FACULDADE DE INFORMÁTICA

PUCRS – Brazil

http://www.inf.pucrs.br

Estudo comparativo sobre Sistemas Tutores Inteligentes Multiagentes Web

Willian Bolzan and Lúcia Maria Martins Giraffa

TECHNICAL REPORT SERIES

Number 024

July, 2002

Contact:
willian@inf.pucrs.br
http://www.inf.pucrs.br/~willian

giraffa@inf.pucrs.br
http://www.inf.pucrs.br/~giraffa

Willian Bolzan is a graduate student of Urcamp - RS - Brazil. He is a member of the GIE research group (Computer Science applied in Education Research Group) since 2001. He receives a graduate research grant from CAPES to support his research.

Lucia Maria Martins Giraffa works at PUCRS/Brazil since 1986. She is a titular professor and leader of the GIE group. She develops research in Multi-agent systems, Artificial Intelligence applied in Education and design of educational software. She received her Ph.D. in 1999 at UFRGS (Brazil).

Published by the FACIN – PUCRS
Av. Ipiranga, 6681
90619-900 Porto Alegre – RS – Brazil

ESTUDO COMPARATIVO SOBRE SISTEMAS TUTORES INTELIGENTES MULTIAGENTES WEB

Relatório Técnico Nº 024/2002
Willian Bolzan (Mestrando)[1]
Lúcia Maria Martins Giraffa(Orientador)[2]

1. Introdução

Atualmente, muitos dos ambientes de ensino-aprendizagem computadorizados na modalidade de Sistemas Tutores Inteligentes (STI), utilizam a tecnologia de agentes no seu projeto. Esta abordagem, orientada a agentes, substitui os módulos da arquitetura tradicional (Vide Figura 1) por uma sociedade de agentes que trabalham de forma cooperativa usando diversas técnicas de Inteligência Artificial (IA) e integrados como um Sistema Multiagente (SMA).

Neste Relatório Técnico (RT) apresenta um estudo sobre os Sistemas Tutores Inteligentes (STI) para a Web que utilizam a tecnologia de agentes no seu projeto e desenvolvimento. É importante salientar que este estudo não será restrito apenas a ambientes classificados explicitamente pelos autores como STI, mas analisa ambientes de ensino-aprendizagem que foram concebidos utilizando os princípios de uma arquitetura de STI. Isto é, ambientes onde existam características inerentes ao STI, tais como: um assistente pessoal, um mediador, etc.

Utilizamos a expressão "tecnologia de agentes", ao invés de "paradigma de agentes" porque ainda não existe consenso na área de IA para este novo padrão. Segundo Thomas Kuhn [KUH1997], paradigmas são as realizações científicas universalmente reconhecidas que, durante algum tempo, fornecem problemas e soluções modulares para a comunidade de praticantes de uma ciência. Portanto, tecnologia de

[1] e-mail: willian@inf.pucrs.br
[2] e-mail: giraffa@inf.pucrs.br

agentes não pode ser definido desta forma por não ter uma especificação, um modelo padrão totalmente estabelecido e aceito.

Este trabalho está organizado da seguinte maneira. A seção 2 apresenta uma breve introdução aos Sistemas Tutores Inteligentes e Sistemas Multiagentes. A seção 3 descreve os STI desenvolvidos com a tecnologia de Agentes. A seção 4 apresenta um estudo comparativo sobre os STI descritos na seção 3. As seções 5 e 6 apresentam respectivamente as considerações finais e a bibliografia utilizada para a redação deste trabalho.

2 Sistemas Multiagentes e Sistemas Tutores Inteligentes

Os SMA constituem-se numa área de pesquisa da Inteligência Artificial Distribuída (IAD). A abordagem de agentes se preocupa em estudar o comportamento de uma sociedade constituídas por agentes, que possuem autonomia, e têm como objetivo realizar tarefas que não são possíveis de serem realizadas individualmente ou coletivamente.

Segundo Russell & Norvig [RUS1995], um Agente é um sistema capaz de perceber através de sensores as informações do ambiente onde está inserido e reagir através de atuadores. Um agente pode ser definido como uma entidade de software que exibe um comportamento autônomo , que está situado em algum ambiente sobre o qual é capaz de realizar ações para alcançar seus próprios objetivos de projeto e a partir do qual percebe alterações [WOO1995, ZAM2000]. Um agente é um software que possui um conjunto de propriedades específicas associadas ao seu objetivo/papel na sociedade multiagente onde está inserido. O seu objetivo/papel vai determinar as propriedades que deve ter. Porem, já existe consenso que um agente deve ter no mínimo: autonomia, reatividade e habilidade social (comunicar-se com outros agentes do ambiente).

Numa abordagem clássica para a área de agentes encontramos a definição de Wooldridge [WOO1995], que visualiza um agente como sendo uma entidade com capacidade de resolução de problemas encapsulada. Neste contexto, define-se agente como tendo as seguintes propriedades:

- **autonomia**: executam a maior parte de suas ações sem interferência direta de agentes humanos ou de outros agentes computacionais, possuindo controle total sobre suas ações e estado interno;

- **habilidade social**: por impossibilidade de resolução de certos problemas ou por outro tipo de conveniência, interagem com outros agentes (humanos ou computacionais), para completarem a resolução de seus problemas, ou ainda para auxiliarem outros agentes;
- **capacidade de reação**: percebem e reagem à alterações no ambiente em que estiverem inseridos;
- **capacidade pró-ativa**: agentes, do tipo deliberativo, além de atuar em resposta às alterações ocorridas em seu ambiente, apresentam um comportamento orientado a objetivos, tomando iniciativas quando julgarem apropriado (no caso, aplicado apenas aos agentes cognitivos).

A justificativa de aplicação da tecnologia de agentes na concepção de Sistemas de Informação é justificada quando o problema possui as seguintes características [JEN1996]:

- o domínio envolve distribuição intrínseca dos dados, capacidade de resolução de problemas e responsabilidades;
- necessidade de manter a autonomia de subpartes, sem a perda da estrutura organizacional;
- complexidade nas interações, incluindo negociação, compartilhamento de informação e coordenação;
- impossibilidade de descrição da solução do problema *a priori*, devido à possibilidade de pertubações em tempo real no ambiente e processos de negócio de natureza dinâmica.

Logo, os softwares educacionais podem se utilizar da tecnologia de agentes porque possuem todas essas características.

As propostas de utilização de arquiteturas SMA em STI trazem uma grande vantagem em relação as arquiteturas tradicionais de STI por apresentarem uma flexibilidade maior no tratamento dos elementos que compõem o sistema. Além disso, o fato de usarmos agentes para modelar os seus componentes possibilita o agrupamento da arquitetura tradicional (um módulo = um agente) ou na explosão de cada módulo em vários agentes, como por exemplo o trabalho de [COS1995; COS1996]. Neste último caso, o refinamento pode chegar até os estados mentais de um agente, [GIR2001a].

Segundo [GIR2001a], a modelagem de STI numa arquitetura funcional de agentes é mais do que uma abordagem generalista. Teoricamente a modelagem não apresenta limites para o número de agentes que podem participar do processo de

aquisição do conhecimento. Processo este que ocorre através de negociação dos papéis dos agentes tanto tutores, como aprendizes.

A utilização de técnicas de IA (Inteligência Artificial) em programas destinados a auxiliar no processo de ensino-aprendizagem remonta do início da década de 70, como o trabalho de Carbonel [CAR1970].

A proposta de Carbonel foi modelar os sistemas educacionais de maneira a levar em consideração o tipo de aluno que está interagindo com o sistema. Ou seja, ele lançou a idéia dos sistemas educacionais adaptativos, sob o ponto de vista de tutoria (Assistência ao aluno).

As pesquisas de Carbonell criaram um marco histórico para os STI, o SCHOLAR [CAR1970]. O ESCHOLAR é um clássico considerado o clássico dos STI. Maires detalhes leia [GIR1998]. Conforme [GIR2001a], a pesquisa do grupo de Carbonel visava atender ao grupo de programas onde existia um conteúdo modelado com um determinado objetivo educacional e tentar adaptar a apresentação deste conteúdo ao perfil de cada aluno. Isto é, um sistema que procurasse se adaptar (em algum grau) ao usuário e não se comportasse de maneira igual com todos os usuários (padrão usado na época para os CAI (*Computed Assisted Instruction*)).

Originalmente estes sistemas foram chamados de ICAI (*Intelligent* CAI) e, durante muito tempo, foram citados na literatura da área como sinônimos de STI. Hoje esta associação é muito questionada por vários pesquisadores e STI deixou de ser sinônimo de ICAI. Como esta não é uma questão relevante para este trabalho, não será abordada.

Se considerarmos que estudantes possuem diferentes estilos ou formas de aprendizagem, é importante que um STI seja capaz de fornecer instrução individualizada, ou seja, deve ser capaz de adaptar suas instruções para satisfazer os estilos individuais de cada aprendiz. Este é o princípio pedagógico geral que rege a concepção de um STI.

STI são sistemas computacionais com modelos de conteúdo instrucionais que especificam o que ensinar e estratégias de ensino que especificam como ensinar [MUR1999].

Segundo [OLI1994, FRE2000], os STI possuem uma organização básica com alguns componentes funcionais que podem ser observados na maioria dos casos: **Módulo do Aluno**, **Módulo Tutor**, **Módulo de Domínio** e **Interface**. A Figura 1, apresenta a arquitetura multipartida onde os módulos estão representados com suas

inter-relações e em seguida há uma pequena descrição dos módulos que compõe a arquitetura clássica de um STI:

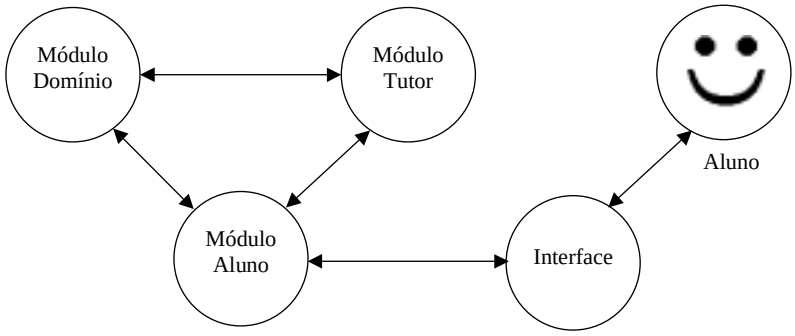

Figura 1: Arquitetura clássica de um STI

- **Módulo do Aluno**: este módulo armazena informações específicas para cada estudante de forma individual. No mínimo, este módulo deve manter um histórico sobre como o estudante está trabalhando no material em questão. É interessante também manter registro sobre os erros do estudante, [THI1999]. Segundo [GIR2001], o módulo do aluno representa o conhecimento e as habilidades cognitivas do aluno em um dado momento. Contém uma representação do estado do conhecimento do aluno no momento qu interage com o STI. A partir desse modelo e do conteúdo representado na base do domínio, o sistema deve ser capaz de inferir a melhor estratégia de ação a ser utilizada para cada aluno.
- **Módulo Tutorial**: O módulo tutorial, ou módulo pedagógico oferece uma metodologia para o processo de aprendizado. Possui o conhecimento sobre as estratégias e táticas para selecioná-las em função das características do aluno. As entradas deste módulo são fornecidas pelo Módulo do Aluno.
- **Módulo de Domínio**: O módulo de domínio armazena a informação que o tutor está ensinando. A modelagem do conhecimento a ser disponibilizado é de grande importância para o sucesso do sistema como um todo. Deve-se procurar uma representação do conhecimento que esteja preparada para o crescimento incremental do domínio.
- **Interface**: Intermedia a interação entre o tutor e o aluno. Segundo [THI1999], a complexidade para a implementação deste módulo é bastante variável, podendo ser desde simples janelas de diálogo até linguagem natural

e reconhecimento de voz. Outra questão a ser considerada é a aplicação de realidade virtual para permitir uma imersão total do estudante no sistema.

Conforme Giraffa [GIR2001], esta proposta trouxe grandes avanços à modelagem de ambientes educacionais, pois separou o domínio da sua forma de manipulação. Permitindo, assim, que estratégias de ensino fossem associadas em função das informações oriundas da modelagem do aluno e relacionadas com que o domínio (conteúdo) é organizado.

Esta arquitetura clássica, foi ampliada por Self [SELF1999 *apud* GIR2001], para tornar uma arquitetura tripartida associada ao modelo de interação que ocorre ao longo de uma sessão de trabalho entre o aluno e o ambiente. A Figura 2 mostra a arquitetura tripartida proposta por Self para os STI.

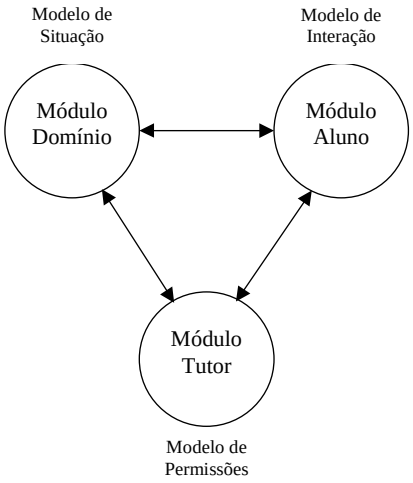

Figura 2: Arquitetura proposta por Self para STI

Segundo [GOU2000], o módulo de domínio não é mais uma forma de tornar as informações interrelacionadas, mas sim um modelo dos aspectos do conhecimento sobre o domínio que o aluno pode acessar durante as interações com o STI (Modelo da Situação). O módulo do estudante não mais relaciona somente as informações sobre a análise das interações do aluno com o domínio, mas busca uma contextualização maior destas interações em função das ações do aluno, o contexto em que elas ocorrem e a estrutura cognitiva do aluno naquele momento (Modelo de Interação). E o módulo tutor deixou de ser o responsável pela seleção do conteúdo e estratégias para se tornar aquele que conduz o aluno de acordo com objetivos e desafios educacionais que o ambiente proporciona ao aluno (Módulo de Permissões).

Agora, relacionando STI com agentes citamos Claude Frasson [FRA2000]: "*Uma das mais promissoras aplicações de agentes autônomos está provavelmente na educação e treinamento*" (p.554).

Existem diversos exemplos na literatura sobre a utilização de agentes em sistemas educacionais. Segundo Shoham [SHO1993 *apud* THI1999], uma sociedade de agentes para aprender e ensinar pode ser a solução para a construção de ambientes de ensino e aprendizagem, se os agentes trabalham de uma maneira concorrente e autônoma para alcançar seus objetivos. Os agentes em um ambiente de ensino/aprendizagem são considerados autônomos porque: as atividades dos agentes individuais não requerem constante supervisão externa (humana), e não há (ou deveria haver) autoridade central projetada para controlar todas as interações desempenhadas entre os agentes.

