TEREZINHA DE FÁTIMA FARIA

UM AMBIENTE INTERATIVO MULTIAGENTES PARA O ENSINO DE ESTRUTURA DA INFORMAÇÃO

FLORIANÓPOLIS

2001

UNIVERSIDADE FEDERAL DE SANTA CATARINA

PROGRAMA DE PÓS-GRADUAÇÃO
EM ENGENHARIA ELÉTRICA

UM AMBIENTE INTERATIVO MULTIAGENTES PARA O ENSINO DE ESTRUTURA DA INFORMAÇÃO

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Mestre em Engenharia Elétrica

TEREZINHA DE FÁTIMA FARIA

Florianópolis, Abril de 2001.

UM AMBIENTE INTERATIVO MULTIAGENTES PARA O ENSINO DE ESTRUTURA DA INFORMAÇÃO

Terezinha de Fátima Faria

'Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia Elétrica, Área de Concentração em *Controle, Automação e Informática Industrial*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina'.

_____

Guilherme Bittencourt, Dr.
Orientador

_____

Aguinaldo Silveira e Silva, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

_____

Guilherme Bittencourt, Dr.
Presidente

_____

Evandro de Barros Costa, Dr.

_____

Jorge Muniz Barreto, Dr.

Marcelo Ricardo Stemmer, Dr.

"... o confronto entre homem e máquina, longe de reduzir o homem à máquina, revela o que há de irredutível no homem, o que o distingue da máquina, aquilo em que sua atividade não pode ser comparada com um simples cálculo, que a memória humana não resulta de mera estocagem de informações análoga à que se produz na memória de um computador.

... a inteligência artificial é um subterfúgio, um artifício destinado a dominar as máquinas atribuindo-lhes uma inteligência".

Jean-Gabriel Ganascia

"Porque a nossa capacidade vem de Deus".

2 Coríntios, 3:5

Aos meus pais
Raimundo e Ermelinda

E aos meus irmãos
Carlinhos e Gaspar

Agradecimentos

A Deus, pela vida e por tantas oportunidades de aprendizado e crescimento.

Aos meus pais Raimundo e Ermelinda, e aos meus irmãos Carlinhos e Gaspar, pelo apoio e carinho constantes e indiscutíveis, sem os quais eu realmente não poderia imaginar a minha vida.

Ao meu orientador Guilherme Bittencourt, pelo apoio e compreensão constantes, pela oportunidade de realizar este trabalho e por todos os ensinamentos.

A Guilherme Weigert, Leonardo Bitsch, Carlos Alexandre de Miranda Silva, Gustavo Adolpho Rangel Monteiro, Luciana Bolan Frigo, cujas colaborações com certeza foram de grande importância para a realização deste trabalho.

Aos professores Evandro de Barros Costa, Jorge Muniz Barreto e Marcelo Ricardo Stemmer, membros da banca examinadora deste trabalho, pelas valiosas contribuições.

Ao professor Marcos Augusto Francisco Borges, meu orientador de Iniciação Científica, que me ajudou a crer que seria possível ir além, pelo indiscutível apoio.

Aos professores Gualberto Rabay Filho e Marcelo Galvão Fonseca, pelo grande e constante incentivo.

Ao amigo Renato Donizete Vilela de Oliveira, pela valiosa ajuda e amizade. Também aos amigos Fred Henrique Souza Paes, Jeruza Marchi e Carlos Ogawa, pelos bons momentos compartilhados.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq - pelo apoio financeiro durante parte do trabalho.

Enfim, o meu agradecimento a todos aqueles que direta ou indiretamente contribuíram para a realização deste trabalho, com seus esforços, dedicação e amizade.

Resumo da Dissertação apresentada à UFSC como parte dos requisitos necessários para a obtenção do grau de Mestre em Engenharia Elétrica.

UM AMBIENTE INTERATIVO MULTIAGENTES PARA O ENSINO DE ESTRUTURA DA INFORMAÇÃO

Terezinha de Fátima Faria

Abril/2001

Orientador: Guilherme Bittencourt, Dr.

Área de Concentração: Controle, Automação e Informática Industrial.

Palavras-chave: Inteligência Artificial em Educação, Ambientes Interativos de Aprendizagem.

Número de Páginas: 60.

Este trabalho apresenta o desenvolvimento, conforme a metodologia proposta no modelo MATHEMA, de um sistema tutor inteligente para o ensino de Estrutura da Informação. O sistema tutor proposto utiliza técnicas de Inteligência Artificial Distribuída, mais especificamente seguindo a abordagem de sistemas multiagentes, e encontra-se no contexto dos ambientes interativos de aprendizagem assistidos por computador. O desenvolvimento deste ambiente faz parte das atividades de um projeto de Pesquisa em Informática na Educação apoiado pelo CNPq-ProTeM-CC. O sistema será testado no âmbito da disciplina de Fundamentos da Estrutura da Informação, aplicada ao curso de Engenharia de Controle e Automação da Universidade Federal de Santa Catarina. Os testes serão feitos com o objetivo de se verificar o comportamento do ambiente desenvolvido, tanto do ponto de vista computacional quanto educacional, e fornecerão novas fontes de dados para a avaliação dos resultados obtidos a partir dos testes locais, efetuados pela equipe de desenvolvimento. A metodologia e ferramentas utilizadas se mostraram bastante adequadas aos objetivos do trabalho. A metodologia proporciona princípios interessantes em relação à problemática do desenvolvimento de sistemas tutores inteligentes, focando questões referentes à adaptabilidade de tais sistemas ao perfil de um aprendiz e à interação entre entidades humanas e artificiais neste processo adaptativo. As ferramentas apresentam os requisitos necessários para o alcance de um importante objetivo do trabalho, que é a aplicação do ambiente desenvolvido ao contexto da Educação a Distância.

Abstract of Dissertation presented to UFSC as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering.

A MULTI-AGENT INTERACTIVE ENVIRONMENT TO TEACH INFORMATION STRUCTURE

Terezinha de Fátima Faria

April/2001

Advisor: Guilherme Bittencourt, Dr.
Area of Concentration: Control, Automation and Industrial Computing.
Keywords: Artificial Intelligence in Education, Learning Interactive Environments.
Number of Pages: 60.

This research presents the development of an intelligent tutoring system to teach Information Structure, it is based on MATHEMA model. The proposed tutoring system uses techniques of Distributed Artificial Intelligence, more specifically it adopts the multi-agent systems approach in context of computer-aided learning interactive environments. The system's development belongs to the activities in a research project of Computing in Education supported by the Brazilian agency CNPq. The system is going to be tested in the Information Structure course of the Automation and Control Engineering undergraduate degree of the Federal University of Santa Catarina. These tests will be done to verify the system behavior, in computational and educational aspects, and it will supply others data sources for evaluation of the results got in the local tests. The local tests were made by the development group. The methodology and tools used in this project seemed suitable to the goals of the work. The methodology has interesting issues about the development of intelligent tutoring systems, it treat with points in respect to adaptability of these systems to the apprentice profile, and interactions among human and artificial entities in this adaptative learning process. The tools have the necessary requisites to get an important goal of this work, which is its application to the Distance Education context.

Sumário

Lista de Figuras

Capítulo 1

Introdução

A Inteligência Artificial (IA) tem o objetivo de estudar e criar modelos para processos cognitivos, busca entender entidades inteligentes. Assim, uma razão para estudá-la é aprender mais sobre nós mesmos [1]. São diversas as áreas cujas teorias influenciam e contribuem decisivamente com a IA. Da Filosofia, surgiu a discussão que deu origem a esta área, criando-se teorias de raciocínio e aprendizagem. A Psicologia proporciona meios para investigar a mente humana, e uma linguagem científica para expressar os resultados obtidos. Da Matemática, tem-se os recursos necessários à formalização das teorias resultantes dos estudos nesta área. E a Ciência da Computação proporciona ferramentas que tornam possível a implementação dos modelos criados, ou seja, que possibilitam fazer da IA uma realidade. Maiores detalhes sobre estas e outras disciplinas que influenciam a IA podem ser encontrados em [1]. Esta é uma área que pode ser analisada sob diversos pontos de vista, e seguindo uma visão pragmática, pode-se dizer que seu objetivo é tornar os computadores mais úteis através da implementação de sistemas mais eficientes.

É adequado dizer que o computador tem o objetivo de automatizar tarefas realizadas por seres humanos, principalmente aquelas demoradas e repetitivas. O computador provoca uma transformação nos tipos de trabalho, no relacionamento com a informação. Sem o auxílio de uma máquina, diversos resultados não poderiam ser obtidos em tempo ideal e com a precisão necessária, configurando um cenário já conhecido até mesmo na época inicial do uso de computadores. A IA vem contribuir com este objetivo,

buscando incorporar alguma inteligência aos sistemas desenvolvidos para controle da máquina, de forma que possam ser mais eficientes e alcancem uma certa independência em relação ao controle humano. Pois se tarefas são automatizadas, melhor ainda que isto seja feito com eficiência. A IA e a Ciência da Computação foram crescendo juntas, trocando influências, atingindo objetivos e enfrentando desafios. Por isto em alguns pontos, a história destas áreas se confunde. Tem acontecido muitas vezes que uma técnica originalmente desenvolvida para propósitos da IA tem sido aceita como parte da "corrente principal" da Ciência da Computação ou Engenharia de Software, e é provável que isto continue a acontecer [2].

Devido as suas características, percebe-se que as teorias da IA se aplicam a diversas áreas onde o computador já é utilizado como ferramenta de apoio a alguma atividade. Tem-se como exemplo a Educação e a Educação a Distância, sendo esta última uma área que tem se expandido em proporções consideráveis. O uso de meios computacionais para o oferecimento de ensino/aprendizagem a distância tem proporcionado inovações interessantes a este processo. As redes de computadores possibilitam o rápido acesso a informações dos mais variados tipos e origens, ampliando as possibilidades na busca do conhecimento. Além disso, diversos recursos computacionais contribuem para uma melhor apresentação de materiais.

Uma preocupação dos pesquisadores em Educação refere-se aos papéis assumidos pelas diversas entidades envolvidas no processo educacional. As pesquisas concluem que o aprendiz deve participar ativamente deste processo, e não apenas receber informações prontas, ou seja, é preciso haver interação entre as entidades. A mesma idéia se aplica aos ambientes computacionais que se destinam a desenvolver processos de aprendizagem, deixando claro a importância dos ambientes interativos de aprendizagem mediados por computador.

Este trabalho apresenta a especificação de um sistema tutor inteligente para a disciplina de Fundamentos da Estrutura da Informação, aplicada ao curso de Engenharia de Controle e Automação da Universidade Federal de Santa Catarina. Esta especificação está baseada na proposta do ambiente MATHEMA e faz parte das atividades do projeto Math_Net - *Uma abordagem via sistemas multiagentes para concepção e realização de ambientes interativos de aprendizagem cooperativa assistidos por computador* [3] - um projeto de Pesquisa em Informática na Educação apoiado pelo CNPq-ProTeM-CC e cujo tema central é a concepção e o desenvolvimento de um modelo computacional para

Ambientes Interativos de Ensino/Aprendizagem Cooperativa com base em múltiplos agentes artificiais e humanos, dispostos em uma estrutura de rede de computadores. Pretende-se com isto, investigar e desenvolver um ambiente interativo e distribuído de suporte à aprendizagem cooperativa e colaborativa, integrando alunos, professores e sistemas computacionais [4].

Um dos maiores desafios apontados ao desenvolvimento de Sistemas Tutores Inteligentes refere-se à capacidade de adaptação do sistema ao perfil cognitivo do aprendiz [5], ou seja, a possibilidade do sistema adaptar suas ações tutoriais às necessidades individuais de um aprendiz (tal como o sistema as percebe) durante o processo de interação de ensino/aprendizagem. Um dos fortes desafios está na administração da complexidade do conhecimento que o sistema deveria lidar para promover instruções apropriadas aos aprendizes humanos [6].

Para se interagir adequadamente com um aprendiz é necessário que se conheça o seu posicionamento durante o processo de aprendizagem, ou seja, é importante perceber seus conhecimentos em relação ao domínio a ser trabalhado, e seu desenvolvimento em relação ao conteúdo exposto. Parte da pesquisa em sistemas que aplicam IA em Educação (IA-ED) busca soluções efetivas para este problema, conhecido como problema da adaptação, procurando responder a três questões clássicas: O que ensinar? A quem ensinar? Como ensinar? Estas questões referem-se, respectivamente, às modelagens do conhecimento do domínio de aplicação, do estado cognitivo do estudante e de estratégias pedagógicas. O modelo que fundamenta o projeto em questão busca soluções neste sentido, propondo um processo de modelagem distribuída de apoio ao diagnóstico das ações do aprendiz [7]. Ainda com base na problemática colocada anteriormente, a Inteligência Artificial Distribuída (IAD), através de uma abordagem multiagentes, representa um recurso apropriado para o tratamento da complexidade citada. A IAD é uma área que se utiliza de tecnologias de redes e de sistemas distribuídos.

1.1 Motivação e objetivos

Os esforços realizados quanto ao estudo e desenvolvimento de técnicas de utilização do computador como uma ferramenta de apoio ao processo de ensino/aprendizagem tiveram seu início pouco tempo depois do desenvolvimento do

computador digital e têm hoje uma longa história. Ao longo desta história, foram investigados diversos enfoques, levando em conta tanto os desenvolvimentos tecnológicos na Ciência da Computação, quanto os trabalhos na área de Educação. Os estudos realizados com este objetivo passaram a constituir uma área que se tornou conhecida como Informática na Educação.

Atualmente, a necessidade de contribuições originadas neste contexto faz-se mais notória, visto que o computador tem cada vez mais se tornado um importante instrumento no processo educacional. Considerando-se, por exemplo, o crescente estudo e desenvolvimento de tecnologias que possibilitam a realização de educação a distância mediada por computador, pode-se ter uma idéia do quanto este tem se tornado um instrumento de fundamental importância. Mas o uso do computador neste processo é uma questão que também gera polêmicas. A este respeito existem comentários interessantes tais como em [8], dizendo que as tecnologias de comunicação não substituem o professor, mas modificam algumas das suas funções. O professor se transforma agora no estimulador da curiosidade do aluno por querer conhecer, por pesquisar, por buscar a informação mais relevante. Observando-se as considerações anteriores, pode-se concluir que cada vez mais se torna necessário o desenvolvimento de sistemas que possam auxiliar o processo educacional e que alcancem resultados efetivos [7].

Conforme citado em [4], a pesquisa e o desenvolvimento do ambiente aqui apresentado são realizados por um consórcio constituído por professores e pesquisadores das Universidades Federais de Alagoas (UFAL), de Santa Catarina (UFSC) e do Maranhão (UFMA).

O modelo de ambiente proposto está sendo aplicado na criação de Ambientes Interativos de Aprendizagem em domínios particulares, comprometidos com níveis de escolaridade que vão desde o primeiro grau até cursos de graduação e pós-graduação. As tarefas sob responsabilidade do grupo da UFSC têm os seguintes objetivos particulares:

- Estudo da Sociedade de Agentes de um ponto de vista interno.
- Desenvolvimento de aplicações específicas com o ambiente.
- Experimentação do ambiente nos domínios desenvolvidos.

As atividades realizadas a fim de se cumprir tais tarefas são descritas nos capítulos 3 e 4.

1.2 O modelo utilizado

O ambiente MATHEMA foi desenvolvido no contexto dos Sistemas de Inteligência Artificial em Educação (IA-ED), mais especificamente tratando da concepção de ambientes interativos de aprendizagem sob um enfoque multiagentes. Trata-se de um contexto interdisciplinar, envolvendo áreas como Inteligência Artificial Distribuída, Educação e Psicologia. O MATHEMA constitui um modelo de ambiente interativo de aprendizagem mediada por computador, sendo concebido para apoiar atividades que venham a favorecer a realização de interações adaptativas e seus desdobramentos no processo de aprendizagem. Como um modelo conceitual, este ambiente busca proporcionar uma arquitetura alternativa para orientar o desenvolvimento de sistemas IA-ED particulares e inclui uma metodologia para representação de conhecimento do domínio de aplicação baseado em três dimensões: contexto, profundidade e lateralidade.

O objetivo é envolver o aprendiz em uma relação de interação com uma sociedade de agentes tutores artificiais, implementada por um sistema multiagentes, visando posicioná-lo em situações de aprendizagem a partir do seu envolvimento em atividades de resolução de problemas. Os agentes tutores podem cooperar entre si ou com uma sociedade de especialistas humanos. Neste contexto, o MATHEMA representa um modelo de sistema IA-ED distribuído, onde agentes tutores se comunicam através de troca de mensagens [5].

