Desenvolvimento de uma arquitetura de comunicação de um agente pedagógico inteligente em Java

Alisson Oldoni[1], Anita Maria da Rocha Fernandes [1], Janice Inês Deters [2]

[1]Laboratório de Inteligência Aplicada – Universidade do Vale do Itajaí (UNIVALI)
Caixa Postal 360 – 88.302-202 – Itajaí – SC – Brasil
{aoldoni,anita.fernandes}@univali.br
[2]Instituto de Engenharia Biomédia – Universidade Federal de Santa Catarina (UFSC)
88.040-900 – Florianópolis – SC – Brasil
jan@ieb.ufsc.br

Resumo

O projeto "Sistemas Virtuais de Ensino Baseados na Internet para Suporte à Educação e Treinamento na Área de Saúde", coordenado pelo Instituto de Engenharia Biomédica da Universidade Federal de Santa Catarina, engloba o desenvolvimento de um Sistema Tutor Inteligente de neurofisiologia que utiliza a tecnologia de agentes pedagógicos inteligentes. O estudo realizado para a escolha da ferramenta de sistema especialista baseado em regras a ser utilizada na inteligência do agente não é uma tarefa trivial e constituiu uma das partes mais fundamentais da modelagem do projeto. Este artigo apresenta este estudo das ferramentas, a ferramenta escolhida e a arquitetura de comunicação desenvolvida.
Palavras-chave: Agentes Pedagógicos, Sistemas Tutores Inteligentes, JCLIPS.

Abstract

The "Web Based Virtual Education Systems to Support Education and Training in Heath Area" project, coordinated by the Instituto de Engenharia Biomédica, an institute inside the Universidade Federal de Santa Catarina, encloses the development of an Intelligent Tutoring System that focuses on the neurophysiology study and uses the intelligent pedagogical agents technology. The study made for the choice of the rule-based expert system that will be used on the agent's intelligence is not a trivial task and constitutes one of the most fundamentals parts of the project modeling. This article shows this study, the chosen tool and the developed communication architecture.
Key-words: Pedagogical Agents, Intelligent Tutoring Systems, JCLIPS.

1. Introdução

A sociedade moderna requer um volume crescente de conhecimento, não somente para o trabalho profissional, mas também para as necessidades do dia a dia. Para manter a qualificação da força de trabalho em um ambiente competitivo e em constante mudança, a educação continuada e à distância se torna essencial [2]. O conceito de ambientes de e-learning pretende fornecer às pessoas um treinamento de alto nível, porém, evitando tarefas intelectuais tediosas e repetitivas, à respeito de um grande volume de informações finamente estruturado e altamente organizado [3].

Atualmente a maiorias dos modelos de EAD (Educação a Distância) disponíveis se respaldam fortemente na metáfora das salas de aula presenciais e nem sempre capturam a especificidade e o potencial educacional da Internet. Os sistemas de EAD devem suprir ao máximo as dificuldades causadas pela separação física do aluno e do professor e dos demais colegas, requerendo ferramentas eficientes, de adaptabilidade, interação, comunicação e motivação. Uma das ferramentas que se aplica muito bem neste sentido são os Sistemas Tutores Inteligentes (STI), que utilizam agentes inteligentes (tutores).

Um agente é tudo o que pode ser considerado capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por intermédio de atuadores, como visto na Figura 1 [16]. Assim como um agente humano, que tem olhos, ouvidos entre outros órgãos sensoriais para captar o que acontece ao seu redor, um agente de software, por exemplo, capturaria as seqüências de teclas digitadas, conteúdo de arquivos e pacote de redes como entradas sensoriais e, no lugar de braços ou bocas (os atuadores humanos),

os atuadores deste agente de software seriam mensagens exibidas na tela, arquivos gravados e arquivos enviados pela rede.