Segundo [GÜR98], o uso de agentes na concepção de sistemas educacionais traz algumas vantagens, tais como: reagir às ações do usuário, credibilidade, modelagem de sistemas colaborativos multi-usuário e modularidade, pelo fato de que cada agente é um módulo único e independente do outro ficando mais fácil adicionar outros agentes a estes sistemas . Para [GIR1998] as vantagens são as seguintes:

- conhecimento pode ser distribuído entre vários "tutores", cada um com suas crenças, desejos, objetivos, emoções e planos de ação. Esta distribuição cria maiores oportunidades de variar técnicas pedagógicas;
- o aprendiz interage com um tutor de forma mais flexível;
- aprendiz pode passar conhecimentos ao tutor que serão repassados a outros aprendizes.

Conforme [WEB2001], as tecnologias baseadas na Web em conjunto com metodologias multiagentes formam uma nova tendência na modelagem e desenvolvimento para ambientes de aprendizagem. A Educação baseada na Web tem sido extensivamente pesquisada, onde os benefícios de aprendizagem são grandes. Como por exemplo, alcance da informação sem condicionamento ao espaço físico, facilidade de atualizar o conteúdo, etc. Entretanto, as metodologias multiagentes tem surgido com uma alternativa para conceber aplicações de aprendizagem distribuída, devido ao conjunto de características inerentes ao conceito de SMA e as peculiaridades de uma sociedade de agentes A principal razão para isto deve-se ao fato que esta tecnologia lida muito bem com aplicações criticas, tais como: distância, cooperação entre diferentes entidades e integração de diferentes componentes de software

Segundo [VAS2001], no futuro, os ambientes da aprendizagem estarão acessíveis a qualquer lugar e a qualquer hora. Os estudantes desses ambientes estarão distribuídos no espaço e no tempo. Logo, trabalhos que utilizam arquiteturas multiagentes oferecem uma promissora abordagem para o projeto desses ambientes, desde que estes ambientes sejam distribuídos. Com a modularidade e a uniformidade dos agentes e com a padronização os protocolos de interação, o nível de escalabilidade e interoperação podem ser alcançados, o que não pode ser conseguido tão facilmente com o uso de outras técnicas. As arquiteturas multiagentes permitem o constante crescimento e a heterogeneidade do ambiente de software. Atualmente, não há muitos ambientes distribuídos de aprendizagem baseados em arquiteturas multiagentes. Isto deve-se a vários fatores:

- A complexidade em modelar e implementar STI e ambientes de ensino-aprendizagem inteligentes. Sua arquitetura é mais complexa e a modelagem dos seus componentes e interrelações é demorada, necessitando de uma grande quantidade de trabalho cooperativo em equipe interdiciplinar;
- A tecnologia de agentes aplicada a tais ambientes agrega mais complexidade ao seu projeto e pode ser considerada como um fato mais recente. Trabalhos mais significativos remontam aos últimos 5 anos de pesquisa. Como se pode observar na análise dos anais dos principais Congressos e eventos da área, tais como: *Intelligent Tutoring Systems* – ITS' (1996, 1998 e 2000), *Artificial Intelligence in Education* – AIED' (1997, 1999 e 2001),), *Brazilian Symposium on Artificial Inteligence* – SBIA' (1995 e 1998),), Simpósio Brasileiro de Informática na Educação – SBIE' (1998, 1999, 2000 e 2001), *WorkShops* de Ambientes de Aprendizagem Baseados em Agentes - SBIE´ (1999, 2000 e 2001), *The International Journal of Artificial Intelligence in Education* – IJAIED.

Os avanços mais recentes no campo dos ambientes de aprendizagem inteligentes, têm proposto o uso de arquiteturas baseadas em sociedades de agentes. Os princípios dos sistemas multiagentes têm mostrado um potencial bastante adequados ao desenvolvimento de sistemas de ensino, devido ao fato de a natureza do problema de ensino-aprendizagem ser mais facilmente resolvido de forma cooperativa, [BIC1998]. Além disso, ambientes de ensino baseados em arquiteturas multiagentes possibilitam suportar o desenvolvimento de sistemas de forma mais robusta, mais rápida e com menores custos.

O próximo capítulo apresenta o resultado da revisão bibliográfica realizada nos principais congressos da área de STI e *Workshops* recentes onde foram apresentados os trabalhos que representam o estado da arte neste segmento de pesquisa.

3 STI desenvolvidos com a tecnologia de Agentes

Nesta seção serão descritos alguns STI desenvolvidos com a utilização da tecnologia de Agentes. Este levantamento terá como finalidade principal, fornecer elementos para uma análise a fim de identificar padrões utilizados no desenvolvimento destes sistemas.

3.1 WHITE RABBIT

White Rabbit [THI2000] é um sistema desenvolvido pelo Departamento de Informática e Pesquisa Operacional da Universidade de Montreal (Montreal – Canadá), este sistema tem como objetivo aumentar a cooperação entre um grupo de pessoas pela análise de suas conversações. Cada usuário é assistido por um agente inteligente o qual estabelece um perfil de seus interesses. Com o comportamento móvel e autônomo o agente pesquisa agente pessoais de outros usuários, afim de encontrar aquele que tenham interesses comuns e então os colocam em contato.

Um agente Mediador é usado para facilitar a comunicação entre os agentes pessoais e para realizar agrupamentos nos perfis que eles tenham recolhidos. Esta abordagem usa agentes inteligentes para descobrir os interesses particulares de um grupo de pessoas trabalhando em um domínio particular com a intenção de colocá-los em contato para aumentar o nível de cooperação. Os agente analisam a conversação entre os usuários através de um chat para construir para cada um deles, um perfil de seus interesses.

A Figura 4 apresenta a arquitetura do agente Pessoal e a Figura 3 apresenta a arquitetura geral do sistema que é constituída por 6 seções principais, que são:

- Uma camada *Voyager*[3] dá aos agentes mobilidade e autonomia;

[3] *Voyager* é uma arquitetura proprietária para agentes móveis da *ObjectSpace* (http://www.objectspace.com). A plataforma *Voyager* tem como objetivo dar suporte a objetos móveis e agentes autônomos. Desenvolvida em Java, esta plataforma suporta vários padrões, tais como: CORBA, COM e RMI. No contexto deste projeto, o módulo de comunicação esta baseado inteiramente nesta arquitetura, permitindo aos agentes do sistema comunicar-se uns com os outros

- O servidor *chat* que organiza o fluxo de mensagens através da rede;
- Uma interface dedicada para o usuário e o administrador do sistema. Permite que o usuário envie e receba mensagens, consulte e modifique seu perfil e permite que o administrador observe e ajuste os parâmetros do processo de *clustering*[4] e alterar a base de conhecimento;
- Um agente Pessoal para cada usuário que realiza o serviço de análise e apresentação do serviço, Figura 4;
- Uma base de conhecimento, onde são alocadas diferentes palavras-chave sobre o domínio do conhecimento e seus links;
- Um agente Mediador para dedicar-se ao processo de *clustering* e facilitar a comunicação entre os agentes.

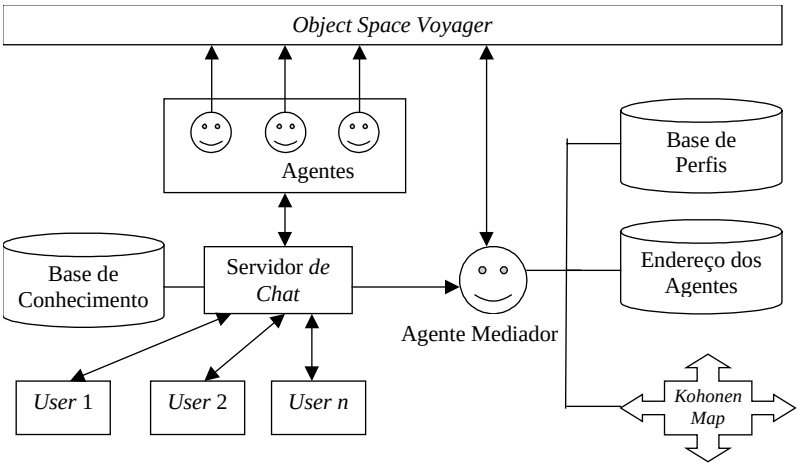

Figura 2: Arquitetura geral do sistema *White Rabbit*

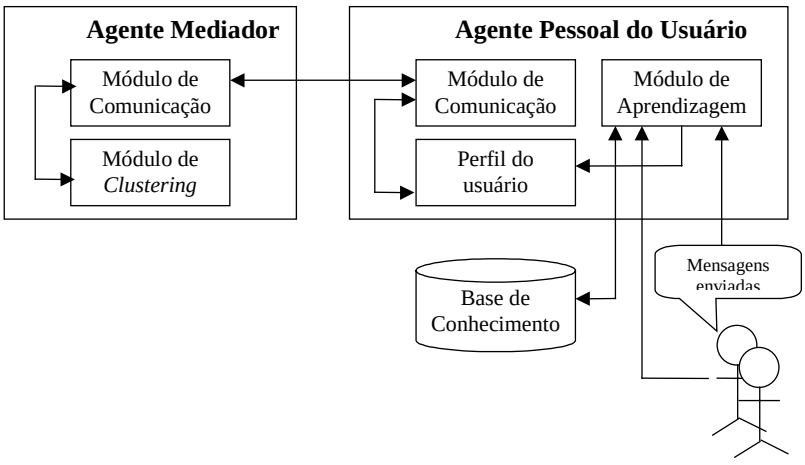

Figura 4: Arquitetura do agente Pessoal do sistema White Rabbit

[4] Processo de Agrupamento.

O perfil do usuário contém todas as informações relevantes sobre o interesse do usuário na escolha do domínio o qual permitirá aos agentes encontrar similaridades e consequentemente realizar agrupamentos coerentes.

O módulo de aprendizagem é responsável por modificar o perfil do usuário fazendo-o mais acurado e mais realista. Este processo é feito em duas etapas:

- O primeiro passo consiste na aquisição preliminar de informação sobre o usuário através de um questionário. Na primeira vez que o sistema é usado o usuário é convidado a preenchê-lo dando ao sistema palavras-chave refletindo seus interesses (projetos, realizações e experiências);
- O segundo passo é a análise da discussão, a qual consiste na extração de palavras-chave do domínio das mensagens enviadas pelo usuário e a atualização do perfil, incrementando o peso dos conceitos associados.

O perfil do usuário é composto por um conjunto de informações ponderadas, isto é, cada um deles possui um peso associado ao contexto onde estão inseridos. Isto permite um ajuste mais personalizado e detalhado do aluno.

Dois aspectos demonstram a mobilidade e autonomia dos agentes. Primeiro, a analise da discussão dos usuários pelos agentes pessoais. Depois de ter analisado e atualizado os perfis dos clientes, o agente vai para uma outra máquina conectada a rede e encontra outros agentes. A segunda importante evidência da mobilidade acontece quando o usuário pergunta para o segundo usuário quem é o membro do mesmo grupo, como determinado pelo agente mediador. Neste momento a requisição do agente do usuário (A) usará a sua autonomia para mover-se e encontrar o agente associado ao cliente (B). Então o agente (A) terá possibilidade de questionar mais sobre o agente (B). Se o agente)B) aceitar as requisições, então o agente (B) dará ao agente (A) as informações pessoais (nome real, e-mail, descrição do projeto, etc)

Estas duas situações demonstram a força alcançada pela mobilidade e autonomia dos agentes, a qual permite considerável redução do número de mensagens transmitidas na rede, desta maneira o risco de sobrecarga na rede e falta de recursos são reduzidas. Mesmo quando o número de agentes pessoais é alto (o número de agente é igual ao número de usuários)

O algoritmo de *clustering* utilizado pelo sistema é baseado no *Map Kohonen*[5] . O *Map Kohonen* é composto de duas camadas distintas: a camada de entrada e a camada de projeção. Cada neurônio é a entrada que representa um atributo ou dimensão dos dados de entrada. Cada um dos neurônios é conectado a todos os neurônios da camada de projeção e cada conecção esta associada a um peso, geralmente um número entre 0 e 1. Com esse vetor, cada neurônio tem a possibilidade de computar seu nível de ativação.

No sistema, o *Map Koronen*, é encontrado no agente Mediador o qual recebe os perfis dos usuários trazidos pelo agente pessoal, em seguida o perfil é adicionado à base contendo todos os perfis. Esses perfis são então convertidos um a um em pesos em um vetor que então será usado para ativar valores das unidade de entrada da rede neural. A rede então é treinada através dos conjunto de perfis. Um *cluster* é determinado para cada perfil pelo nível de ativação e determina o maior para cada um.

3.2 LeCS

O LeCS (*Learning from Case Studies*) foi desenvolvido pelo Departamento de Computação e Estatística da UFSC (Santa Catarina – Brasil), em conjunto com a Universidade do Vale do Itajaí (Univali) e a Unidade de Aprendizagem Baseada em Computador da Universidade de Lees (Leeds – UK), LeCS é um sistema inteligente de aprendizagem à distância, o qual, segundo [ROS2000], possui uma arquitetura baseada em um Sistema Federativo de agentes.

A Figura 5 descreve a arquitetura e a estrutura da comunicação usada, a qual estabelece que a comunicação não acontece diretamente entre os agentes, mas através de um Facilitador (*Facilitator*). O *Facilitator* é um programa especial (implementado como um agente) que guarda a informação sobre cada agente no sistema e é responsável pelo roteamento dessas mensagens, trabalhando como um *broker*. Há dois banco de dados implementados: o primeiro é o próprio *Facilitator* que armazena todas informações necessárias para rotear as mensagens, e o segundo é um mecanismo de *logs* que armazena todas as trocas de mensagens.

A arquitetura inclui três classes de agentes: Agente Interface, Agente de Informação e Agente Conselheiro.