A arquitetura utilizada como base para este trabalho se mostra muito interessante e adequada face a uma análise dos aspectos que compõem a problemática abordada pela área de Informática na Educação, pois esta arquitetura integra atividades de entidades humanas e artificiais no processo educacional. Se algumas vezes a idéia de se criar um sistema computacional que possa ensinar (ou, falando de uma maneira construtivista, facilitar o processo de aprendizagem) é considerada pretensiosa, a proposta do ambiente MATHEMA se mostra bastante coerente ao enfatizar o acompanhamento, por parte de uma Sociedade de Especialistas Humanos e do próprio aprendiz, das atividades desenvolvidas em tal ambiente. E além disso, esta arquitetura define uma metodologia distribuída para as modelagens necessárias, oferecendo contribuições no que se refere ao problema da adaptação.

1.3 O ambiente MathTutor

O ambiente desenvolvido, chamado MathTutor, baseia-se no modelo e objetivos anteriormente apresentados, adequando-se também aos requisitos necessários para aplicação ao contexto de Educação a Distância, visto que para a implementação, preocupou-se com a escolha de ferramentas que possam garantir a portabilidade do sistema e que possibilitem a criação de uma interface adequada às necessidades de tal contexto. As características estruturais e funcionalidades do sistema tutor inteligente desenvolvido são apresentadas no capítulo 3 desta dissertação e os aspectos referentes à implementação são discutidos no capítulo 4.

O domínio escolhido para ser tratado pelo sistema tutor proporcionou algumas situações desafiadoras, pois não se trata de conhecimentos que possam ser representados (pelo menos não de uma maneira trivial) com base em padrões e fórmulas, como seria o caso de alguns domínios matemáticos, mas de um domínio em que os problemas apresentados são solucionados de maneiras particulares, fugindo a padronizações.

1.4 Organização da dissertação

A presente dissertação está organizada da seguinte maneira: no capítulo 2 é apresentada a fundamentação teórica para o trabalho desenvolvido. De maneira sucinta, tem-se um histórico sobre a utilização da Informática como apoio ao processo educacional, expondo o contexto no qual o modelo MATHEMA foi desenvolvido, sendo que em seguida, este modelo é descrito, relatando-se suas características e funcionalidades. Ainda no capítulo 2 encontram-se noções a respeito de projetos instrucionais, associadas às características do MATHEMA e do MathTutor. O capítulo 3 descreve o MathTutor e como o domínio de conhecimento tratado foi modelado e distribuído entre os membros da sociedade de agentes tutores artificiais. As características do ambiente MATHEMA são associadas às necessidades do MathTutor. O capítulo 4 apresenta detalhes relacionados à implementação do MathTutor. Por fim, o capítulo 5 relata as conclusões e trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Neste capítulo apresenta-se um breve histórico a respeito da utilização de técnicas computacionais no processo educacional. Faz-se uma rápida discussão a respeito de Inteligência Artificial e apresenta-se um breve histórico dos Ambientes de Aprendizagem Assistidos por Computador, comentando algumas transformações ocorridas quando houve a união destes ambientes às técnicas de Inteligência Artificial, o que originou a área conhecida como Inteligência Artificial em Educação. Tais ambientes, em um momento posterior, passaram a incorporar os resultados da pesquisa relacionada à Inteligência Artificial Distribuída, que tem nos Sistemas Multiagentes uma de suas abordagens. Estas noções são importantes para a contextualização do ambiente MATHEMA. As idéias são apresentadas sucintamente, mas o texto contém referências a materiais onde tais discussões são mais detalhadas. Em seguida o ambiente MATHEMA é descrito. Por fim apresenta-se algumas idéias referentes a projetos instrucionais.

2.1 Histórico sobre a utilização da Informática como apoio ao processo educacional

Os sistemas desenvolvidos com a finalidade de constituir auxílio ao processo educacional se apoiaram em propostas bastante variadas, refletindo de alguma maneira

diversas concepções pedagógicas, as quais tratam diferentemente do relacionamento aprendiz/professor/máquina. O desenvolvimento de *software* educativo traz como importante requisito o conhecimento de diversas áreas, tais como Psicologia, Educação e Ciência da Computação, constituindo portanto, uma área altamente interdisciplinar. Algumas idéias importantes sobre áreas de conhecimento que se uniram ao processo de construção de *software* educacional são apresentadas a seguir.

2.1.1 Inteligência Artificial

A área de conhecimento denominada Inteligência Artificial (IA) teve sua origem oficial no ano de 1956, em uma conferência de verão que aconteceu em Dartmouth College, USA, para a qual vários pesquisadores importantes se propunham a realizar um estudo sobre o tema "inteligência artificial". Tal área sempre foi causadora de grandes polêmicas, desde seu nome até seus objetivos e metodologias, levando algumas vezes a promessas exageradas e conseqüentemente, a algumas decepções [9].

Várias definições operacionais foram dadas à IA, tendo em vista a impossibilidade de se obter uma definição formal. Uma das muitas definições que podem ser citadas é:

> "Inteligência Artificial é o estudo das faculdades mentais através do uso de modelos computacionais" [10].

O objetivo central da IA pode ser definido como a criação de modelos para a capacidade cognitiva e a implementação de sistemas computacionais baseados nestes modelos [11]. Ganascia [12] diz que o homem empresta inteligência à máquina para melhor dominá-la.

Mas existem questões de alta complexidade envolvidas na criação de tais modelos, visto que a capacidade de aprender é apenas um dos aspectos da inteligência. Portanto, além de uma máquina "aprender" algo com base em modelos predeterminados, existe uma série de aspectos particulares do que se pode definir como inteligência, geralmente difíceis de se expressar em formas computáveis. Uma ampla exposição sobre IA pode ser encontrada em [1, 12, 13, 14]. Sloman [2] fala de IA sob dois enfoques: como Ciência e como Engenharia, comentado sobre áreas de aplicação, ferramentas e linguagens etc.

Linhas de pesquisa

As principais linhas de pesquisa na criação de sistemas inteligentes têm a seguinte classificação [9]:

- Linha Conexionista: caracteriza-se pela modelagem da inteligência humana através da simulação dos componentes do cérebro (neurônios e suas interligações). Foi formalizada em 1943 com a proposição do primeiro modelo matemático do neurônio. Esta linha deu origem à área conhecida atualmente como Redes Neurais Artificiais.
- Linha Simbólica: fazem parte desta linha os sistemas especialistas (SE's), que estabeleceram a manipulação simbólica de fatos especializados sobre um domínio como paradigma para o desenvolvimento de sistemas inteligentes do tipo simbólico. Com base em [15] comenta-se a seguir exemplos de SE's bem sucedidos:

O sistema MYCIN [16] foi um dos primeiros SE's. Seu Objetivo é prover conselho a respeito de diagnóstico e terapia de doenças infecciosas. Este tipo de aconselhamento pode ser muito útil, pois nem sempre o médico responsável é um especialista em infecções, principalmente em ambiente hospitalar. Uma seção do sistema inicia-se com um questionário, a ser respondido pelo usuário, a respeito do paciente. Informações como nome, idade, sexo, tempo de manifestação dos sintomas, resultados de exames, etc. são solicitadas. A partir destas informações, e utilizando sua base de regras, o sistema é capaz de estabelecer um diagnóstico e propor uma terapia adequada.

O sistema DENDRAL [17] não segue a estrutura tradicional dos SE's, pois ele integra três programas independentes, dos quais apenas dois são baseados em regras. Os três programas têm as seguintes funções: (i) Planejamento: determinação das combinações de átomos consistentes com um conjunto de regras heurísticas sobre espectroscopia de massa. As restrições impostas às estruturas são de dois tipos: determinação de fragmentos moleculares que devem estar necessariamente presentes ou ausentes da estrutura final. (ii) Geração: construção de todas as estruturas moleculares que obedecem às restrições inferidas na parte de planejamento, isto é, que incluem todos os fragmentos necessários e nenhum dos proibidos. Este processo é realizado por um algoritmo tradicional.

Originalmente foi utilizado um algoritmo devido a Lederberg e mais tarde o programa CONGEN. (iii) Teste: classificação das estruturas geradas através da simulação de seu comportamento em um espectrógrafo de massa. As estruturas cujos espectros simulados se aproximam do espectro real são classificadas com um escore mais alto. A simulação é realizada utilizando-se regras que prevêem a posição de picos no espectro a partir da estrutura molecular.

O sistema PROSPECTOR [18] foi desenvolvido no SRI International (USA) com o objetivo de auxiliar geologistas envolvidos em prospecção mineral. A principal função do sistema é determinar a correspondência entre dados que descrevem uma determinada situação com modelos que descrevem classes disjuntas de situações possíveis. Os modelos são descrições formais dos tipos mais importantes de depósitos minerais e os dados de entrada se referem a observações geológicas de superfície.

- Linha da Computação Evolutiva: esta linha de pesquisa baseia-se em mecanismos evolutivos encontrados na natureza, sendo um de seus modelos mais conhecidos representado pelos algoritmos genéticos [11].

O ambiente de que trata este trabalho utiliza um arcabouço para a construção de sistemas especialistas, para armazenar conhecimentos a respeito dos módulos do domínio, do estudante e pedagógico, e para manipular estes conhecimentos, proporcionando um processo de interação dinamicamente adaptativo. Cada um dos agentes tutores tem presente em sua arquitetura um sistema tutor.

Se observações são feitas a respeito dos acontecimentos de diferentes períodos do desenvolvimento de pesquisas em IA, nota-se que os objetivos iniciais eram difíceis de se atingir, principalmente naquela época, quando ainda se teria um longo caminho para chegar ao atual estágio das pesquisas, que ainda hoje têm grandes desafios. Atualmente pretende-se através da IA, desenvolver ferramentas que possam ser úteis a diversas tarefas desempenhadas pelos seres humanos, simulando suas maneiras de tratar determinadas situações. Os frutos destas pesquisas estão cada vez mais presentes no nosso dia-a-dia, e nas mais diversas áreas, apresentando aplicações muito interessantes, que muitas vezes nos permitem alcançar, através de máquinas, locais inacessíveis aos seres humanos.

2.1.2 Ambientes de Aprendizagem Assistidos por Computador

Os ambientes de aprendizagem assistidos por computador (AAAC) originaram diferentes enfoques distribuídos em momentos cronologicamente distintos. Trabalhos como [5, 19, 20] apresentam discussões sobre o histórico de tais ambientes e também sobre paradigmas educacionais. Com base em [5], apresenta-se os enfoques que surgiram segundo uma divisão em três períodos. Em um primeiro momento consideram-se os enfoques surgidos até meados da década de 70, sendo eles:

- Sistemas de Instrução Assistida por Computador (*Computer Aided Instruction - CAI*): baseados em instrução programada, segundo uma abordagem behaviorista [21]. Não consideram as características cognitivas individuais dos estudantes.
- Micromundos [22]: baseados no princípio da construção de conhecimento, tendo como uma das primeiras concretizações o ambiente LOGO. Estes ambientes baseiam-se nas idéias do construtivismo [23].
- Simuladores e Jogos Educacionais: os simuladores envolvem a criação de modelos do mundo real ou imaginário, oferecendo ao aluno a possibilidade de desenvolver hipóteses, testá-las, analisar resultados e refinar conceitos. Os jogos educacionais visam proporcionar mais motivação, seguindo a idéia de "aprender brincando".

Lembrando que a entrada em um novo momento não implica o abandono de atividades de momentos anteriores, notando-se em muitos casos, uma extensão ou abordagens híbridas, passamos a considerar os enfoques surgidos no segundo momento da história de tais ambientes, compreendidos entre a segunda metade da década de 70 e o fim da década de 80. Neste momento definem-se duas fortes alianças para o desenvolvimento deste tipo de sistema, sendo uma delas a aliança com as áreas de IA e Psicologia Cognitiva e por outro lado, o aparato tecnológico mais apropriado (*hardware* e *software*) mais possantes, tecnologia de comunicação mais sofisticada, interfaces gráficas, Internet etc).

Surgem então, os *Intelligent Computer Aided Instruction* (*ICAI*) ou Sistemas Tutores Inteligentes (*Intelligent Tutoring System - ITS*), cuja característica básica é a representação de conhecimentos relacionados às três clássicas questões do chamado problema da adaptação: o que ensinar?, a quem ensinar? e como ensinar? Desta maneira,

busca-se promover instrução individualizada, ou seja, de acordo com o perfil cognitivo do aprendiz.

2.1.3 Inteligência Artificial em Educação

Esta aliança com a IA ocasionou o surgimento da área que passou a ser conhecida como Inteligência Artificial em Educação (*Artificial Intelligence in Education - AI-ED*), originada da combinação de métodos e técnicas de IA na concepção de ambientes de ensino/aprendizagem assistidos por computador.

Quanto aos aspectos educacionais, no contexto da pesquisa IA-ED considera-se a questão da aprendizagem como atividade fundamental em Educação, ao contrário, por exemplo, dos sistemas CAI's que priorizam o ensino. Já alguns micromundos descartam a atividade de ensino e as demais propostas geralmente se posicionam em um quadro intermediário. Vários modelos podem ser seguidos no tratamento da questão da aprendizagem, sendo estes divididos em abordagens tais como:

- Behaviorista [21]: baseada em um modelo do tipo estímulo-resposta. Tal abordagem tem perdido espaço, ao contrário das abordagens cognitivistas.
- Cognitivista: nesta abordagem situa-se a linha construtivista [23], considerando o conhecimento como o resultado de um processo de construção pessoal.
- Situacionista (contextualista) [24]: considera que a aprendizagem está sempre localizada num contexto e é permanentemente evolutiva, sendo um processo de iniciação, integração e colaboração no interior de diferentes comunidades de prática.

No terceiro momento histórico, abrangendo desde a década de 90 até a atualidade, destaca-se a tendência para modelos computacionais baseados na noção de cooperação, ocorrendo a evolução dos STI's tradicionais para Ambientes Interativos/Inteligentes de Aprendizagem (*Interactive/Intelligent Learning Enviroment - ILE*) ou ainda, Sistemas Tutores Cooperativos.

2.1.4 Dos STI's Clássicos para os ILE's

Um STI pode ser definido como um sistema computacional que visa reproduzir o comportamento atribuído a um professor humano competente em um domínio particular, sendo tal competência normalmente vinculada à capacidade de prover instrução individualizada ao aprendiz (adaptação dinâmica). Para tanto, um STI inclui conhecimentos para tratar as questões referentes ao problema da adaptação, questões estas mapeadas aos módulos do domínio, aprendiz e pedagógico, respectivamente (estes módulos são apresentados a seguir). Para o tratamento de tais questões os STI's clássicos envolvem domínios de estudo como IA, Psicologia Cognitiva e Pedagogia. A IA permite a representação e manipulação dos conhecimentos, a Psicologia Cognitiva oferece suporte na busca do entendimento sobre indivíduos e a Pedagogia oferece métodos de ensino.

Diversos foram os trabalhos desenvolvidos na linha dos STI's e que poderiam ser aqui referenciados, como por exemplo o sistema SCHOLAR [25] para o ensino de geografia, cuja comunicação com o aprendiz acontece via um diálogo em linguagem natural. Outro exemplo é o sistema chamado PROUST, que encontra erros não sintáticos de programação Pascal feitos por programadores não experientes. Quando encontra um erro, o sistema indica a linha onde este foi encontrado e sugere como corrigi-lo, após conhecer os objetivos dos programas, descritos em uma linguagem própria do ambiente [20]. O projeto GUIDON [26] levou ao desenvolvimento de um sistema para o ensino de estratégias de diagnóstico para estudantes de medicina, a partir do sistema especialista MYCIN (descrito na seção "Inteligência Artificial" deste capítulo). Um exemplo interessante no contexto do MathTutor é o ELM-ART (ELM Adaptive Remote Tutor) [27], um STI para o ensino de programação em linguagem Lisp [28], desenvolvido para o contexto da WWW.

Arquitetura geral de um STI

Uma arquitetura bem aceita para um STI é apresentada a seguir, baseando-se nos três módulos citados anteriormente.

- Módulo do Modelo do Domínio: este módulo é responsável pela questão O que ensinar?, contendo o conhecimento especializado sobre um domínio de aplicação. Serve de base para a operacionalização das funções pedagógicas (por exemplo, resolução de problemas, diagnóstico cognitivo e instrução).
- Módulo do Modelo do Aprendiz: responsável pela questão A quem ensinar?, constitui uma representação abstrata que o sistema constrói sobre o conhecimento e desempenho do aprendiz, devendo determinar o estado cognitivo deste (o que sabe, o que não sabe, o que entendeu mal). Serve como apoio às decisões do sistema quanto a estratégias de ensino adequadas ao aprendiz em determinada situação, servindo também de ajuda ao diagnóstico e remediações em relação ao seu desempenho. Este módulo é de fundamental importância para a adaptação do sistema ao aprendiz.
- Módulo do Modelo Pedagógico: também chamado de módulo do modelo de tutor ou instrutor, é responsável pela questão Como ensinar?, estando relacionado às estratégias de ensino/aprendizagem a serem adotadas sobre o domínio de aplicação. Este módulo contém a funcionalidade básica relativa à decisão sobre qual atividade pedagógica apresentar e como apresentá-la, a partir de informações geradas pelos modelos do aprendiz e do domínio.

