Construindo um Sistema Tutor Inteligente a partir de um Tutorial

João Carlos de C. e S. Ribeiro[1], Lucia Maria Martins Giraffa[1], Úrsula A. L. Fernandes Ribeiro[2]

[1]Faculdade de Informática (FACIN) – Programa de Pós-Graduação em Ciência da Computação (PPGCC) Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Av. Ipiranga, 6681 – CEP 90619-900 – Porto Alegre – RS – Brasil.

[2]Faculdade de Administração Contabilidade e Informática (FACI)
PUCRS – Campus Uruguaiana
BR 472 Km 7 Cx. Postal 249 – CEP 97500-970 – Uruguaiana – RS – Brasil.

{jribeiro, giraffa}@inf.pucrs.br, ursula@pucrs.campus2.br

Abstract*. This paper describes our proposal to wrap up the educational software named "Matematiquinha". The wrapping process applied on Matematiquinha allows us to identify the set of efforts and problems related to achieve such goal. Our research project will test how we can reuse some features and results previously achieved with Matematiquinha environment. Some definitions and concepts related to "wrapping" process are also presented.*

Resumo. *Este trabalho descreve o projeto de agentificação do software educacional denominado Matematiquinha, desenvolvido para suporte ao processo de aprendizagem de conteúdos da matemática elementar. O trabalho utiliza os conceitos de agentificação oriundos da Engenharia de Software. Este projeto nos auxiliou a identificar esforços, restrições e problemas inerentes ao processo de agentificação de um ambiente educacional. Alguns conceitos sobre agentificação são apresentados de maneira a melhor contextualizar o trabalho.*

1. Introdução

Segundo [Jennings 2000], a abordagem multiagente para o projeto e desenvolvimento de software se constitui numa nova proposta para modelar sistemas complexos. O objetivo desta abordagem é construir sistemas compostos de múltiplas entidades resolvedoras de problemas (agentes) que interagem entre si, aumentando o desempenho geral do programa. O escopo de cada entidade é limitado, diminuindo a complexidade e permitindo unidades de processamento menores e mais confiáveis. Um agente é um sistema computacional encapsulado situado em um determinado ambiente, capaz de agir de forma flexível e autônoma. Segundo [Bordini, Vieira and Moreira 2001], agentes são entidades de software autônomas que interagem através de um ambiente compartilhado por uma sociedade que possui outros agentes e processos. O comportamento dos agentes é definido através por um repertório pré-definido de ações. O projeto de um agente e o ambiente onde ele está inserido determinam as propriedades que o agente terá [Giraffa 2002].

A agentificação de sistemas existentes é uma tendência que se faz presente nas corporações modernas, especialmente naquelas onde já existem sistemas consolidados. O termo agentificação é uma tradução livre do original "*wrapping*". A agentificação é um processo associado ao "empacotamento" de um sistema já existente, afim de que ele se comporte como um sistema multiagente (SMA). Isto é, ele exiba propriedades e características que permitam inseri-lo dentro de um contexto SMA. Parte-se do pressuposto de que não faz sentido projetar ou construir um agente, se ele não coexistir com outros agentes. Ou seja, ele pertença a uma sociedade de agentes onde sua interação proverá a solução de um dado problema.

As referências relativas ao processo de agentificação, encontradas na literatura, são associadas, na sua maioria, a aplicações industriais e comerciais. Grande parte da análise e dos conceitos utilizados neste projeto baseiam-se nos trabalhos de Landauer e Bellman [Bellman 1990] [Landauer 1990b] [Landauer 1190a] [Landauer and Bellman 1996c] [Landauer and Bellman 1992] [Landauer and Bellman 1996a] [Landauer and Bellman 1997a] [Landauer and Bellman 1998b] [Landauer, Bellman and Gilliam 1993].