[5] *Kohonen Maps* é um tipo de rede neural que realiza aprendizagem não-supervisionada

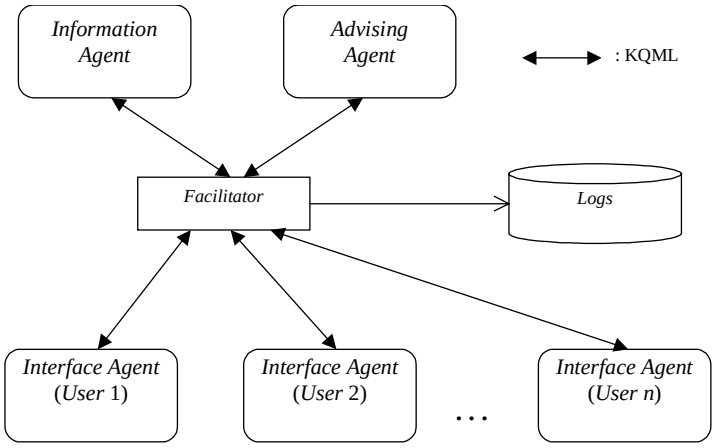

Figura 3: Arquitetura do sistema LeCS

- ***Interface Agent*** - (Agente Interface): Todas as intervenções do sistema são apresentadas através deste agente. O Agente Interface armazena as informações sobre os usuários individualmente: o que é digitado no editor de texto, o número de participantes do *chat*, a etapa em que o indivíduo esta trabalhando, a resposta a cada questão e o tempo gasto em cada etapa. Baseado nestas informações, o Agente Interface faz intervenções sobre tempo e participação. Utiliza o banco de dados que contém informações adicionais: o endereço dos agentes, o nome pelo qual ele é conhecido, etc.
- ***Information Agent*** – (Agente Informação): este agente lida com o domínio e o conhecimento pedagógico. O Agente Interface e o Agente Conselheiro podem acessar o Agente Informação. A informação armazenada por este agente é dividida em duas diferentes categorias: material didático e base de conhecimento. O material didático consiste de páginas HTML, imagens e textos. A base de conhecimento refere-se ao domínio e representação das soluções dos casos desenvolvidos. Além disso, este agente armazena as interações do *chat*.
- ***Advising Agent*** – (Agente Conselheiro): enquanto que o Agente de Informação tem acesso ao banco de dados e as bases de conhecimento, o Agente Conselheiro tem um mecanismo para entender sobre suas ações e para reconhecer situações onde algum suporte é necessário. O Agente Conselheiro executa o algoritmo para gerar uma árvore de soluções com a informação fornecida pelo Agente Interface e retorna a representação para este agente. Quando aplicável ele gera uma intervenção sobre o mal

entendimento de caso de estudo e envia uma requisição para uma intervenção do Agente Interface.

Comunicação dos agentes está baseada na *Agent Comunication Language* (ACL) e as mensagens trocadas entre os agentes usam o formato da *Knowledge Query and Manipulation Language* (KQML).

O sistema foi implementado na linguagem Delphi, numa arquitetura cliente-servidor. O servidor hospeda as seções, as quais são associadas com um grupo de estudantes trabalhando cooperativamente. O cliente roda na máquina do estudante permitindo dessa forma que vários estudantes possam participar de uma mesma sessão.

A Figura 6 mostra a interface com o usuário do sistema LeCS, a qual é composta da seguinte maneira: uma lista de participantes, um browser, uma área para a representação gráfica da solução, um *chat*, um editor de texto colaborativo e uma área de intervenções do sistema. O *browser* é usado para acessar *web pages* que apresentam o conteúdo dos estudos de caso e os passos para guiar o desenvolvimento da solução deste casos. O *chat* é o lugar onde pode acontecer as discussões sobre o assunto. O editor de texto é usado para responder as questões colocadas em cada etapa.

Figura 6: Interface com o usuário do sistema LeCS

Segundo [ROS2000, ROS2000a], o LeCS pode ser caracterizado como um sistema inteligente para o ensino a distância. LeCS dá suporte à aprendizagem colaborativa através da *World Wide Web* (WWW) usando o método de ensinar com estudos de casos (*Case Based Resoning* - CBR). O cenário de aprendizado para o uso do sistema é um grupo de alunos que está geograficamente disperso, cursando uma disciplina (por exemplo engenharia, medicina, administração, etc.) de um curso a

distância onde o método de aprender através de estudos de casos. O sistema inclui as ferramentas necessárias para desenvolver a solução para um determinado caso (a ser modelado pelo professor) e desempenha funções que dão apoio ao processo de aprendizado.

3.3 LANCA

No projeto LANCA [FRA1998], os autores procuram expor o porque do uso de agentes inteligente em STI e como podem ser adaptados para aprendizagem à distância. O projeto foi desenvolvido pelo grupo do Departamento de Informática e Pesquisa Operacional da Universidade de Montreal (Montreal – Canadá) e da Unidade de Informática da Universidade de Pau (Bayonne – França). LANCA apresenta as principais características dos agentes para um ambiente de aprendizagem à distância, bem como suas funções em ambientes distribuídos. Propõem, também uma arquitetura para o ambiente com a especificação dos papéis dos diferentes agentes inteligentes que compõem a sociedade.

A arquitetura proposta inclui quatro agentes cognitivos: *pedagogical agent*, *dialog agent*, *negotiating agent* e o *moderator agent*.

- ***Pedagogical Agent*** (Agente Pedagógico) – O papel deste agente é supervisionar a aprendizagem local. Ele fornece ao aprendiz a estratégia pedagógica para ajuda-lo em situações de resolução de problemas. Ele utiliza:
    - Um modelo do aprendiz , inicialmente, resultante das requisições endereçadas ao aprendiz, e progressivamente atualizadas. O modelo do aprendiz inclui várias informações, tais como: identificação pessoal, preferências de aprendizagem, eficiência de abordagens pedagógicas específicas ou estilos, performace alcançada em cada curso e nível de finalização;
    - Um ITS Process (analizador), que interpreta as ações de cada aprendiz e compara-os com um conjunto de soluções possíveis;
    - Diversas estratégias de aprendizagem, as quais podem ser ativadas pelo agente pedagógico ou pelo aprendiz.

O agente pedagógico detecta a fraqueza do aprendiz utilizando-se do analisador e o tipo de explicação, o qual estimula a descoberta da solução. Ele

grava as ações do aprendiz para fornecer uma ajuda futura para este tipo de aprendiz. E ele pesquisa novas lições pertinentes, controla o progresso do aprendiz e em caso de queda da performance, ele é capaz de mudar a estratégia de ensino.

- ***Dialog Agent*** (Agente Diálogo) – o papel do agente Diálogo é fornecer ajuda e explicação adicional ao aprendiz quando a a explicação do agente pedagógico é inadequada. Permite integração de diferentes explicações com diferentes modos de comunicação selecionadas de acordo com as necessidades do usuário. As principais funções deste agentes são:
    - No modo síncrono ele usa uma sala de chat para a comunicação com outros aprendizes via seu Agente Pedagógico. Diferentes pontos de visão e soluções podem ser comparados pelos aprendizes, contribuindo para resolver conflitos entre explicações contraditórias usadas pelsos ITS locais;
    - No modo Assíncrono ele acessa a base de conhecimento através de uma interface Web. A base de conhecimento inclui cursos, base de explicações, perfis de diferentes aprendizes e estratégias de aprendizagem, tais como: o book, o companion ,o tutor e o troublemaker.

- ***Negotiating Agent*** (Agente Negociador) – o papel do agente negociador esta baseado no principio que ajudas adicionais são solicitadas pelo aprendiz que pode perguntar ao Agente Companheiro ou a alguém, no caso da Web (através de e-mail) sobre um tópico específico. Entretanto, o tempo que o aprendiz gastaria pesquisando adequada reduziria o progresso no curso, então um agente específico (o Agente Negociador), representará o aprendiz.

- ***Moderator Agent*** (Agente Moderador) – o papel deste agente é avaliar e melhorar a funcionalidade do sistema. Ele objetiva:
    - Selecionar entre as diferentes explicações dadas para o aprendiz qual foi a mais eficiente(usando estatísticas), e então atualizar a base de conhecimento de explicações e o tipo de perfil do aprendiz;
    - Determinar o custo das explicações dos Agentes Negociadores.

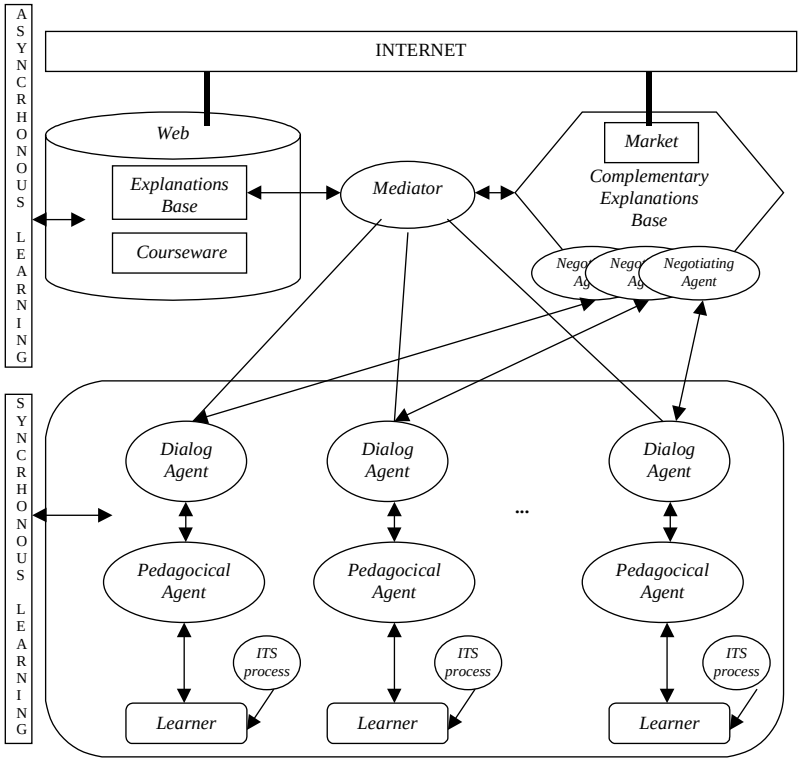

Figura 7: Arquitetura do sistema LANCA

Neste projeto são utilizadas quatro estratégias de ensino: o *book*, o tutor, o *companion* e o *troublemaker*.

- **O *Book*** (Livro) – é o conjunto de índices do curso, relacionado com o entendimento de conceitos específicos;
- **O *Tutor*** (Tutor) – é uma estratégia diretiva na qual o sistema dá respostas ou conselhos para as requisições do aprendiz;
- **O *Companion*** (Companheiro) – é um colega vitual capaz de discutir com o aprendiz, dando algumas dicas e perguntando algumas questões focando a atenção em um ponto específico;
- **O *Troublemaker*** (Criador de caso) – é um companheiro particular que envolve o aprendiz por propor soluções que são algumas vezes verdadeiras mas outras vezes erradas. Desta maneira faz o aprendiz pensar e defender seu ponto de vista, aumentando sua motivação.

3.4 Baguera

O projeto Baguera [WEB2001], foi desenvolvido pela Universidade de Grenoble (Grenoble – França), cujo objetivo é desenvolver uma fundamentação teórica e

metodológica para guiar a concepção e modelagem de ambientes de aprendizagem. A plataforma Baguera está fundamentada no princípio que a função educacional do sistema está nas interações organizadas entre os componentes: agentes e humanos e, não meramente na funcionalidade de uma de suas partes.

O primeiro resultado desse projeto inclui uma arquitetura multiagente baseada na Web para ambientes de aprendizagem e um protótipo para a aprendizagem de geometria.

A plataforma Baguera foi desenvolvida usando JatLite (http://java.stanford.edu/). Cada agente foi extendido por um módulo de interação que fornece suporte ao gerenciamento de protocolos entre os agentes. Os agentes possuem habilidade de comunicar-se com outros agentes, raciocinar e tomar decisões. A comunicação entre agentes está baseado na Teoria dos Atos de Fala , em adequação com o padrão FIPA-ACL (http://www.fipa.org/).

A arquitetura multiagente da plataforma Baguera foi concebida pela metodologia AEIO (Agent, Environment, Interactions, Organisation), uma metodologia para a análise e projeto orientados a agentes proposta por Demazeau [DEM1995]. Como resultado desse processo, estudantes e professores interagem com diferentes agentes. Cada estudante é apoiado por três agentes artificiais, que são:

- ***Companion Student*** (Companheiro do Estudante) – É um agente associado com a interface do estudante. Monitora as ações do estudante, notificando outros agentes quando necessário. Este agente controla o acesso a pasta eletronica do estudante e pode interagir com outros agentes: Agentes Companheiros Professores e Agentes Companheiros Estudantes. Dessas interações, o agente traz ao estudante informações sobre o ambiente e permite a comunicação com professores e outros estudantes conectados.
- ***Tutor Agent*** (Tutor) – Esse agente pode interagir com o Agente Mediador, Agente Assistente, Agentes Tutores e Agentes Companheiros. Pode acessar a pasta do estudante para retomar, adicionar exercícios e atualizar o histórico do estudante. Estes agentes são capazes de modelar as concepções do estudante.
- ***Mediator Agent*** (Agente Mediador) – O objetivo deste agente é enviar as soluções do estudante a um solucionador de problemas apropriado. Este agente implementa técnicas para analisar e apresentar provas e pode interagir com outros agentes.

Similarmente, cada professor é apoiado por dois agentes artificiais:

- ***Companion Teacher*** (Companheiro do Professor) – Este agente é associado com a interface do Professor. Ele traz para o usuário informações sobre o ambiente de aprendizagem. Este agente media a comunicação com outros humanos e agentes artificiais, como edição de novas atividades para os estudantes, distribuição de tais atividades ao estudantes e supervisão dos trabalhos feitos pelos estudantes. O Agente Companheiro Professor pode interagir com o Agente Assistente, Agente Tutor e qualquer Agente Companheiro.
- ***Assistant Agent*** (Agente Assistente) – Um agente assistente é também um tipo de agente pessoal, o qual tem como objetivo assistir o professor com a criação e distribuição de novas atividades, que são armazenadas na pasta eletronica do professor. Este agente controla o acesso a pasta eletronica do professor e pode interagir com o Agente Tutor, outros Agentes Assistentes e companheiros. Além disso, estes agentes são notificados de novas conecções de estudantes e professores pertencentes a classe dos professores.

O protótipo do ambiente para a ensinar prova de uma determinada geometria está baseado na plataforma multiagente apresentada. Os usuários podem ter acesso ao ambiente de aprendizagem através de qualquer browser que suporta Applets Java, acessando a seguinte URL: http://www-baghera.imag.fr.

A relação dos estudantes é apresentada, primeiramente, como mostrado na Figura 8. Desta janela os estudantes têm o acesso à sua classe, os exercícios a serem resolvidos e, na parte inferior da janela, os estudantes podem estabelecer um diálogo simples com seu agente pessoal. Uma vez que um estudante escolheu um exercício para trabalhar, EuclideJava (Figura 9) é iniciado.