Na evolução para os ILE's, nota-se que as pesquisas sobre os STI's voltam-se para um estilo de interação mais flexível, considerando o aprendiz como um elemento ativo e com certa autonomia no processo de aprendizagem. Portanto, os ILE's representam uma transformação dos STI's tradicionais para atender às mudanças ocorridas, sendo o aprendiz colocado em situações de descoberta, envolvendo atividades de resolução de problemas. Enquanto nos STI's tradicionais o sistema detém sempre o controle da interação, nos ILE's a iniciativa é normalmente compartilhada com o aprendiz.

Ainda no contexto do terceiro momento da história dos Ambientes de Aprendizagem Assistidos por Computador, nota-se que, mais recentemente, tais ambientes começam a dar ênfase à tecnologia de Computação Distribuída, sendo dotados de suporte para a tecnologia CSCW (*Computer Supported Cooperative Work*). Neste mesmo período os STI's e ILE's passam a incorporar resultados da Inteligência Artificial Distribuída [29] através de uma abordagem de sistemas Multiagentes.

2.1.5 Inteligência Artificial Distribuída

A Inteligência Artificial Distribuída (IAD) representa a união de técnicas computacionais de IA e de sistemas distribuídos. Seu objetivo é estudar o conhecimento e as técnicas de raciocínio que podem ser necessárias ou úteis para que agentes computacionais participem de sociedades de agentes [29]. Em Marietto [30] encontra-se um breve histórico sobre IAD.

Enquanto a IA clássica baseia-se no comportamento individual, cuja ênfase é colocada na representação de conhecimento, a IAD baseia-se no comportamento social, enfatizando as ações e interações entre agentes. Em tal disciplina busca-se desenvolver métodos e técnicas relacionadas à solução de problemas complexos, trabalhados sob uma metáfora de inteligência coletiva, sendo que tais soluções podem envolver conhecimento em várias especialidades. Portanto, considera-se na resolução de problemas um comportamento inteligente e cooperativo por parte de suas entidades, dispostas em um sistema distribuído. A inteligência pode residir em unidades separadas (agentes) ou emergir de suas interconexões.

Algumas motivações para investimentos em IAD que podem ser apontadas são os ganhos em relação à Engenharia de *Software* em aspectos como modularidade, manutenção, paralelismo, desenvolvimento e robustez, e o potencial que pode oferecer às implementações da idéia de adaptação nos sistemas adaptativos em geral.

A IAD pode ser considerada sob dois enfoques:

- Resolução Distribuída de Problemas (*Distributed Problem Solving - DPS*): em que já se supõe a existência de um problema, para então definir os agentes segundo as necessidades de tal problema.
- Sistemas Multiagentes: para os quais não necessariamente existe um problema definido a princípio.

2.1.6 Sistemas Multiagentes

Uma definição do termo agente bem aceita pela comunidade de SMA é apresentada a seguir:

> "Um agente é definido como sendo uma entidade (real ou abstrata) capaz de agir sobre ela mesma e sobre seu ambiente, dispondo de uma representação parcial deste ambiente, podendo comunicar-se com outros agentes, e cujo comportamento é conseqüência de suas observações, de seu conhecimento e das interações com outros agentes" [31].

Existem duas abordagens para a concepção de um SMA em relação aos agentes:

- Agentes Reativos: ou não inteligentes, cujo comportamento inteligente resulta de um comportamento global inteligente, a partir dos comportamentos individuais reativos.
- Agentes Cognitivos/Intencionais: ou inteligentes, possuem representação de si mesmos, dos demais agentes e do ambiente, ao contrário dos agentes reativos. Neste modelo a ativação do sistema é resultado da comunicação direta através de troca de mensagens, baseando-se em modelos de sociedades humanas. Esta é a abordagem utilizada na concepção do ambiente MATHEMA.

A articulação dos agentes de uma sociedade buscando solucionar um problema torna necessária a consideração de alguns conceitos como organização (de um grupo de agentes, objetivando resolver um determinado problema, onde a idéia é realizar uma decomposição em subproblemas, repartidos entre os agentes envolvidos), controle (centralizado ou distribuído), cooperação (mecanismos que viabilizam as atividades cooperativas entre agentes) e comunicação (mecanismos responsáveis pela comunicação entre agentes, normalmente baseados em troca de mensagens ou compartilhamento de informações). Um sistema multiagentes oferece ganhos relacionados à complexidade, modularidade, reusabilidade e incrementabilidade do sistema.

2.2 O ambiente MATHEMA

Esta seção apresenta o ambiente MATHEMA [5], modelo utilizado para orientar o desenvolvimento do ambiente MathTutor. O MATHEMA foi definido no contexto dos sistemas IA-ED, ou seja, conforme apresentado anteriormente, sistemas de aprendizagem

assistida por computador que utilizam técnicas de IA. Mais especificamente, o MATHEMA utiliza técnicas de IAD, segundo uma abordagem multiagentes. [30, 32] apresentam arquiteturas para sistemas multiagentes, que inclusive utilizam as mesmas ferramentas para o armazenamento e manipulação do conhecimento e comunicação entre os agentes, utilizadas no presente trabalho.

Considerando-se que é necessário haver processos mais efetivos de interação entre sistema e aprendiz, e levando-se em conta também questões relacionadas ao problema da adaptação, é proposto um modelo estrutural para o domínio de conhecimento.

2.2.1 Modelagem do conhecimento sobre um domínio

O conhecimento sobre um domínio a ser modelado no ambiente MATHEMA é abordado segundo duas dimensões: uma visão externa e uma visão interna. Desta forma, busca-se uma estruturação apropriada a uma maior efetividade no processo de aprendizagem.

A visão externa mostra o conhecimento sob um aspecto de diferentes subdomínios para o domínio alvo. O particionamento é de natureza epistemológica, estando comprometido com alguma visão particular do domínio. São definidas três dimensões de conhecimento:

- Contexto (ponto de vista): esta dimensão compõe-se de diferentes contextos ou pontos de vista sobre um domínio de conhecimento, constituindo-se de representações (abordagens) diferentes de um mesmo objeto de conhecimento.
- Profundidade: dimensão relativa a um contexto particular, referindo-se a alguma forma de refinamento na linguagem de percepção, ou seja, estratificação dos vários níveis epistemológicos de percepção do objeto de conhecimento.
- Lateralidade: dimensão referente aos conhecimentos afins de suporte a um determinado objeto de conhecimento do domínio, proveniente de uma visão particular de contexto e profundidade.

Definidos os subdomínios referentes ao domínio de conhecimento tratado, deve-se analisá-los internamente como sendo constituídos por um conjunto de unidades pedagógicas, definidas em função de objetivos de ensino/aprendizagem específicos,

associados a um *curriculum*. As unidades pedagógicas são relacionadas segundo uma ordem definida com base em critérios pedagógicos e a cada uma corresponde um conjunto de problemas. A cada problema está associado um conhecimento de suporte a sua resolução, que pode incluir conceitos, exemplos, contra-exemplos, dicas etc.

A adoção de uma abordagem baseada em agentes deve-se ao fato de que tal abordagem se mostra adequada ao tratamento da complexidade envolvida no conhecimento e sua manipulação, e também devido às características do modelo do domínio.

2.2.2 A sociedade de agentes tutores artificiais

Com base na modelagem do conhecimento, a Sociedade de Agentes Tutores é definida. A cada subdomínio de um domínio D é associado um agente específico, obedecendo a seguinte relação:

$D_{ij} \rightarrow AT_{ij}$

A mesma idéia é considerada no tratamento do conhecimento lateral, ou seja, a cada visão de lateralidade é atribuído um agente:

$dl_{ijk} \rightarrow AT_{ijk}$

Portanto, a sociedade de agentes tutores artificiais (SATA) em relação a um domínio de conhecimento D é definida da seguinte maneira:

$SATA = AT \cup ATL$

onde AT significa o conjunto dos agentes tutores relativos a um domínio D, e ATL representa o conjunto dos agentes tutores associados a um domínio DL.

2.2.3 A arquitetura do ambiente MATHEMA

Para a definição da arquitetura deste ambiente considerou-se que há influência no desempenho das entidades computacionais havendo a inclusão de entidades humanas (especialmente aquelas que monitoram o desempenho da sociedade de agentes, realizando melhorias sobre esta sempre que necessário). Portanto, tal arquitetura baseia-se no princípio da integração de entidades humanas e artificiais dispostas a interagir cooperativamente, sendo definida da seguinte forma:

Arq-Mat = {AH, SATA, SEH, AI, AM, ME}

onde:

AH representa um Aprendiz Humano, um agente ativo no processo de aprendizagem que deverá agir sob assistência de um agente tutor.
SATA representa uma Sociedade de Agentes Tutores Artificiais, sendo a cada um destes agentes atribuído um subdomínio do conhecimento modelado. Estes agentes podem cooperar entre si.
SEH indica uma Sociedade de Especialistas Humanos, que é fonte de conhecimento para a SATA e realiza operações de manutenção sobre a mesma. Em caso de falha crítica da SATA, a SEH deve de algum modo prestar assistência ao aprendiz.
AI representa o Agente de Interface, encarregado da interação entre o aprendiz e a SATA e de auxiliar na definição de um agente supervisor para o aprendiz.
AM representa o Agente de Manutenção, responsável pela interface entre SEH e SATA, provendo meios para a realização de operações de manutenção e para facilitar o processo de aquisição de conhecimento.
ME indica um Motivador Externo, representando entidades humanas externas que podem motivar o aprendiz a trabalhar no MATHEMA.

A SATA é um Sistema Tutor Multiagentes (STMA), formado por uma sociedade estruturada de agentes que cooperam entre si e com a SEH, através de linguagens e protocolos estabelecidos. Um agente toma decisões quanto à necessidade de cooperação baseado na constatação de conhecimento sobre uma tarefa, ignorância parcial ou total.

Cada um destes agentes possui um conjunto de habilidades (métodos para resolver uma determinada classe de tarefas) formado por resolução de problema, diagnóstico e instrução. A composição de uma tarefa inclui a informação da habilidade e recursos necessários. Além do autoconhecimento os agentes tutores têm uma representação do conhecimento social, permitindo solicitar cooperações baseadas neste conhecimento. A figura 2.1 apresenta a arquitetura do ambiente MATHEMA.

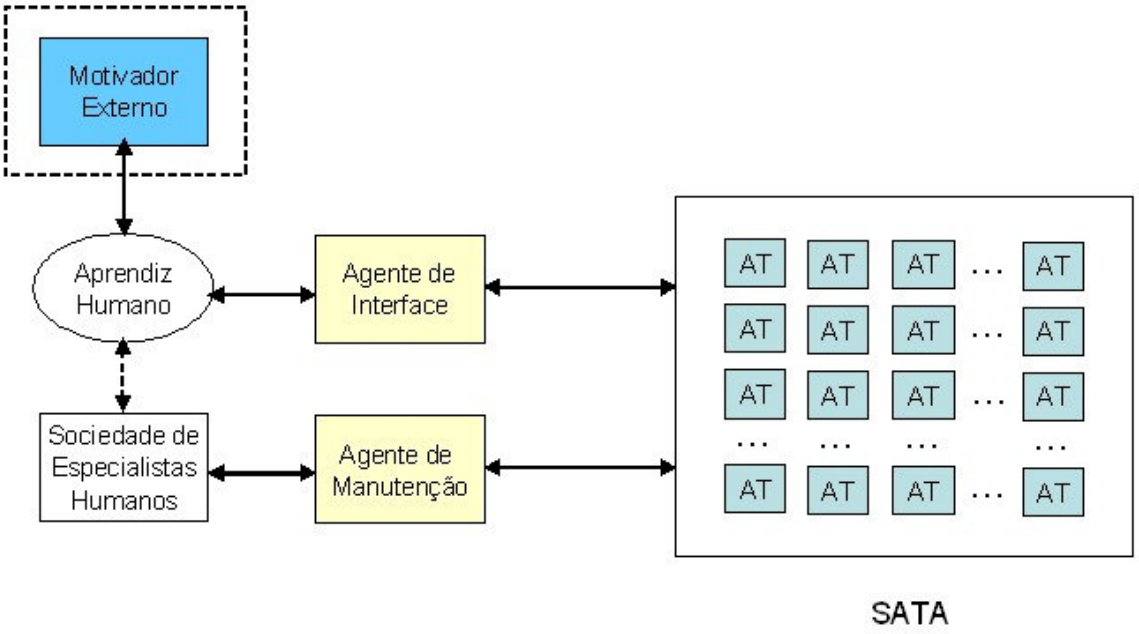


Figura 2.1 Arquitetura do ambiente MATHEMA.

Funcionalidade geral

A funcionalidade pretendida pelo MATHEMA sugere situações como esta apresentada a seguir:

Inicialmente, pode existir um aprendiz interessado em trabalhar no ambiente MATHEMA, tendo sido incentivado pelo Motivador externo. A esta situação inicial segue-se uma interação dialógica entre o aprendiz e o agente de Interface (AI), sendo que o aprendiz informa ao AI os seus objetivos. O AI por sua vez, deve prover informações sobre o ambiente computacional e deve auxiliá-lo mediante análise de seus objetivos a escolher um supervisor na SATA. A partir deste ponto, inicia-se um processo de interação cooperativa e didática entre o aprendiz e o agente Supervisor, sendo que este passa a ser o responsável pelo aprendiz. Durante a interação, situações com diferentes níveis de

complexidade podem ocorrer. Seguem alguns exemplos em ordem crescente de complexidade:

- Interação entre o aprendiz e seu Supervisor.
- Demanda por participação de outros agentes, além do Supervisor, no processo interativo.
- Agentes não conseguem atender as requisições do aprendiz.

No último caso constata-se que a sociedade de agentes não tem o conhecimento necessário para realizar determinada atividade, assim, notifica-se o aprendiz a respeito da impossibilidade, talvez apenas momentânea, de atender sua solicitação e contata-se a Sociedade dos Especialistas Humanos para que sejam feitas as intervenções necessárias para a correção do problema (que neste caso representaria a ampliação dos conhecimentos da SATA). A SEH deve ser informada via agente de Manutenção, e tendo solucionado o problema, deve notificar o agente Supervisor.

A idéia exposta anteriormente compõe o funcionamento geral da arquitetura do MATHEMA, mas um subconjunto destas interações é enfatizado no modelo, conforme mostrado na figura 2.2, a saber:

- Interações entre um aprendiz e a SATA através de um agente tutor.
- Interações entre os agentes tutores.
- Interações entre SATA e SEH.

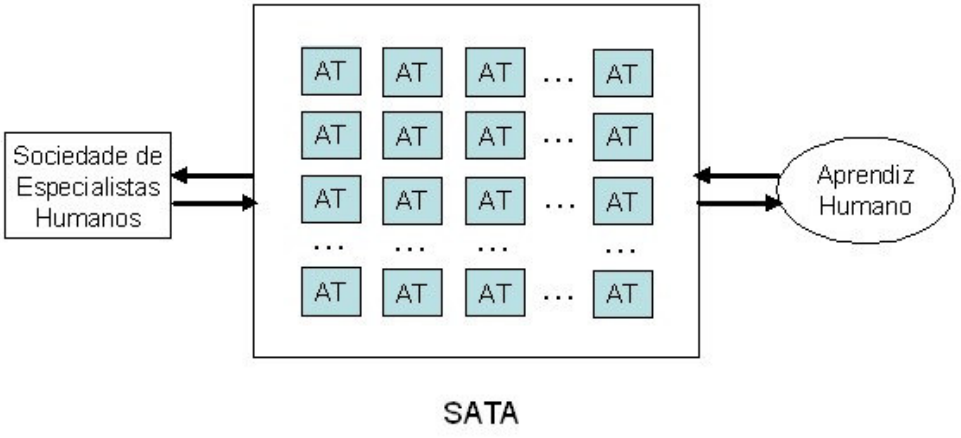


Figura 2.2 Interações no ambiente MATHEMA.

As interações dinâmicas entre um aprendiz e um agente tutor ocorrem num contexto de resolução de problemas, ativando diferentes funções pedagógicas durante o processo adaptativo. O agente responsável pela interação com o aprendiz pode solicitar a cooperação de outros agentes quando verificar que não pode realizar determinadas tarefas. As interações entre as sociedades SATA e SEH ocorrem quando um agente tutor faz alguma solicitação à SEH ou quando a SEH constata que existe necessidade de intervenções. Esta constatação pode vir da observação do processo interativo entre aprendiz e SATA, a partir de uma determinada fonte, podendo esta ser algo como um mecanismo de registro (*log*).

