XML que contém as seguintes partes conforme figura 2 [JavaTM, 2002]:

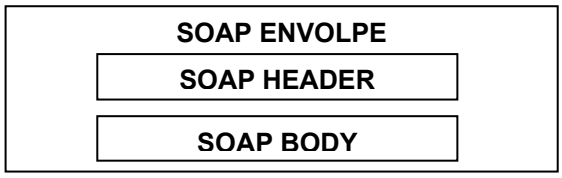


Figura 2: Formato da mensagem SOAP

- Envelope: bloco externo que representa a mensagem;
- Cabeçalho (Header): bloco genérico para adicionar funcionalidades a uma mensagem SOAP, define atributos e funcionalidade. Obrigatório ou opcional.
- O Corpo (Body): bloco para as informações a serem enviadas ao destinatário.

3.1 Aspectos de Implementação

Para a concepção dos campos da mensagem foi realizada uma análise dos principais requisitos necessários a transmissão. Como resultado obteve-se os requisitos de performance, eficiência, funcionalidade, flexibilidade e alcance no ambiente gerenciado.

A proposta se concentra nas seguintes ações :

- Requerer ações corretivas para dispositivos;

- Reportar mudanças nas configurações; e
- Registrar os recursos gerenciados;

Na construção dos DTDs (*Document Type Definitions)* foram definidos os tipos de elementos e atributos necessários para o documento XML válido, bem como a ordem necessária para o funcionamento. A seguir serão apresentados os componentes: cabeçalho e corpo da mensagem.

3.2 Componentes do Cabeçalho

Descrição dos componentes formadores do cabeçalho da mensagem de Requisição/Atualização. Na figura 3 temos a representação gráfica.

- Campo Origem: máquina que gerou a solicitação ou atualização ao ponto de decisão PDP;

- Campo Destino: responsável pelo endereço da PDP, podendo ser um endereço IP ou URL;

- Campo Tipo de Mensagem: identifica qual a finalidade da mensagem. No modelo criado existem dois tipos:

  - Mensagem de Atualização: O PEP informa a situação do dispositivo ao PDP;
  - Mensagem de Requisição: solicitação de uma política corretiva ao PDP para estabelecer o funcionamento pré-determinado pelo administrador;

- Campo Tempo: indica o instante de tempo (dd/mm/aa) e (hh/mm/ss) do envio da mensagem ao PDP. Possibilita um controle mais efetivo;

- Campo ReqID: responsável por identificar a solicitação para o PDP.

- Campo SNY (sincronização): define um número de seqüência das mensagens para controle das políticas pelo PDP.

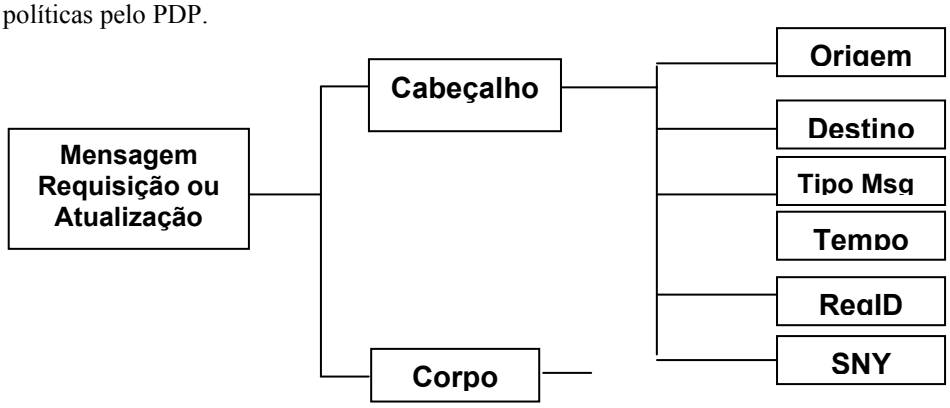


Figura 3: Componentes do Cabeçalho da Mensagem.

3.3 Componentes do Corpo

Descrição dos componentes formadores do corpo da mensagem de Requisição/Atualização e representação gráfica na figura 4.

- Campo Tipo de Mensagem: classifica os acontecimentos no dispositivo PEP, onde pode haver grande variedade de ações referentes ao gerenciamento;

- Campo Prioridade: define qual mensagem terá preferência no envio pelo PDP;

- Campo Tempo: tempo da detecção da degradação da política no dispositivo.

- Campo Parâmetro: Dividido em duas partes;

- Campo ParamentroID: são os parâmetros solicitados pela mensagem de requisição para envio de uma política de correção;
- Campo Valor: valores dos parâmetros para correção da degradação da política.

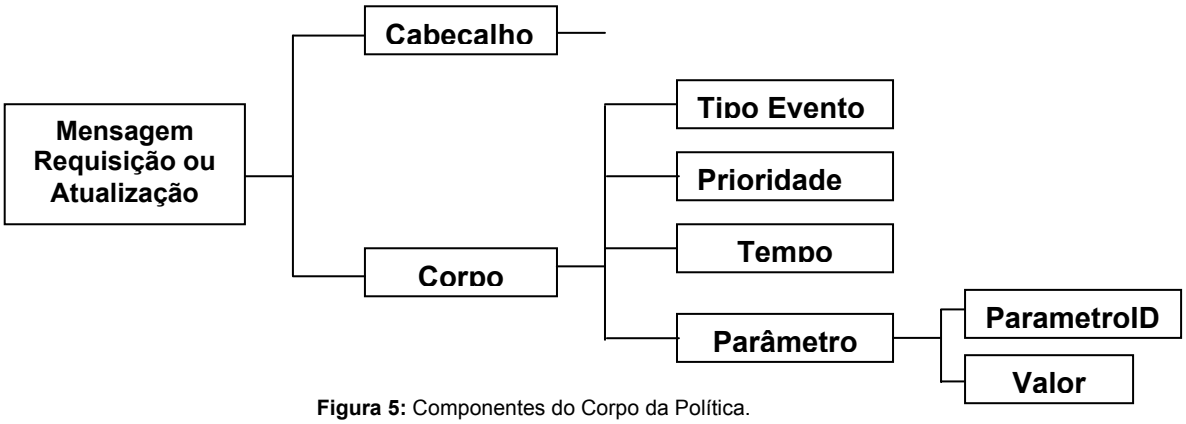


Figura 5: Componentes do Corpo da Política.