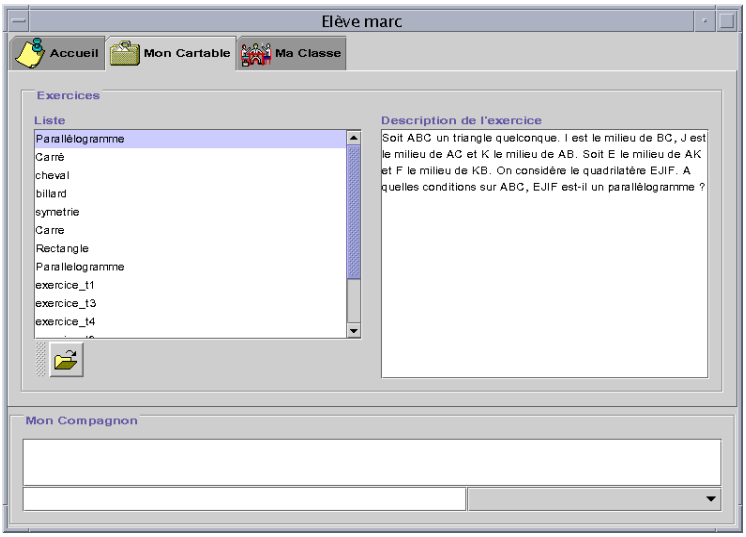

Figura 8: Interface geral do estudante no sistema Baguera

A Figura 9 mostra uma solução construída por um estudante. No alto da janela há um menu e a indicação do exercício. O resto da janela é dividido em dois lados. O lado esquerdo é usado para os estudantes construir a prova de geometria. Na parte direita um objeto geométrico é colocado para a manipulação geométrica. A manipulação geométrica é feita por Cabri-Cabri-Java (http://www.cabri.net/cabrijava/).

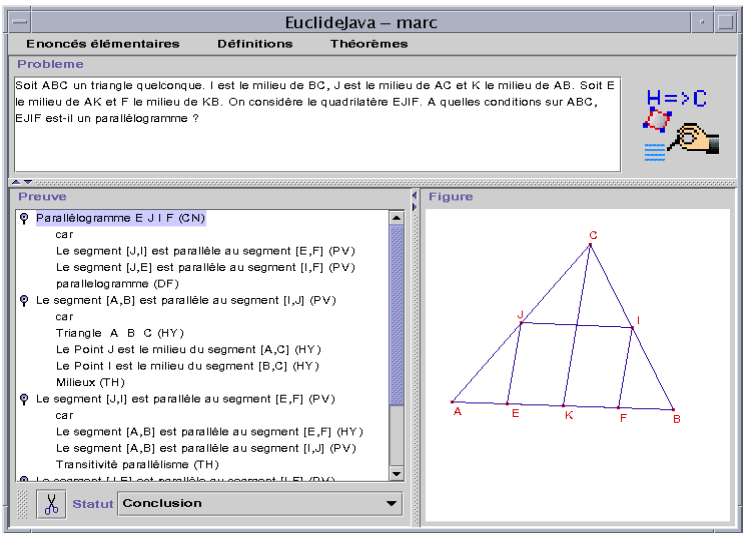

Figura 9: Interface de resolução de problemas no ambiente Baguera

A interface do professor é similar à interface do estudante. Funcionalidades extras da aplicação permitem ao professores inserir e distribuir problemas, supervisionar estudantes enquanto estão trabalhando e trocar mensagens. Estudantes e professores são dinamicamente organizados em classes virtuais conforme seu nível. Os membros (alunos) de uma mesma classe podem trocar mensagens.

3.5 I-Help

O I-Help [VAS2001] foi desenvolvido pela Universidade de Saskatchewan – Canadá), este projeto descreve uma infraestrutura multiagente para o I-Help, um ambiente de aprendizagem baseado na Web para auxiliar aprendizes na solução de problemas. O sistema contém uma variedade de recursos da aprendizagem, fóruns, materiais *on-line*, *chat*, etc.

Para ilustrar a funcionalidade do I-Help imaginemos o seguinte cenário. Um estudante tem uma questão e este delega a tarefa de encontrar ajuda a seu agente pessoal. O agente pessoal tenta encontrar um outro agente (agente de aplicação ou um outro agente pessoal) que ofereça os recursos da informação relacionados ao pedido da ajuda. Podem ser também recursos humanos (representado no sistema por seus agentes pessoais), isto é, os estudantes que estão online e são capazes de ajudar a solucionar a questão. A Figura 10 apresenta a interface gráfica do ambiente I-Help.

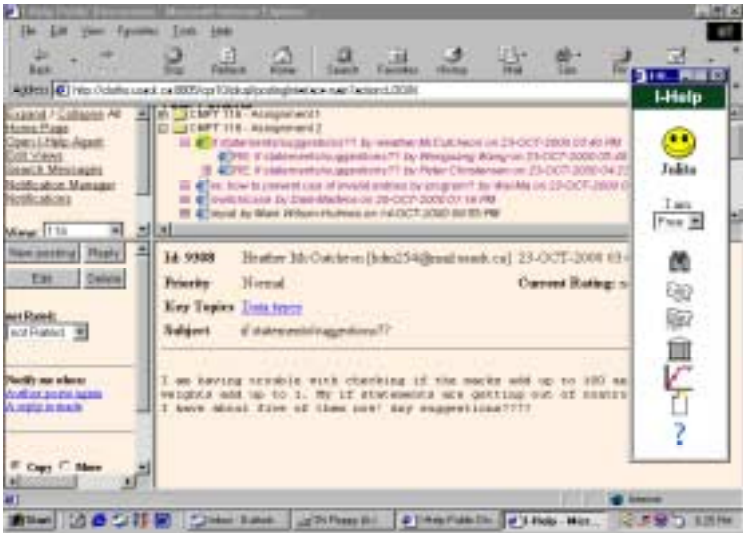

Figura 10: Interface do sistema I-Help

O sistema I-Help é baseado em uma arquitetura multiagente, consistindo em agentes pessoais (usuários e humanos) e em agentes de aplicação (Figura 11). Estes agentes usam uma ontologia e uma linguagem de comunicação comum. Cada agente controla recursos específicos do usuário (ou da aplicação) que ele representa, incluindo por exemplo, os conhecimentos do usuário sobre determinados conceitos, ou os materiais instrutivos que pertencem a uma aplicação. Os agentes usam seus recursos para conseguir os objetivos de seus usuários, seus próprios objetivos, e objetivos de outros agentes. Todos os agentes são autônomos.

Cada agente possui um modelo de seu usuário e de outros agentes que ele encontrou e negociou. Os agentes comunicam-se com outros agente e com Agentes *Matchmaker* (combinadores) para procurar por recursos apropriados para seus usuários, dependendo do tópico da ajuda requisitada. Se um recurso eletrônico for encontrado (representado por agentes de aplicação), o agente pessoal apropria-se do recurso e apresenta-o ao usuário em um browser. A ajuda é arranjada (negociada) inteiramente pelos agentes pessoais, livrando o aprendiz da necessidade de negociar e pensar sobre o "custo" da ajuda. Desta maneira os agentes pessoais negociam a ajuda de seus usuários em um "mercado virtual de ajuda".

Arquiteturas multiagentes envolvem vários níveis de organização, incluindo negociação entre os agentes. Desta maneira, consegue-se um sistema distribuído, multi-usuário, multiaplicação, adaptável e auto-organizado que suporta alocação de recursos de ajuda (outros usuários, aplicações e informações).

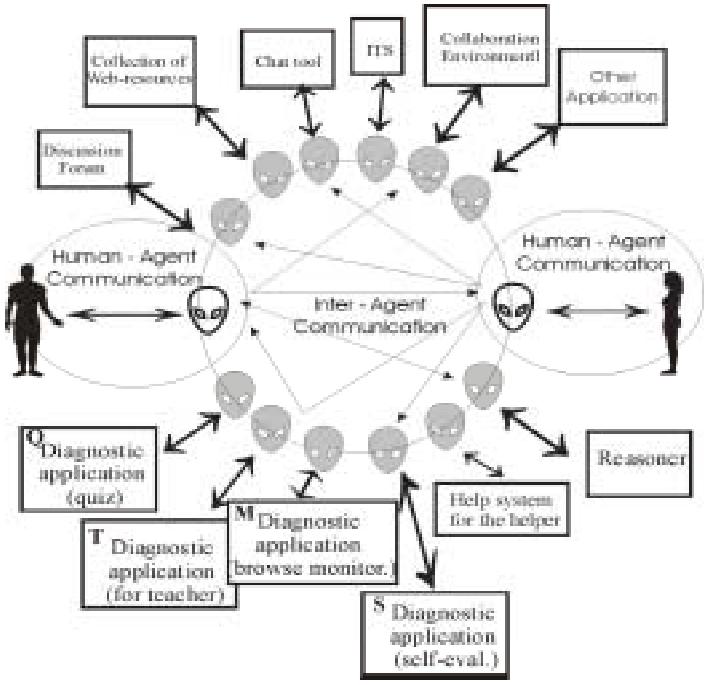

Figura 11: Arquitetura do sistema I-Help

3.6 The Explanation Agent

O Explanantion Agent ou Agente de Explicações de [ZOU2000], desenvolvido pelo Departamento de Informática e Pesquisa Operacional da Universidade de Montreal (Montreal – Canadá), tem como objetivo principal prover respostas ou explicações sobre o conteúdo com maior qualidade, identificando problemas que possam ocorrer

durante o processo de explicação ou resolução de problemas. Ele tem dois objetivos específicos: descobrir a fonte do mal entendimento do aprendiz através do modelo do estudante, e ajudar o projetista do curso a adaptar suas explicações de acordo com estas observações. É utilizado a teoria de Mapas Conceituais para estruturar as explicação em uma representação formal. Esta representação é usada pelo Agente Explicação para fazer suas deduções sobre conceitos mal entendidos pelo aprendiz.

O termo Agentes Pedagógicos, é usado para referir-se a agentes criados para ajudar pessoas iniciantes no processo de aprendizagem, por meio da interação com o aprendiz.

Agentes Pedagógicos exibem um conjunto de características: Eles podem adaptar suas interações à necessidades do aprendiz, e o estado corrente do ambiente. Eles podem colaborar com aprendizes e outros agentes e são capazes de providenciar um contínuo *feedback* ao aprendiz. E eles podem mostra-se como figuras vivas, introduzindo emoções e aspectos sociais em suas interações e ralações com o aprendiz.

A arquitetura do ambiente de aprendizagem é uma arquitetura cliente-servidor dividida em duas partes (Figura 12):

- Um editor no servidor onde o curso, exercícios e explicações são criadas;
- O ambiente de aprendizagem onde o Agente Explicação está envolvido;

O Editor do ambiente possui os seguintes componentes:

- **O Modelo de domínio**: é composto por todos os cursos e exercícios disponíveis. Cada curso pode ser representado numa forma de rede semântica. A menor unidade de dados na representação é chamada de conceito. Cada exercício é composto de um rótulo, possíveis respostas e um conjunto de erros pré-definidos;
- **O Agente Servidor**: O Agente Explicação é armazenado no servidor. Em cada sessão de aprendizagem ele é enviado para a máquina do aprendiz;
- **O Modelo do Estudante**: o modelo do Estudante é um modelo de perturbação e mais precisamente um modelo de *bug*. A técnica comum para implementar um modelo de perturbação é representar o conhecimento do especialista e então argumentar sua representação com o conhecimento explicito do provável conceito mal entendido. Neste sistema o projetista tem que elaborar uma biblioteca de *bugs* para cada exercício enumerando um conjunto de *bugs* que ele notou através de suas experiências;

- **A Base de Conhecimento**: é usada para armazenar cursos, exercícios e explicações;
- **O Editor de Explicações**: é usado para produzir explicações na forma de textos ou *web pages*. Estas páginas podem conter imagens, vídeos e simulações virtuais. Depois de projetar a explicação na forma de textos ou *web pages*, o projetista tem que decompo-los em conceitos chaves e links semânticos entre esses conceitos.

Cursos, exercícios e explicações são apresentados no ambiente de aprendizagem que é composto da seguinte forma:

- **Módulo Diálogo**: este módulo interage com o aprendiz através de uma sessão de aprendizagem: inicializando um diálogo, criando e atualizando o modelo do estudante e manipulando o processo de comunicação;
- **Agente Explicação**: na seção de aprendizagem, o Agente Explicação apresenta explicações adaptadas ao modelo do aprendiz. Ele envia informações relacionadas com a performance do aprendiz ao servidor e todas as informações necessárias para detectar as incompreenções do aprendiz.

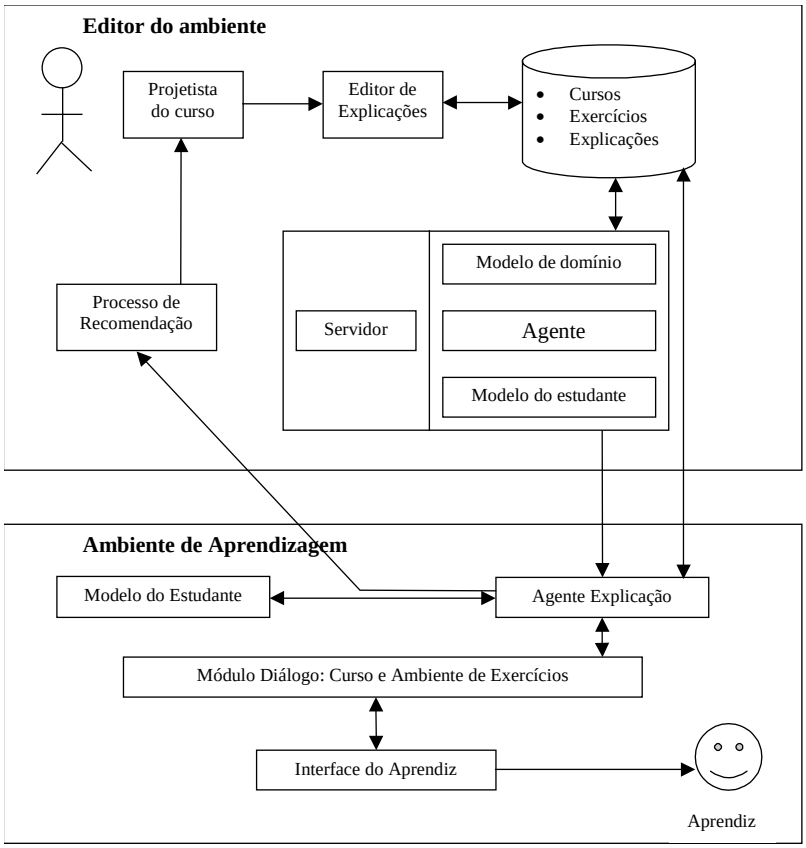

Figura 12: Arquitetura do ambiente Explanation Agent

A arquitetura do agente é dividida em camadas. Uma de cognição, outra de explicação e outra de percepção (Figura 13). Cada uma sintetizando uma ou várias das características do agente. As características do agente são: reatividade, mobilidade, autonomia, aprendizagem, dedução e capacidade de compreensão.