2.2.4 Modelo de agente tutor

A arquitetura de um agente tutor pode ser vista em dois níveis de abstração: níveis macro e micro. Segundo o nível macro um agente é composto por três componentes principais: o sistema tutor, social e de distribuição. O sistema tutor interage diretamente com o aprendiz e armazena os conhecimentos que o agente possui para efetuar operações pedagógicas no domínio de aplicação. O sistema social possui bases de conhecimento e mecanismos de raciocínio necessários ao comportamento cooperativo entre os agentes tutores. O sistema de distribuição manipula a troca de mensagens entre agentes, através do meio de comunicação.

O nível micro apresenta uma visão interior da arquitetura de um agente tutor. A figura 2.3 mostra os componentes citados no nível macro detalhados por esta visão interior.

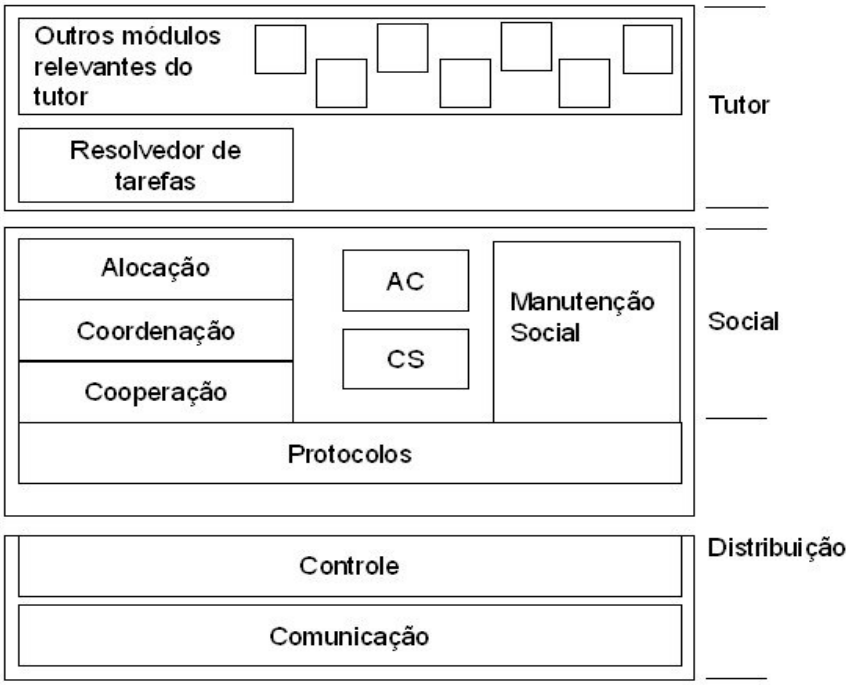


Figura 2.3 Arquitetura de um Agente Tutor: visão interna

O resolvedor de tarefas é responsável por desempenhar um conjunto de tarefas pedagógicas e identificar aquelas que não são de sua competência. O autoconhecimento é um modelo que o agente tem de si mesmo, enquanto o conhecimento social (CS) é um modelo dos outros agentes tutores. Estes módulos de conhecimento são consultados sempre que é necessário identificar um agente que possa resolver uma determinada tarefa pedagógica. Alocação é o módulo que efetua a seleção dos agentes aptos a resolver uma determinada tarefa. Coordenação é um mecanismo que interpreta a estrutura de tarefas (útil apenas em situações que envolvem tarefas decompostas). Cooperação é o módulo responsável por promover a execução de uma tarefa, encaminhando-a para algum agente da sociedade (inclusive ele próprio). Manutenção Social permite a realização de operações que podem incluir entrada, saída e atualização de agentes, com a finalidade de manter os CS's atualizados. O módulo Protocolos diz respeito às instâncias de diálogos de interação em execução. O módulo Controle é responsável pela intermediação entre o Sistema Social (através do módulo responsável pelas instâncias de diálogo em execução) e o Sistema de Distribuição. O módulo Comunicação realiza a distribuição e coleta de mensagens, fazendo a mediação entre o Sistema de Distribuição e o meio de comunicação. O modelo MATHEMA define detalhadamente a funcionalidade das atividades cooperativas.

O modelo adotado para um agente tutor baseia-se em um modelo de agentes cognitivos e é composto como especificado a seguir:

MAG = {ST, CS, AC, ME}

onde:

ST é um sistema tutor, um componente especializado em um domínio.
CS representa o conhecimento social, modelado por CS = (AC_1, AC_2, ..., AC_n) onde cada AC_k representa o autoconhecimento de um outro agente tutor da SATA.
AC representa o autoconhecimento, definido por AC = (AgentId, LH) onde AgentId é um identificador único para um agente e LH é uma lista de habilidades.
ME indica o mundo externo à SATA, sendo definido por ME = (AgInt, AgMnt) onde AgInt é o agente de interface e AgMnt é o agente de manutenção.

O MATHEMA inspira-se numa abordagem construtivista, recebendo também influências da teoria de Vygotsky [33], a respeito dos aspectos sociais envolvidos no processo de interação, combinando, em seu modelo de ensino-aprendizagem, o qual é cooperativo, a aprendizagem pela ação e por instrução. A aprendizagem pela ação enfatiza características exploratórias em situações de resolução de problemas, conduzindo a uma forma de aprendizagem por descoberta a partir da ação, inspirada de um certo modo, na categoria micromundo. A aprendizagem por instrução tem um aspecto mais instrucional, proveniente dos STI's clássicos.

Arquitetura de um sistema tutor

Formalmente, um sistema tutor é definido como:

ST = {Med, Raciocinadores, Bases de Conhecimento}

onde:

Med representa o módulo Mediador, um mecanismo pedagógico que deve reagir a cada ação vinda do aprendiz.

Raciocinadores são os mecanismos provedores de funções pedagógicas.
Bases de Conhecimento representa o módulo que mantém os conhecimentos de suporte à operacionalização dos raciocinadores, incluindo os conhecimentos sobre a estrutura pedagógica (*curriculum*), o domínio e o aprendiz.

O módulo Raciocinadores possui três submódulos: Especialista, Tutor e Modelagem do Aprendiz. O primeiro deles é um sistema especialista que realiza funções como resolução de problemas, tanto aqueles apresentados ao aprendiz quanto aqueles propostos por este. Possui os submódulos: Resolvedor de Problemas, Explicador, Diagnóstico Cognitivo e Remediação. Tutor é o módulo que seleciona os recursos pedagógicos, escolhidos sobre o *curriculum*, e realiza atividades instrucionais. O módulo de Modelagem do Aprendiz constrói o modelo do aprendiz, contando com resultados do módulo de diagnóstico cognitivo. É de sua responsabilidade identificar os acertos, conhecimentos adquiridos, erros e mau entendimentos apresentados pelo aprendiz. O módulo Mediador decide sobre o tipo de intervenção a ser realizada para atender ao aprendiz, possuindo conhecimento sobre os Raciocinadores e as Bases de Conhecimento. Este módulo tem também o papel de interação com o Sistema Social.

Cada agente possui um sistema tutor inteligente associado, responsável por interagir diretamente com o aprendiz. Neste sistema o domínio de conhecimento, o aprendiz e ainda outros componentes são modelados de forma distribuída, buscando atingir um processo educacional que gere resultados efetivos, através de uma solução eficaz para a questão da adaptabilidade do sistema ao estado cognitivo do aprendiz.

2.2.5 Modelos distribuídos

O modelo distribuído do domínio é um dos módulos mais críticos de um STI, considerado como base para se buscar melhorias no processo de interação entre aprendiz e tutor.

A modelagem do aprendiz em relação ao domínio alvo é construída também de maneira distribuída, a partir dos resultados obtidos pela participação de vários agentes no processo interativo.

Através desta estrutura baseada em múltiplos agentes que interagem entre si, pode-se realizar a tarefa de Instrução de maneira distribuída, tal como as outras atividades pedagógicas.

2.3 Projeto instrucional

As idéias apresentadas nesta seção baseiam-se em [30]. Foram selecionadas as idéias mais relevantes em relação à aplicação desenvolvida.

A expressão do processo de ensino/aprendizagem em formas computáveis encontra uma série de dificuldades, visto que é necessário considerar um grande número de variáveis que influenciam a dinâmica de um ambiente que implemente tais idéias. Por este motivo, várias pesquisas são desenvolvidas com o objetivo de desenvolver teorias instrucionais viáveis de serem implementadas. As idéias associadas a projetos instrucionais mostram-se adequadas para aplicação ao desenvolvimento de STI's por considerar a aprendizagem como um processamento de informações.

Projetos instrucionais envolvem, além de questões relacionadas à interação professor/aluno, eventos que possam ter um efeito direto ou indireto no processo de aprendizagem. Define-se como projeto instrucional a estruturação destes eventos de instrução. Em tal projeto são tratadas as condições de aprendizagem, divididas em condições internas, relacionadas aos estados mentais do aprendiz (conhecimentos previamente adquiridos) e externas, que representam as ações intencionais que objetivam favorecer o processo de aprendizagem. O estudo de tais condições ajuda a definir como ocorre o processo de aprendizagem (condições internas) e o processo de ensino (condições externas), cuja união resulta no processo de instrução.

Os eventos de instrução podem ser realizados por professores, por um STI, por materiais voltados ao ensino personalizado etc., e são independentes do domínio. A idéia é que processos internos podem ser influenciados por processos externos em situações de aprendizagem, considerando-se que cada aprendiz tem uma maneira diferenciada de processar informações.

A instrução é planejada visando alcançar um conjunto de objetivos instrucionais, que são definidos como atividades humanas que podem ser desenvolvidas em situações de aprendizagem. Algumas capacidades humanas que se considera ser necessário desenvolver

e possíveis de se realizar computacionalmente em um processo de aprendizagem, estão relacionadas ao conhecimento declarativo (saber que), conhecimento procedural (saber como fazer alguma coisa de forma intelectual) e estratégias cognitivas (estratégias pelas quais o aprendiz exerce controle sobre seu aprendizado, lembrando, repensando seu comportamento). Os conhecimentos declarativo e procedural podem ser objetivos instrucionais, ou seja, podem ser utilizados na apresentação do domínio de conhecimento tratado pelo sistema. As estratégias cognitivas podem ser utilizadas pelo STI para tomar decisões instrucionais. O conhecimento declarativo pode ser considerado como base para a formação das habilidades intelectuais e das estratégias cognitivas. As estratégias cognitivas devem monitorar e guiar o aprendizado dos outros tipos de capacidades, podem ser consideradas como uma técnica instrucional.

O aprendizado de um novo conhecimento declarativo necessita de uma estruturação que permita que tal conhecimento seja inserido de forma lógica, enquanto o aprendizado de habilidades intelectuais é influenciado pela recuperação de outras habilidades intelectuais consideradas pré-requisitos (habilidades mais simples). Existem três condições externas que devem ser propiciadas para auxiliar a aprendizagem de habilidades intelectuais:

- “Ensinar” habilidades consideradas como pré-requisitos na ordem especificada.
- Garantir que os conhecimentos e habilidades necessários sejam relembrados.
- Apresentar vários exemplos e contra-exemplos, considerando que a generalização é uma característica essencial do aprendizado de habilidades intelectuais.

Devido à dificuldade de se expressar computacionalmente diversos pontos das teorias de aprendizagem, é necessário analisar estas teorias, identificando os pontos que podem ser modelados e implementados de forma computacional.

Capítulo 3

O MathTutor

Neste capítulo apresenta-se a aplicação da proposta do modelo MATHEMA ao contexto específico do curso de Estrutura da Informação. São descritos aqui a funcionalidade do sistema implementado, a modelagem do conhecimento tratado por este sistema, a estrutura da Sociedade de Agentes Tutores Artificiais e exemplos de interação entre o sistema tutor e o aprendiz.

3.1 Comentários gerais

Esta seção apresenta as atividades que foram realizadas visando cumprir as tarefas atribuídas à equipe da Universidade Federal de Santa Catarina (UFSC), no contexto do projeto Math_Net, conforme colocado no capítulo 1. As seções seguintes descrevem estas atividades de maneira detalhada.

Quanto ao desenvolvimento de aplicações específicas com o ambiente, foi implementado um protótipo de um Sistema Tutor Inteligente para o ensino de Fundamentos da Estrutura da Informação. Esta disciplina será descrita na próxima seção. Com base nesta aplicação, realizou-se o estudo da Sociedade de Agentes de um ponto de vista interno, definindo-se a modelagem do conhecimento tratado pelo sistema. Esta modelagem foi feita de acordo com a distribuição das aulas e com o livro texto adotado. Referente à experimentação, com o objetivo de se obter novas fontes de dados para a

avaliação dos resultados alcançados a partir dos testes locais, o sistema será testado em sala de aula com os alunos da disciplina de Fundamentos da Estrutura da Informação, dando seqüência às atividades do projeto Math_Net.

3.2 O domínio da aplicação

A disciplina de Fundamentos da Estrutura da Informação é oferecida no segundo semestre do primeiro ano do Curso de Engenharia de Controle e Automação da UFSC e tem como objetivo introduzir os conceitos básicos de procedimento, processo e estrutura de dados [7]. De duas aulas semanais de duas horas cada, uma é teórica e a outra prática, realizada em laboratório equipado de computadores ligados à Internet e à disposição dos alunos. As aulas práticas são baseadas na linguagem de programação *Common Lisp* [33], e o livro adotado é "*Structure and Interpretation of Computer Programs*" [34]. Atualmente, o curso dispõe de uma página própria [35], contendo ementa, exercícios e ponteiros para outras páginas relevantes, por exemplo páginas de manuais ou livros. Esta página foi utilizada como base para o sistema, sendo fonte de materiais referentes à linguagem Lisp, de exercícios propostos aos alunos e outros materiais pedagógicos. A seguir, apresenta-se a modelagem do domínio segundo as duas formas de visualização do conhecimento sugeridas pelo modelo MATHEMA.

3.3 Modelagem do conhecimento sobre Estrutura da Informação

A modelagem multidimensional de domínio, sugerida pelo modelo, foi aplicada ao domínio tratado pelo MathTutor, sendo feita com base no livro utilizado na disciplina de Fundamentos da Estrutura da Informação e também na estrutura do curso. A seguir apresentam-se, respectivamente, as visões externa e interna do domínio modelado. Este modelo está descrito também em [7].

3.3.1 Visão externa

Naturalmente, o domínio (D) do sistema consiste em Fundamentos da Estrutura da Informação. A própria distribuição atual das aulas, divididas em teóricas e práticas, sugere os contextos sob os quais o domínio pode ser analisado:

Visão Teórica (C_1) e
Visão Prática (C_2).

As profundidades relativas a tais contextos podem ser identificadas como:

Abstração Procedural (P_{11}, P_{21}) e
Abstração de Dados (P_{12}, P_{22}),

pois em ambos os contextos a abstração procedural é mais simples, mais concreta e mesmo historicamente anterior à abstração de dados, que a incorpora e refina. Estes contextos e profundidades definem os subdomínios que constituem D:

d_{11} = Visão Teórica da Abstração Procedural
d_{21} = Visão Prática da Abstração Procedural
d_{12} = Visão Teórica da Abstração de Dados
d_{22} = Visão Prática da Abstração de Dados

Para enfocar com sucesso os conceitos envolvidos no domínio é necessário algum conhecimento adicional, as lateralidades de acordo com o modelo MATHEMA. Este conhecimento está ligado principalmente à história da Matemática, Cálculo Numérico e à Teoria da Computabilidade, no caso do contexto teórico, e à utilização de linguagens de programação em geral e da linguagem Lisp em particular, no caso do contexto prático. Associadas à visão teórica da abstração procedural (d_{11}) podemos citar, por exemplo, os seguintes tópicos:

Ordens de crescimento (L_{111}) e
Modelo de substituição (L_{112}).

Já associadas à visão prática do mesmo contexto (d_{21}) apresenta-se:

Motivos para utilizar a linguagem Lisp (L_{211}) e
Avaliação de expressões pelo interpretador Lisp (L_{212}).

Como exemplos de lateralidades da visão teórica da abstração de dados (d_{12}) tem-se:

O significado dos dados (L_{121}) e
Encapsulamento e informação escondida (L_{122}).

Finalmente, algumas lateralidades associadas à visão prática da abstração de dados (d_{22}) consistem em:

Pares, listas e memória virtual (L_{221}) e
Propriedade de fechamento (L_{222}).