3.4 Formato em XML da Mensagem de Requisição e Resposta

Na figura 6 temos a mensagem de Requisição em XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 2 U (http://www.xmlspy.com) by pino (pino) -->
<!ELEMENT Request (Cabecalho?, Corpo?)>
<!ELEMENT Cabecalho (#PCDATA)>
<!ATTLIST Cabecalho
        Origem CDATA #REQUIRED
        Destino CDATA #REQUIRED
        TipoMsg CDATA #REQUIRED
        Tempo CDATA #REQUIRED
        ReqID CDATA #REQUIRED
>
<!ELEMENT Corpo (Evento+)>
<!ELEMENT Evento (Parametro+)>
<!ATTLIST Evento
        TipoEvento CDATA #REQUIRED
        Prioridade CDATA #REQUIRED
        Tempo CDATA #REQUIRED
<!ELEMENT Parametro (#PCDATA)>
<!ATTLIST Parametro
        ParametroID CDATA #REQUIRED
        Valor CDATA #REQUIRED
>
```

Figura 6: Mensagem de Requisição em XML.

4 Geração de Regras/Políticas

No gerenciamento de redes de computadores existem diferentes características peculiares a cada ambiente. Para construção de políticas eficientes é fundamental entender todos os requisitos necessários às aplicações e ao tráfego de dados na rede. A políticas de gerenciamento trabalham na perspectiva de otimizar o funcionamento do gerenciamento, eficiência do administrador, e possibilitar a integração do gerenciamento entre diferentes domínios administrativos. Com relação a políticas de gerenciamento entre diferentes domínios administrativos, a grande empenho das empresas na integração de soluções PBNM e também por parte comunidade científica. As políticas são construídas a partir das características de cada rede gerenciada. Exemplos abaixo, mostra a estrutura da geração de políticas e sua aplicabilidade.

Neste exemplo de gerenciamento de desempenho foi determinado que as máquinas com os números IP ao lado, terão sua Largura de Banda reduzida e aumentada a taxa de perda de pacotes. Com as regras desta política, as máquinas trabalharão com 30 kbps, proporcionando maior Largura de Banda para outras máquinas pertencentes à rede.

Política: Gerenciamento de Desempenho
Se (Trafego <= 150 Kbps) **e** (Perda >= 20%)
e (IP = 150.162.6.125) - Máquinas da Rede
e (IP = 150.162.6.132) - Máquinas da Rede
e (IP = 150.162.8.166) - Máquinas da Rede
então
Largura de Banda = 80 Kbps
Perda de Pacotes = 30 %

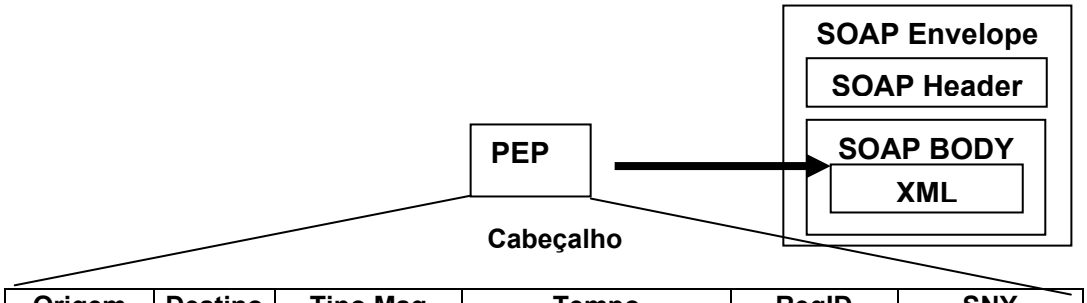
5 Distribuição e Aplicação das Políticas de Gerenciamento

Para construção do protocolo para distribuição de políticas de gerenciamento, é necessária a abstração de alguns pontos do extenso framework da IETF. Com isto se pode limitar melhor o foco do trabalho. Como mencionado na descrição funcional, a área de concentração do trabalho está entre o PDP e o PEP. Onde se desenvolve toda a troca das informações e o uso efetivo do protocolo de distribuição sugerido.

A proposta deste protocolo destaca–se para a distribuição de políticas "*Pull*" e para a configuração de dispositivos "*outsourcing*". A distribuição de política "*Pull*" prevê que a ferramenta de gerenciamento informe ao PDP a localização da política, e a configuração do dispositivo "*outsourcing*" define a requisição da política a partir do PEP para a PDP. As abstrações mencionadas são: a ferramenta de gerenciamento e a monitoração dos dados dos dispositivos. Onde a ferramenta de gerenciamento foi abstraída pelo fato do objetivo ser o protocolo de distribuição das políticas, no qual se concentrou a função de ferramenta de gerência e repositório de políticas no PDP. Foi adotada essa postura para dar maior ênfase no protocolo proposto e diminuir muito da complexidade do framework da IETF.

Completando, a monitoração pelo fato de existir vários mecanismos já amplamente utilizados e contextualizados em diversos trabalhos. Contudo podemos citar o SNMP e o WMI e também a construção de agentes para este fim. Para exemplificar como o protocolo de distribuição de política trabalha, será demonstrado detalhadamente o seu procedimento. O cenário criado para este exemplo utiliza as políticas de videoconferência descritas acima. O desenvolvimento da implementação está em sua fase final utilizando Java API for XML *Messaging* (JAXM) para efetuar a troca de mensagem baseada em SOAP/XML.

Cenário: O Dispositivo constata uma condição não prevista pelo gerenciamento. O PEP cria um documento XML contendo as informações atuais, requerendo uma política corretiva. A mensagem é encapsulada utilizando o SOAP e transmitida sobre o HTTP pelo requisitante. A infraestrutura HTTP envia a mensagem para ao servidor de políticas. A figura 7 demonstra o formato do protocolo e todas as informações ocorrentes no dispositivo para envio ao PDP.



Cabeçalho

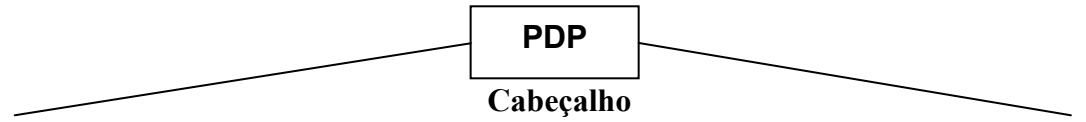
Origem	Destino	Tipo Msg	Tempo	ReqID	SNY
PEP A1	PDP	Requisição	04/03/04 – 11:45 hs	ReqA1	001

Corpo

Tipo Evento	Prioridade	Tempo (min)	Parâmetro	
Desempenho	01	00:05	**ParametroID**	**Valor**
			Bandwith	50 Kbps
			Delay	300 ms
			Perda de Pacotes	35 %

Figura 7: Mensagem de requisição do PEP.

PDP é responsável pelo processamento da requisição e por gerar uma resposta. A resposta também será encapsulada pelo SOAP e utilizará os mesmos campos do cabeçalho da requisição com valores correspondentes. Na figura 8 a demonstração da mensagem de resposta, contendo no cabeçalho informações do dispositivo e a política com novos parâmetros de desempenho.



Cabeçalho

Origem	Destino	Tipo Msg	Tempo	ReqID	SNY
PDP	PEP A1	Resposta	04/03/04 – 11:56 hs	ReqA1	002

Corpo

Política	Parâmetro	
Desempenho PEP A1	**ParametroID**	**Valor**
	Bandwith	200 Kbps
	Delay	100 ms
	Perda de Pacotes	10 %

Figura 8: Mensagem de resposta do PDP.

6 Conclusão

A distribuição de políticas de gerenciamento como foi proposto, permite o gerenciamento entre domínios. Uma vez que todos domínios que utilizam COPS precisam de um PDP local, o protocolo concebido se destaca por não ter essa condição, por flexibilidade de sua arquitetura e o gerenciamento de dispositivos distribuídos. Por trabalhar na camada de aplicação é possível agregar novas funções diferentemente do padronizado que trabalha na camada de transporte.

Outro ponto de favorecimento é o meio de comunicação utilizado para distribuição das políticas o HTTP, por sua portabilidade e extensibilidade. Complementando, a capacidade de desenvolvimento e evolução do XML são uma das peças fundamentais para a flexibilidade do protocolo. A utilização do protocolo SOAP para definir o formato XML para troca de mensagens é decisiva para a distribuição, além de proporcionar independência de plataforma e utilizar protocolo de grande credibilidade como o HTTP e SMTP. Estes fatores, juntamente com os *Web services*, contribuem para que a utilização do SOAP aumente como tecnologia de comunicação para sistemas distribuídos .