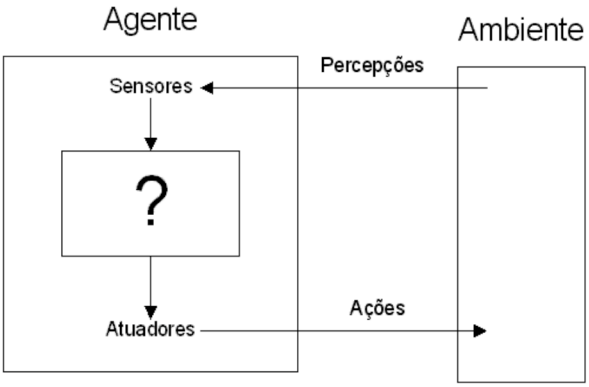

Figura 1 - Agentes interagem com ambientes por meio de sensores e atuadores.

Os agentes pedagógicos também ajudam trabalhando na interface, e são vistos como uma moderna abordagem para interface dos STI, facilitando a interação do usuário com a sistema tutor. A interação com o computador hoje em dia se dá por meios não naturais e uma interação natural seria por meios de linguagem, expressões faciais e linguagem corporal [1]. Uma linguagem simplesmente textual pode não detectar sarcasmos, referências literárias e figurativas. Sendo assim, para melhorar a interação humano-computador, os agentes podem ter personagens gráficos que usam de recursos como animação, sons e texto, originando os agentes pedagógicos "animados" ou "de interface".

A tecnologia de agentes tem sido incorporada na modelagem do STI e nos ambientes educacionais na Internet [13]. O agente, no STI, teria uma intenção além de simplesmente guiar o aluno no melhor uso do programa, mas também orientar o aluno pedagogicamente sobre o domínio da aplicação. Tem-se então a missão de fazer o agente saber o que está ensinando, definir estratégias de como melhor ensinar, e perceber se o aluno aprendeu ou não.

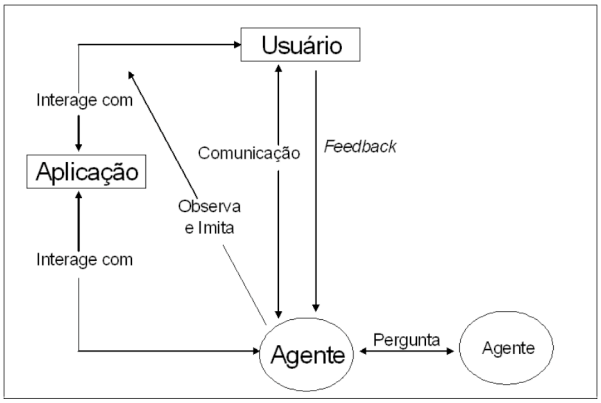

Figura 2 - Como os agentes de interface funcionam.

Um agente pedagógico trabalhando na interface observa e monitora as ações tomadas pelo usuário na interface (como se fosse um agente humano "olhando sobre o ombro do usuário"), ensinaria novos "atalhos" e sugeriria novas formas de realizar tarefas [12]. Além disso, o agente se comportaria como um assistente pessoal autônomo, que coopera com o usuário na realização de tarefas do software. Seu comportamento pode ser visto na Figura 2.

2. Portal "Saúde+Educação"

O portal Saúde+Educação, desenvolvido pelo projeto "Sistemas Virtuais de Ensino Baseados na Internet para Suporte à Educação e Treinamento na Área de Saúde" do IEB (Instituto de Engenharia Biomédica) fomentado pela FINEP (Financiadora de Estudos e Projetos), engloba a construção de um STI na área Biomédica que disponibilizará tutoriais on-line de Neurofisiologia.

Este trabalho tem como meta criar uma arquitetura de comunicação e interação dentro deste tutorial de Neurofisiologia. O agente do projeto tem como objetivo maximizar o desempenho do aluno nas auto-avaliações do STI. Para chegar-se a isso, a construção do agente pedagógico deve levar em conta algumas informações que serão cuidadosamente obtidas no sistema. Fundamentalmente, devem-se registrar todos os passos do aluno dentro do STI. É isso que fará com que o agente saiba quais conteúdos o aluno já visualizou, para então orientá-lo no caso de um mau desempenho na auto-avaliação.