3.3.2 Visão interna

Definidos os subdomínios, estes podem ser associados a um *curriculum*, identificando as unidades pedagógicas que o constituem. Com base na experiência didática acumulada e na estrutura do livro texto, chegou-se à seguinte estrutura curricular: 7 unidades pedagógicas associadas tanto à visão teórica (d_{11}) quanto à visão prática (d_{21}) da abstração procedural e 8 unidades pedagógicas associadas às visões teórica (d_{12}) e prática (d_{22}) da abstração de dados.

As 7 unidades pedagógicas associadas a d_{11} e d_{21} são:

Funções Primitivas,
Funções Compostas,
Iteração e Recursão,
Funções como Objetos Manipuláveis,
Funções como Argumentos,
Funções como Métodos Gerais e

Funções como Resultados de Funções.

Já as 8 unidades pedagógicas associadas a d_{12} e d_{22} são:

Dados Compostos,
Barreiras de Abstração,
Dados Hierárquicos,
Seqüências como Interfaces Convencionais,
Dados Simbólicos,
Múltiplas Representações para Dados Abstratos,
Programação Direcionada a Dados e Aditividade e
Sistemas com Operações Genéricas.

3.4 Funcionalidade geral

Esta seção traz alguns comentários buscando descrever, de maneira geral, a funcionalidade do MathTutor. Inicialmente, o aprendiz deve se identificar como usuário do ambiente. De posse da identificação, o sistema verifica os dados a fim de descobrir se o usuário já possui um registro. Caso o registro solicitado não seja encontrado, o potencial usuário recebe uma mensagem indicando que é necessário cadastrar-se no sistema. Logo que o cadastro for efetuado, o novo usuário recebe uma mensagem indicando que está apto a utilizar o MathTutor, ou seja, pode iniciar uma sessão de trabalho (*login*). Sempre que uma nova sessão é iniciada, o tutor recupera as informações particulares do aprendiz que a iniciou. Estas informações constituem o modelo criado pelo sistema a respeito das ações deste aprendiz, e são modificadas ao longo do processo de aprendizagem. Para o caso de um usuário novo, o sistema irá sugerir sempre o material referente à primeira unidade pedagógica, conforme o modelo do domínio, caso este usuário não faça um pedido diferente.

Após este processo inicial, um aprendiz pode ter acesso ao ambiente, conhecendo seu escopo, obtendo auxílio para sua utilização e contato com os especialistas humanos, equipe de desenvolvimento etc. A partir dai, diferentes situações podem ser provocadas por ambos, aprendiz e sistema.

3.4.1 Situação 1: O aprendiz informa suas preferências

O aprendiz pode informar ao sistema sobre suas preferências através do envio de um texto, falando sobre o assunto que gostaria de explorar, recebendo o retorno adequado, de acordo com os objetivos definidos para o processo de aprendizagem, como mostrado na figura 3.1. Ou seja, o sistema fará uma avaliação do texto enviado, identificando palavras-chave no mesmo. Mediante consulta às bases de conhecimento, compostas por fatos sobre o domínio de aplicação e diversas regras, se for constatado que o assunto solicitado faz parte de seu escopo, estando presente em alguma unidade pedagógica ou lateralidade, será retornado o material pedagógico disponível. Caso contrário, o aprendiz será informado de que não é possível responder a sua solicitação, a qual será analisada pela SEH e se for relevante para o domínio, será e acrescentada ao sistema. Todo o material acessado por um aprendiz é "observado" pelo tutor e passa a fazer a parte do modelo do aprendiz, ou seja, toda a trajetória percorrida no sistema é armazenada, pois esta informação é de grande importância para a análise de seu desempenho e de seu perfil.

A consideração do perfil de aprendizagem de um aprendiz é importante, pois cada um apresenta um ritmo próprio de desenvolver atividades, tem preferência por um determinado estilo de apresentação de material, tem níveis diferentes de conhecimento e dificuldades. Por estas razões, a trajetória fornece dados interessantes, através dela pode-se descobrir quais os tópicos que foram mais acessados, exercícios que ocasionaram maior número de erros, pode indicar ainda se o aprendiz não costuma acessar muitos exemplos, indo mais diretamente à prática etc.

Figura 3.1 Exploração do ambiente com base nas preferências do aprendiz.

3.4.2 Situação 2: O aprendiz define como quer explorar o ambiente, sem interferências do Tutor

O ambiente proporciona também que o aprendiz siga uma ordem qualquer para consulta ao material disponível, apresentando-lhe todos os ponteiros para os tópicos tratados. Mas ainda assim, toda a trajetória é armazenada, pois sempre que um aprendiz inicia uma sessão de trabalho, o tutor faz uma análise do seu modelo, buscando alertar sobre falhas e orientar a continuação do processo. A figura 3.2 mostra esta situação.

Figura 3.2 Exploração livre.

3.4.3 Situação 3: O aprendiz é orientado pelo MathTutor

O MathTutor é capaz de intervir nas atitudes do aprendiz devido às "observações" que efetua sobre o mesmo, como apresentado na figura 3.3. Com base nas conclusões a que chega o sistema, podem ser sugeridos tópicos específicos, ou atividades (leitura, exercícios etc.). Esta intervenção ocorre da seguinte maneira: considerando os aspectos colocados na situação 1, o MathTutor sugere o material ou atividade que seja mais adequado à situação atual. Por exemplo, se o aprendiz apresenta dificuldades ao solucionar um problema utilizando Lisp, o tutor sugere o acesso às lateralidades que tratam o assunto.

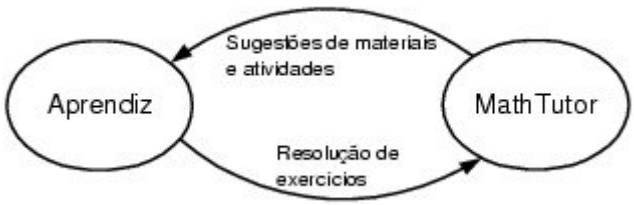


Figura 3.3 Exploração orientada pelo MathTutor.

3.4.4 Exploração das habilidades da SATA

Conforme citado no capítulo 2, os agentes que compõem a SATA têm diferentes habilidades (capacidade para executar atividades pedagógicas): instrução, diagnóstico e resolução de problemas. Para um aprendiz iniciante o tutor sempre irá sugerir uma determinada seqüência para a exploração destas diferentes habilidades. É aconselhável que se leia um determinado texto que expõe conceitos teóricos a respeito de um determinado assunto. Em seguida, com uma suposta base teórica suficiente, seria interessante acessar alguns exemplos que apliquem esta teoria, o que, conforme discutido em alguns paradigmas educacionais, torna o aprendizado mais efetivo, considerando-se que conceitos desconectados de alguma prática ou realidade são mais difíceis de ser realmente aprendidos.

Após a exploração das habilidades referentes ao contexto teórico do assunto tratado, passa-se às habilidades de diagnóstico, quando o aprendiz é solicitado a criar soluções para um determinado problema e submetê-las para avaliação, que neste caso significa a compilação e execução de algoritmos pelo CLisp [28] através do sistema tutor. Ou seja, o MathTutor é capaz de simular um compilador da Linguagem CLisp. Tanto a ativação do compilador quanto o retorno de resultados ao aprendiz, ocorrem através da interface do tutor, não sendo necessário nenhuma troca de ambiente para a execução das diversas atividades pedagógicas permitidas.

A ativação das habilidades de diagnóstico pode ocorrer a partir das seguintes situações:

- Por vontade do próprio aprendiz, que pode selecionar os ponteiros que lhe interessam. Se por algum motivo o tutor lhe sugere algo diferente de um exercício, ele pode buscar o ponteiro desejado e acessá-lo.
- Por decisão do tutor, após análise do modelo do aprendiz, concluindo que os conceitos teóricos necessários já foram apresentados.
- Por uma “pergunta” indireta ao aprendiz. Neste caso, um ponteiro referente a exercícios sobre o tópico em estudo é sempre apresentado, dando ao aprendiz a possibilidade de selecioná-lo a qualquer momento.

Mas vale ressaltar que esta seqüência não é obrigatória, é apenas aconselhável, pois situações diversas podem ocorrer devido aos diferentes perfis de usuário envolvidos no processo. Situações possíveis são apresentadas a seguir:

- Se o aprendiz considera necessário apenas ler um determinado texto e partir diretamente para os exercícios, ele pode tomar tal atitude. O tutor irá observar estes fatos, ou seja, se detecta que um aprendiz geralmente não acessa exemplos e que mesmo assim tem um bom aproveitamento, ele deixará de sugerir exemplos para este aprendiz (pois isto parece desnecessário). O tutor pode agir assim até que esta situação mude e os exemplos possam auxiliar o processo, o que é possível em tópicos que envolvam maior complexidade em seus conceitos. O fato de se deixar de sugerir exemplos, no contexto anterior, pode se fundamentar também em um ponto da teoria de projetos instrucionais, que diz que um dos fatores que aumentam a percepção seletiva é o significado que tem uma determinada tarefa para um indivíduo.
- O aprendiz pode até mesmo não acessar nenhuma das habilidades de instrução dos agentes, fazendo somente os exercícios sugeridos. Isto pode ocorrer, por exemplo, com um aprendiz que resolve utilizar o MathTutor e já conhece parte dos conceitos trabalhados, e devido a este conhecimento prévio, para algumas unidades pedagógicas considere necessário apenas fazer os exercícios.
- Se o aprendiz não costuma acessar exemplos, mas apresenta muitos erros ao resolver exercícios, o tutor deverá orientá-lo a explorar melhor as habilidades de instrução.

Existem ainda as habilidades de resolução de problemas, que sofrem restrições quanto à proposta do modelo, devido a algumas particularidades do domínio considerado neste trabalho. O modelo MATHEMA possui várias características que se adaptam perfeitamente a domínios formais, criando sistemas capazes de trabalhar graciosamente com questões da Matemática, por exemplo. Mas para o caso de Estrutura da Informação, algumas destas características se tornam difíceis de serem implementadas, pois este não é um domínio que obedece a padrões rigorosos. Torna-se difícil avaliar as respostas dos aprendizes aos problemas sugeridos pelo tutor, é também difícil apresentar soluções aos problemas propostos pelos alunos, são muitas as variáveis a serem consideradas. Sintetizar

e formalizar o conhecimento de um professor que cria soluções algorítmicas e também analisa estas soluções, não é uma tarefa trivial. Portanto, levou-se em consideração neste trabalho todos estes fatos, o que fez do MathTutor um sistema baseado no MATHEMA, mas com várias características próprias.

Basicamente, o aprendiz solicita que o sistema tutor lhe apresente soluções algorítmicas para um determinado problema. É esperado que este problema faça parte de uma biblioteca, onde são armazenados os problemas e soluções conhecidos pelo tutor. Esta solicitação é feita enviando-se um texto ao sistema, contendo a descrição do problema que se deseja ver solucionado. O texto é avaliado mediante a identificação de palavras-chave e o tutor então verifica a biblioteca de funções, através das regras em suas bases de conhecimento, a fim de atender ao pedido. Isto pode levar às seguintes situações:

- Se for possível, o tutor então retorna através da interface, uma implementação em linguagem de programação para o problema. As soluções consistem em funções pré-programadas, armazenadas em uma biblioteca, conforme citado anteriormente. Pois sabe-se que "ensinar" programação a um agente envolveria uma complexidade considerável, pelo menos para o domínio aqui abordado, um domínio que explora diversos conceitos relativos a processos e dados computacionais.
- Caso não seja possível atender ao pedido, o aprendiz é notificado e este fato passa a constar de uma base de conhecimento especial, que se comporta como um arquivo de registros (*log*), sempre observado pelos especialistas humanos, que procuram identificar mudanças necessárias ao ambiente.

O ambiente capta grande quantidade de informações sobre as ações do aprendiz, como sua trajetória (seqüência de consulta às habilidades dos diferentes agentes tutores) e erros apresentados em resolução de exercícios. A partir destes dados coletados infere suas preferências, com base em critérios pré-estabelecidos em fatos e regras em suas bases de conhecimento. Com isto, busca-se alcançar um nível satisfatório de adaptação ao estilo de aprendizagem e necessidades do aprendiz.

A seqüência de apresentação, inicialmente sugerindo textos teóricos, exemplos e exercícios, parece adequada do ponto de vista de projetos instrucionais, pelo fato de que um exemplo após um texto proporciona uma certa repetição, fazendo com que um conteúdo seja lembrado, e os exercícios podem efetuar uma associação com materiais previamente aprendidos.

3.5 A sociedade de agentes tutores

Conforme o modelo adotado, a cada subdomínio associa-se um agente tutor dentro da sociedade de agentes. Tal agente é então responsável por desempenhar as atividades pedagógicas adequadas no que diz respeito ao subdomínio sob sua responsabilidade. Para o domínio aqui modelado, os agentes foram definidos como é apresentado na tabela 3.1.

Agente	**Apelido**
AT_{11}	TProcedural
AT_{21}	PProcedural
AT_{12}	TDados
AT_{22}	PDados

Tabela 3.1 Identificação dos agentes.

Conforme o MATHEMA, os conhecimentos laterais associados ao domínio alvo também recebem um agente específico para tratá-los. Na sociedade definida para atuar sobre Estrutura da Informação estão presentes os seguintes agentes "laterais":

$$AT_{111}, AT_{112}, AT_{211}, AT_{212}, AT_{121}, AT_{122}, AT_{221}, AT_{222}$$

Portanto, a sociedade de agentes tutores para esta aplicação é definida como segue:

$$SATA = \{AT_{11}, AT_{21}, AT_{12}, AT_{22}, AT_{111}, AT_{112}, AT_{211}, AT_{212}, AT_{121}, AT_{122}, AT_{221}, AT_{222}\}$$

De acordo com a linguagem social definida pelo MATHEMA e com a modelagem do conhecimento apresentada na seção 3.3, mostra-se a seguir a representação do autoconhecimento e habilidades de dois agentes do MathTutor. Os detalhes sobre a implementação do sistema são apresentados no capítulo 4. A base de conhecimento dos agentes foi implementada em JESS [36] – esta ferramenta é discutida no capítulo 4.

Inicialmente o *template* Agente define as informações que caracterizam esta entidade. Cada agente tem um identificador (ID), um apelido e algumas habilidades. A cada habilidade associa-se uma lista de identificações que representam os assuntos que fazem parte do conhecimento daquele agente.

```
(deftemplate Agente
  (slot ID)
  (slot Apelido)
  (slot Habilidade)
  (multislot Identificacoes))
```

Neste exemplo, o agente AT_{11}, responsável pelo subdomínio d_{11}, possui a habilidade instrução para as 7 unidades pedagógicas associadas a d_{11} e d_{21}. Na seqüência é apresentado o significado para cada identificação.

InstFuncPrim: texto sobre o assunto da unidade pedagógica Funções Primitivas.
ExFuncPrim: exemplo sobre Funções Primitivas.
InstFuncComp: texto sobre o assunto da unidade pedagógica Funções Compostas.
ExFuncComp: exemplo sobre Funções Compostas.
InstItRec: texto sobre o assunto da unidade pedagógica Iteração e Recursão.
ExItRec: exemplo sobre Iteração e Recursão.
InstFuncObj: texto sobre o assunto da unidade pedagógica Funções como Objetos Manipuláveis.
ExFuncObj: exemplo sobre Funções como Objetos Manipuláveis.
InstFuncArg: texto sobre o assunto da unidade pedagógica Funções como Argumentos.
ExFuncArg: exemplo sobre Funções como Argumentos.
InstFuncMGerais: texto sobre o assunto da unidade pedagógica Funções como Métodos Gerais.
ExFuncMGerais: exemplo sobre Funções como Métodos Gerais.
InstFuncRFunc: texto sobre o assunto da unidade pedagógica Funções como Resultados de Funções.
ExFuncRFunc: exemplo sobre Funções como Resultados de Funções.