Este artigo está organizado em 6 seções. A seção 2 apresenta algumas considerações sobre o processo de agentificação. A seção 3 descreve o ambiente original do Matematiquinha. A seção 4 apresenta uma arquitetura multiagente para o Matematiquinha levando em consideração o processo de agentificação. A seção 5 apresenta as considerações finais. A seção 6 apresenta as referências bibliográficas utilizadas na elaboração deste texto.

2. Agentificação e suas Implicações

De uma maneira simplificada, agentificar um programa consiste em colocar no sistema uma camada capaz de habilitá-lo a interagir com outros agentes e processos [Wooldridge and Jennings 1997]. Tornando o sistema, no nível funcional e operacional, um SMA.

A agentificação geralmente ocorre por meio da agregação de uma camada (ou várias camadas) à arquitetura original. A idéia básica é habilitar componentes dos sistemas pré-existentes a comunicarem-se e cooperarem com outros agentes e processos. Desta Forma, a funcionalidade do software pré-existente pode ser estendida para trabalhar com outros componentes do outro software recentemente desenvolvido [Wooldridge and Jennings 1998]. Considerando-se que, este novo programa seja modelado e projetado dentro do contexto da abordagem de SMA.

Sob o ponto de vista externo, todos os elementos agora "encapsulados" passam a ser vistos como agentes. A agentificação desempenha uma função de interpretação em dois sentidos:

- trata as solicitações externas de outros agentes mapeando-as dentro das chamadas, no código legado;
- trata as solicitações do código legado e as mapeia dentro de um conjunto apropriado de comandos que permitem a comunicação dos agentes.

Esta possibilidade de se agentificar sistemas legados permite-nos reaproveitar sistemas já existentes. Segundo [Genesereth and Ketchpel 1994], podemos propor três abordagens diferentes para a agentificação de sistemas pré-existentes:

- *Transducer*: Consiste em implementar um programa tradutor que atua como um mediador entre um programa existente e outros agentes. O tradutor aceita mensagens dos outros agentes, traduz as mensagens para o protocolo de comunicação do programa nativo e, as passa para outro programa ou ambiente. O tradutor, então, aceita as respostas do programa, traduz para uma linguagem ACL (*Agents Communication Language*)[1] e envia as mensagens resultantes para outros agentes. Esta abordagem tem a vantagem de não requerer conhecimento do outro programa que não seja seu comportamento de comunicação. Ela é, conseqüentemente, especialmente utilizada em situações nas quais o código do programa é indisponível ou delicado para modificações.
- *Wrapper*: Consiste na implementação de uma camada adicional para o programa existente. Provendo-o com a capacidade de se comunicar através de uma linguagem ACL. Significa injetar código dentro de um programa para permitir que ele se comunique em ACL.
- *Rewrite*: Consiste em reescrever o programa original aproveitando a modelagem dos dados e as funcionalidades definidas para este. Pode-se questionar tão drástica opção. Porém, a agentificação neste caso aproveitaria todo o conhecimento modelado e adaptaria o código para a nova abordagem.

Landauer & Bellman [Landauer and Bellman 1996c] defendem uma nova abordagem para o desenvolvimento, integração e administração de sistemas heterogêneos baseada em duas classes de entidades de software: *Wrapping Knowledge Bases* (WKBs) e *Problem Managers* (PMs). Os autores partem da idéia de agentificar um sistema, já existente, através da reutilização quase total do código, mantendo suas funcionalidades e características. Logo, a agentificação abrange muito mais que simplesmente definir como utilizar um recurso. Ela provê informações para auxiliar a decidir quando o sistema está apropriado, e, se ele pode ser utilizado na solução de determinada classe de problemas.

Na prática, a agentificação cria um conjunto de interfaces personalizadas que permitem utilizar recursos em um sistema heterogêneo. Agentificar não é simplesmente agregar uma interface para um recurso. Ela é uma interface para a "utilização" de um determinado recurso. O enfoque está na forma que o recurso é utilizado. Ou seja, o estudo das maneiras que os recursos podem ser utilizados. Haverão agentificações separadas para diferentes usos comuns de ferramentas complexas. Similarmente, combinações de recursos que freqüentemente trabalham juntos podem ter uma única agentificação. Em adição, várias agentificações separadas podem representar diferentes caminhos para utilizar um único recurso.