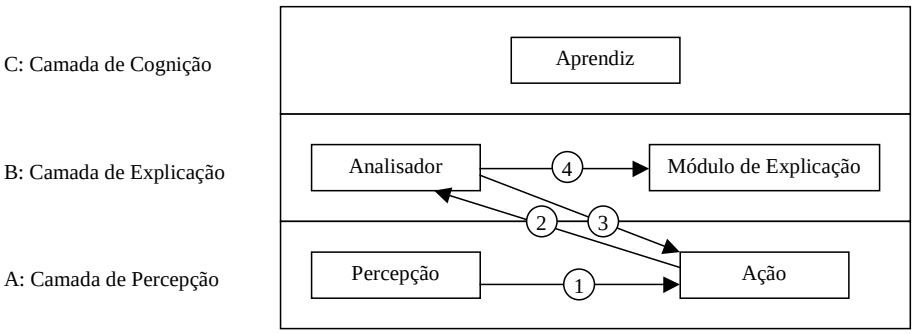

Figura 13: Arquitetura do Agente Explicação

3.7 AME-A

AME-A [DAM1997, DAM1998, PER2001] é um ambiente multiagente de ensino-aprendizagem, no qual se propõe o estudo e o desenvolvimento de um sistema educacional interativo para o ensino à distância. A proposta é o ensino genérico e adaptável às características psico-pedagógicas do aprendiz.

As características principais do sistema são a aprendizagem estática e a aprendizagem dinâmica. A aprendizagem estática corresponde a primeira interação do aprendiz com o ambiente, onde um agente modela o aprendiz conforme suas características afetivas, motivação e nível de conhecimento. A aprendizagem dinâmica ocorre durante a interação, quando é validado o modelo de aluno e estratégias pedagógicas em vigor.

Este ambiente utiliza a abordagem de sistemas multiagentes conforme mostra a Figura 14. Cada agente trabalha concorrentemente, realizando suas tarefas e trocando mensagens entre si, com o intuito de que o aprendiz atinja uma aprendizagem efetiva. As características psico-pedagógicas são relevantes para o ensino adaptado e viabilizam a apresentação do material instrucional de uma maneira individualizada.

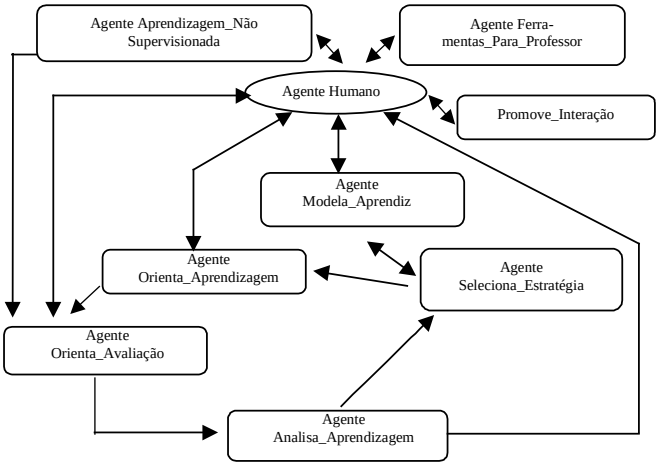

Figura 14: Arquitetura do ambiente AME-A

No AME-A [DAM1998a], cada agente é responsável por suas tarefas e agem continuamente no ambiente, com a finalidade de cooperar para promover uma aprendizagem inteligente e adaptável às características dos aprendizes.

A seguir são definidos todos os agentes envolvidos no ambiente [DAM1999]:

- **Agente Aprendizagem_Não_Supervisionada**: é responsável pela aprendizagem livre e sem supervisão quando o aprendiz assim desejar.
- **Agente Promove_Interação**: auxilia a interação aluno x aluno, aluno x professor e professor x professor.
- **Agente Ferramentas_Para_Professor**: o professor possui uma ferramenta para auxiliá-lo na integração do material no banco de dados do ambiente.O agente Ferramentas_Para_Professor tem como função orientar o professor e armazenar o material do curso. Inicialmente o professor cadastra-se no sistema, podendo incluir novos materiais, alterar e aperfeiçoar o material existente.
- **Agente Modela_Aprendiz**: O ambiente propõe-se a personalizar o ensino conforme um modelo de aluno gerado pelo agente Modela_Aprendiz. Para o modelo de aluno adotou-se a combinação de quatro pares de perfis estudados por Carl Jung (Extrovertido-Introvertido; Sensitivo-Intuitivo; Emocional-Racional; Perceptivo-Julgador), gerando 16 perfis psicológicos. Através de um questionário que é dado ao aprendiz no início do curso, o agente Modela_Aprendiz verifica em qual dos perfis o aluno se enquadra. Essas

informações que caracterizam o aluno são passadas para o agente Seleciona_Estratégia.

- **Agente Seleciona_Estratégia**: Para o perfil do aprendiz, foram definidos métodos de ensinar e características relevantes. Durante a interação do aluno no ambiente, o agente pode mudar o perfil do aprendiz, caso perceba alguma modificação em seu comportamento, e esta informação é passada ao agente Seleciona_Estratégia. O agente Seleciona_Estratégia muda a estratégia para um determinado aluno quando o agente Modela_Aprendiz, ou o agente Analisa_Aprendizagem, informarem alguma alteração na aprendizagem, motivação e personalidade do aluno. Neste trabalho o agente Seleciona_Estratégia, seleciona uma estratégia híbrida com base em características do aluno e características das estratégias que se enquadram para um determinado perfil de aluno.
- **Agente Orienta_Aprendizagem**: busca o endereço no banco de dados e apresenta o material, conforme plano de ensino, isto é, a seqüência de ações (táticas) geradas pelo agente Seleciona_Estratégia. O material de ensino, também denominado domínio do conhecimento (conceitos gerais, conceitos mais abrangentes, exemplos, mensagens, alertas, gráficos, tabelas, etc.), está armazenado sob a forma de diferentes mídias (vídeo, texto, gráfico, áudio, etc.), as quais são utilizadas para a sua apresentação. No banco de dados encontram-se os endereços das páginas conforme uma estrutura pré-definida, a qual retrata o plano de ensino com todas as possibilidades de estratégias, onde o professor disponibilizou através do agente Ferramentas_Para_Professor.
- **Agente Orienta_Avaliação**: informa ao agente Orienta_Avaliação a sessão atual, informações do aprendiz e objetivo em curso. Com essas informações, o agente Orienta_Avaliação busca e apresenta o material de avaliação correspondente.
- **Agente Analisa_Aprendizagem**: analisa e verifica a aprendizagem durante a interação do aprendiz, e informa ao agente Seleciona_Estratégia. Caso o aprendiz não estiver aprendendo, o agente seleciona outra estratégia de ensino.

A análise deste ambiente mostrou a possibilidade de utilização de múltiplas estratégias de ensino, selecionadas em função de parâmetros que o agente seleção_estratégia recebe de outros agentes.

Na implementação deste ambiente, usou-se a linguagem Java para manter a independência de plataforma e o acesso através do WWW. Para armazenar o conhecimento, um banco de dados foi definido em um servidor, permitindo o acesso dos diversos seguimentos da nossa comunidade.

3.8 Eletrotutor

O Eletrotutor III é a 3º versão do STI Eletrotutor proposto por Silveira [SIL1992] em sua dissertação de mestrado. Foi seguido do Eletrotutor II uma versão para a Web onde não funcionava como um STI e sim apenas um tutorial na Web.

Esta 3º versão foi desenvolvida no Instituto de Informática da UFRGS, por [BIC1998, BIC1998a]. O Eletrotutor III implementa um ambiente distribuído de ensino-aprendizagem inteligente (*Intelligent Learning Environment* - ILE) baseado em uma arquitetura multiagente, na qual os agentes possuem as seguintes características: perceber dinamicamente as condições do ambiente; tomar decisões para afetar condições do ambiente; interpretar percepções, resolver problemas, extrair inferências e determinar ações. O ambiente Eletrotutor aborda o conteúdo constituído por alguns capítulos de Eletrodinâmica, um capítulo da Física que estuda alguns fenômenos da Eletricidade e aborda as relações entre algumas grandezas elétricas como Corrente Elétrica, Tensão, ou Diferença de Potencial, Resistência e Potência Elétrica.

A sociedade de agentes é composta por sete agentes, cada um deles possui uma função específica e como objetivo principal tem-se o aprendizado do aluno.

Neste ambiente é de vital importância a coordenação do comportamento dos agentes e da maneira pela qual eles compartilham seus conhecimentos, objetivos, habilidades e seus planos para, em conjunto, tomar as ações necessárias para solucionar um problema. Para que diferentes agentes autônomos possam cooperar mutuamente a fim de atingirem seus objetivos é necessário que a sociedade possua organização (arquitetura) e comunicação. A organização diz respeito à natureza e à função da sociedade e de seus elementos constituintes e a comunicação é o principal instrumento que os agentes utilizam para desenvolver a coordenação de suas ações [SIL1999].

A fim de poder atuar sobre o ambiente, cada agente possui uma representação interna parcial do mundo que o rodeia. Para isso, empregou-se a metáfora de estados

mentais para modelar a base de conhecimento que representa os estados do ambiente onde o agente está inserido.

A sociedade de agentes proposta na terceira versão do Eletrotutor contém agentes autônomos que comunicam-se uns com outros, cada agente possui funções e objetivos dentro de sua especialidade. A Figura 15 representa a arquitetura elaborada neste trabalho, baseada em [SIL1998].

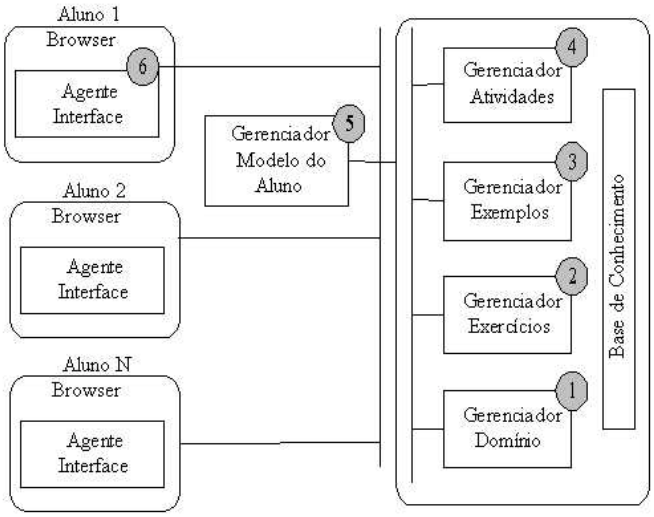

Figura 15: Arquitetura do ambiente Eletrotutor III

O ambiente de aprendizado inteligente proposto, seguindo a Figura 15 contém um agente responsável pela recuperação do conhecimento do domínio sobre cada ponto a ser apresentado ao aluno, agentes responsáveis pela tarefa de propor exercícios e avaliação de respostas, exemplos ao aluno e atividades extras.

As características dos agentes do ambiente Eletrotutor III são definidas nas seções a seguir:

- **Agente Gerenciador Domínio**: O agente Gerenciador Domínio possui a função de recuperar informações referentes ao domínio sobre cada ponto a ser apresentado ao aluno. Ele pode trocar mensagens com os agentes Gerenciador Modelo do Aluno e Agente Interface. Como características básicas, temos: colaboração (autonomia, habilidade social, cooperação), orientado por objetivo e possui continuidade temporal;
- **Agente Gerenciador de Exercício**: O agente Gerenciador Exercício possui a tarefa de propor exercícios e avaliar respostas do aluno. A troca de mensagens pode ser feita com o agente Gerenciador Modelo do Aluno ou

com o Agente Interface. Possui como características básicas a colaboração, adaptabilidade e continuidade temporal;

- **Agente Gerenciador de Exemplo**: O agente Gerenciador Exemplo é responsável pela tarefa de propor exemplos ao aluno. Pode trocar mensagens com os agentes Gerenciador Modelo do Aluno e Agente Interface. Como características básicas o agente Gerenciador Exemplo é colaborador, adaptável e continuidade temporal;
- **Agente Gerenciador de Atividades**: O agente Gerenciador Atividade é responsável pela tarefa de propor atividades extras ao aluno. Pode trocar mensagens com os agentes Gerenciador Modelo do Aluno e Agente Interface. Suas características básicas são: colaboração, adaptabilidade e continuidade temporal;
- **Agente Gerenciador do Modelo do Aluno**: O agente Gerenciador Modelo do Aluno é responsável por construir e manter uma base de conhecimento que modele o estado cognitivo dos alunos que estejam conectados ou que tenham estado conectados ao sistema. Quando a sessão tutorial inicia o ele recebe informação do Agente Interface , sobre a identificação do aluno, para que possa recuperar o Modelo do Aluno correspondente e conforme o andamento das lições decidir qual a estratégia de ensino a ser utilizada com este aluno. Possui como características a colaboração, adaptabilidade, racionalidade e continuidade temporal;
- **Agente Interface**: O Agente Interface possui como função o controle do browser no ambiente do aluno. As mensagens são recebidas e enviadas a todos os agentes através do agente Gerenciador de Comunicação. Este agente não possui inteligência, apenas repassa as atividades do aluno na interface do sistema. Como características, possui reatividade e autonomia;
- **Agente Gerenciador de Comunicação**: O agente Gerenciador de Comunicação é responsável pela comunicação do agente Interface com os demais agentes. Recebe e envia mensagens aos agentes Gerenciador Modelo do Aluno e Agente Interface, conforme necessidades do Gerenciador de Domínio, não trata mensagens, só as repassa. Não possui nenhuma inteligência. Como características, possui reatividade, habilidade social, autonomia e continuidade temporal.

Com relação a implementação, o ambiente foi desenvolvido na linguagem Java, com o objetivo de manter a independência entre plataformas e o acesso através da Internet. Os agentes trocam mensagens através do *Remote Method Invocation* (RMI). As mensagens trocadas seguem um protocolo baseado na linguagem KQML. A base de conhecimento está armazenada em um banco de dados relacional, o Oracle versão 8.0.4.

3.9 Um modelo computacional baseado na teoria de Vygotsky

Este trabalho está sendo desenvolvido pelo PGIE-UFRGS, PPGC-UFRGS e FACIN-PUCRS, o qual descreve uma proposta de um framework para uso da Tecnologia da Informação na educação. Esta baseado na teoria socio-cultural interacionista de Vygotsky e é projetado como uma sociedade multiagente para suportar a aprendizagem à distância.

O objetivo deste trabalho é propor um ambiente que privilegie a colaboração como forma de interação social através do uso de linguagens, símbolos e sinais, [AND2001]. Para suportar essa aprendizagem colaborativa, é apresentado uma sociedade formada pelos seguintes agentes artificiais: Agentes ZPD (*Zone of Proximal Development*), Agente Mediador, Agente Semiótico e Agente Social.