O conhecimento social de um agente traz informações sobre o conhecimento dos outros agentes da sociedade. Se um agente não pode resolver sozinho uma determinada tarefa, ele usa estas informações para identificar aqueles que podem cooperar com a busca da solução.

```
(Agente
     (ID AT11)
     (Apelido TProcedural)
     (Habilidade Instrucao)
     (Identificacoes InstFuncPrim ExFuncPrim InstFuncComp
          ExFuncComp InstItRec ExItRec InstFuncObj ExFuncObj
          InstFuncArg ExFuncArg InstFuncMGerais ExFuncMGerais
          InstFuncRFunc ExFuncRFunc))
```

O agente AT_{21}, responsável pelo subdomínio d_{21}, possui as habilidades diagnóstico e resolução de problemas para as 7 unidades pedagógicas associadas a d_{11} e d_{21}. Cada identificação é explicada a seguir. Para a habilidade diagnóstico tem-se:

ExercFuncPrim: exercício sobre Funções Primitivas.
ExercFuncComp: exercício sobre Funções Compostas.
ExercItRec: exercício sobre Iteração e Recursão.
ExercFuncObj: exercício sobre Funções como Objetos Manipuláveis.
ExercFuncArg: exercício sobre Funções como Argumentos.
ExercFuncMGerais: exercício sobre Funções como Métodos Gerais.
ExercFuncRFunc: exercício sobre Funções como Resultados de Funções.

```
(Agente
     (ID AT21)
     (Apelido PProcedural)
     (Habilidade Diagnostico)
     (Identificacoes ExercFuncPrim ExercFuncComp ExercItRec
          ExercFuncObj ExercFuncArg ExercFuncMGerais
          ExercFuncRFunc))
```

As identificações para a habilidade resolução de problemas são:

ProbFuncComp: problema sobre Funções Compostas.
ProbItRec: problema sobre Iteração e Recursão.

```
(Agente
     (ID AT21)
     (Apelido PProcedural)
     (Habilidade Resolucao)
     (Identificacoes ProbFuncComp ProbItRec))
```

3.6 Exemplo de interação entre o MathTutor e o aprendiz

O aprendiz interessado em interagir com o MathTutor poderá solicitar a realização de diferentes atividades pedagógicas através das habilidades dos sistemas tutores presentes na SATA.

O processo interativo, para o tratamento de cada uma das unidades pedagógicas pode ser iniciado com uma apresentação teórica, seguida de um exemplo e de um exercício. O sistema armazena grande quantidade de informações a respeito das interações de um aprendiz e utiliza estas informações para inferir suas preferências. Este novo conhecimento é utilizado em interações futuras.

A apresentação teórica é realizada através das habilidades de Instrução dos agentes. No presente caso, considerando-se como exemplo a unidade pedagógica referente a Iteração e Recursão, conforme observa-se na modelagem do conhecimento do domínio, estas habilidades estão associadas ao agente responsável pela Visão Teórica da Abstração Procedural (subdomínio d_{11}), para o caso da profundidade Abstração Procedural. Um exemplo de habilidade de Instrução seria a apresentação, através da interface, de um texto introdutório a respeito do tópico em questão, como mostrado na figura 3.4.

A um mesmo tema podem estar associadas diversas habilidades referentes a apresentações teóricas, mais ou menos detalhadas. Outro tipo de habilidade seria a apresentação de um exemplo, como apresentado na figura 3.5.

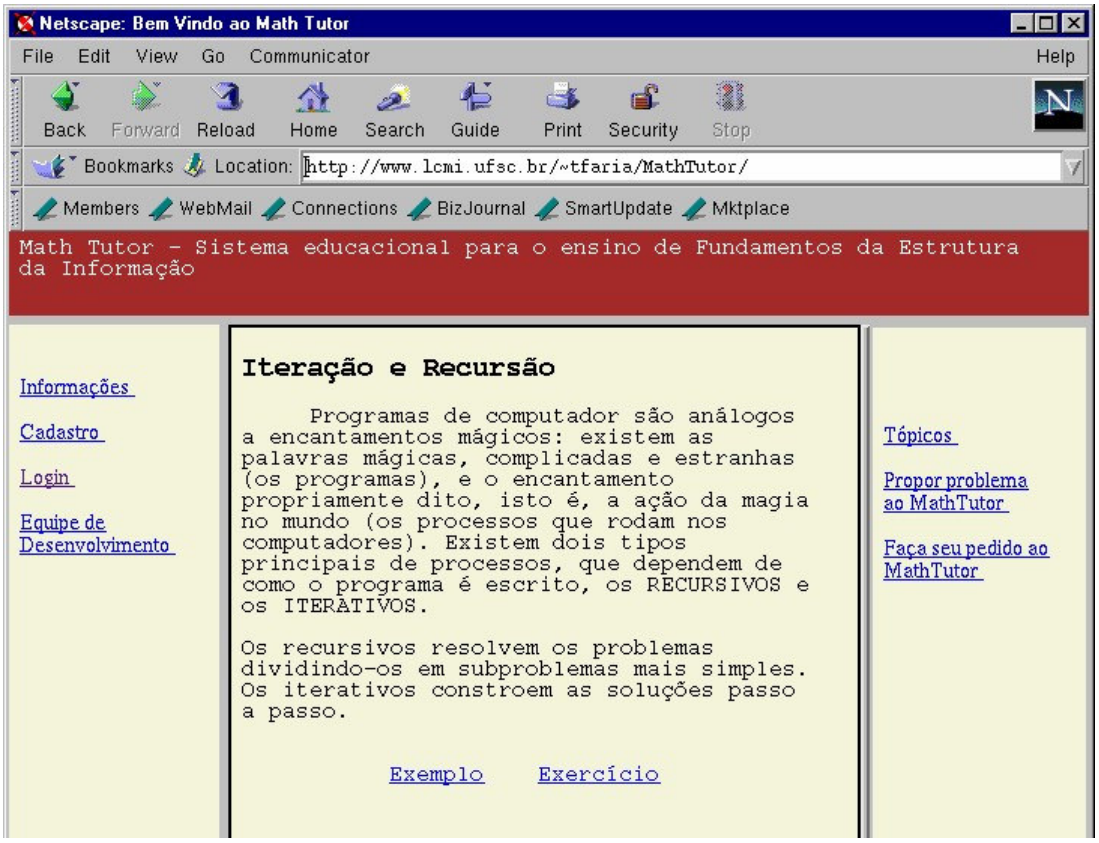


Figura 3.4 Habilidade de Instrução sob a forma de um texto teórico.

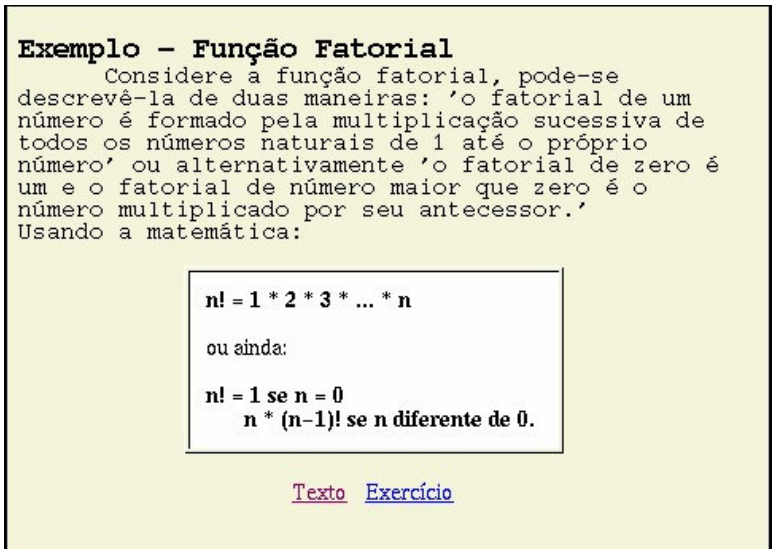


Figura 3.5 Habilidade de Instrução sob a forma de um exemplo.

Depois da apresentação teórica e de alguns exemplos, o agente responsável pelo subdomínio d_{11}, de acordo com o modelo do aprendiz e com critérios pré-estabelecidos em

suas bases de conhecimento, pode julgar adequado enviar uma mensagem para o agente responsável pela Visão Prática da Abstração Procedural (subdomínio d_{21}), de maneira que este proceda aos exercícios. Esta decisão pode resultar de uma iniciativa do aprendiz, de uma consulta ao aprendiz sobre seu interesse em continuar a interação ou do próprio agente, por considerar que o aprendiz já possui a base teórica necessária. Um exercício possível neste contexto é mostrado na figura 3.6.

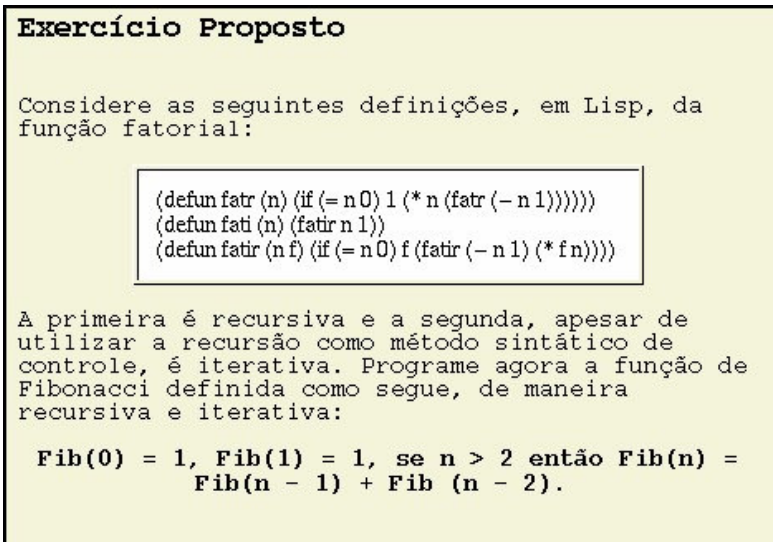


Figura 3.6 Apresentação de um exercício.

O aprendiz pode solicitar uma explicação teórica sobre a utilização da recursão como controle para a iteração, sendo neste caso, novamente requerida a cooperação do agente responsável pela Visão Teórica da Abstração Procedural. O aprendiz pode também resolver o problema e solicitar ao sistema tutor que avalie a solução que ele criou, sendo tal avaliação relacionada às habilidades de Diagnóstico. O diagnóstico é definido a partir da execução do algoritmo proposto pelo aprendiz (para o MathTutor adota-se a linguagem *Common Lisp* para a criação dos algoritmos). As figuras 3.7, 3.8 e 3.9 trazem um exemplo de utilização das habilidades de Diagnóstico.

Basicamente, o aprendiz envia ao sistema um algoritmo escrito em linguagem Lisp (figura 3.7), que é então submetido ao compilador para avaliação. Após a compilação do algoritmo, é retornado ao aprendiz o resultado desta ação. Este resultado pode ser um retorno de função sintaticamente correta (figura 3.8) e portanto aceita pelo CLisp, ou um erro (figura 3.9), ambos apresentados como parte de uma página HTML, através da interface. Se a função foi definida, então são retornados também os dados referentes ao

resultado de sua execução. Alguns erros lógicos também são tratados e quando ocorrem, um texto explicativo a respeito é apresentado.

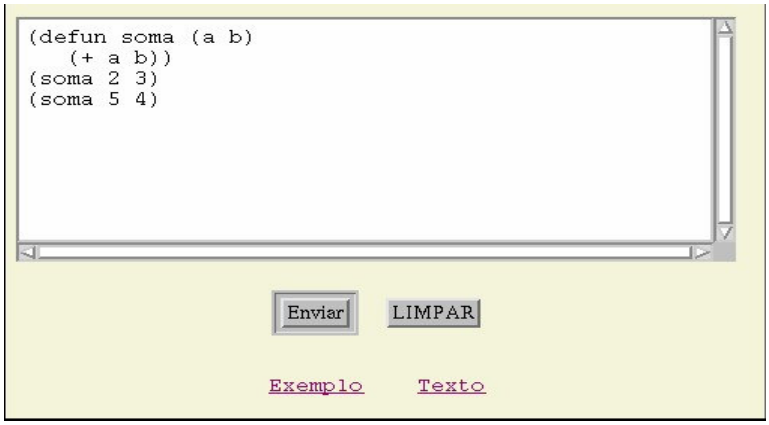


Figura 3.7 Envio de dados que ativam as habilidades de Diagnóstico.

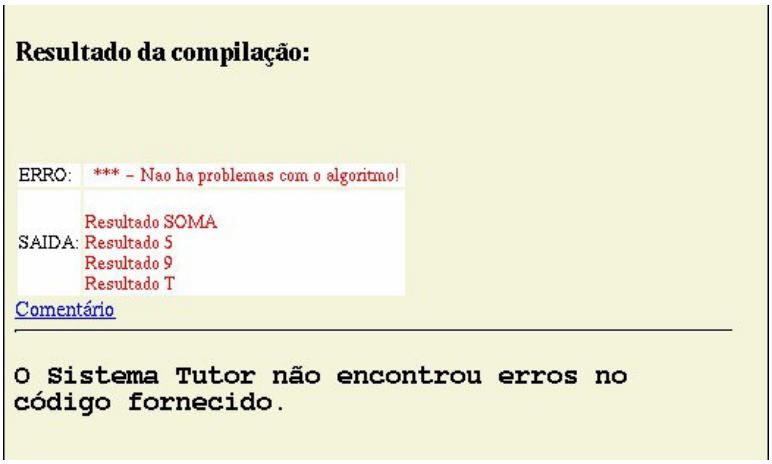


Figura 3.8 Retorno enviado ao aprendiz após a realização do diagnóstico.

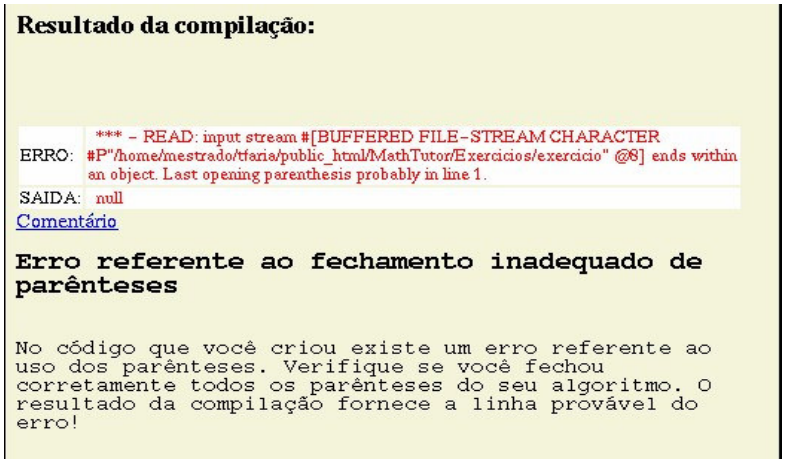


Figura 3.9 Retorno enviado ao aprendiz, informando erro no código.

O aprendiz pode também sentir-se inseguro quanto à codificação do exemplo em Lisp ou sobre sua utilização para a realização do exercício, neste caso seria ativada alguma lateralidade adequada, isto é, uma habilidade de instrução sobre algum conhecimento que não pertence diretamente ao domínio da aplicação.

Outra situação possível é a solicitação por parte do aprendiz que o sistema tutor resolva um determinado problema, o que acionaria alguma das habilidades de Resolução de Problemas. Como explicado anteriormente, o problema solicitado deve ser conhecido pelo sistema, fazendo parte de algo como uma biblioteca de problemas. Os diferentes agentes podem cooperar entre si buscando alcançar a solução. Caso não se tenha o conhecimento necessário para tal atividade notifica-se o aprendiz e contata-se a Sociedade dos Especialistas Humanos. A figura 3.10 mostra um exemplo de utilização das habilidades de Resolução de Problemas, contendo resposta à solicitação de uma função que calcule o *j-ésimo* número da *i-ésima* linha do Triângulo de Pascal.

No Triângulo de Pascal os números que ficam nos lados do triângulo são iguais a um, e os números internos são a soma dos dois números diretamente acima deles. Veja a figura abaixo do início do Triângulo de Pascal:

```
    1
   1 1
  1 2 1
 1 3 3 1
```

A função abaixo fornece a implementação da fórmula do Triângulo de Pascal onde linha representa a linha e coluna representa a coluna. Apesar de a implementação parecer perfeita, ela contem um erro muito frequente. Este erro esta na formulação de suas condições. Repare que não podemos tomar um valor para a coluna maior que o valor para a linha, pois assim existiriam números fora do Triângulo de Pascal. A seguir apresentamos a implementação contendo o erro citado:

```
(defun tp ( linha coluna )
   (cond   ((= 1 coluna ) 1)
           ((= linha coluna ) 1)
(t (+ (tp (- linha 1) (- coluna 1))
      (tp (- linha 1) coluna )))))
```

O algoritmo a seguir mostra a implementação correta desta função. Para corrigí-la acrescentamos uma condição que retorna um aviso em caso de valores de coluna maiores que valores de linha.

```
(defun tp ( linha coluna )
   (cond   ((> coluna linha ) "O valor da coluna nao pode ser maior que o valor da
linha" )
           ((= coluna 1) 1)
           ((= coluna linha ) 1)
(t (+ (tp (- linha 1) (- coluna 1))
      (tp (- linha 1) coluna )))))
```

Figura 3.10 Retorno para um pedido de resolução de um problema.

A cada passo do processo interativo, através dos dados coletados sobre o aprendiz - como sua trajetória (materiais acessados) e erros em exercícios -, o sistema tutor infere suas preferências e necessidades, sugerindo sempre uma atividade adequada para a seqüência do processo. Esta sugestão pode não ser seguida pelo aprendiz, caso ele prefira explorar outros tópicos, o que também é permitido. A figura 3.11 mostra este exemplo de interação através de gráficos.