A abordagem de agentificação pode ser aplicada em qualquer sistema que possua requisitos de heterogeneidade. Como por exemplo, para sistemas de diferentes linguagens, para diferentes processos dentro do mesmo sistema, diferentes programas

[1] Considerar as recomendações da FIPA (*Foundation for Intelligent Physical Agents*) onde a linguagem de comunicação padrão é observada. Maiores detalhes em http://www.fipa.org/

computacionais, ou combinações de outros recursos de software para suportar análise e modelagem de sistemas complexos.

Agentificações são mecanismos de baixa complexidade, porém, poderosos. Exemplos de como agentificações podem mudar a concepção de verificação e validação de grandes sistemas podem ser obtidos em [Bellman 1990], [Landauer 1990b] e [Landauer and Bellman 1996a]. Para obter-se exemplos de arquitetura de simulação de sistemas de larga escala, sugere-se [Landauer and Bellman 1992] e [Landauer and Bellman 1997a]. Outros modelos baseados em Engenharia de Software, consultar [Landauer and Bellman 1998b]. E, para exemplos de engenharia de sistemas com questões envolvendo sistemas computacionais complexos, consultar os trabalhos de [Landauer and Bellman 1996c], [Landauer and Bellman 1996a] e [Landauer, Bellman and Gillian 1993].

3. O Ambiente do Matematiquinha

O software denominado Matematiquinha, desenvolvido por [Ribeiro and Ribeiro, 2002], tem a finalidade de auxiliar o professor no ensino do sistema decimal e das operações de adição e subtração para crianças das séries iniciais. O sistema foi projetado tendo como base o jogo educacional conhecido como Material Dourado. O Material Dourado é composto por um grande número de cubos e barras confeccionados na maioria das vezes em madeira, representando a unidade, a dezena e a centena. É utilizado no Ensino Fundamental como recurso ao ensino de Matemática fundamental. Possibilita ao aluno construir, de forma prática, o sistema de numeração decimal, através da manipulação de materiais concretos.

Cada caixa de Material Dourado é composta por um total de seiscentas e onze peças confeccionadas em madeira, dentre elas: quinhentos cubos pequenos que representam as unidades; cem barras que representam as dezenas; dez quadrados que representam as centenas e um cubo grande que representa o milhar. Tendo por base esta metodologia, desenvolveu-se um ambiente computadorizado para suportar atividades práticas visando ampliar o número de usuários sem a necessidade de a escola adquirir vários conjuntos concretos deste material.

A interface do Matematiquinha (figura 1) utiliza muitos ícones com o objetivo de facilitar a interação com o usuário. Possibilitando, assim, uma melhor compreensão através da sua visualização. O programa é destinado a crianças, que já saibam ler e escrever. A interface apresenta uma descrição textual da ação dos ícones disponibilizados. Esta descrição é ativada quando o mouse passa sobre o ícone desejado.