O STI, então, contará com a intervenção de um agente pedagógico, chamado Bernardo, em todas as interações do usuário com o sistema, disponibilizando os conteúdos divididos em tópicos em um tutorial. Cada tópico, além de ter o conteúdo, será dividido nos seguintes sub-itens:

- Mão na Massa: ao selecionar este item o sistema exibirá um desafio prático para o usuário resolver;
- Correlação Clínica: neste item o usuário encontrará informações relacionadas à prática clínica;
- História: neste item será exibida a contextualização histórica do ítem selecionado.

Além disso o sistema tutor disponibilizará uma auto-Avaliação, que permitirá ao usuário responder questões objetivas diversas apresentadas aleatoriamente ao usuário e resultará em um nível de desempenho. Em caso de desempenho baixo do aluno na auto-avaliação, o agente pedagógico apresentará alternativas ao usuário as quais ele deverá acessar para reforçar o seu estudo. Estas alternativas apresentadas pelo agente caso o aluno tenha um desempenho baixo levará em conta o que o aluno já leu, e quais dos itens relacionados ele já observou, sendo que o uso dos sub-itens relacionados (Mão na Massa, Correlação Clínica e História) também é levado em conta. O agente pedagógico animado de interface, então, acessará o sistema especialista para descobrir qual deverá ser a sua atuação e estratégia sempre que o aluno realizar a auto-avaliação.

3. Concepção visual

Para a modelagem e implementação da personagem que compõe o agente proposto "Bernardo", foram realizadas algumas entrevistas e pesquisas, resultando nas seguintes características:

- Ser o mais próximo da realidade, do sexo feminino ou masculino, de corpo inteiro e proporcional ao tamanho da interface;
- Possuir expressões faciais dos mais variados sentimentos como alegria, admiração, tristeza, entre outros; expressando emoções de acordo com o momento, não interrompendo o usuário em seu aprendizado com intervenção escrita ou falada;
- Interagir com o usuário, mudando de postura e face sutilmente, entrando em ação e mudando de posturas de acordo com a situação; e
- Falar e escrever, usando balões de texto e/ou recursos de som. Gesticular, mover-se no ambiente. Ter reações diversas para não tornar a interação com usuário, monótona. Por exemplo, usar a comunicação escrita num primeiro contato, na segunda abordagem dar um sorriso, dicas, entre outros.

Além das diretrizes apresentadas anteriormente, o personagem "Bernardo" foi idealizado tendo como base os seguintes objetivos visuais: passar a imagem de um pesquisador; ter características físicas semelhantes às características dos brasileiros; ser simpático e prender a atenção do usuário; usar roupas alegres e jovens. Foram utilizados traços arredondados, semelhantes aos de "cartoon", dando ao agente uma fisionomia jovial.

O contorno dos membros reafirma essa proposta, juntamente com o uso de cores vivas em suas vestimentas, que são trocadas de acordo com a situação (Figura 3).

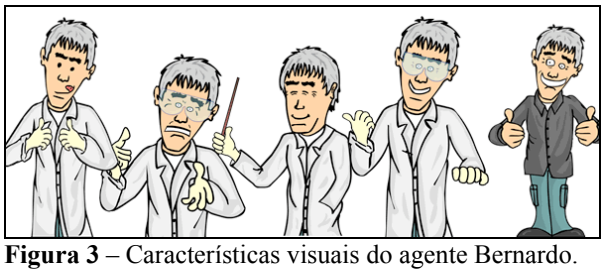

Figura 3 – Características visuais do agente Bernardo.

4. Concepção funcional: escolha da ferramenta de sistema especialista

Quanto às tecnologias e ferramentas escolhidas pelo projeto para desenvolvimento do portal foram prezadas ferramentas de uso livre, portanto o portal utiliza a linguagem Java [17], com as técnicas de JSP (Java Server Pages) [18] e JSF (Java Server Faces) [19].

Das técnicas existentes para raciocínio do agente, o projeto escolheu o uso dos Sistemas Especialistas (SE), pois a idéia de orientação vinda do agente vai se basear em algumas várias observações feitas, tais como "o aluno já visitou a correlação clínica?". Estas regras são do tipo "se <condição envolvendo fatos> então", o que se encaixa no perfil da técnica de regras do sistema especialista.