A configuração dos dispositivos mediante o recebimento das políticas não é abordado neste artigo. Pois o protocolo utilizado tem a função de troca de mensagens entre o PEP e o PDP. Todavia, os mecanismos para estabelecer as políticas requisitadas aos dispositivos podem ser criados (agentes) ou adaptados aos já existentes (SNMP, WMI, etc.). As contribuições desta pesquisa estão na distribuição de políticas através do XML, onde foi otimizado o funcionamento do framework da IETF, possibilitando gerenciamento distribuído, extensibilidade, flexibilidade, portabilidade e por fim capacidade de evolução.

Em trabalhos futuros parâmetros de performance deverão ser pesquisados para atestar a eficiência do protocolo. Também construir um repositório utilizando o modelo CIM e acrescentar uma variedade de dispositivos gerenciados bem como políticas de gerenciamento mais abrangentes.

Referências

BOYLE, J., et al., **The COPS (*Common Open Policy Service*) Protocol**, February 1999,

CLEMENTS, Tom. **Overview of SOAP.Java Developer Connection**, 2002.

DURHAM, D.; BOYLE, J.; COHEN, R.; HERZOG, S.; RAJAN, R.; and SASTRY, A.; (2000) "**The COPS (*Common Open Policy Service*) Protocol**", RFC 2748, Network Working.

STRASSNER, J.; "**Policy-Based Network Management**": Solutions for the Next Generation: Morgan Kaufmann, 2003.

SLOMAN, M.; and LUPU, E.; "**Security and Management Policy Specification,**"*IEEE Network*, pp. 10-19, 2002.

HEWLETT-PACKARD. **HP OpenView PolicyXpert Homepage**.
Disponível em: <http://www.openview.hp.com/ >.

CISCO SYSTEMS. **Cisco QoS Policy Manager (QPM) Homepage**. Disponível em: <http://www.cisco.com/ >.

SLOMAN, M,; "**Policy Based Management of Telecommunication Systems and Networks**", presented at First UK Programmable Networks and Telecommunications Workshop, Hewlett-Packard Laboratories, Bristol, 1998.

FLEGKAS, P.; TRIMINTZIOS, P.; PAVLOU, G.; ANDRIKOPOULOS, I.; and CAVALCANTI, C. F.; "**Policy-based Extensible Hierarchical Network Management**," presented at Proceedings of Workshop on Policies for Distributed Systems and Networks (Policy 2001), Bristol, U.K. P. Flegkas et al., "Design and Implementation of a Policy-Based Resource Management Architecture", IEEE/IFIP IM, 2003.

WAHL, M.; HOWES, T.; Critical Angle Inc; "**Lightweight Directory Access Protocol**" RFC 2251. March 2001, http://www.ietf.org/rfc/rfc2251.txt

Ferramenta CASE JM Designer

Jonathan Manoel Borges (FURB)
jmborges@inf.furb.br

Categoria: Banco de Dados / Engenharia de Software

Linguagem de programação: Object Pascal (Delphi)

Sistema operacional: Windows

Palavras-chave: Ferramenta Case, Diagrama Entidade Relacionamento

1 Contexto

Neste trabalho é apresentada uma ferramenta CASE que auxilia no desenvolvimento de Diagramas de Entidade Relacionamento. A ferramenta JM Designer é uma ferramenta visual de fácil manipulação que auxilia tanto no projeto quanto no desenvolvimento, pois gera código SQL para a implementação do banco de dados.

Surgiu de um trabalho acadêmico envolvendo as disciplinas de Banco de Dados e Engenharia de Software. Foi apresentada em forma de seminário na disciplina de Banco de Dados II (professor Alexander Roberto Valdameri) e referente ao trabalho de desenvolvimento de uma ferramenta CASE na disciplina de Engenharia de Software (professor Everaldo Arthur Grahl).

Optou pelo desenvolvimento desse software por dois motivos. Grande parte teórica, base para o desenvolvimento do software, havia sido vista nas disciplinas de Banco de Dados I e II e na disciplina de Engenharia de Software havia sido feita uma análise desse tipo de ferramenta, destacando os pontos fortes e fracos de algumas ferramentas do gênero.

Um dos objetivos do trabalho é motivar os acadêmicos no desenvolvimento de ferramentas desse porte. Para isso, a ferramenta é um software livre (Open Source) com o código fonte disponível na internet pela URL http://www.inf.furb.br/~jmborges/JMDesigner, permitindo que outros acadêmicos dêem continuidade no projeto para, quem sabe, tornar-se uma ferramenta livre conceituada no mercado nacional, levando consigo o nome da FURB.

2 Desenvolvimento

O software segue o padrão das ferramentas de Diagrama Entidade Relacionamento. Suas funcionalidades básicas são incluir tabela com seus respectivos campos e relacioná-las. Estas opções encontram-se tanto no menu ferramentas quanto na barra de tarefas. As tabelas, os campos e os relacionamentos podem ser editados (duplo clique) e excluídos (selecionando e pressionando DEL).

As funcionalidades apresentadas anteriormente dizem respeito à edição do Diagrama Entidade Relacionamento (etapa de projeto), porém a ferramenta auxilia também na etapa de desenvolvimento, gerando script SQL para implementação do Banco de Dados Relacional, além de documentação em HTML.

O script SQL gerado trás comentários e pode ser executado no banco Mysql. Também trás a opção de criar índices automáticos em campos de chave estrangeira (Foreign Key), além de poder escolher o tipo de tabela do Mysql, entre os tipos MYISAM, InnoDB, HEAP, BDB, ISAM e MERGE.

A documentação em HTML gerada pelo software é bem completa, trazendo todos os comentários, tabelas, campos e relacionamentos. A figura do Diagrama também é inclusa na documentação e utiliza a técnica de "mapa" para poder navegar na documentação através da figura. Nesta técnica atribui-se âncoras a determinadas áreas da figura. Ou seja, o usuário ao clicar na figura da tabela, a página HTML desloca para a descrição da tabela. O mesmo ocorre com os relacionamentos.

Em função do tempo para desenvolvimento, foi escolhido o ambiente de desenvolvimento Delphi, que é uma ferramenta de rápido desenvolvimento, pois possui uma série de componentes prontos, o que facilita no processo de desenvolvimento da interface do software.

Os principais componentes utilizados no projeto foram o TStringList (uma lista de objetos) para implementar a estrutura de dados e o TCanvas (exibe figuras e imagens em geral) utilizado na parte gráfica.

A estrutura de dados do software pode ser dividida em duas partes. Os objetos bases do software e as listas de objetos.

As listas de objetos, compostas pelas classes TTableList, TFieldList e TRelationList, herdadas do componente TStringList do Delphi, implementam listas dos objetos bases dos softwares. Suas funções são adicionar, remover e retornar objetos das listas.

As classes TTable, TField e TRelation implementam os objetos bases do software: tabela, campo e relacionamento. A *tabela* armazena nome, comentário e a lista de campos, além da posição da tabela na área de trabalho. O *campo* armazena nome, tipo, tamanho e a lista de relacionamentos, além da tabela a qual o campo faz parte. O *relacionamento* armazena as tabelas e os campos envolvidos no relacionamento.

Algumas técnicas de computação gráfica foram utilizadas no desenvolvimento do trabalho. São procedimentos para desenhar as tabelas e os relacionamentos no Canvas (área de trabalho).