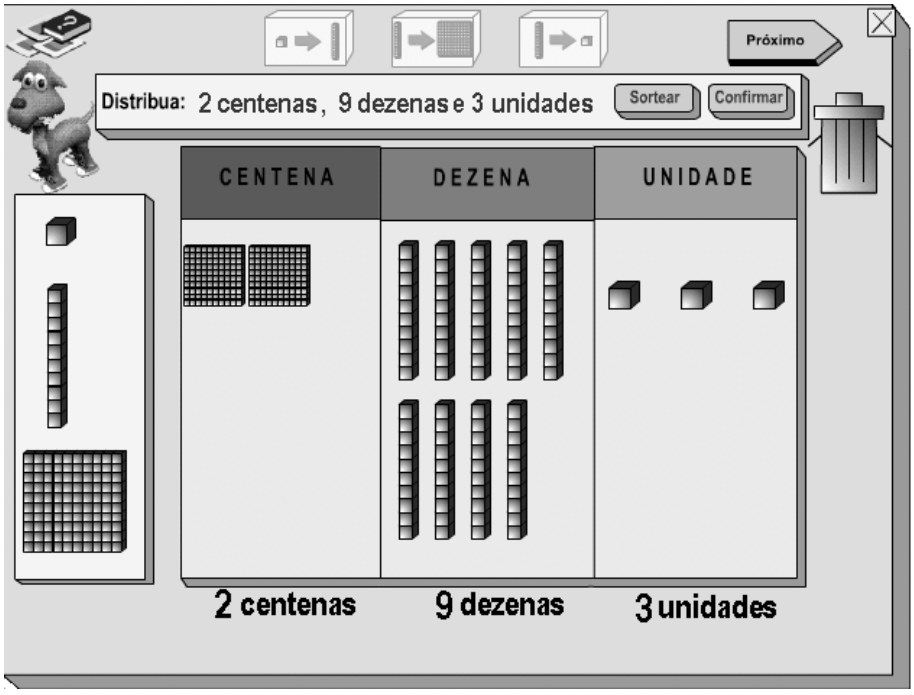

Figura 1. Apresentação da Interface do nível 1

O ambiente foi projetado para auxiliar a introduzir os conceitos relativos ao sistema decimal a partir do conceito de trocas. Ou seja, há um campo específico, onde o aluno insere as unidades, as dezenas e as centenas. É importante ressaltar que esta é uma das fases mais importantes para que o aluno possa construir o conhecimento. Para efetuar a troca, o aluno faz uso de procedimentos. Estes procedimentos o induzem a concluir que a troca não é uma simples substituição de objetos, mas sim, de grandezas. O objetivo é ilustrar a possibilidade de trocar dez objetos de um campo de menor grandeza por um único objeto pertencente ao campo de maior grandeza e vice-versa.

Os exercícios disponíveis no ambiente são organizados em níveis de complexidade diferenciados. É necessário realizá-los de forma seqüencial e ordenada. Maiores detalhes podem ser obtidos em [Ribeiro and Ribeiro 2002] e [Ribeiro, Giraffa and Ribeiro 2002]. O Matematiquinha foi testado com alunos em situação de sala de aula real, nas turmas da especialista que auxiliou a modelagem do sistema.

Este projeto tem por objetivo atender as dificuldades encontradas pela professora em acompanhar os alunos nas atividades de laboratório, quando da utilização do Matematiquinha. O professor trabalha com uma turma de 35 alunos. Sendo impossível atender cada aluno, durante todo o tempo. A opção para resolver esta limitação é utilizar um Assistente Virtual (AV). O AV registra as informações geradas durante as interações do aluno com o sistema, sendo capaz de auxiliar o aluno em situações de erro e/ou dúvidas mais freqüentes.

4. Uma Arquitetura Multiagente para o Matematiquinha

O Matematiquinha Multiagente (MM) é baseado na arquitetura de um STI. Existe uma base de domínio expressa por um conjunto de exercícios, organizada em função das atividades e objetivos do professor que atua em sala de aula presencial. Utilizou-se o

trabalho de [Schuck and Giraffa 2001] para identificar as funcionalidades necessárias para um assistente.

No MM existe a figura do tutor e do assistente. O tutor engloba as funções do tradicional módulo de estratégias de ensino previsto em arquiteturas de STI [Giraffa 1999]. Ele é o responsável pela elaboração do plano de trabalho para o aluno. Para tal ele utiliza as informações armazenadas no banco de informações do aluno. Esta base contém as informações relativas ao perfil de cada aluno, considerando os seguintes aspectos: tempo de execução das tarefas, quantidade de vezes que a tarefa foi refeita até encontrar a resposta correta, uso do sistema de ajuda, tipo de ajuda selecionada e complexidade dos exercícios selecionados.