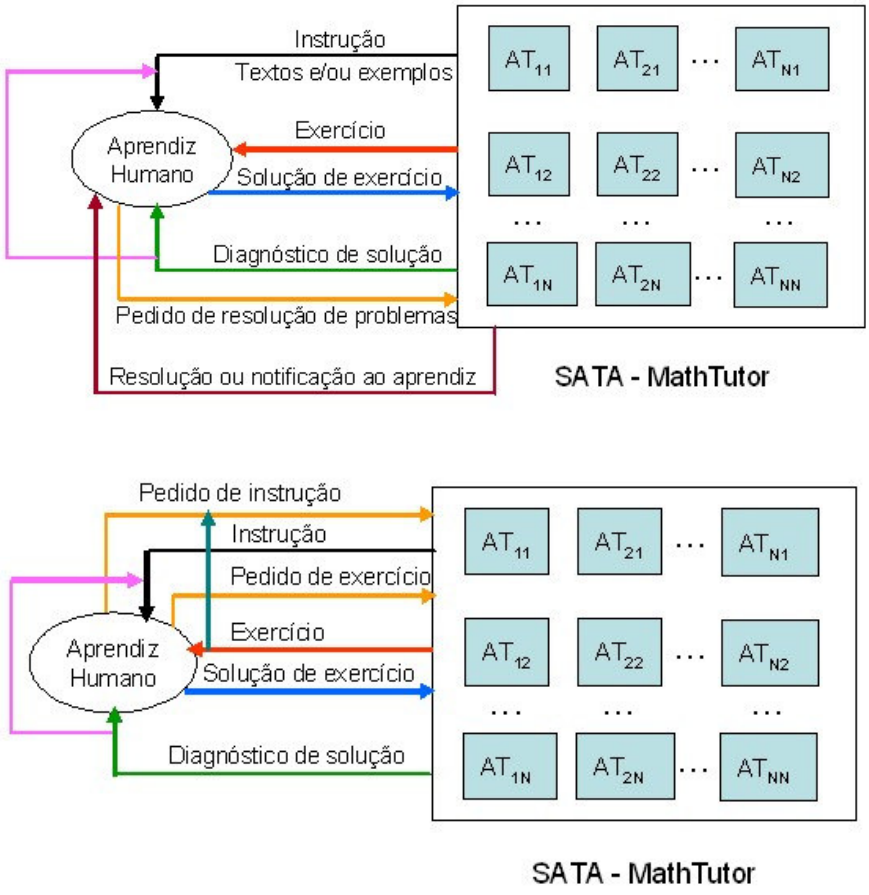


Figura 3.11 Interações entre o MathTutor e um aprendiz .

3.7 Cooperação entre os agentes

Apresenta-se nesta seção algumas situações em que é necessário haver cooperação entre os agentes do MathTutor.

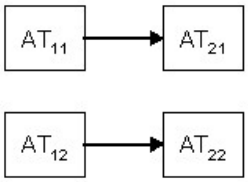
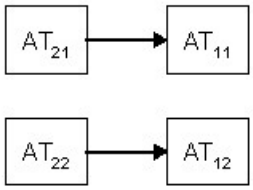
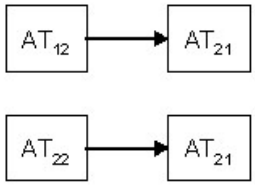
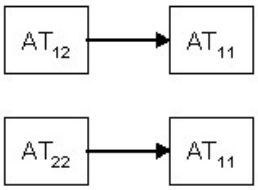
Agentes envolvidos	**Situações**
	Solicitação de exemplos quando o aprendiz pede mais informações.
	Solicitação de instruções específicas baseadas no diagnóstico de erros das respostas do aprendiz.
	Solicitação de exemplos de procedimentos quando o aprendiz tem dificuldade com os aspectos procedurais de construtores e seletores.
	Solicitação de instruções específicas sobre os aspectos procedurais de construtores e seletores.

Tabela 3.2 Situações de cooperação.

Capítulo 4

Desenvolvimento do MathTutor

Devido às características apresentadas pelo contexto de aplicação (o estudo de Estrutura da Informação utilizando a linguagem de programação Lisp), e também pelo objetivo de se aplicar o sistema implementado a cursos a distância, a interação entre diversas ferramentas constitui um requisito para a implementação.

Inicialmente, buscou-se definir a interface através da utilização de *applets* Java, mas devido ao seu modelo de segurança, encontrou-se uma série de dificuldades que levaram à conclusão de que esta não seria uma opção adequada. Quando uma interface é definida através de *applets*, os arquivos que implementam o sistema são carregados para a máquina do cliente que o acessa via internet em um servidor. Então todo o processamento é feito do lado do cliente, utilizando seu sistema de arquivos. Por esta razão, existe um modelo de segurança que policia o acesso ao sistema de arquivos, caso contrário, o cliente estaria exposto às ações de códigos estranhos, que poderiam se instalar e danificar sua máquina. Adequar um sistema a este modelo não é algo trivial, envolvendo questões como assinaturas digitais [37]. Além disso, em um sistema complexo, os arquivos que implementam *applets* são normalmente muito grandes, o que torna o processo de carregamento (*download*) muito demorado para o usuário.

Por estas razões, passou-se a considerar o uso de *servlets* [38], uma tecnologia Java que representa uma alternativa interessante na criação de interfaces para sistemas aplicados à *web*, que proporcionou soluções satisfatórias e eficientes. *Servlets* são classes Java que constituem uma aplicação que faz todo o processamento do lado do servidor, sem a

necessidade do uso do sistema de arquivos do cliente. A interface pode ser implementada em linguagem HTML, tendo seu código escrito dentro das próprias classes Java que implementam o sistema. Ao contrário, em *applets*, quando se faz necessário executar uma aplicação via *web*, cria-se uma página comum e esta página invoca uma aplicação em forma de *applet*.

4.1 Ferramentas utilizadas

As ferramentas utilizadas no projeto foram escolhidas levando em conta os seguintes critérios: serem de domínio público, apresentarem uma ampla utilização contando com listas de usuários e boa documentação, e serem portáveis para diversas plataformas. As ferramentas escolhidas foram:

- Java [39]: considerada a principal linguagem para aplicações utilizando a Internet. Adequada também ao desenvolvimento de aplicações para uso local, é uma ferramenta que dispõe de poderosos recursos, além de ser independente de plataforma.
- JESS (*Java Expert System Shell*) [36]: arcabouço para sistemas especialistas baseado no sistema CLIPS [40], implementado em Java. Suporta o desenvolvimento de sistemas especialistas baseados em regras. Em termos simples, isto significa que a proposta do JESS é aplicar continuamente um conjunto de regras (que podem ser comparadas a sentenças *if-then*) a um conjunto de dados (lista de fatos). Esta ferramenta utiliza-se de um método muito eficiente, conhecido como algoritmo *Rete* – latin para *net* – para controle do sistema. Possibilita fácil integração com outros sistemas desenvolvidos em Java, e através dela foram desenvolvidos os sistemas especialistas associados a cada agente do MathTutor.
- JAtLite (*Java Agent Template, Lite*) [41]: implementação em Java da linguagem KQML (*Knowledge Query and Manipulation Language*) [42], utilizada na comunicação entre agentes. É um pacote de programas que permite aos usuários criar facilmente novos agentes de *software* que se comunicam robustamente sobre a internet. Provê uma infra-estrutura básica na qual agentes se registram em um roteador/facilitador usando nome e senha, se conectam e desconectam da internet,

enviam e recebem mensagens, transferem arquivos, e invocam outros programas ou funções nos vários computadores onde estão sendo executados. Esta estrutura básica pode então ser expandida para se adequar a um problema específico. Os agentes criados com o JATLite não necessariamente são inteligentes, para assim serem é necessário incorporar alguma técnica de IA. No presente caso, utilizou-se a técnica de sistemas especialistas (criados a partir do JESS).

- CLisp [28]: implementação do padrão *Common Lisp*. Esta é a linguagem adotada no Curso de Fundamentos da Estrutura da Informação, encontra-se disponível para diversas plataformas. Esta ferramenta é utilizada pelo sistema tutor para realizar a compilação e execução dos algoritmos propostos pelos aprendizes.

4.2 Implementação

Com base na arquitetura do MATHEMA, vale ressaltar que o módulo referente ao Sistema de Distribuição de um agente tutor tem as suas funções realizadas pela ferramenta JATLite, no que se refere aos seus submódulos. O JATLite apresenta uma estrutura através de *templates*, permitindo o reaproveitamento de códigos que implementam as funcionalidades necessárias a sistemas multiagentes, como descrito na seção anterior.

Diversas outros módulos do sistema tutor foram influenciados pelas características do domínio de aplicação, que não é um domínio formal, resultando em algumas diferenças em relação ao modelo MATHEMA. Um agente no MathTutor não contém o módulo Coordenação, visto que não é possível haver decomposição de tarefas. A figura 4.1 mostra a arquitetura de um agente considerando o que foi exposto acima.

A arquitetura geral do sistema é apresentada na figura 4.2. O sistema é composto por uma comunidade de agentes, que mantêm bases de conhecimento (BC) referentes ao domínio de aplicação, ao modelo do aprendiz e ao modelo pedagógico. Estas BC's são compostas por fatos e regras. Cada agente é responsável por uma parte do domínio de conhecimento, conforme definido na modelagem.

Os agentes necessitam interagir com o interpretador *Common Lisp*, enviando as soluções propostas pelo aprendiz e retornando os resultados da avaliação do interpretador, acrescentando comentários, exemplos etc, buscando tornar mais fácil a compreensão por parte do aprendiz.

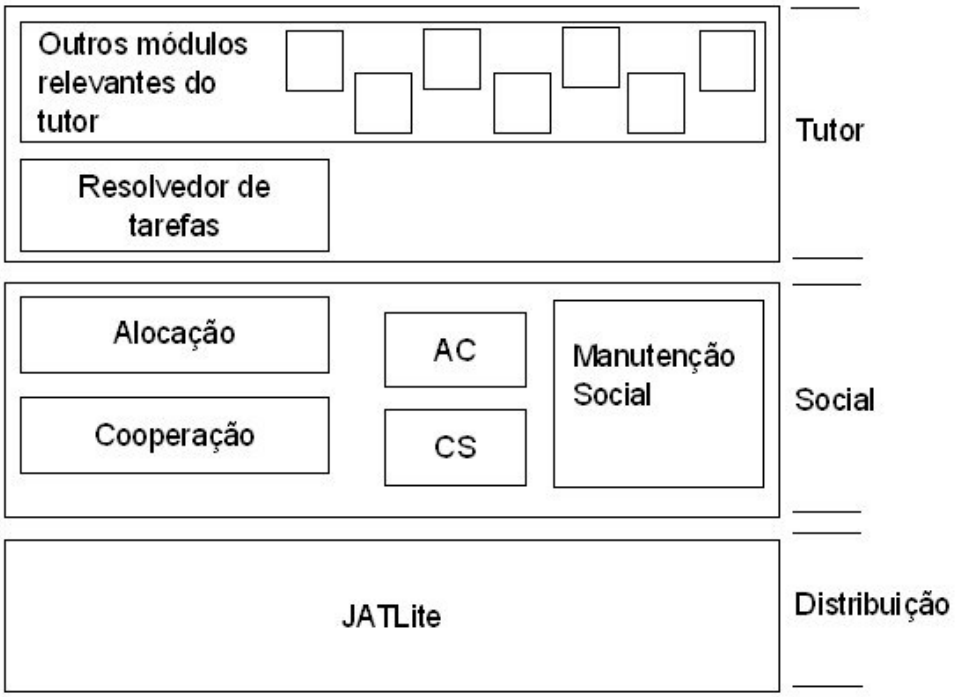


Figura 4.1 Arquitetura de um agente para o MathTutor.

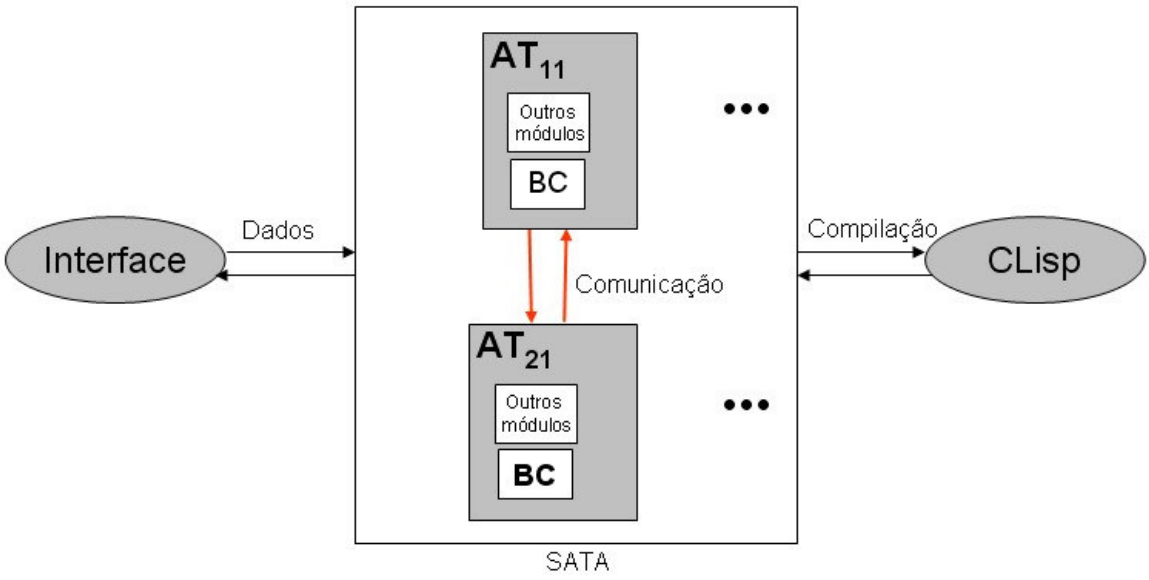


Figura 4.2 Arquitetura geral do sistema.

Todas as interações entre aprendiz e sistema tutor ocorrem via navegador *web*, portanto, a interface é implementada em linguagem HTML. A figura 4.3 mostra alguns detalhes referentes à implementação do MathTutor.

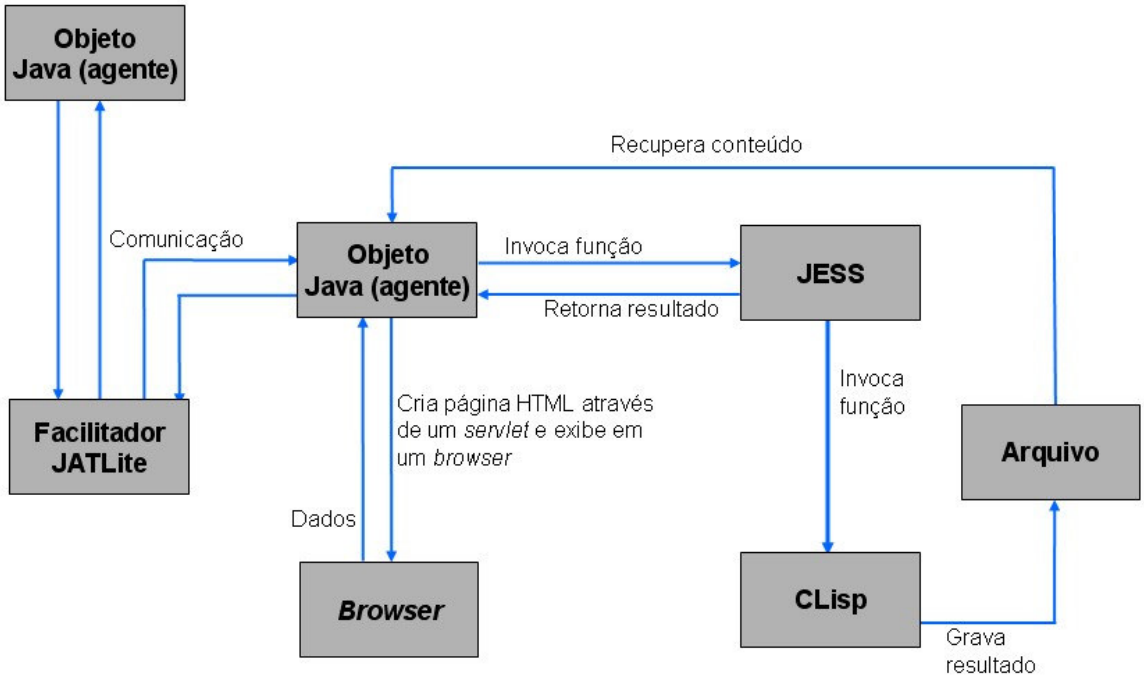


Figura 4.3 Alguns detalhes sobre a implementação do sistema tutor.