Para validação e verificação destas regras no sistema, é necessário um motor de verificação de regras de sistema especialista. Tais ferramentas são softwares que combinam fatos e regras de modos diversos, embutem uma máquina de inferência específica, assim como uma representação própria de conhecimento, e permite a construção de sistemas especialistas que utilizem a sua representação de conhecimento e máquina de inferência [11].

O projeto levou em conta o estudo de ferramentas de sistema especialista disponíveis, tendo como intuito utilizar a melhor ferramenta possível que se adequasse aos requisitos do projeto. O sistema especialista para este projeto deve seguir os seguintes requisitos: ser compatível com a linguagem Java; ser possível seu uso pela Internet; e ser de uso livre. Para ser de uso livre um software deve ter as licenças comuns de uso livre, por exemplo, GNU (GNU is Not Unix) [4] ou GPL (General Public License) [5].

Sendo assim, foram estudadas as seguintes ferramentas:

- CLIPS (C Language Integrated Production System): uma ferramenta que tem sido desenvolvido pela NASA (National Aeronautics and Space Administration) desde 1985 [15];
- WebCLIPS: CGI (Common Gateway Interface) para o CLIPS, desenvolvido por Michael Giordano [6]; e
- JCLIPS (CLIPS for Java): Ponte em software entre o Java e o CLIPS, desenvolvido por Maarten Menken [10].

Inicialmente o projeto levava em conta a utilização do JESS (Java Expert System Shell) [7], desenvolvido pela Sandia National Laboratories, nos Estados Unidos, porém seu uso ficou automaticamente fora de cogitação pela sua licença não ser livre. A seguir tem-se as informações obtidas das ferramentas estudadas.

4.1. CLIPS

O CLIPS é uma ferramenta sem custo de distribuição que permite o total desenvolvimento de sistemas especialistas baseados em regras. Ele pode ser embutido em código procedural, chamado como uma sub-rotina, e integrado com linguagens como C, Java e FORTRAN (Formula Translator). Ele é facilmente extensível através do uso de protocolos já fortemente definidos.

O CLIPS é desenvolvido na linguagem C e tendo como preocupação principal a portabilidade entre vários sistemas operacionais, o que se torna uma das suas grandes vantagens. Ele compila em vários sistemas operacionais e o fato de ele ser desenvolvido em C não impede um possível encapsulamento em Java, pois ele pode ser compilado como uma biblioteca.

O CLIPS funciona permitindo a entradas de regras e fatos através de código armazenado em arquivos texto com a extensão ".clp" ou com código digitado no próprio console que é fornecido juntamente com a biblioteca.

A documentação para orientação de uso do CLIPS é abrangente e orienta a utilização sob vários aspectos, desde a construção de sistemas especialistas até a expansão do CLIPS e seu encapsulamento em outras ferramentas de desenvolvimento.

4.2. WEBCLIPS

O WebCLIPS cria uma interface CGI (Common Gateway Interface) para o CLIPS, fornecendo uma comunicação com a Web. O WebCLIPS é implementado em C e há versões para plataformas UNIX e Windows. Sua distribuição é gratuita e ele se encontra sob licença GNU.

Os scripts rodados em WebCLIPS devem atender as mesmas especificações do CLIPS, tendo como único elemento incomum o fato dos resultados obtidos no WebCLIPS terem algumas "tags" HTML incluídas para a amostragem apropriada no navegador Web. As páginas e formulários são gerados dinamicamente em HTML pelo WebCLIPS, sendo que, para isso, ele obtém as informações vindas das variáveis da requisição enviada ao servidor.

Seu funcionamento se dá a partir do momento que uma requisição é feita ao CGI. A partir dela o WebCLIPS identifica qual o arquivo de regras que deve ser usado, tenta montar os fatos vindos da requisição, e verifica se este mesmo cliente já não tinha fatos armazenados no servidor em um "cookie". Os fatos então são carregados e o CLIPS é iniciado e o processamento das regras e dos fatos é feita. Depois da máquina de inferência terminar o seu trabalho, o sistema então atualiza sua memória de trabalho com os resultados, monta e envia a saída com os devidos comandos HTML para o cliente.