O agente assistente é o responsável pela operacionalização do plano do tutor para um determinado aluno. O tutor permanece no servidor. Para que um determinado aluno possa utilizar o sistema ele deve ter sido previamente cadastrado pelo professor. O professor é o único usuário com acesso ao módulo de manutenção. Podendo inserir, retirar e atualizar componentes, na base de alunos e de exercícios. Na base de exercícios vamos encontrar o tipo de ajuda necessária para que o agente assistente auxilie o aluno. O aluno, ao acessar o ambiente, tem seu código de usuário e senha verificados. Ocorre, então, a ativação do agente assistente que detém agora, o plano de trabalho criado pelo tutor. Este agente trabalha com o aluno armazenando as informações do aluno e repassando-as para o tutor que as atualiza a cada sessão.

Como se prevê que o sistema esteja disponível na máquina do servidor da rede da escola. A arquitetura geral do sistema pode ser expressa pela figura 2.

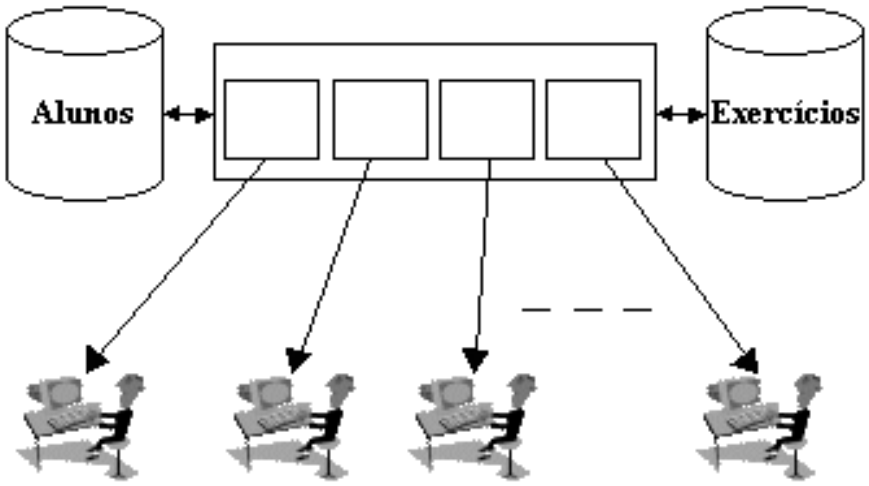

Figura 2. Arquitetura geral do sistema na rede da escola

A tarefa de assistência e gerenciamento de todas as informações do aluno é feita pelo agente assistente. O Agente Assistente (representado na interface por um mascote) apresenta os exercícios, controla as informações disponibilizadas na interface, faz as mudanças necessárias associadas a cada execução dos exercícios e as envia para o tutor. A partir de um banco de personagens, o aluno poderá configurar a aparência do mascote. O mascote poderá ficar visível na interface. Dependo da escolha do aluno. Sob o ponto de vista do aluno, o mascote é um companheiro que o auxilia quando ele

necessita. Sob o ponto de vista do sistema, o mascote (visível ou não) trabalha em parceria com o agente tutor.

Uma vez conectado ao ambiente, o aluno passa a interagir com a interface que contém no seu interior o agente assistente. De posse do plano de trabalho elaborado pelo tutor, o agente assistente pode passar ao aluno as informações necessárias ao início das suas atividades. O agente assistente, além de prestar assistência ao aluno tem a função de monitorar as informações do usuário associadas ao modelo do aluno. O tutor tem o conjunto de informações referentes ao usuário disponível na base de perfis de alunos. Esta base é atualizada pelo agente assistente, ao final de cada sessão de trabalho. Desta forma, o tutor pode atualizar seu plano de ação para o aluno. Considera-se uma sessão de trabalho aquela onde o aluno executa o plano criado pelo tutor com uma conexão (*log*).