Para suportar o modelo coletivo de aprendizagem à distância, foi utilizada a teoria formulada por Vygotsky, como base da fundamentação teórica da proposta. Um importante conceito nesta teoria é que as atividades mentais são baseadas em relacionamentos sociais entre o indivíduo e o ambiente e este relacionamento é mediado por um sistema simbólico.

Um outro fundamental conceito nesta teoria é a *Zone of Proximal Development* (ZPD) – Zona de Desenvolvimento Próximo, onde o nível de desenvolvimento do aluno: o *Level of Real Development* (LRD) que refere-se as funções que o usuário já processou e o *Level Potencial Development* (LPD) que determina as funções que ele pode desenvolver.

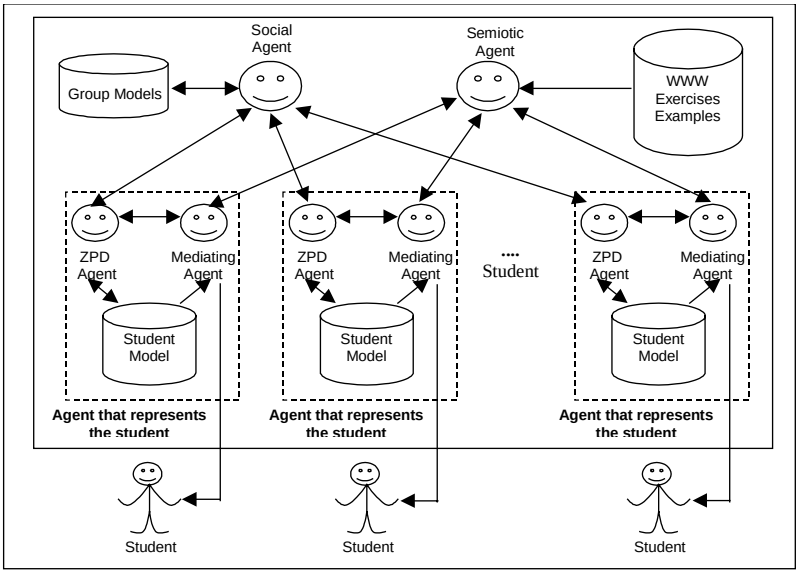

Figura 16: Arquitetura do ambiente proposto por [AND2001]

O modelo pedagógico desta pesquisa esta baseado numa forma colaborativa de aprendizagem que é alcançado através da interação social. As interações podem ser de vários tipos, considerando critérios como temporalidade, número de participantes, reciprocidade, hierarquias e até critérios baseados em comportamentos: personalidade, motivação, estado emocional, etc.

O sistema é composto por quatro tipos de agentes artificiais: o Agente ZPD, o Agente Mediador, o Agente Social e o Agente Semióta, bem como agentes humanos (Estudantes ou aprendizes).

- ***ZPD Agent*** (Agente ZPD) – são resposáveis por observar o desenvolvimento real do estudante e propõe atividades que tornaria suas capacidades reais. É resposável por estimular aquelas funções que ainda não estão maturadas, mas estão no processo de desenvolvimento. Este agente pode ter funções como: variar o grau de controle das atividades, considerar tarefas gradualmente, ou modificar as formas de ajuda e/ou suporte. Para auxiliar no processo de aprendizagem , um ZPD deve ter um modelo do estudante, identificando suas habilidades e deficiências, o qual será construído pela observação das interações do estudante.
- ***Mediating Agent*** (Agente Mediador) – É responsável pela interface entre o sistema e o estudante. Além da função de mediar a interação entre o estudante com o ZPD, o Agente Mediador deve ter acesso ao modelo do estudante ajudando a predição do comportamento do estudante o qual

permite detrminar as melhores ações para serem executadas para auxiliar o processo de aprendizagem do estudante.

- ***Semiotic Agent*** (Agente Semiótico) – Para o Agente Mediador completar seu papel, é necessário a iNtervenção de uma estimulação externa (sinais) para o estudante. O Agente Semiótico auxilia a atividade cognitiva do estudante introduzindo esses elementos, por exemplo: figuras, sons, textos, números, etc.
- ***Social Agent*** (Agente Social) – O Agente Social conhece todos os Agentes ZPD da sociedade e também tem conhecimento da presença de Agentes Sociais. Sua função é estabelecer a integração da sociedade e construir modelos de grupos de estudantes. Uma de suas atividade de Agente social é investigar a existÊncia de estudantes que tenham o conhecimento necessário, crenças e tipos de personalidade que seriam melhor apropriados para uma cooperação entre os estudantes.
- ***Human agents*** (Agentes Humanos) – Os agentes humanos são vistos como agentes que estabelecem relacionamento social com cada um conforme suas características pessoais e personalidade. Então é importante que a característica pessoa e a personalidade do estudante estejam contidas no modelo, estes traços afetaram a interação diretamente através dos papéis que cada estudante assumirá. Estes papéis determinam a afeição que acontecerá no grupo de estudantes.

O ambiente será implementado em Java e utiliza KQML para o processo de comunicação.

Este trabalho envolve vários projetos menores relacionados a teses de doutorado (PGIE e PGCC/UFRGS) e mestrado (PGCC/UFRGS). Sendo que um deles encontramos a pesquisadora do GIE-FACIN/PUCRS, Profa. Adja Ferreira, autora da proposta.

O trabalho de coordenação das contribuições individuais deverá impactar e modificar a arquitetura aqui apresentada. Apesar de ser um trabalho inicial nos trouxe as contribuições necessárias para o tipo de trabalho que se pretende neste estudo.

4. Estudo Comparativo sobre as arquiteturas selecionadas

Nesta seção apresentamos o resultado do estudo comparativo dos ambientes identificados na seção anterior.

Os resultados foram organizados em um quadro comparativo onde são destacados o conjunto de aspectos considerados relevantes para nosso estudo. A fim de se identificar o conjunto de requisitos utilizados pelos autores para modelar/implementar os ambientes inteligentes Web que utilizam a tecnologia de agentes.

O preenchimento do quadro segue a seguinte convenção: cada célula possui uma descrição de como o STI atende este critério (por exemplo, "KQML" para o critério "Tipo de comunicação entre os agentes"). No caso do critério Atividades dos Agentes (Tipo, Nome e atividade) inserimos os símbolos: ① e/ou ②, para indicar o tipo de agente, de acordo com as características que apresentam em relação às atividades que desempenham no ambiente de ensino-aprendizagem. O símbolo ① identifica que o agente em questão é do tipo Executor de Tarefas e o símbolo ② identifica que o agente é tipo Assistente. Esta classificação será detalhada e justificada na seção 4.1.

Utilizamos (?), para indicar, simplificadamente, a dúvida dos autores deste texto (autor e orientadora) em relação ao posicionamento do STI em relação à um determinado aspecto, e também pelo fato de não conseguirmos, um a partir das informações disponíveis nos textos e/ou desenhos, se as conclusões que chegamos seriam corretas ou não. Em alguns casos colocamos nossa interpretação a partir das informações disponíveis, por julgarmos que o conjunto de indicadores nos dava um alto grau de certeza.

Após a apresentação dos quadros que mostram as funcionalidades identificadas nos ambientes, passaremos a explicar cada um cada um dos itens considerados para compor o estudo comparativo. Os itens que foram considerados relevantes para esse estudo são:

- **Objetivo**: visa identificar o objetivo do ambiente;
- **Domínio** (Conteúdo): visa identificar o conteúdo a ser trabalhado no sistema;
- **Quantidade, nome e atividades dos agentes**: visa identificar o número de agentes associados e respectivas atividades dentro da sociedade multiagente;

- **Tipo de comunicação**: visa identificar se existe ou não um padrão utilizado para a comunicação entre os agentes;
- **Linguagem e/ou Ferramenta de Implementação**: visa identificar o tipo de ambiente utilizado para implementar os ambientes;
- **Tipo de arquitetura SMA utilizada**: visa identificar se os autores utilizaram ou explicitaram o tipo de arquitetura para a sociedade de agentes que está relacionada com a proposta do ambiente;
- **Estratégias de Ensino utilizada**: visa identificar o tipo de estratégias de ensino utilizada para auxiliar a promover a aprendizagem do conteúdo (domínio) nos usuários dos ambientes;
- **Tipo de Modelo do Aluno**: visa identificar o tipo de modelo do aluno criado/desenvolvido a partir das interações do sistema com o usuário em função dos modelos encontrados na literatura de STI;
- **Interface Gráfica**: visa identificar se o sistema utiliza ou não interfaces gráficas e o grau de adaptabilidade do sistema que ela reflete (exibe) para o usuário;
- **Ferramentas auxiliares**: visa identificar quais as ferramentas auxiliares mais utilizadas em ambientes de ensino-aprendizagem.

Quadro 1: Funcionalidades identificadas nos ambientes

Nome / tens	White Rabbit	LeCS	AME-A	Lanca	Baguera
Objetivo	Aumentar a cooperação entre um grupo de pessoas pela análise de suas conversações.	Dar suporte à aprendizagem colaborativa através da WWW.	Propõe-se ao ensino genérico e adaptável às características psico-pedagógicas do aprendiz	Expor que agentes inteligentes em STI podem ser adaptados para aprendizagem à distância	Desenvolver um fundamentação teórica metodológica para guiar concepção e modelagem d ambientes de aprendizagem.
Domínio (Conteúdo)	Independente de domínio	Independente de domínio	Independente de domínio	Independente de domínio	Independente de domínio
Quantidade de agentes	2	3	8	4	5
Atividades dos agentes Tipo, nome e atividade)	① Agente Pessoal: responsável por obter informações dos alunos; gerenciar a interface gráfica, etc. ② Agente Mediador: facilita a comunicação entre os agentes pessoais; gerencia o processo de *clustering* para a construção do modelo do aluno, etc.	① Agente Interface: armazena as informações obtidas pela interação com a interface do ambiente; realiza intervenções sobre o tempo e participação, etc. ② Agente Informação: lida com o domínio e o conhecimento pedagógico. ② Agente Conselheiro: realiza intervenções quando um mal entendimento do aluno é percebido.	② Agente Aprendizagem Não_Supervisionada: gerencia a aprendizagem livre (sem supervisão). ① Agente Promove Interação: auxilia na interação do participantes (Aluno e professores). ② Agente Ferramentas Para Professor: orientar o professor e armazenar o material do curso. ② Agente Modela Aprediz: personaliza o ensino conforme o modelo do aluno. ② Agente Seleciona Estratégia: seleciona a estratégia apropriada conforme o perfil do aluno. ① Agente Orienta Aprendizagem: busca e apresenta o material de ensino. ② Agente Orienta Avaliação: auxilia no processo de avaliação. ② Agente Analisa Aprendizagem: analisa e verifica a aprendizagem durante a interação.	② Agente Pedagógico: supervisionar a aprendizagem. ② Agente Diálogo: fornece ajuda e explicação ao aluno. ① Agente Negociador: negocia informações na Web com outros agentes. ① Agente Moderador: avaliar e melhorar a funcionalidade do sistema.	① Agente Companheiro d Estudante: monitora as ações d estudante, etc. ② Agente Tutor: modela a ações do aluno, etc. ② Agente Mediador intermedia as soluções do alun com um Solucionado apropriado, etc. ① Agente Companheiro d Professor: interface do sistem com o professor, auxilia professor sobre o processo d aprendizagem, etc. ① Agente Assistente: agent pessoal do professor, auxilia n distribuição de novas atividades etc.

Quadro 2: Funcionalidades identificadas nos ambientes (continuação)

Nome / tens	White Rabbit	LeCS	AME-A	Lanca	Baguera
Tipo de comunicação entre os agentes	?	KQML	?	?	KQML
Linguagem e/ou Ferramenta de desenvolvimento	Supostamente Java, JavaScript e HTML	Delphi	Java	?	JATLite
Tipo de arquitetura/sociedade SMA utilizada	Sociedade heterogênea e aberta	Sistema Federativo de Agentes	Sociedade heterogênea e fechada	Sociedade heterogênea e fechada	?
Estratégias de Ensino utilizada	Múltiplas estratégias de ensino	Múltiplas estratégias de ensino	Múltiplas estratégias de ensino	Múltiplas estratégias de ensino	?
Tipo de Modelo do Aluno	Rede neural	?	Modelos pré-definidos, 4 pares de modelos: Extrovertido – Introvertido; Sensitivo – Intuitivo; Emocional - Racional; Perceptivo – Julgador, gerando 16 perfis psicológicos, classificados por uma Rede Neural.	?	?
Interface Gráfica	Gráfica interativa	Gráfica interativa	Gráfica interativa	Gráfica interativa	Gráfica interativa
Ferramentas auxiliares	*Browser* e *chat*	*Browser*, *chat* e editor de texto.	?	*Browser* e *chat*	*Browser*

Quadro 3: Funcionalidades identificadas nos ambientes

Nome / tens	*Explanation Agent*	Eletrotutor	I-Help	Modelo computacional de Vygotsky
Objetivo	Prover respostas ou explicações sobre o conteúdo com maior qualidade, identificando problemas que possam ocorrer durante o processo de explicação ou resolução de problemas.	Desenvolver um instrumento para verificar a eficácia do uso de diferentes abordagens de ambientes de ensino por computador na escola	Auxiliar estudantes na solução de problemas através da Web	Propor um ambiente que privilegie a colaboração como forma de interação social através do uso de linguagens, símbolos e sinais.
Domínio (Conteúdo)	Independente de domínio	Física (Eletrodinâminca)	Independente de domínio	Independente de domínio
Quantidade de agentes	1 para cada usuário do sistema	7	3	5
Atividades dos agentes (Nome e função)	① ② Agente Explicação: apresenta explicações adaptadas ao modelo do aprendiz.	① Agente Gerenciador Domínio: recupera informações referentes ao domínio, etc. ② Agente Gerenciador de Exercício: propõe exercícios e avalia respostas. ① Agente Gerenciador de Exemplo: apresenta exemplos ao aluno. ② Agente Gerenciador de Atividades: propõe atividades extras ao aluno. ② Agente Gerenciador do Modelo do Aluno: constrói e mantém o modelo do estado cognitivo do aluno. ① Agente Interface: controla a interface do ambiente. ① Agente Gerenciador de Comunicação: gerencia a interface do ambiente com os demais agentes.	① ② Agente Pessoal: controla recursos específicos dos usuários (alunos ou professores) ① ② Agente de Aplicação: controla recursos específicos das aplicações.	② Agente ZPD: responsável por observar o desenvolvimento e propor atividades. ① Agente Mediador: é responsável pela interface entre o sistema e o estudante. ② Agente Social: estabelece a integração da sociedade e constrói modelos de grupos de estudantes. ② Agente Semiótico: auxilia na atividade cognitiva do estudante. ① Agente Humano: estabelece relacionamento social com cada agente conforme suas características pessoais.