Entre outros, podemos destacar o procedimento de "redesenho". Este é utilizado para atualizar as dimensões das tabelas e relacionamentos e executado sempre que um objeto gráfico (tabela ou relacionamento) é modificado. Ao executar este procedimento, percorre-se uma lista (TTableList) onde estão armazenadas todas as tabelas do projeto. Para cada tabela são executadas três ações: remove a tabela do Canvas, recalcula as dimensões da tabela e redesenha a tabela no Canvas. O mesmo acontece com os relacionamentos.

3 Resultados

Como resultado do desenvolvimento, encontramos uma ferramenta de fácil manipulação que pode ser utilizada no desenvolvimento de Diagramas Entidade Relacionamento. Gera script SQL para o banco de dados Mysql, além de gerar documentação bem completa em HTML.

Algumas restrições encontradas na versão atual servem como sugestões para as próximas versões da ferramenta:

- uma tabela não permite chave primária com mais de um campo
- os tipos de dados dos campos estão restritos a apenas cinco tipos do banco Mysql: CHAR, DATE, INTEGER, NUMERIC, TIME e VARCHAR

Pretende-se ainda, para as próximas versões, permitir que o software salve os projetos em XML, compatível com outras ferramentas, como por exemplo, o DB Designer.

4 Referências

VALDAMERI, Alexander R. **ARV – Alexander Roberto Valdameri**. Disponível em: http://www.inf.furb.br/~arv. Acesso em 20 jun 2004.

Baba XP: encante seus clientes com *extreme programming*

Giovane Roslindo Kuhn (FURB)
brain@netuno.com.br

Vitor Fernando Pamplona (FURB)
vitor@javafree.com.br

Categoria: Engenharia de Software.

Linguagem de programação: Java

Sistema operacional: Qualquer um que suporte JRE 1.4

Palavras-chave: *Extreme Programming*, Engenharia de Software, Qualidade de Software, Metodologia de Desenvolvimento, Processos Ágeis de Desenvolvimento, XP, Requisitos de Software.

1 Contexto

O *Extreme Programming* (XP) é um processo para desenvolvimento ágil de software voltado para projetos cujos requisitos mudam freqüentemente, desenvolvimento orientado a objetos, equipes pequenas (12 desenvolvedores) e desenvolvimento iterativo. Este processo busca assegurar que o cliente receba o máximo de valor a cada dia de trabalho da equipe.

O XP se baseia em quatro valores fundamentais: *feedback* (cliente gera constante retorno a equipe de desenvolvimento), comunicação (boa comunicação para agilizar o retorno do cliente a equipe), simplicidade (implementar o necessário para atender a necessidade do cliente) e coragem (conseguir colocar em prática todos os valores e práticas do XP). Além dos valores o XP possui resumidamente as seguintes práticas: cliente presente, jogo planejamento, *stand up meeting*, programação em par, desenvolvimento guiado pelos testes, *refactoring*, código coletivo, código padronizado, *design* simples, metáfora, ritmo sustentável, integração contínua e *releases* curtos.

Devido a crescente utilização do XP nas empresas desenvolvedoras de *software* e a falta de qualidade nas ferramentas existentes no mercado, constatou-se a necessidade de desenvolvimento de uma ferramenta para controle de projetos baseada em práticas de *Extreme Programming*, chamada **Baba XP**, e apresentada como trabalho para a disciplina de Requisitos de *Software*.

O *software* tem como objetivo controlar um projeto, bem como as estórias (similares aos casos de uso, porém descritos pelo cliente) e testes de aceitação requisitados pelo cliente. Os desenvolvedores podem ter controle de estimativas, prioridades das estórias que devem ser implementadas e relacionamentos entre as mesmas.

Em cada projeto é disponibilizada uma tabela para análise de produtividade dos desenvolvedores, sendo uma ótima alternativa para detectar o conhecimento e agilidade das duplas formadas e dos desenvolvedores envolvidos no projeto.

O *software* deve ser simples o suficiente para que os desenvolvedores XP se agradem e aceitem a ferramenta, já que muitos deles evitam *softwares* burocráticos e com itens desnecessários. Outra questão importante do projeto é que o mesmo execute independente da plataforma utilizada. Baseado nos incentivos federais aos *softwares* livres o **Baba XP** está aberto a comunidade, onde qualquer desenvolvedor interessado pode auxiliar na evolução da ferramenta.

2 Desenvolvimento

Para atender os requisitos levantados, a linguagem de programação escolhida para o desenvolvimento é o Java, pela sua independência de plataforma e forte aderência a comunidade de *softwares* livres.

Por ser uma ferramenta simples, optou-se por uma aplicação mono-usuário e *desktop*. Este tipo aplicação é de fácil instalação e configuração, sem a necessidade de um servidor. A *API (Application Programming Interface)* utilizada para o desenvolvimento da aplicação *desktop* é o *Swing*, uma *API* bastante completa, porém um pouco complexa de ser utilizada.

Para a persistência dos dados o requisito simplicidade não foi ferido, já que foi escolhido o conjunto de *API's* do *Prevayler* (http://www.prevayler.org), um projeto brasileiro para persistência de objetos Java e com extrema facilidade de uso. A utilização desta *API* brasileira é um dos grandes pontos de inovação do projeto.

O projeto ainda tem mecanismos de internacionalização, permitindo que o mesmo seja traduzido para outros idiomas sem a necessidade de alterar seu código fonte.

3 Resultados

O projeto está incubado no *Java.net* com o nome ***babaxp*.** O *Java.net* é um repositório de projetos Java disponibilizados para a comunidade e o endereço do projeto **Baba XP** é http://babaxp.dev.java.net/

Recentemente o *Java Tools Community*, um grupo que especifica padrões para ferramentas feitas em java, lançou um *newsletter* semanal escolhendo os melhores projetos pelo seu tempo de vida. Na categoria incubado o Baba XP ganhou o destaque de *Great Idea* (https://javatools.dev.java.net/newsletter/20040807.html).

O Baba XP Manager está implementado, mas muito longe de ser finalizado. O projeto estará em constante evolução para atender as necessidades desta incrível metodologia e pode ser melhorado por qualquer pessoa, através de sugestões e idéias ou até mesmo com o desenvolvimento de novas funcionalidades na ferramenta, pois o **Baba XP** é da comunidade !

4 Bibliografia

ASTELS, David; MILLER, Granville; NOVAK, Miroslav. **Extreme programming: guia prático.** Rio de Janeiro – RJ: Campus, 2002.

TELES, Vinícius Manhães. **Extreme programming: aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade.** São Paulo - SP: Novatec Editora Ltda, 2004.

Robô simulado que encontra espelhos: utilizando esquemas motores

Giovane Roslindo Kuhn (FURB)
brain@netuno.com.br

Categoria: Robótica, Inteligência artificial.

Linguagem de programação: Java.

Sistema operacional: Qualquer um que suporte JRE 1.4.

Palavras-chave: Esquemas motores, inteligência artificial, *TeamBots*, robótica.

1 Contexto

O trabalho tem como objetivo desenvolver um robô que encontre um espelho no ambiente de uma casa, utilizando o simulador TeamBots (http://www.teambots.org). O robô simulado deve ser o *Nomad150* e arquitetura para desenvolvimento do robô deve ser com esquemas motores.