A coreografia que se estabelece durante as interações de cada aluno vai gerar um conjunto de informações a serem enviadas para o tutor. O tutor possui internamente uma base de estratégias, conforme ilustra a figura 3.

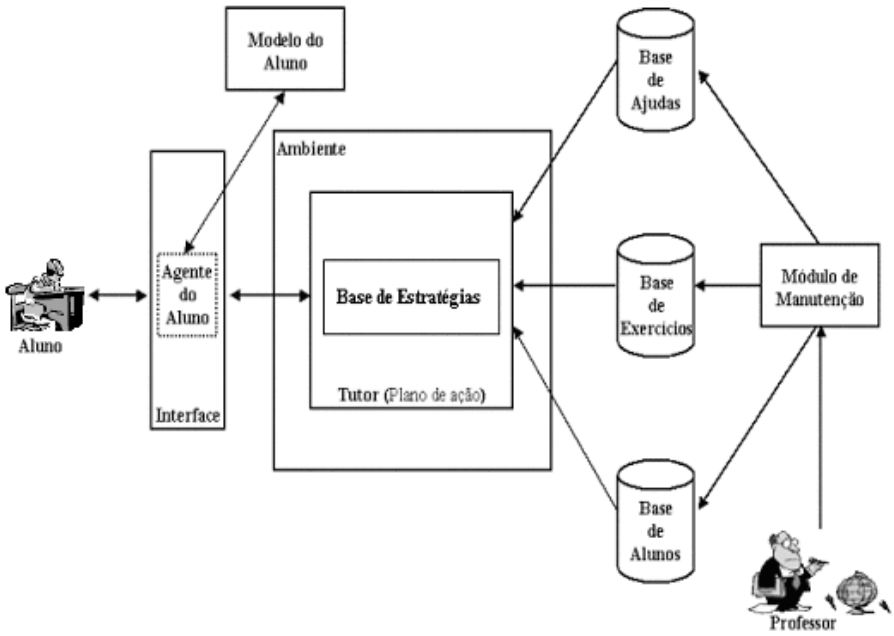

Figura 3. Arquitetura Geral do Sistema

O Tutor faz a manutenção da base de exercícios. Também, é responsabilidade do tutor verificar na base de ajudas, a existência de elementos de ajuda correspondentes com o padrão de exercícios selecionados. O ambiente possui um módulo de aquisição de conhecimento para que se modifique a base de estratégias. O comportamento do tutor é modelado através de regras.

Quando o aluno encerra sua sessão de trabalho, o agente assistente faz o *upload*[2] das informações de cada aluno no servidor. O professor possui autorização para fazer

[2] Transferir arquivos de dados de um computador pequeno para uma CPU principal.

alterações nas bases de exercícios e ajudas. Ele pode, também, obter um relatório da base de alunos.

O projeto leva em conta, principalmente, as funcionalidades do Matematinha já existentes, que não só devem ser preservadas, como estendidas para trabalhar em sintonia com os novos componentes. O esforço desprendido para se efetuar o processo de agentificação é considerável. Foi necessário adicionar ao programa original novas propriedades e possibilidade de comunicação, através de uma linguagem de comunicação entre agentes. Permitindo a interação com os novos elementos do ambiente. O ambiente está em fase final de implementação.

5. Considerações Finais

A utilização de agentes no projeto e desenvolvimento de software educacionais, especialmente nos ambientes cooperativos de suporte à aprendizagem, possibilita o aumento das possibilidades pedagógicas do ambiente. A tecnologia de agentes proporciona um aumento na qualidade do produto final (interface), permitindo o incremento das interações entre os usuários e o sistema, o tratamento do arquivo de *log* (registro das interações), a exploração de diferentes mídias associadas à representação múltipla dos conteúdos com um menor custo computacional [Giraffa 1999] e [Chaves and Bercht 2000]. A utilização desta tecnologia resgata uma série de limitações que os projetistas de software educacional vinham enfrentando em função da tecnologia disponível.