4.3.JCLIPS

JCLIPS é um software livre, sob licença GNU, que permite programadores Java a usarem a ferramenta CLIPS em combinação com Java através da incorporação do motor CLIPS em uma classe Java. Sendo assim, torna-se possível controlar o motor CLIPS através do Java, podendo-se então carregar arquivos CLIPS, executar o motor, executar comandos arbitrários, e receber informações de volta do CLIPS no formato texto. O JCLIPS pode ser embutido em uma aplicação Java que seja entregues através de uma página Web via Applet.

As aplicações são escritas normalmente em Java e utiliza-se dois arquivos JCLIPS (um ".dll" em Windows ou ".so" em Linux e um ".jar") para estabelecer sua ponte entre Java e CLIPS. O arquivo ".dll" contém toda a distribuição CLIPS compilada pra Windows, e não é necessária a obtenção do CLIPS separadamente. O arquivo ".jar" completa a parte Java desta ponte. O JCLIPS é conhecido por rodar em ambas plataformas Windows e Linux.

A integração do JCLIPS com qualquer aplicação Java é simples, bastando adicionar os arquivos contendo a classe Java e a biblioteca CLIPS em um mesmo diretório do classpath Java, ou adicionar o diretório da aplicação e do arquivo na variável de ambiente classpath, utilizada pelo Java. A classe a ser instanciada chama-se "JClips". Esta classe contém os métodos para configuração e controle do ambiente CLIPS.

Só é possível um ambiente CLIPS rodando simultaneamente e, portanto, só uma ponte entre CLIPS e Java. Para enfatizar isso, a única instancia dessa classe é automaticamente criada e uma referencia pode ser obtida através do método "getInstance".

5. Ferramenta escolhida

Extraindo informações destas quatro ferramentas, tem-se uma tabela comparativa (Tabela 1) sobre a compatibilidade das ferramentas com a linguagem Java e a possibilidade de seu uso pela Internet.

Ferramenta	**Compatibilidade com Java**	**Possibilidade de uso na Internet**
CLIPS	Sim, mas requer esforço extra de programação	Sim
WebCLIPS	Não	Sim
JESS	Sim	Sim
JCLIPS	Sim	Sim

Tabela 1 – Compatibilidade com Java e uso pela Internet

Na Tabela 2 tem-se o comparativo entre a qualidade da documentação das ferramentas analisadas e da existência ou não da necessidade da compra de licenças ou outros recursos para a utilização do software.

Ferramenta	**Documentação**	**Licença**
CLIPS	Boa	Livre
WebCLIPS	Média	Livre
JESS	Boa	Paga
JCLIPS	Regular	Livre

Tabela 2 – Qualidade da documentação e tipo de licença

A ferramenta escolhida para o desenvolvimento do agente foi a JCLIPS. O JCLIPS levou em conta os requisitos básicos de fácil acoplamento com Java e de software livre, seguindo a filosofia do projeto de desenvolvimento com software livre. Este é um dos pontos fortes do JCLIPS em relação a concorrentes diretos em Java, como o JESS, que é pago.

Mesmo que a única ferramenta analisada com uso na Internet nativo seja a WebCLIPS (as outras necessitam a utilização de uma linguagem "server-side"), a intenção é propriamente o uso de uma linguagem "server-side", o JSP, para uma integração mais fácil com o banco de dados.

A deficiência na documentação específica para JCLIPS não se sobressai aos outros itens, isso porque boa parte da documentação do CLIPS (informações da linguagem de especificação das regras, informações sobre funcionamento, etc.) se aplica ao uso do JCLIPS.

6. Arquitetura desenvolvida

A arquitetura de comunicação se dará com uma animação em SWF (Small Web Format) embutida num frame HTML na parte cliente, gerado pelo JSP. Essa animação então se comunicará por requisições para o servidor para processamento de regras no JCLIPS e busca de feições, diálogos e sons, como visto na Figura 4. A escolha do formato SWF se dá pela sua facilidade de lidar com gráficos, sons, texto e interações em interfaces Web o que é imprescindível para a personagem gráfica do agente e suas feições. Arquivos nesse formato são desenvolvidos com a ferramenta Macromedia Flash [9].