2 Desenvolvimento

Inicialmente foram criados três esquemas motores:

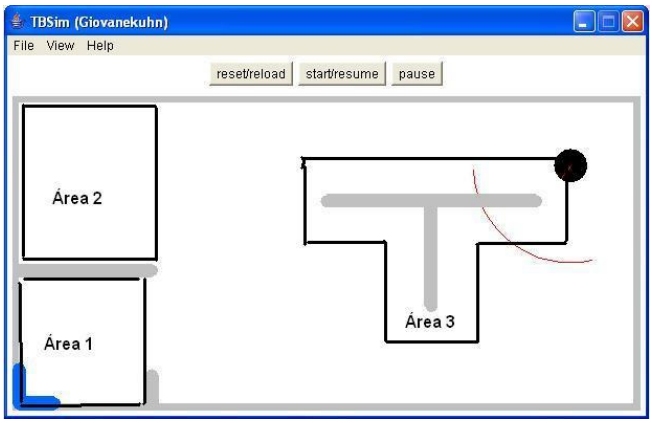


Figura 1

Esquema 1 - Atrator: colocado no objetivo do robô (espelho), desta forma o robô estará sendo atraído pelo objetivo, seu campo de atuação está restrito a **Área 1** da **Figura 1** (vide anexo) e seu fator de influência é o maior de todos os esquemas**.** No restante do ambiente, por estar sendo utilizada a classe *v_LinearAttraction_v* do *TeamBots*, este atrator gera um vetor constante no valor de 1,0.

Esquema 2 - Repulsor: colocado em todas as paredes do ambiente e utilizando a classe *v_Avoid_va* do *TeamBots*, seu campo de atuação é pequeno, porém seu fator de influência é alto para que quando o robô estiver próximo as paredes possa ser "repelido".

Esquema 3 - Noise: esquema utilizado para "desempatar" os dois esquemas descritos acima, seu fator de influência é baixo.

Este conjunto de esquemas não se mostrou eficiente pelos seguintes aspectos:

Problema 1: o vetor constante gerado pelo **Esquema 1** quando o robô está fora da **Área 1** é muito alto, fazendo com que o robô em alguns casos colida contra a parede.

Problema 2: o robô ao entrar na **Área 2** não consegue contornar a parede para entrar na **Área 1,** devido ao forte fator de influência do **Esquema 1.**

Para resolver o **Problema 1** foi alterado o **Esquema 1** criando a classe *MyAttractor*, esta classe tem as mesmas características da *v_LinearAttraction_v* do *TeamBots,* porém quando o robô esta fora da área de atuação do atrator, o vetor gerado é configurável e não mais constante em 1,0.

O **Problema 2** foi resolvido alterando o **Esquema 2**, inicialmente com a classe *v_SwirlLeft_va* do *TeamBots*, desta forma o robô ao se aproximar de uma parede não era mais "repelido" e sim "induzido" para esquerda, com isso o robô ao encontrar uma parede seguia a mesma.

Com esta nova implementação, onde o robô seguia as paredes até achar o espelho, o robô encontrou o espelho em todos os casos exceto quando se aproximava da **Área 3**. Como o robô seguia as paredes, ficava em *loop* contornando as paredes da **Área 3.** Para resolver este novo problema foi criado um sistema de navegação

Para evitar que o robô ficasse em *loop* (andando pelo mesmo caminho), foi criada a classe *MyNavigator*. O simulador *TeamBots* gera um passo do robô a cada 100ms. e a cada passo do robô o sistema de navegação verifica se o robô encontrou uma nova zona (uma área quadrada por onde o robô já caminhou) ou se já não havia passado pela ponto em que se encontra.

Caso o robô caminhe por uma zona que já tenha passado antes em um determinado tempo, ele entra no modo chamado de "**Fuga do loop".** Para isto foi criado um novo esquema (**Esquema 4 – Repulsor**) colocado em todas as paredes do ambiente e quando o robô entra no modo **"Fuga do loop"** a influência do **Esquema 2** é zerada e a do **Esquema 4** é elevada, desta forma o robô pára de seguir as paredes e passa a fugir delas. O robô se mantém neste modo por alguns segundos e depois retorna ao processo inicial.

3 Resultados

Esquemas motores se mostraram bastante eficientes para a resolução deste tipo de problema em ambientes simulados, porém o fato do robô entrar em loop em determinados pontos do ambiente, exigiu uma lógica de navegação para evitar este tipo de acontecimento, com esta implementação o robô encontrou o espelho em todos os casos submetidos e com ponto de saída em vários locais do ambiente.

4 Bibliografia

FERRARI Giulio. **Programming lego mindstorm with java**. Rochland – United States: Syngress Publishing, Inc, 2002.

MURPHY Robin. **Introductions to AI robotics.** Cambridge – United States: Mit, 2000.

Bomba de água controlada pelo PC

Alexandre Helfrich (FURB)
Christian Rogério Câmara de Abreu (FURB)
Juliane Menin (FURB)
{crca, helfrich,juliane }@inf.furb.br

Categoria: Integração H/S

Linguagem de programação: Delphi

Sistema operacional: Windows

Palavras-chave: controlador ON-OFF

1 Contexto

O principal objetivo do software é controlar o nível de água em uma cuba. Para isto serão utilizados um sensor de pressão e um dispositivo *ON/OFF*. A comunicação entre os componentes do sistema é feita via porta paralela de um computador tipo PC. O software no computador controlará os demais componentes do sistema, além de exibir graficamente o comportamento do nível de água.

Segundo Ogata (1993), em um controlador do tipo *ON/OFF*, o elemento atuante possui apenas duas posições fixas que são: ligado e desligado. Em um sistema para controle de nível de água a partir de bombas que utiliza controlador *ON/OFF*, este atua sobre as bombas ligando-as ou desligando-as. Sendo assim, a taxa de entrada de água na cuba ou é uma constante positiva ou é nula.

O software foi desenvolvido para a disciplina de Automação e Controle na Universidade Regional de Blumenau.

2 Desenvolvimento

Para que seja feito o controle do nível de água dentro de uma cuba de vidro, a seguinte regra foi definida: se a água estiver abaixo de 10% da capacidade da cuba, a bomba de água é acionada e se estiver acima de 90%, é automaticamente desligada. Para automatizar este controle, foi construído um software que interage com um sensor de pressão e um controlador *ON/OFF*.

Um sensor de pressão foi acoplado na cuba. A medida que a cuba vai enchendo de água, o sensor é alterado, até um limite estabelecido, onde a bomba é desligada. A partir da leitura do sensor, o software determina o nível de água na cuba, verificando se há necessidade de ligar ou desligar a bomba de água.

Na cuba de vidro representada na figura 1, o orifício presente é a entrada do fluxo de água da bomba. Havendo a necessidade de evitar o vazamento de água, definiu-se como 21 cm a altura máxima que o líquido chegará. Estes 21 cm correspondem a 90% da capacidade da cuba.

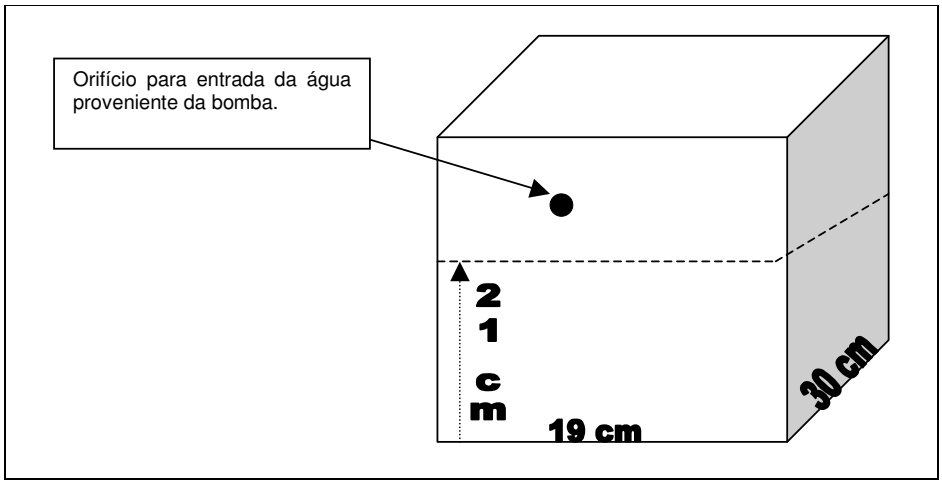


Figura 1 – Cuba de vidro

A tabela 1 apresenta informações detalhadas da situação em que a bomba de água será ligada (nível de água em 10% da cuba) e da situação em que será desligada (nível de água em 90% da cuba). Ainda, também são apresentadas as respectivas leituras no sensor de pressão nestas situações.

Quantidade de água	***Nível de água***	***Altura da água***	***Leitura do sensor de pressão***	***Ação realizada na bomba***
2.388 ml	10%	4 cm	17 kg/cm^2	ligar
12.537 ml	90%	21 cm	19 kg/cm^2	desligar

Tabela 1 – Informações sobre níveis de água na cuba

O software foi desenvolvido no ambiente Delphi 5. Para acesso a porta paralela do computador foi utilizado o componente IOPort (VILELA, 1999).

3 Resultados

Através de um sensor de pressão foi possível determinar o nível da água na cuba de vidro. Com o controlador *ON/OFF* foi possível controlar o nível da água e, através de um computador ligar ou desligar a bomba de água. Desta forma, conseguiu-se automatizar com sucesso o controle do nível de água na cuba.

4 Referências