A agentificação dos sistemas permite o reuso e adaptação de sistemas já existentes e uma ampliação do seu potencial. No caso dos sistemas educacionais isto traz vantagens significativas, uma vez que permite a personalização das atividades, a adaptação em função das necessidades do usuário (aluno) e, a utilização do sistema fora do escopo de uma aplicação local.

A agentificação de um software educacional parece trazer contribuições significativas à pesquisa na área de software educacional. Cabe salientar, que não é um processo simples e rápido. A identificação dos componentes passíveis de serem reutilizados e, a compatibilização de tecnologias, envolvem muito tempo e estudo. É fundamental a existência de uma boa documentação do sistema original. No caso do Matematiquinha, parte dos autores deste artigo desenvolveram o projeto original. Como o objetivo desta pesquisa era justamente verificar a questão do volume de trabalho demandado para se agentificar um software educacional, a escolha de um ambiente conhecido e com completa documentação foi fundamental.

Muitas questões ainda estão em aberto e nos desfiarão por muito tempo no que concerne ao projeto de ambientes educacionais. Os componentes afetivos que devem ser considerados no processo de ensino-aprendizagem, a origem da motivação e, muitas outras possibilidades que não se esgotam numa pequena lista. Mas elas também podem ser contempladas com uso da tecnologia de agentes.

Um projeto deste porte não se consegue levar a termo sem uma equipe interdisciplinar. Este trabalho conta com a participação de uma professora de ensino básico, bolsistas de iniciação científica (adaptação do código fonte original), e o um mestrando e respectiva orientadora, que funcionam como os especialistas em Informática na Educação.

O custo estimado de tempo para modelagem do ambiente e construção do protótipo é de aproximadamente 18 meses. Logo, isto explica a pouca oferta de ambientes com estas características. Estima-se que após a fase de testes do protótipo, o sistema em versão completa, consumirá mais 12 meses para sua efetiva utilização.

Na seqüência deste trabalho buscar-se-á definir que padrões devem ser seguidos para documentação do processo de agentificação e, como tratar da questão da atualização do plano do tutor numa mesma sessão de trabalho do aluno. Isto é, como modificar o plano gerado para o aluno de maneira *on-line*, atualizando e adaptando o plano à medida que o aluno realiza etapas do plano inicial. Esta é uma tarefa de maior porte que será abordada na seqüência da pesquisa, em um novo projeto.

Referências bibliográficas

Augustine, C. H. (1996) "Métodos Modernos Para o Ensino da Matemática", Rio de Janeiro: Ao Livro Técnico, 1996.

Bellman, K. L. (1990) "The Modelling Issues Inherent in Testing and Evaluating Knowledge-Based Systems" In Special Issue: Verification an Valiation of Knowledge Based Systems,Expert Systems With Application Journal 1:109-215.

Bordini, R. H. and Vieira, R.; Moreira, Á. F. (2001) "Fundamentos de Sistemas Multiagentes", In Anais do XXI Congresso da Sociedade Brasileira de Computação (SBC2001), Volume 2, XX Jornada de Atualização em Informática (JAI), 30 de julho - 3 de agosto, Fortaleza - CE, Brasil. Sociedade Brasileira de Computação. chapter 1, 3-41.

Chaves, T. H. C.and Bercht, M. (2000) "Elementos de Um Ambiente Multiagente com Intermediação para Suporte à Aprendizagem" , In Anais do Simpósio Brasileiro de Informática na Educação (SBIE), 08 - 10 de novembro, Maceió - AL, Brasil. Sociedade Brasileira de Computação. pp. 141-147,.

Genesereth, M. R. and Ketchpel S. P. (1994)"Software Agents", In Communications of the Association for Computing Machinery, July, pp 48-53.