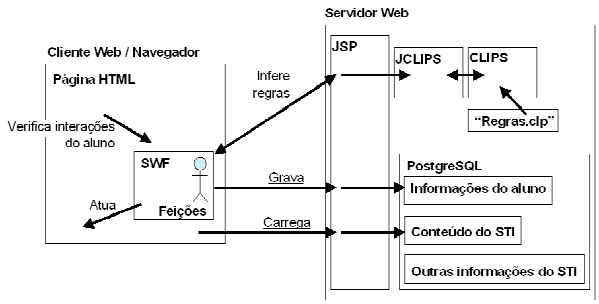

Figura 4 - A arquitetura do agente

Na Figura 4 tem-se a seta "verifica interações do aluno" e a seta "atua", que indicam a parte da arquitetura de comunicação que interage diretamente com o navegador Web do aluno. Esta interação é possível graças ao formato SWF, que também possui uma forte linguagem de programação, chamada ActionScript [8], e que permite a personagem gráfica do agente observar o ambiente HTML (HyperText Markup Language) (sensores) através da comunicação direta com o Javascript [14] e conversar como usuário através de sons, desenhos, animações e textos (atuadores).

A seta "grava", também na Figura 4, faz relação a gravação e obtenção das informações do aluno necessárias para a posterior inferências das regras (no momento em que o aluno finalizar a auto-avaliação). Já a seta "carrega" indica que o agente identificou (capturou pelo sensor) que o aluno clicou em algum item e comunicou ao navegador Web para que carregue o item correspondente. Por último, a seta nomeada "infere regras" indica o momento em que o aluno finalizou a realização da auto-avaliação. Ali o agente inicia a inferência das regras contidas em um arquivo de regras no servidor ("Regras.clp") sobre as informações do aluno existentes no banco de dados (seta "grava") e obtém a resposta do que fazer para orientar o aluno com estratégias de aprendizado.

Somadas estas funcionalidades, pode-se observar no agente os seguintes sensores:

- Integração Javascript/Actionscript: verifica as interações do aluno com o navegador quanto a requisição de conteúdo;
- Integração Javascript/Actionscript: verifica as interações do aluno com o navegador quanto a realização de auto-avaliação; e
- Identificador das estratégias vindas do JCLIPS: recebe e trata as respostas do JCLIPS.

Também pode-se observar no agente os seguintes atuadores:

- Integração Javascript/Actionscript: indica ao navegador qual conteúdo carregar quando requisitado pelo aluno;
- Exibidor de animações e texto no SWF: faz a atuação entre a personagem gráfica do agente e o aluno, mostrando mensagens, caixas de diálogo e suas respectivas expressões e gestos;
- Gravador de informações do aluno: grava as informações obtidas do aluno no banco de dados no servidor através de uma requisição HTTP (Hypertext Transfer Protocol) ao JSP no servidor; e
- Requisitor de estratégias do JCLIPS: requisita estratégias de ensino ao JCLIPS através de uma requisição HTTP ao JSP no servidor, que por sua vez o repassa ao JCLIPS.

7. Conclusões

A contribuição desta pesquisa é o desenvolvimento de um ambiente de ensino/aprendizagem usando ferramentas "open source" e utilizando realidade virtual para tornar agradável a comunicação entre os "estudantes" e o professor.

Para testar o sistema, atualmente está sendo desenvolvido o conteúdo instrucional sobre homeostase. A equipe de criação de conteúdo é

composta por pessoas das áreas de ciência da computação, design e saúde.

Quando o ambiente estiver com o conteúdo de homeostase totalmente concluído, serão feitos testes de com dois grupos distintos de alunos: um grupo referente aos acadêmicos do curso de medicina da Universidade Federal de Santa Catarina que possuem a disciplina de Fisiologia em seus currículos e um outro grupo com os alunos do Curso de Pós Graduação em Engenharia Biomédica que possuem a disciplina de Fisiologia como obrigatória.