OGATA, Katsuhiko. **Engenharia de controle moderno**. Rio de Janeiro: Prentice Hall do Brasil, 1993, 81 p.

VILELA, Eduardo Divino Dias. **Programação Delphi para eletrônica – parte 2**, São Paulo, 1999. Disponível em: <www.py4flr.hpg.ig.com.br/informatica/delphi2.pdf>. Acesso em: 12 de setembro de 2004.

Minilogger

Ariberto Montibeller Júnior (FURB)
Aurélio Faustino Hoppe (FURB)
Fernando dos Santos (FURB)
Karly Schubert Vargas (FURB)
{amjunior, aureliof, fds, karly}@inf.furb.br

Categoria: Integração Hardware/Software.

Linguagem de programação: Java, Basic.

Sistema operacional: Windows.

Palavras-chave: Datalogger, Microcontrolador.

1 Contexto

A cada dia torna-se mais imprescindível à captação e o armazenamento de dados. A captação pode ser feita de forma manual ou através de algum equipamento que permita obter as informações de forma precisa e segura. Para ajudar nisso estão sendo desenvolvidas tecnologias de auxílio a coleta e a análise dos dados, por exemplo o datalogger, instrumento eletrônico que grava dados de medidas durante certo tempo. Tipicamente, os dataloggers são dispositivos pequenos, movidos a bateria e equipados com um microprocessador, um armazenador de dados e um sensor. A maioria dos dataloggers utiliza-se de um software que descarrega os dados coletados em um computador. Por serem de porte pequeno, são ideais para atividades que são realizadas em locais remotos ou em lugares onde é necessário armazenagem de dados e não se dispõem de recursos, pois basta uma bateria para que o datalogger funcione.

O datalogger funciona da seguinte maneira: ele é conectado a um computador onde é informado que tipo de dados ele deve armazenar, depois de selecionados os dados ele é desconectado do computador e pode começar a coleta de dados, onde cada dado coletado é gravado na memória. Após a coleta dos dados o datalogger é conectado em um computador e através de um software seus dados são descarregados para o computador.

Havendo esta necessidade, foi desenvolvido na disciplina de Automação e Controle um minilogger. Trata-se de uma versão compacta de um datalogger. O minilogger desenvolvido captura dados através de um sensor, armazenando-os e através de um software permite que os mesmos sejam descarregados no computador. Através de um software específico no computador os dados obtidos são visualizados através de uma listagem ou gráfico, que relaciona tempo *versus* unidade de medida.

2 Desenvolvimento

Uma das etapas do desenvolvimento consistiu na confecção do hardware. O ADC0832, que é um conversor analógico digital em que pode ser acoplado qualquer sensor desde que ele atue numa escala de 0 a 5 volts, é ligado ao PIC16F84A que recebe os dados coletados pelo sensor. Para o armazenamento dos dados é utilizado o 24LC64 (MICROCHIP, 2004) com capacidade de 64 Kbytes, onde são armazenados os dados, data e hora em que foram coletados. Com o armazenamento correto dos dados é necessário transmiti-los para o computador através da interface serial RS232. O MAX232 (MAXIM, 2004a) faz o interfaceamento entre o padrão TTL do PIC e o padrão RS232 do PC. Para implementação do relógio, utilizou-se o DS1307 (MAXIM,

2004b) que conecta-se ao PIC utilizando o mesmo barramento da memória, pois tanto o relógio quanto a memória usam o protocolo de comunicação I2C (PHILIPS, 2004). A figura 1 representa a ligação entre os componentes.

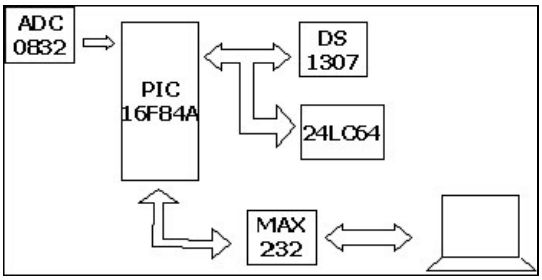


Figura 1: Representação dos componentes do minilogger

Outra etapa foi o desenvolvimento de rotinas para comunicação com o minilogger a partir de um PC. Todas as rotinas que estão envolvidas na comunicação entre o PC e o microcontrolador do minilogger foram agrupadas em uma classe Java. Esta classe utiliza a API Java para Comunicação Serial e Paralela (SUN, 2004) através das portas de comunicação de um PC.

Por fim, desenvolveu-se uma interface gráfica para o usuário. O usuário pode configurar o minilogger (setar hora e data, por exemplo) e também efetuar a leitura dos dados gravados no minilogger. A exibição dos dados em uma listagem ou um gráfico, que relaciona tempo *versus* unidade de medida do sensor utilizado, é disponibilizada através do software. A criação e exibição do gráfico foi desenvolvida com a biblioteca JfreeChart (GILBERT, 2004).

3 Resultados

Atingiu-se todos os objetivos do projeto, sendo que é possível capturar dados através de um sensor. Estes dados são armazenados numa memória dentro do minilogger. Quando conectado a um computador é possível descarregar os dados utilizando o software desenvolvido e visualizar os dados na forma de listagem ou gráfico.

A biblioteca gráfica, JfreeChart, é bem simples de usar, porém apresentou algumas limitações no manuseio das informações e na personalização da interface gráfica.

4 Bibliografia

GILBERT, David; MORGNER, Thomas. **JfreeChart**, [s.l.], 2004. Disponível em: <http://www.jfree.org/jfreechart/index.html>. Acesso em: 10 ago. 2004.

MAXIM. **MAX220 – MAX249**, Sunnyvale , 2004a. Disponível em: <http://pdfserv.maxim-ic.com/en/ds/MAX220-MAX249.pdf>. Acesso em: 15 ago. 2004.

_____. **DS1307**, Sunnyvale, 2004b . Disponível em: <http://pdfserv.maxim-ic.com/en/ds/DS1307.pdf>. Acesso em: 15 ago. 2004.

MICROCHIP. **24LC64**, Chandler, [2004]. Disponível em: <http://www.microchip.com/download/lit/pline/memory/arc/21189f.pdf>. Acesso em: 16 ago. 2004.

PHILIPS. **Philips semiconductors – I2C**, [Eindhoven]**,** [2004]. Disponível em: <http://www.semiconductors.philips.com/buses/i2c/>. Acesso em: 12 ago. 2004.

SUN. **Java communications API**, Santa Clara, 2004. Disponível em: <http://java.sun.com>. Acesso em: 12 ago. 2004.

Sistema para acompanhamento de empresas incubadas

Alexandro Deschamps (FURB)
xando@terra.com.br

Allan Dalmarco (FURB)
allan@senior.com.br

Categoria: Sistemas de informação.

Linguagem de programação: PHP, Javascrip.

Sistema operacional: Sistema WEB independente de plataforma.

Palavras-chave: Acompanhamento de Projetos, NAP.

1 Contexto

Para auxiliar e automatizar o exercício das atividades de coordenação e tutoria de projetos desenvolvidos por empresas incubadas no CRIEM (Centro de Referência em Incubação e Empreendimento) foi proposto na disciplina Gerência de Projetos de Informática o desenvolvimento de um sistema para realizar o acompanhamento e controle destes processos.

O CRIEM se constitui em uma das linhas de ações do Instituto GENE e suas atividades são regidas pelo regulamento do NAP (Núcleo de Acompanhamento de Projetos).

O NAP é constituído por uma coordenação e duas tutorias, acadêmica e de negócios, e tem por objetivo apoiar os projetos incubados em relação a aspectos técnicos e comerciais através da prestação de assessoria especializada e sistemática nestas áreas (MENDES, 2004, p. 6).