Giraffa, L. M. M. (2002) "Inteligência Artificial Distribuída e os agentes", http://www.edukbr.com.br, May.

Giraffa, L. M. M. (1999) "Uma Arquitetura de Tutor Utilizando Estados Mentais", Porto Alegre: CPGCC/UFRGS. (Tese de Doutorado).

Jennings, N. R. (2000) "On Agent-Base Software Engineering", In Artificial Intelligence, 117:227-296,.

Landauer, C. (1990b) "Correctness Principles for Rule-Based Expert Systems", Special Issue: Verification and Validation of Knowledge Based Systems, Expert Systems With Apllications Journal 1:291-316

Landauer, C. (1990a). "Wrapping the Mathematical Tools" In Proc. 1990 SCS Eastern Multi Conference, 23-26 April, Nashville, TN, Simulation Series, 22:261-266.

Landauer, C.; Bellman K. L (1996c) "Constructed Coplex Systems: Issues, Architectures and Wrappings" In Proc.EMCRS 96: Thirteenth European Meeting on

Cybernetics and Systems Research, Symposium on Complex Systems Analysis and Design, 9-12 April, 233-238, Vienna.

Landauer, C.; Bellman K. L. (1992) "Integrated Simulation Environmens", In Proc. DARPA Variable-Resolution Modeling Conference, 5-6 May, Herndon, Va Conference Proc., P. K. Davis and R. Hillestad (eds.). 409-431.

Landauer, C.; Bellman K. L (1996a) "Knowledge-Based Integration Infrastructure for Complex Systems", Internat. J. Intelligent Control and Systems 1:133-153.

Landauer, C.; Bellman K. L (1997a) "Model-Based Simulation Design With Wrappings", In Proc. OOS'97: Object Oriented Simulation Conference, WMC'97: 1997 SCS Western Multi-Conference, 12-15 January, 169-174.

Landauer, C.; Bellman K. L (1998b) "Wrappings for Software Development", In 31st Hawaii Conferece on System Sciences, Volume III: Emerging Technologies, 6-9 January, Kona, HI pp 420-429.

Landauer, C.; Bellman K. L.; Gilliam, A. (1993) "Software Infrastructure for System Engineering Support", In Proceedings AAAI '93 Workshop on Artificial Intelligence for Software Rengineering, 12 July, Washington, D. C.

Ribeiro, J. C. C. S.; Ribeiro, U.A. L. F. (2002) "Matematiquinha — Uma Ferramenta De Auxilio Ao Ensino De Matemática Para Séries Iniciais", III Simpósio de Informática do Planalto Médio (SIPM'2002), 21 –24 de maio, Passo Fundo – RS.

Ribeiro, J. C. C. E. S., Giraffa. L. M. M., Ribeiro, U. A. L. F.(2002) "Agentificando um Ambiente Computadorizado para suporte a Aprendizagem de Matemática Elementar", In VII Simpósio De Informática E II Mostra Regional De Software Acadêmico – Revista Hífen – Pontifícia Universidade Católica do Rio Grande do Sul, Uruguaiana, p 165-170.

Schuck, P. W.; Giraffa, L. M. M. (2001) "Construindo um Sistema Tutor Inteligente para suporte ao ensino de Matemática Financeira: da modelagem à validação" In Anais do XII Simpósio Brasileiro de Informática na Educação (SBIE), 21 - 23 de novembro, Vitória - ES, Brasil. Sociedade Brasileira de Computação. pp. 418-426,.

Wooldridge, M.; Jennings, N. R. (1997) "Agent-Based Software Engineering", IEE Proc. Software Engineering 144 (1) 26-37

Wooldridge, M.; Jennings, N. R. (1998) "Pitfalls of Agent-Oriented Development." In Proceedings of the second International Conference on Autonomous Agents (Agents 98), pages 385-391, Minneapolis/St Paul, MN, May.