Quadro 4: Funcionalidades identificadas nos ambientes (continuação)

Nome / tens	*Explanation Agent*	Eletrotutor	I-Help	Modelo computacional de Vygotsky
Tipo de comunicação entre os agentes	?	KQML	KQML	KQML
Linguagem e/ou Ferramenta De desenvolvimento	?	Java	?	Java
Tipo de arquitetura/sociedade SMA utilizada	Sociedade heterogênea e aberta	Sistema Federativo, sociedade heterogênea e fechada	Sociedade heterogênea, aberta e baseada em leis.	?
Estratégias de Ensino utilizada	Múltiplas estratégias de ensino	Múltiplas estratégias de ensino	Múltiplas estratégias de ensino	Múltiplas estratégias de ensino
Tipo de Modelo do Aluno	Modelo de Pertubação (Buggy)	Modelo de Pertubação (Buggy)	?	?
Interface Gráfica	Gráfica interativa	Gráfica interativa	Gráfica interativa	Gráfica interativa
Ferramentas auxiliares	?	*Browser*	*Browser*, fórnuns e *chat*	?

4.1 Resultado da análise comparativa

O estudo apresentado neste trabalho foi resultado da avaliação de alguns STI existentes selecionados com os critérios apresentados na introdução deste trabalho. Esse estudo nos permitiu levantar requisitos e funcionalidades necessários para a definição de aspectos relevantes para o projeto de STI que utilizam a tecnologia de agentes para sua concepção. A seguir apresentaremos a análise comentada dos itens discutidos na seção anterior.

Quanto ao objetivo, a maioria dos ambientes na Web, não apresentam uma proposta para um conteúdo específico, ou seja, suas propostas são independentes do domínio/conteúdo.

Nos STI tradicionais, os domínios modelados eram essencialmente domínios lógicos, bem estruturados e muito restritos, tais como: ensino de Matemática, Física, Linguagens de Programação, etc.

Com a incorporação da tecnologia de Agentes no projeto e desenvolvimento de STI, verificou-se que o conteúdo do domínio passou a ser variado, ou seja, o conteúdo passou a ser modelado independentemente do domínio.

Observa-se que a quantidade, atividades e até mesmo a denominação dada para os agentes é variada. No entanto podemos agrupá-los em dois grandes conjuntos.

Segundo Vassileva [VAS1997; VAS2001], os agentes quando desempenham tarefas e/ou atividades em aplicações voltadas para suporte ao processo de ensino-aprendizagem são denominados de Agentes Pedagógicos.

Afim de melhor auxiliar o tipo de atividade dos Agentes Pedagógicos nos ambientes analisados, caracterizamos um conjunto de possíveis funcionalidades ou atividades que os agentes podem desempenhar num ambiente dessa natureza (Aplicação Educacional), de acordo com as atividades comuns apresentadas pelos agentes nos ambientes analisados (ver critérios técnicos nos quadros 1 e 2). A Figura 17 apresenta a subdivisão das atividades de um Agente Pedagógico.

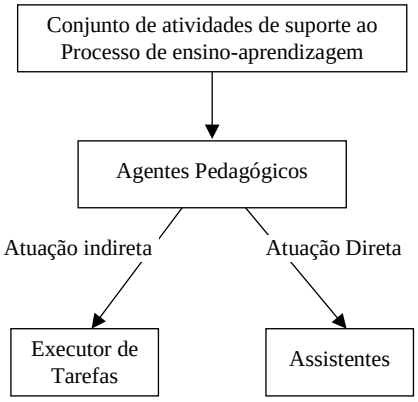

Figura 17: Subdivisão para um Agente Pedagógico no processo de ensino-aprendizagem.

De acordo com a Figura 17, podemos observar que conjunto de atividades que dão suporte ao processo de ensino-aprendizagem dos Agentes Pedagógicos, pode ser de duas formas distintas de atuação: atuação direta e atuação indireta. Na atuação direta, o processo se dá pelas seguintes tarefas: coleta de informações (especialmente na Web); enviar mensagens para outros agentes ou para o aluno; monitorar a interface para identificar o tipo de atividade que o aluno está desempenhando; etc. Todas essas atividades são realizadas pelo Executor de Tarefas.

É importante salientar que o papel do Executor de Tarefas não está relacionado diretamente com a parte pedagógica, ou seja, a seleção e adoção de estratégias do estudante.

Já na atuação indireta, o processo é realizado, pelo que denominamos de Assistentes, cujas tarefas são: intermediar e/ou analisar o fluxo de informações entre os estudantes; selecionar informações/materiais; relacionar e determinar estratégias de ensino para os estudantes e realizar o processo de modelagem do aluno. O que se observou é que os agentes do tipo Assistentes na realidade executam as tarefas do módulo tutor, na arquitetura clássica de STI.

A troca da palavra “Tutor” para “Assistente” se justifica pela troca de paradigma educacional. Lembrar que nas décadas de 70 e 80, do século XX, o paradigma utilizado era comportamentalista. A difusão e adoção do paradigma construtivista, a partir da década de 90, do mesmo século, mudou a concepção do papel do professor.

Isto pode ser claramente observado nas propostas mais modernas, com a referência ao Agente Assistente (Mediador, Moderador, Gerenciador, Negociados, Facilitador, etc).

O Agente Executor de Tarefas pode buscar informações, executar instruções solicitadas pelo Agente Assistente. Basicamente, ele faz o papel de auxiliar do Agente Assistente, desempenhando tarefas que são importantes para a tomada de decisão ou execução do (s) plano (s) adotado pelo Agente Assistente.

Ambientes que possuem mais de um agente, praticamente exigem troca de informações. Neste sentido, torna-se claro a necessidade de uma Linguagem de Comunicação comum. A linguagem de comunicação entre os agentes, utilizada na grande maioria dos ambientes estudados foi a linguagem KQML.

A Linguagem de Implementação utilizada determina a portabilidade, performance, bem como recursos audiovisuais que podem ser utilizados no sistema. Dentre os STI citados, os ambientes portáveis são White Rabit, AME-A, Eletrotutor III, e o Modelo Computacional proposto por [AND2001], que foram desenvolvidos em Java (Ver critérios técnicos no Quadro 1) e que, portanto, podem ser executados em qualquer navegador World Wide Web (browser) ou sistema operacional que possua suporte à Java. O ambiente LeCS foi desenvolvido utilizando a linguagem Delphi. Já o ambiente Baguera foi desenvolvido com a Ferramenta JATLite, a qual possui um conjunto de programas Java, que possibilitam a criação rápida de agentes que podem comunicar-se pela Internet.

Segundo [OLI1996], existem três critérios para classificação de uma sociedade de agentes:

a) Quanto ao tipo de agentes:

- Sociedades homogêneas: os agentes são todos do mesmo tipo, ou seja, possuem arquiteturas idênticas;
- Sociedades heterogêneas: existem agentes de diferentes tipos na sociedade.

b) Quanto à migração de agentes:

- Sociedades fechadas: há um número fixo e único de agentes na sociedade;
- Sociedades abertas: o número de agentes nesta sociedade pode variar, pois podem entrar novos agentes ou sair agentes da sociedade.

c) Quanto à presença de regras de comportamento:

- Sociedades baseadas em leis: existem regras que determinam o comportamento dos agentes;
- Sociedades sem lei: quando não há regras para reger os agentes da sociedade.

A maioria dos trabalhos analisados não descrevem a forma como esta organizada e/ou estruturada a sociedade ou a arquitetura do sistema. Entretanto, podemos supor que

vários ambientes apresentam características em que sua sociedade é considerada uma sociedade heterogênea e fechada e possuem regras de comportamento para seus agentes.

Arquitetura de agentes refere-se ao modo de organização dos agentes dentro de um sistema e como estão estruturados seus relacionamentos e interações. Assim como existem diversas arquiteturas de software, o mesmo ocorre com relação as arquiteturas de agentes, as quais possuem certas características que permitem a avaliação de sua qualidade e eficácia.

Para exemplificar, tomemos como exemplo a arquitetura de Sistema Federado, ou Federativo. Conforme [THI1999], a interação dos agentes é assistida por um programa especial denominado Facilitador, o qual oferece um conjunto de serviços de coordenação. A Figura 18 apresenta uma arquitetura baseada em um Sistema Federativo.

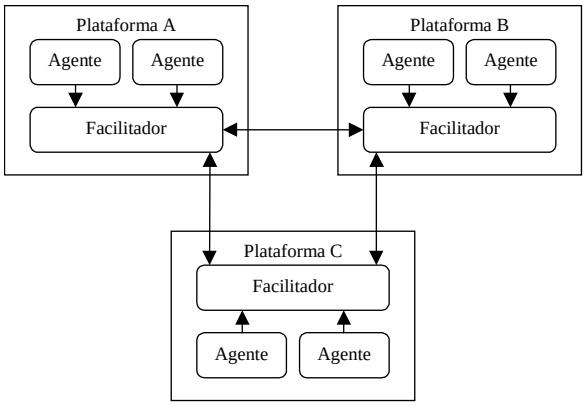

Figura 18: Arquitetura de um Sistema Federativo

O Facilitador atua como um mediador, roteando mensagens (solicitações e respostas) de acordo com seu conhecimento interno. Os ambientes que adotaram o Sistema Federativo para modelar sua arquitetura foram, o ambiente LeCS e o Eletrotutor.

Uma outra arquitetura bastante conhecida e utilizada é a arquitetura *Blackboard* (Quadro negro). Conforme [JAQ1999], *Blackboard* é uma estrutura única e compartilhada entre vários agentes, onde as informações serão escritas e lidas durante o desenvolvimento das tarefas. Como não há comunicação direta entre os agentes, eles devem consultar a estrutura de tempos em tempos para verificar se existe alguma informação destinada a eles. O esquema da estrutura *Blackboard* pode ser visualizado na Figura 29.

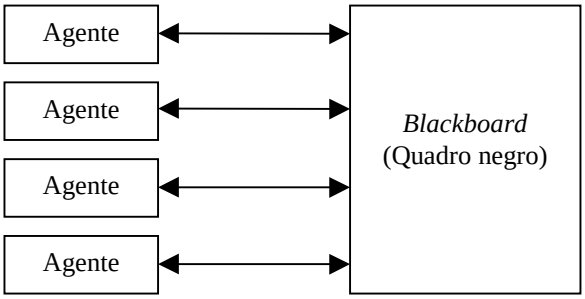

Figura 19: Arquitetura Blackboard

As Estratégias de Ensino podem ser vistas como “esquemas de planos” que definem formas de apresentar o material instrucional ao aluno [GIR1997].

Segundo [GIR1998], a seleção do conjunto de estratégias de ensino que será utilizada no STI é um aspecto muito importante para garantir a qualidade pedagógica do ambiente de ensino-aprendizagem. A seleção de uma estratégia depende de diversos fatores, tais como: o nível de conhecimento do estudante, o domínio, a motivação e as características afetivas do mesmo e outras.

Pode-se dizer que os STI apresentam, na sua maioria, os seguintes tipos de estratégias de ensino [GIR1998], [GIR2001]:

- **Socrático** – o material apresentado é projetado para induzir os estudantes a identificar enganos e interpretações errôneas acerca do conteúdo. Essa estratégia baseia-se no erro do estudante;
- **Reativos** – as lições reagem às perguntas do estudante e hipóteses que simulam os efeitos das idéias do estudante, apresentando as respectivas implicações baseadas em regras prévia;
- **Treinamento** (*Coaching*) – o sistema utiliza um conjunto de regras de produção para escolher a forma mais apropriada de instrução para um determinado estudante;
- **Assistente** – o tutor age como um participante em uma conversação com estudantes e auxilia-os a clarificar suas idéias utilizando suas próprias perícias;
- ***Troublemaker*** - esta estratégia sugere que o computador possa simular distintos comportamentos: dar uma resposta errada a um problema para forçar o aprendiz a reagir e a propor uma solução correta, ou esperar pela solução do aprendiz e dar uma sugestão ou solução errada..

Muitos sistemas selecionam e adotam mais de uma estratégia. Isso ocorre porque os sistemas geralmente tem mais de um objetivo, princípios diferentes de instruir, diferentes métodos de estruturar o conhecimento, etc.

Os ambientes de ensino-aprendizagem computadorizados devem oferecer um estudo individualizado ao aluno e, para tanto, todos os agentes possuem, mantém e atualizam os dados dos alunos em um Modelo de Aluno que guarda informações sobre o nível de conhecimento e as preferências deste aluno. Os agentes usam as informações do modelo do aluno, para definir qual o melhor conteúdo e/ou exercício a ser exibido para o aluno.

Segundo [GIR1998], modelo do aluno representa o conhecimento e as habilidades cognitivas do aluno em um dado momento. A partir desse modelo e do conteúdo a ser ensinado, o sistema deve ser capaz de inferir a melhor estratégia de ensino a ser utilizada.

Nos ambientes analisados são utilizadas diversas maneiras para construir o modelo do aluno. Listamos a seguir algumas delas:

- incluir um reconhecimento de padrões aplicados ao histórico das respostas fornecidas por ele;
- comparar a conduta do aluno com a de um especialista e verificar os pontos em comum;
- acrescentar as preferências do aluno;
- incluir seus objetivos particulares;
- observar as coisas que o aluno sempre costuma esquecer quando interage com o tutor;

Segundo [GIR2000], o modelo do aluno pode ser representado sob alguns modelos de descrição, tais como: Modelo diferencial, Modelo de *Overlay* ou Superposição, Modelo de Perturbação (*Buggy*), Modelo de Simulação, Modelo de Crenças e Modelo de Agentes.

Na grande maioria dos trabalhos avaliados, não foi encontrada a citação explícita do tipo de modelo do aluno. Entretanto, todos trabalham, basicamente, no tratamento do erro cometido pelo aluno, quando está resolvendo um exercício ou uma tarefa proposta. A forma como este erro é tratado utiliza muito dos modelos de Buggy, ou Perturbação utilizados nos STI clássicos.

Somente, nos ambientes White Habbit e AME-A, o modelo do aluno é construído sob uma rede neuronal construída através do algoritmo de aprendizagem *Map Koronen* para a criação de *clusters* para cada perfil.

Para qualquer sistema interativo, a Interface Gráfica é de suma importância. Em relação à esse aspecto, todos os ambientes analisados possuem interface gráfica interativa. No desenvolvimento de STI a preocupação com este aspecto não é diferente. Pelo contrário, segundo [GIR2000], é na interação que o sistema tutor exerce duas de suas principais funções, que são: apresentação do material instrucional e a monitoração do progresso do estudante através da recepção da resposta do aluno.