O sistema desenvolvido foi denominado NAP On-line e usa como marca o logotipo do CRIEM. Para atender as necessidades do NAP o sistema conta com um site WEB onde são registradas as principais informações de empresas e projetos. Os tutores e a coordenação podem realizar os registros de suas reuniões, emitir pareceres e gerenciar planos de ação, tornando possível medir e avaliar a evolução de empresas e projetos através de indicadores de desempenho. Recursos para gerenciamento das atividades de tutores e o armazenamento de uma base histórica dos registros lançados desde o inicio ao encerramento do projeto, e que são de suma importância para coordenação, podem ser consultados a qualquer momento.

De forma geral o sistema atende aos seguintes requisitos:

- Cadastramento das empresas (registro da certidão da empresa);
- Controle de acesso por usuário
- Registro das capacitações obtidas pelos participantes das empresas;
- Armazenamento de artefatos como documentos e similares, fornecendo opção para download dos mesmos (contrato social, plano de negócio e outros planos da empresa);
- Registrar a cronologia (datas importantes) da empresa no CRIEM;
- Permitir o acompanhamento e controle das empresas incubadas na forma de quadro de encontros, registro de reuniões (datas e anotações), relatório de avaliação, relatório de evolução e controle dos planos de ações.

2 Desenvolvimento

No processo de modelagem e levantamento dos requisitos foi utilizada a ferramenta Power Designer 10, na qual foram desenvolvidos o diagrama entidade-relacionamento e o modelo físico dos dados.

Na criação e design da interface foi utilizado como base o site do CRIEM, procurando manter o mesmo padrão visual, respeitando seus os princípios ergonômicos e de navegação.

A linguagem de programação PHP foi utilizada para a manipulação e extração das informações, que depois de moldadas são persistidas em um banco de dados MySQL. Ambos, PHP e MySQL foram escolhidos por serem softwares livres e por estarem disponíveis no ambiente da FURB (Universidade Regional de Blumenau).

Foram utilizados recursos para geração dinâmica de páginas HTML através de scripts PHP, com rotinas construídas e moldadas de forma genérica, tornando mais rápido o seu desenvolvimento.

Além de fornecer a estrutura de programação para o desenvolvimento de software, o PHP fornece suporte para acessar uma grande faixa de bancos de dados. Quanto a ambiente de servidor, o PHP pode ser usado em Linux, FreeBSD e Windows, entre outros (SCHWENDIMAN, 2001).

Houve uma grande preocupação referente à reutilização e padronização de rotinas, evitando ao máximo o retrabalho, garantindo assim fácil manutenção e compreensão do código fonte.

Por se tratar de um sistema com interface WEB o desempenho foi um dos pontos mais consistidos durante a fase de testes e controle de qualidade.

As ferramentas utilizadas durante o processo de desenvolvimento foram o PHP Editor, PHP Edit, Macromedia Dreamweaver MX Trial Version, Top Style Lite e MySQL Front.

3 Resultados

O sistema NAP On-line possui independência com relação ao sistema operacional utilizado nas estações de trabalho bem como no servidor, reduzindo significativamente o custo de implantação.

Apresenta interface WEB amplamente customizável, utilizando estilos em cascata (CSS), permitindo personalização completa e oferecendo a cada cliente uma solução que traz a sua identidade visual.

O sistema fornece controle de acesso por usuário ou grupo de usuários, onde cada consulta é moldada conforme a permissão destinada ao usuário que esta operando.

Em suas consultas, mostra de forma clara e simplificada as informações de maior interesse no acompanhamento de projetos, conforme a necessidade de cada usuário.

Reduz significantemente o volume de papéis utilizados, pois o usuário dispõe de consultas on-line onde as informações relatam à realidade atual do projeto.

4 Referências

MENDES, R. B. et al. **Plano de Gerência do Projeto NAP on-line.** 2004. 43 f. Trabalho Final da Disciplina de Gerência de Projetos de Informática. (Bacharelado em Sistemas de Informação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

SCHWENDIMAN, Blake. **PHP4 – guia do programador**. Rio de Janeiro: Ciência Moderna, 2001. 799 p.

Ambiente do Empreendedor: Ambiente de Aprendizagem para auxiliar na disciplina de Empreendedor em Informática

Jonathan Manoel Borges (FURB)
jmborges@inf.furb.br

Oscar Dalfovo (FURB)
dalfovo@furb.br

Categoria: Informática na Educação

Linguagem de programação: PHP

Sistema operacional: Multiplataforma (WEB)

Palavras-chave: Empreendedor em Informática, Ambiente de Aprendizagem

1 Contexto

Este trabalho tem como objetivo apresentar um ambiente de aprendizagem para auxiliar na disciplina de Empreendedor em Informática, ministrada no curso de Ciências da Computação da Universidade Regional de Blumenau (FURB).

A disciplina de Empreendedor em Informática, na FURB, surgiu em 1996 e vem obtendo resultados desde então. O principal objetivo da disciplina é desenvolver o espírito empreendedor nos alunos. Como resultado, ao longo da disciplina, é desenvolvido um Plano de Negócio, documento o qual serve como base para criação de uma empresa.

O ambiente surgiu de um Trabalho de Conclusão de Curso, desenvolvido pelo acadêmico Alencar Barbieri e orientado pelo professor Oscar Dalfovo. Posteriormente, foi ampliado e implantado pelo acadêmico Jonathan Manoel Borges. Seu principal objetivo é auxiliar os alunos na construção do Plano de Negócio, permitindo que o professor acompanhe as atividades dos alunos, além de servir como um portal onde os alunos encontram informações e materiais relacionados à disciplina.

2 Desenvolvimento

No desenvolvimento do ambiente de aprendizagem foram utilizados, a linguagem de programação PHP, o banco de dados MYSQL, script HTML e a ferramenta Power Designer (editor de Diagramas Entidade Relacionamento). O ambiente está rodando no servidor Apache localizado no LCI (Laboratório de Computação e Informática) da FURB e pode ser acessado pela URL: campeche.inf.furb.br/empinf/ambiente.

O Ambiente de Aprendizagem para auxiliar na disciplina de Empreendedor em Informática acha-se dividido em três partes: o ambiente disponível ao professor, o ambiente disponível ao acadêmico e a feira simulada.

2.1 Ambiente disponível ao professor

Após entrar no ambiente, é apresentado ao professor a sua área de trabalho, onde o mesmo tem acesso às ferramentas do ambiente.

Na ferramenta *avaliações* o professor insere as avaliações, incluindo provas e trabalhos, descrevendo o conteúdo da avaliação. O professor também decide se, para a avaliação, o aluno deve entregar um arquivo pelo o ambiente. É nesta ferramenta que o professor insere as notas das avaliações para cada aluno, além de poder verificar o conteúdo do arquivo postado pelo aluno.

Na ferramenta *plano de negócio* o professor consulta o plano de negócio dos alunos, podendo visualizá-los em RTF ou HTML, além de ter o controle das informações do plano de negócios do acadêmico. O plano de negócio é dividido em módulos e em cada módulo há várias questões. O professor cadastra os módulos, as questões e exemplos de respostas para orientar os alunos na construção do plano de negócio.