O teste buscará avaliar o sistema quanto ao aspecto ergonômico, bem como em relação à aceitação de uma ferramenta de ensino pelos alunos. Além disto, outro teste será feito com um grupo de alunos para analisar a melhoria do processo de ensino/aprendizagem com o uso do ambiente.

Referências

[4].BIGUS, Joseph P., BIGUS, J. *Constructing intelligent agents using JAVA*. 2. ed. New York: John Wiley & Sons, 2001.

[1].CRISTEA, P. D., TUDUCE, R. Intelligent eLearning environments. In: *Proceedings* of *3-rd european conference on e-commerce / e-activities / e-working / e-business, on-line services, virtual institutes and their influences on ahe economic and social environment*, Bucharest, 2002.

[2].DOWLING, C. Intelligent pedagogical agents in online learning environments. In: *Proceedings of 16rd ICEUT: international conference on educational uses of information and communication tehcnologies*, Pequim, 2000.

[11].FREE SOFTWARE FOUNDATION. The GNU operating system. *Home Page.* 2006. http://www.gnu.org/ (8/jun./2006).

[12].FREE SOFTWARE FOUNDATION. Welcome to GPLv3. *Home Page*. 2006. http://gplv3.fsf.org/ (8/jun./2006).

[14].GIORDANO, M. CLIPS interface and sample projects. *Home Page.* 2003. http://clipsinterface.sourceforge.net/WebCLIPS/wchome.htm (9/abr./2001).

[16].JESS. JESS: the expert system shell for the Java platform. *Home Page*. 2006. http://www.jessrules.com/ (7/jun./2006).

[18].MACROMEDIA. ActionScript language reference. *Home Page*. 2004. http://download.macromedia.com/pub/documentation/en/flash/mx2004/fl_actionscript_reference.pdf (19/jun./2006).

[17].MACROMEDIA. Using Flash. *Home Page.* 2004. http://download.macromedia.com/pub/documentation/en/flash/mx2004/fl_usingas_in_flash.pdf (19/jun./2006).

[15].MENKEN, M. JCLIPS: Clips for Java. *Home Page*. 2005. http://www.cs.vu.nl/~mrmenken/jclips/ (7/jun./2006).

[10].MOURA, M. F.; CRUZ, S. A. B. Estudo de expert system shells para o ambiente de diagnose remota. In: *Relatório Técnico*. Campinas: Embrapa, 2001.

[6].NWANA, H. Software Agents: An Overview. *Knowledge Engineering Review* 11, 3, 1996, 1-40.

[5].PEREIRA, A. S., GEYER, C. F. R. Um agente para seleção de estratégias de ensino em ambientes educacionais na internet. *RITA (Revista de Informática Teórica e Aplicada)* 6, 2, 1999, 89-105.

[19].REFSNES DATA. JavaScript Reference. *Home Page*. 2006. http://www.w3schools.com/jsref/default.asp (13/jun./2006).

[13].RILEY, G. CLIPS: a tool for building expert systems. *Home Page.* 2005. http://www.ghg.net/clips/CLIPS.html (7/jun./2006).

[3].RUSSELL, S., NORVIG, P. *Inteligência Artificial*. 2. ed. Rio de Janeiro: Campus, 2004.

[7].SUN. Java technology. *Home Page.* 2006. http://java.sun.com/ (7/jun./2006).

[8].SUN. Java Server Pages technology. *Home Page*. 2006. http://java.sun.com/products/jsp/index.jsp (7/jun./2006).

[9].SUN. Java Platform, Enterprise Edition (Java EE): JavaServer Faces Technology. *Home Page*. 2006. http://java.sun.com/javaee/javaserverfaces/ (7/jun./2006).

Curriculum Vitae

Anita Maria da Rocha Fernandes é Doutora em Engenharia de Produção pela Universidade Federal de Santa Catarina, líder do Grupo de Inteligência Aplicada (GIA) e professora titular nas disciplinas de Inteligência Artificial I e II do curso de Ciência da Computação da Universidade do Vale do Itajaí (Univali).

Alisson Oldoni é Formando em Ciência da Computação pela Universidade do Vale do Itajaí (Univali) e pesquisador do Grupo de Inteligência Aplicada (GIA) do Curso de Ciência da Computação da Univali.