Pode-se citar algumas funções que a Interface do sistema deve propiciar:

- é necessário evitar que o estudante se entedie, ou seja, é preciso riqueza de recursos na apresentação do material instrucional;
- é desejável que haja facilidade para troca da iniciativa do diálogo: o estudante deve poder intervir facilmente no discurso do tutor, e vice-versa;
- o tempo de resposta deve ser rápido;
- a monitoração deve ser realizada o máximo possível em *background*, para onerar o estudante com questionários excessivos, mas respeitando também a barreira do tempo de resposta;

Devido a isto, em muitos dos sistemas existem agentes executores para fazer estas tarefas.

Em relação às ferramentas auxiliares que foram incorporadas nos sistemas para auxiliar no processo de ensino-aprendizagem, podemos dizer que as ferramentas tais como: *browser* e *chat* são essenciais por tratar-se de ferramentas que dão suporte à comunicação em ambientes baseados na Web.

5 Considerações Finais

OS STI representam uma interessante ferramenta para ambientes de ensino-aprendizagem computadorizados. Entretanto, os maiores problemas associados a estes tipos de sistema são:

- seu alto custo financeiro;
- elevado tempo de desenvolvimento;
- complexidade de modelagem;
- interação em alto grau entre os membros da equipe interdisciplinar.

Outro aspecto importante, resultante de nossa pesquisa, é que o desenvolvimento de STI que utilizam a tecnologia de agentes, não possuem uma metodologia clara e estabelecida. Isto ocorre, também, devido ao fato de a própria tecnologia de agentes não possuir uma especificação ainda clara e concisa. Os STI são uma aplicação como outra qualquer sob o ponto de vista de Engenharia de Software e, como tal, refletem a questão crucial da área de SMA onde a não-padronização da modelagem ou inexistência de metodologias para se especificar SMA, se põe como uma questão em aberto.

Na tentativa de reduzir estes custos, conceitos bem conhecidos da Engenharia de Software como reutilização e modularidade têm sido pesquisados e utilizados. A questão de fundo é desenvolver STI de forma incremental, permitindo uma evolução contínua e baseada numa metodologia que contemple as peculiaridades desta aplicação.

Aliás, uma metodologia de especificação de sistema deve poder permitir detalhar as peculiaridades de cada aplicação. Para tanto, foi desenvolvida uma pesquisa acerca das diferentes metodologias para a especificação de SMA a fim de determinar qual a que seria mais adequada para os STI Multiagentes. Este trabalho será apresentado em um futuro RT.

O extenso trabalho de pesquisa desenvolvido para a elaboração deste estudo permitiu-nos identificar o estado corrente (considerando o período estudado) no que concerne à STI que utilizam a tecnologia de agentes para a sua modelagem e implementação. Espera-se, com este trabalho, ter contribuído para a discussão das questões em aberto envolvendo a modelagem de STI utilizando a tecnologia de agentes.

Além da organização e análise aqui documentada, este RT apresenta o levantamento realizado na busca de elementos freqüentes encontrados em STI Multiagentes, objetivando verificar como os módulos da arquitetura tradicional de um STI foram substituídos por agentes.

Observou-se que a variedade de alternativas é grande. Cada grupo/autor apresenta uma interpretação própria de como a abordagem multiagente se relaciona com cada módulo da arquitetura apresentada nas páginas 7 e 8 deste RT. Este trabalho está vinculado ao projeto MASP (http://www.inf.pucrs.br/~giraffa/masp/).

6 Referências Bibliográficas

[AND2001] ANDRADE, Adja F. de; JAQUES, Patrícia A.; JUNG, João Luiz; BORDINI, Rafael H.; VICARI, Rosa M.. A Computacional Model

of distance Learning Based on Vygotsky´s Socio-cultural Approach. Workshop – Multi-Agent Architectures for Distribuited Learning Environments. Procedings International Conference on AI and Education, San Antonio, Texas, May, 2001. P.33-40.

[BIC1998] BICA, Francine; SILVEIRA, R. A.; VICCARI, R. ELETROTUTOR III – Uma abordagem Multiagentes. IX Simpósio Brasileiro de Informática na Educação/SBIE – Fortaleza, Novembro 1998.

[BIC1998a] BICA, Francine. Eletrotutor III - Uma Abordagem Multiagente para o Ensino à Distância. CPGCC/UFRGS, Porto Alegre, 2001. Tese de Doutorado.

[CAR1970] CARBONELL, J. R. AI in CAI: na artificial intelligence approach to computer assisted instruction. IEEE Transactions on Man Machine Systems, v.11, n.4, p.190-202, 1970.

[COS1995] COSTA, E. B. and LOPES, M. A. and FERNEDA, E.. Mathema: A Learning Enviroment based on a Miltu-Agent Architecture. In Lectures Notes in Artificial Inteligence. Advances in Artificial Inteligence. Procedings of 12th Brazilian Symposium on Artificial Inteligence. SBIA '95, Campinas, Brasil, october 1995. P.141-150.

[COS1996] COSTA, E. B..Um modelo de ambiente Interativo de Ensino-Aprendizagem baseado numa rquitetura Muitl-Agentes. Campina Grande: CPGEE/UFPA, 1996. Exame de Qualificação.

[DAM1997] D'Amico, C. B; Viccari, R. M.; Alvares, L.O. A Framework for Teaching and Learning Environments. In: SIMPÓSIO DE INFORMÁTICA NA EDUCAÇÃO, VIII, 1997, São Paulo, SP.

[DAM1998] D'Amico, C. B.; Pereira, A. S.; Geyer, C. F. R.;Viccari, R. M. Adapting Teaching Strategies in a Learning Environment on WWW. In: Proceedings of the WebNet World Conference of the WWW, Internet & Intranet. Florida, USA. 1998.

[DAM1998a] D'Amico, C. B.; Pereira, A. S.; Geyer, C. F. R.; Viccari, R. M. Adaptive Teaching and Learning Environment for Advanced WEB-Based Educational Applications. In: Proceedings of the II Iberoamerican Workshop on DAI and MAS. Toledo, Espanha. 1998.

[DAM1999] D'AMICO, C.B. Aprendizagem estática e dinâmica em ambientes

multiagentes de ensino-aprendizagem. Porto Alegre: PPGC da UFRGS, 1999. Tese de Doutorado.

[DEM1995] DEMAZEAU, Y. From Interactions to Collective Behaviour in Agent-Based Systems. Im: Proceedings of the 1st. Europeam Conference on Cognitive Science. Saint-Malo, france, 1995.

[FRA1998] FRASSON, Claude; MARTIN, Louis; GOUARDERES, Guy; AIMEUR, Esma. LANCA: A Distance Learning Architecture Based on Networked cognitive Agents. In Lectures Notes in Computer Science. Inteligent Tutoring Systems. Procedings of 4th International Conference, ITS 1998, San Antonio, Texas, August 1998. P.594-603.

[GIR1997] GIRAFFA, Lúcia. M. Martins. Seleção e adoção de estratégias de Ensino em Sistemas Tutores Inteligentes. Porto Alegre. CPGCC/UFRGS, 1997. (Exame de Qualificação)

[GIR1998] GIRAFFA, L. M. M. Uma arquitetura de tutor utilizando estados mentais. Porto Alegre: CPGCC da UFRGS, 1998. Tese de Doutorado.

[GIR1998b] GIRAFFA, L. M. M.; VICCARI, R. M.. Estratégias de ensino em Sistemas Tutores Inteligentes modelados através da tecnologia de agentes. IX Simpósio Brasileiro de Informática na Educação, SBIE'98. Fortaleza. Novembro, 1998.

[GIR1999] GIRAFFA, L. M. M.; VICCARI, R. M.. Estratégias de ensino em Sistemas Tutores Inteligentes modelados através da tecnologia de agentes. Revista Brasileira de Informática na Educação. IE/SBC. (6), 2, 1999.

[GIR2001] GIRAFFA, Lúcia M. M., VICCARI, Rosa M.. Fundamentos dos Sistemas Tutores Inteligentes. Porto Alegre. Capítulo de livro (a ser publicado)

[GIR2001a] GIRAFFA, L. M. M. STI modelados através de uma sociedade de agentes. Capturado em Novembro de 2001. *Online*. Disponível na Internet em: http://www.edukbr.com.br/portal.asp

[GOU2000] GOULART, Rodrigo. O papel do módulo tutor em STI. Porto Alegre. 2000. Trabalho Individual I.

[JAQ1999] JAQUES, Patricia Augustin. Agentes de Software na Monitoração da Colaboração em Ambientes Telemáticos de Ensino. PPGCC/PUCRS, Porto Alegre, 1999. Dissertação de Mestrado.

[JEN1996] JENNINGS, N. R.; FARATIN, P.; JOHNSON, M. J.; O'BRIEN, P.; WIEGAND, M. E. Using Intelligent Agents to Manage Business Processes, Proceedings of First International Conference on The Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM96), London, UK, 345-360. Capturado em Setembro de 2001. *Online*. Disponível na Internet em: http://www.ecs.soton.ac.uk/~nrj/pubs.html

[KUH1997] KUHN, Thomas. A Estrutura das revoluções científicas. Capturado em Novembro de 2001. *Online*. Disponível na Internet em: http://www.sul-sc.com.br/afolha/pag/revolucao_cientifica.htm

[MUR1999] MURRAY, Tom. Authoring Intelligent Tutoring Systems: Na analysis of the state of the art. International Journal of Artificial Intelligence in Education (1999), 10, 98-129

[OLI1994] OLIVEIRA, F. M.. Critérios de equilibração para Sistemas Tutores Inteligentes. Porto Alegre: CPGCC da UFRGS, 1994. Tese de Doutorado.

[OLI1996] OLIVEIRA, Flávio Moreira de. Inteligência Artificial Distribuída. In: ESCOLA REGIONAL DE INFORMÁTICA, 4, 1996, Canoas. Anais... Sociedade Brasileira de Computação, 1996. p. 54-73.

[PER2001] Pereira, A. S.; D'Amico, C. B.; Geyer C. F. R.. Gerenciamento do Conhecimento do ambiente Ame-A. Capturado em Novembro de 2001. *Online*. Disponível na Internet em: http://www.inf.ufrgs.br/~adriana/vcied.doc

[ROS2000] ROSATELLI, M.C., Self, J.A. and Thiry, M. LeCS: a collaborative case study system, in G. Gauthier, C. Frasson and K. VanLehn (eds.), Intelligent Tutoring Systems, Berlin: Springer-Verlag, p232-241, 2000.

[ROS2000a] ROSATELLI, M.C., Self, J.A. and Thiry, M. LeCS: a collaborative case study system. In Lectures Notes in Computer Science. Inteligent Tutoring Systems. Procedings of 5th International Conference, ITS

2000, Montreal, Canada, June 2000. P.242-251.

[RUS1995] RUSSELL, Stuart; NORVIG, Peter. Artificial Intelligence: A Modern Approach. Prentice Hall Series in Artificial Intelligence, New Jersey, 1995.

[SEA1969] SEARLE, J. R. Speach acts: na essay in the philosophy of language. Cambridge: Cambridge University Press, 1969. Tese de Doutorado.

[SEL1999] SELF, John. The defining characteristics of intelligent tutoring systems research: its cara, precisely. International Journal of Artificial Intelligence in Education, Leeds, England. 1999.

[SHO1993] SHOHAM, Y. Agent-oriented Programming. Artificial Intelligence, 60, 51-92, 1993.

[SIL1992] SILVEIRA, R.A. Inteligência Artificial em Educação: um modelo de sistema tutorial inteligente para microcomputadores. Porto Alegre PUCRS, 1992. Dissertação de Mestrado em Educação.

[SIL1998] SILVEIRA, R. A. Ambientes Inteligentes de Ensino-Aprendizagem. Porto Alegre: PPGC da UFRGS, 1998. Exame de Qualificação.

[SIL1999] SILVEIRA, R. Modelagem orientada a agentes aplicada a ambientes distribuídos de ensino. Porto Alegre: PPGC da UFRGS, 1999.

[THI1999] THIRY, Marcelo Comicholi da Costa. Uma Arquitetura Baseada em Agentes para Suporte ao Ensino à Distância. CPGCC/UFRGS, Porto Alegre, 2001. DEPS/PPGEP, 1999. Tese de Doutorado. Capturado em Novembro de 2001. *Online*. Disponível na Internet em: http://www.ecs.soton.ac.uk/~nrj/pubs.html

[THI2000] THIBODEAU, Marc-André; BÉLANGER, Simon; FRASSON, Claude. WHITE RABBIT – Matchmaking of User Profiles Based on discussion analysis Using Intelligent Agents. In Lectures Notes in Computer Science. Inteligent Tutoring Systems. Procedings of 5th International Conference, ITS 2000, Montreal, Canada, June 2000. P.113-122.

[VAS1997] VASSILEVA J. Adaptive Systems and User Modeling on the World Wide Web. The 6-th International Conference on User Modeling, UM'97. Chia-Laguna, Sardinia, June. 1997.

[VAS2001] VASSILEVA, Julita; DETERS, Ralph; GEER, Jim; Mccalla, Gord;

BULL, Susan; KETTEL, Lori. Lessons from Deploying I-Help. Workshop – Multi-Agent Architectures for Distribuited Learning Environments. Procedings International Conference on AI and Education, San Antonio, Texas, May, 2001. P.3-11.

[WEB2001] WEBBER, Carine; BERGIA, Loris; PESTY, Sylvie; BALACHEFF, Nicolas. The Baghera project: a multi-agent architecture for human learning. Workshop – Multi-Agent Architectures for Distribuited Learning Environments. Procedings International Conference on AI and Education, San Antonio, Texas, May, 2001. P.12-17.

[WOO1995] WOOLDRIDGE, M.; JENNINGS N. R.. Intelligent Agents: Theory and Practice. In Knowledge Engineering Review 10(2), 1995. (The PostScript version.). Capturado em Setembro de 2001. *Online*. Disponível na Internet em: http://www.csc.liv.ac.uk/~mjw/pubs/.

[ZAM2000] ZAMBONELLI, F.; Jennings, N. R.; Omicini, A.; Wooldridge, M.. Agent-Oriented Software Engineering for Internet Applications. In A. Omicini, F. Zambonelli, M. Klusch and R. Tolksdorf, editors, Coordination of Internet Agents, Springer Verlag. Capturado em Setembro de 2001. *Online*. Disponível na Internet em: http://sirio.dsi.unimo.it/Zambonelli/pubblica.html

[ZOU2000] ZOUAQ, Amal; FRASSON, Claude; ROUANE, Khalid. The Explanation Agent. In Lectures Notes in Computer Science. Inteligent Tutoring Systems. Procedings of 5th International Conference, ITS 2000, Montreal, Canada, June 2000. P.554-563.