Na opção chamada *ferramentas*, o professor tem o controle de todo o conteúdo disponível no ambiente. É nesta opção que o professor disponibiliza materiais, cadastra suas turmas (grupos de trabalho) e pode alterar seus dados, como email e senha de acesso ao ambiente.

2.2 Ambiente disponível ao acadêmico

Além da opção *ferramentas*, onde o acadêmico pode alterar seus dados, como email e senha, outras opções são apresentadas na área de trabalho do acadêmico.

Na ferramenta *avaliações* o acadêmico pode efetuar a entrega dos arquivos de suas avaliações, além de verificar o conteúdo e a data das mesmas.

Na ferramenta *plano de negócio* o acadêmico constrói seu plano de negócio, respondendo textualmente ou inserindo figuras, as questões previamente cadastradas pelo professor. O acadêmico pode visualizar o plano de negócio em RTF ou HTML.

2.3 Feira simulada

Na feira simulada os acadêmicos podem cadastrar as empresas desenvolvidas em seus planos de negócio juntamente com o endereço do *site* da empresa, para que o público possa ter acesso e verificar as idéias, produtos e serviços desenvolvidos por cada empresa e decidir-se pela melhor através do voto.

Para evitar que um visitante vote mais que uma vez, o mesmo deve informar seu email. Após votar, o ambiente gera uma mensagem para o email do visitante contendo um código secreto. Na mensagem possui um link chamado "*clique aqui para confirmar seu voto*", que submete este código para o ambiente, validando o voto do visitante.

3 Resultados

O grande diferencial do Ambiente do Empreendedor em relação a outros ambientes, trata do Plano de Negócio, onde os acadêmicos desenvolvem o plano a partir de questões previamente cadastradas pelo professor. Tanto o acadêmico quanto o professor podem visualizar o plano em RTF ou HTML.

O Ambiente do Empreendedor foi aprovado e está sendo utilizado na integra por professores da disciplina de Empreendedor em Informática em todo o país. Mostrou que pode auxiliar tanto o professor quanto os acadêmicos, fazendo com que o primeiro tenha um recurso a mais para facilitar e agilizar processos na disciplina.

4 Referências

DALFOVO, Oscar. AMORIN, Sammy Newton. Quem tem informação é mais competitivo. Blumenau: Acadêmica, 2000.

Ferramenta para a Transformação de Autômato Finito Não-Determinístico em Autômato Finito Determinístico

Fernando dos Santos (FURB)
Karly Schubert Vargas (FURB)
{fds, karly}@inf.furb.br

Categoria: Teoria da Computação

Linguagem de programação: Delphi

Sistema operacional: Windows

Palavras-chave: Autômatos, Determinístico, Não-determinístico

1 Contexto

Autômatos finitos não-determinísticos (AFND's) representam adequadamente linguagens regulares. Porém, a implementação dos mesmos é complexa, exigindo algoritmos com *backtracking*. Assim, é mais conveniente para a implementação, encontrar um autômato finito determinístico (AFD) equivalente. Por este motivo, uma das etapas enfrentadas pelos alunos de Linguagens Formais/Compiladores é justamente efetuar a transformação de AFND's em AFD's.

A seguir tem-se uma breve explicação sobre o processo de transformação de AFND para AFD, onde são seguidos três passos (MARTINS, 2002):

1º) determinar a primeira linha da tabela de transição do AFD (Função de Transição, δ) como sendo o conjunto unitário contendo apenas o estado inicial do AFND (Tabela de Transição, δ);

2º) verificar, para cada estado do AFD, se as transições foram determinadas, ou seja, identificar se o rótulo K, possivelmente os rótulos de vários estados do AFND representando a união dos mesmos, de cada transição já é usado como rótulo de algum estado (em alguma linha) da tabela de transição do AFD. Em caso negativo, criar uma nova linha contendo aquele rótulo K e determinar, para cada símbolo de entrada do alfabeto Σ, o conjunto resultante da união dos vários estados que compõem K. Repetir o segundo passo até que todas as transições tenham sido determinadas;

3º) determinar o estado inicial do AFD como sendo a primeira linha da tabela de transições. Determinar o(s) estado(s) final(is) do AFD como sendo todos os estados (todas as linha) rotuladas com conjuntos que contenham pelo menos um estado final F do AFND.

Na figura 1 é mostrada a tabela informada (AFD) e gerada (AFD) no processo de transformação.

AFND

	a	b
> e0	e0, e1	e0, e2
e1	e3	-
e2	-	e3
* e3	e3	e3

AFD

	a	B
> e0	e0e1	e0e2
e0e1	e0e1e3	e0e2
* e0e1e3	e0e1e3	e0e2e3
* e0e2e3	e0e1e3	e0e2e3
e0e2	e0e1	e0e2e3

Figura 1: AFND em forma de tabela, e o AFD equivalente.

Como pode-se notar, o processo que faz as transformações no autômato possui um significativo número de operações, tornando o processo não trivial.

Para facilitar este processo, foi desenvolvido como um trabalho da disciplina de Linguagens Formais do segundo semestre de 2002, uma ferramenta para automatizar o processo de transformação de um AFND em um AFD.

2 Desenvolvimento

O software foi desenvolvido utilizando-se o ambiente de desenvolvimento Borland Delphi 5 (BORLAND SOFTWARE CORPORATION, 1999).

Para especificação do processo de transformação de AFND para AFD foi utilizada a técnica de orientação a objetos. A definição da estruturação de dados utilizada é mostrada na figura 2.

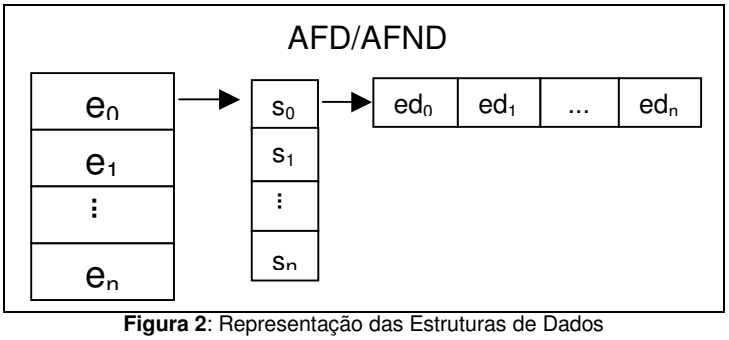


Figura 2: Representação das Estruturas de Dados

Duas classes principais compõem o modelo de classes. Uma delas representa um AFND e a outra um AFD. Cada uma destas classes possui uma lista de estados do autômato (e_0, e_1, ..., e_n). Cada um destes estados possui uma lista de símbolos (s_0, s_1, ..., s_n) que o estado pode reconhecer. Por fim, cada símbolo possui uma lista de estados de destino (ed_0, ed_1, ..., ed_n), armazenando a(s) transição(ões) que pode(m) ocorrer caso o símbolo seja reconhecido. Estados e símbolos também foram implementados através de classes, sendo que cada classe implementa os métodos necessários para a execução do algoritmo de transformação.

A interface do software foi desenvolvida com o objetivo de oferecer grande semelhança com a representação utilizada para descrição de autômatos na notação de tabelas, tanto que a entrada (confecção do autômato) e saída dos dados é feita em tabelas, onde são inseridos/exibidos os estados e as transições do autômato.

3 Resultados

A ferramenta transforma com sucesso um AFND em um AFD, podendo com isso ser utilizada sem nenhum tipo de restrição. Além disso, o modelo de classes desenvolvido é reutilizável em qualquer outro software que necessite transformar AFND's em AFD's.

4 Referências

Borland Software Corporation. **Documentação do Borland Delphi.** V. 5.0. Scotts Valley: Borland, 1999. CD-ROM.

MARTINS, Joyce. **Linguagens formais e compiladores**. Apostila da disciplina de Linguagens Formais do curso de Ciências da Computação da Universidade Regional de Blumenau (FURB). Blumenau, 2002.

Índice de Autores