Os dados enviados ao sistema através da interface ativam diversas funções dos agentes. A partir disto pode-se manipular os dados que compõem o modelo do aprendiz e também ativar as habilidades de diagnóstico dos agentes, realizando tratamento de erros etc. Os agentes devem comunicar-se a fim de que possam solicitar cooperações entre si sempre que necessário, pois cada um é responsável pelo conhecimento referente a uma parte do domínio. As seções seguintes apresentam alguns detalhes referentes à interação entre as diversas ferramentas utilizadas na implementação.

A linguagem Java faz a interligação de todas as ferramentas utilizadas. Além de ser aplicada à construção da interface - através do uso de *servlets*, que por sua vez incorporam os códigos HTML necessários – Java é a linguagem de implementação das ferramentas utilizadas para armazenamento e manipulação do conhecimento e para a comunicação entre os agentes (JESS e JATLite respectivamente). É também indiretamente responsável pela interação do sistema com o CLisp, possibilitada pelo JESS que contém recursos para acesso a outras aplicações.

4.2.1 A construção da interface

Como já foi dito, a interface constitui um componente de difícil implementação. O aprendiz tem liberdade para explorar o conteúdo do sistema, portanto, não se tem total controle de suas atitudes, e além disto, o sistema poderá ser acessado remotamente, o que requer alguns cuidados com a maneira de se disponibilizar as informações.

A interface captura informações sobre a trajetória de um aprendiz no MathTutor, pedidos, exercícios resolvidos e outros dados necessários ao sistema especialista – que faz parte de um agente – para a realização do "raciocínio" e tomada de decisão no processo interativo. Dados são constantemente trocados entre interface e agentes. Desta interação surgem situações complexas no que se refere ao tratamento destes dados, pois cada componente tem seu comportamento específico e deve colaborar para o alcance de um objetivo comum. Após todo o processo de análise de dados recebidos, o que pode envolver a cooperação entre agentes, o resultado é enviado ao aprendiz através da interface, fazendo a exibição do material pedagógico adequado.

Para que todo o percurso realizado pelo aprendiz seja devidamente identificado e armazenado, assim como dados referentes à resolução de problemas, implementou-se diversos componentes da interface como *servlets* e não simplesmente páginas HTML. Desta forma é possível ativar funções que registrem na base de conhecimento diversas informações importantes para o modelo do aprendiz.

4.2.2 Interação entre sistema especialista e CLisp

Mais especificamente sobre a interação com o CLisp, foi preciso definir um "simulador", que faz com que o CLisp leia os dados de entrada a partir de um arquivo onde são gravadas as funções criadas pelo usuário, e realize a compilação. Após a compilação, os resultados são armazenados em arquivos específicos, sendo um destinado aos dados referentes a funções sintaticamente corretas e outro destinado às mensagens geradas devido a erros sintáticos. Estes arquivos são lidos por métodos responsáveis por enviar ao aprendiz os resultados da compilação dos algoritmos por ele apresentados e um retorno adequado sobre suas ações.

Esta interação inicia-se com a exibição de um exercício a ser resolvido, apresentando também uma área para entrada de texto, na qual o aprendiz desenvolve uma solução (conforme é mostrado no capítulo 3). O algoritmo criado pelo aprendiz é enviado ao sistema e armazenado em um arquivo-texto que servirá como entrada para o simulador. Os métodos responsáveis pela análise dos resultados da avaliação do CLisp, lêem o conteúdo dos arquivos que armazenam dados de saída do algoritmo ou erros. Com base no que foi retornado, o agente responsável faz uma explicação sobre a ação do aprendiz e sugere o material pedagógico necessário.

O tratamento de erros consiste da identificação de palavras-chave nas mensagens retornadas pelo CLisp. As palavras-chave são enviadas ao sistema especialista para que este possa fazer as devidas inferências e com base nelas organizar o conteúdo a ser retornado à interface. Esta situação faz parte das habilidades de Diagnóstico presentes nos agentes tutores.

4.2.3 Comunicação entre os agentes

As mensagens enviadas e recebidas pelos agentes que compõem a SATA se referem à troca de conhecimentos importantes para a realização de atividades cooperativas. Como proposto pela funcionalidade do sistema, se um agente não possui competência para realizar determinada tarefa, deve solicitar cooperação dos demais. Uma mensagem pode ser enviada diretamente a um agente identificado através do Conhecimento Social, ou para um agente roteador de mensagens (facilitador - componente do pacote JATLite) que é o responsável pelo encaminhamento das mensagens trocadas pelos agentes. O JATLite proporciona a implementação de um sistema cujos agentes podem estar fisicamente separados, espalhados através da internet, proporcionando vantagens quanto à modularidade e segurança do sistema, permitindo inclusive que os agentes sejam conhecidos por diversos roteadores. A idéia é que, em caso de falha ou sobrecarga de um roteador, exista uma via alternativa para que o sistema não seja prejudicado.

Os agentes trocam mensagens através de KQML [42], que é uma linguagem e protocolo para troca de informações e conhecimento. Um exemplo da interação dos agentes através deste tipo de mensagem é mostrado a seguir.

```
String mens = "(ask-one
                :sender TProcedural
                :receiver PProcedural
                :language JESS
                :content (Pedido(ExercicioItRec))";
```

Uma variável do tipo *string* armazena a mensagem cujos elementos são descritos abaixo.

ask-one: performativa, define o tipo de fala. Neste exemplo o agente TProcedural pede uma resposta ao agente PProcedural. Outros exemplos de performativas: *tell*: para informar algo ao destinatário; *reply*: para responder a uma mensagem recebida.
sender: identifica o remetente da mensagem.
receiver: identifica o destinatário da mensagem.
language: define a linguagem utilizada para expressar o conteúdo da mensagem.
content: conteúdo da mensagem. Neste exemplo um fato em linguagem JESS é enviado ao agente PProcedural, solicitando um exercício para a unidade pedagógica Iteração e Recursão.

O JATLite proporciona métodos para o envio, recebimento e tratamento de mensagens KQML.

4.3 Algumas considerações gerais

Quanto à habilidade de resolução de problemas, conforme exposto anteriormente, existem restrições em relação a sua implementação nas condições definidas pelo modelo MATHEMA, devido às características apresentadas pelo domínio do MathTutor. Tal habilidade neste modelo significa que o sistema tutor deve solucionar um problema enviado pelo aprendiz. No MathTutor estas soluções consistem de funções pré-programadas, no escopo da disciplina.

Quanto às habilidades de diagnóstico, os erros sintáticos são relativamente fáceis de se tratar. Mas quando ocorrem erros lógicos, depara-se novamente com uma questão complicada, pois este tipo de erro nem sempre gera uma mensagem, situação em que o

aprendiz ficaria sem retorno algum. Sendo assim, foi preciso realizar uma série de testes para identificar erros que podem ocasionar uma parada do CLisp, não retornando mensagem alguma ao usuário. Tais situações foram cuidadosamente verificadas e procurou-se abranger o máximo possível de hipóteses neste contexto, a fim de se alcançar um processo de tratamento de erros consistente.

Capítulo 5

Conclusões e trabalhos futuros

5.1 Comentários gerais

O presente trabalho abordou as questões referentes ao desenvolvimento de um sistema tutor para a disciplina de Fundamentos da Estrutura da Informação, baseado no modelo MATHEMA.

O protótipo desenvolvido apresenta considerável abrangência em relação às funcionalidades propostas pelo modelo utilizado. As habilidades presentes nos agentes que compõem a Sociedade de Agentes Tutores foram implementadas considerando-se as características próprias do domínio de aplicação, que por não se tratar de um domínio formal impôs algumas restrições ao sistema. Em domínios formais torna-se mais simples identificar padrões de comportamento, mas o caso de Fundamentos da Estrutura da Informação envolve a criação de algoritmos e neste contexto muitas são as possíveis soluções corretas. Alguns módulos propostos no modelo foram substituídos no MathTutor por um pacote de classes escritas em linguagem Java e que implementam a ferramenta JATLite. Esta substituição se deve ao fato de que a ferramenta corresponde às exigências do modelo e desta maneira pôde-se concentrar esforços em módulos ainda não implementados. Os temas teóricos, os exercícios e mesmo o diagnóstico de erros foi grandemente facilitado pelos dados acumulados durante as aulas da disciplina de Estrutura

da Informação atual. Alguns testes já foram realizados, proporcionando uma verificação do comportamento do sistema, tanto do ponto de vista computacional quanto educacional.

A modelagem multidimensional do conhecimento sobre um domínio pode facilitar a interação com o sistema tutor, tendo em vista a maneira pela qual se pode explorar as informações disponíveis. Como se pode observar pela modelagem apresentada e pela própria funcionalidade pretendida para o sistema, as dimensões referentes a um domínio proporcionam uma estruturação do conhecimento e um modelo de interação que permitem haver adaptação ao estado cognitivo do aprendiz. Quando o sistema tutor considerar necessário, ou por iniciativa do aprendiz, solicita-se as diversas habilidades de cada agente, proporcionando a obtenção das informações desejadas, que podem inclusive prover suporte ao domínio alvo. E ainda relacionado ao problema da adaptação existe o fato de que a modelagem do aprendiz também é feita de maneira distribuída, mantendo em cada agente informações sobre *performance* e preferências de cada aprendiz, no que diz respeito ao subdomínio sob sua responsabilidade. A seqüência de agentes consultados durante o aprendizado pode também fornecer dados valiosos para o modelo do aprendiz. Desta maneira é possível obter um modelo mais real de seu desenvolvimento, tendo por conseqüência uma aprendizagem mais efetiva.

Como já se sabe, outras equipes do projeto Math_Net têm se concentrado em estudos e desenvolvimentos baseados em tópicos específicos, conforme objetivos deste projeto. Existem investimentos relacionados a ferramentas que sirvam como auxílio à criação de agentes, no sentido de facilitar a estruturação do conhecimento em conformidade com a proposta do modelo MATHEMA. Têm sido realizados esforços também na busca de melhorias para o processo de modelagem do aprendiz, objetivando alcançar sempre maior adaptação do sistema às características específicas de cada aprendiz no processo educacional. Alguns dos trabalhos desenvolvidos pelas outras equipes deste projeto são encontrados em [43, 44, 45].

5.2 Trabalhos futuros

Existem ainda questões a serem estudadas e efetuadas quanto à implementação do sistema tutor aqui apresentado. Preocupou-se primeiramente com a criação de um protótipo que apresentasse certa abrangência quanto às funcionalidades propostas pelo modelo, ou

seja, o sistema foi sendo considerado qualitativamente. As funções existentes precisam, a partir de agora, ser aplicadas a todas as unidades pedagógicas definidas na modelagem do conhecimento.

Uma atividade já iniciada e de relevância para o ambiente desenvolvido é a integração deste a um banco de dados. No sistema aqui apresentado, os dados são sempre armazenados em arquivos simples, mas a possibilidade de utilização de um banco de dados é interessante, principalmente pelo aspecto relacionado à segurança e também à adaptação do sistema a um contexto multiusuário. Tais questões não foram solucionadas no presente trabalho devido ao fato de que o sistema envolve comunicação entre diversas ferramentas, e portanto é necessário um tratamento cuidadoso das informações trocadas. Outro fator relevante é o desenvolvimento da interface, que representa um componente crítico e de difícil construção em qualquer sistema tutor. Portanto, acredita-se que não seria viável o envolvimento de todas estas questões no escopo deste trabalho.

A realização de testes com o sistema em sala de aula com os alunos da disciplina de Estrutura da Informação também consiste de uma atividade interessante, visto que estes testes representam novas fontes de dados para avaliação do ambiente desenvolvido.

Outra questão que pode ser apontada é uma avaliação mais detalhada quanto à disponibilização e validação do sistema em sua aplicação a cursos realizados a distância. E, valendo ressaltar que estas atividades terão continuidade no âmbito do projeto Math_Net.

Referências Bibliográficas

[1] RUSSEL, S.; NORVIG, P. *Artificial Intelligence - A Modern Approach*. Prentice Hall, 1995.

[2] SLOMAN, A. *Artificial Intelligence - An Illustrative Overview*. The University of Birmingham. http://www.cs.bham.ac.uk/~axs/courses/ai.html. 2001.

[3] DAS/UFSC. *Projeto MATH-NET "Uma abordagem via sistemas multiagentes para concepção e realização de ambientes interativos de aprendizagem cooperativa assistidos por computador"*. http://www.das.ufsc.br/mathnet/. 2001.

[4] DAS/UFSC. *Projeto MATH-NET "Uma abordagem via sistemas multiagentes para concepção e realização de ambientes interativos de aprendizagem cooperativa assistidos por computador"*. http://www.das.ufsc.br/mathnet/. 2000.

[5] COSTA, E. B. *Um Modelo de Ambiente Interativo de Aprendizagem Baseado numa Arquitetura Multiagentes*. Campina Grande, PB, 1997. Tese de Doutorado em Engenharia Elétrica. Universidade Federal da Paraíba.

[6] SBIE. *Workshop de Ambientes de Aprendizagem Baseados em Agentes*. Maceió, AL, 2000. http://www.inf.pucrs.br/~giraffa/sbie2000. Acessado em 2001.

[7] FARIA, T. F.; BITTENCOURT, G. Um ambiente interativo multiagentes para o ensino de Estrutura da Informação. In: Simposio Brasileiro de Informática na

Educação (XI. Novembro de 2000: Maceió, AL). *Anais do XI Simposio Brasileiro de Informática na Educação*. Maceió, AL: Gráfica e Editora UFAL, 2000. 429-431.

[8] MORAN, J. M. *Novas tecnologias e o reencatamento do mundo*. USP. http://www.eca.usp.br/prof/moran/textos.htm. 2001.

[9] BITTENCOURT, G. *Breve história da Inteligência Artificial*. Universidade Federal de Santa Catarina. http://www.das.ufsc.br/gia/history/. 2001.

[10] CHARNIAK, E.; McDERMOTT, D. *Introduction to Artificial Intelligence*. Reading: Addison-Wesley Publishing Company, 1985.

[11] BITTENCOURT, G. *Inteligência Artificial Distribuída*. Universidade Federal de Santa Catarina. http://www.das.ufsc.br/gia/artigos/tra-iad.ps.gz. 2001.

[12] GANASCIA, J. G. *Inteligência Artificial*. Editora Ática, 1997.

[13] BITTENCOURT, G. *Inteligência Artificial Ferramentas e Teorias*. Florianópolis, SC: Editora da UFSC, 1998.

[14] BITTENCOURT, G. Representação de Conhecimento: da Metafísica aos Programas. In: Jornada de Atualização em Informática (XVII. Agosto de 1998: Belo Horizonte, MG). *Anais do XVII CSBC*. Belo Horizonte, MG: Gráfica da UFMG, 1998. 283-333.

[15] BITTENCOURT, G. *Inteligência Computacional*. Universidade Federal de Santa Catarina. http://www.das.ufsc.br/gia/softcomp/. 2002.

[16] SHORTLIFFE, E. H. *Computer-Based Medical Consultations: MYCIN*. New York: American Elsevier, 1976.

[17] FEIGENBAUM, E.A.; BUCHANAN, B.G.; LEDERBERG, J. On generality and problem solving: A case study using the Dendral program. *Machine Intelligence*, Edinburgh University Press: Edinburgh, GB, Volume 6, 165-190, 1971.

[18] HART, P. E.; DUDA, R. O.; EINAUDI, M. T. Prospector - a computer-based consultation system for mineral exploration. *Mathematical Geology*, 10(5), 1978.

[19] FLEMMING, E. *Um Sistema Tutor Multiagentes no Domínio da Lógica*. Florianópolis, SC, 1998. Dissertação de Mestrado em Engenharia Elétrica. Universidade Federal de Santa Catarina.

[20] BORGES, M. A. F. *O Design Centrado no Aprendiz no Sistema Jonas: Uma Experiência de Desenvolvimento de um Sistema para Formação na Empresa*. Campinas, SP, 1997. Dissertação de Mestrado em Ciência da Computação. Universidade Estadual de Campinas.

[21] SKINNER, B. F. Teaching Machines. *Science*, Vol. 128, 1958.

[22] PAPERT, S. Microworlds: Transforming education. *Artificial Intelligence and Education - Learning Environments and Tutoring Systems*, Vol. 1, 1987.

[23] PIAGET, J. *A Psicologia da Inteligência*. Rio de Janeiro: Zahar, 1977.

[24] CLANSEY, W. J. Guidon-Manage revisited: A socio-technical systems approach. *Journal of Artificial Intelligence in Education*, 4(1), 5-34, 1993.

[25] CARBONELL, J. R. AI in CAI: an artificial intelligence approach to computer-assisted instruction. *IEEE Transactions on Man-Machine Systems*, 11(4), 190-202, 1970.

[26] CLANCEY, W. J. *Knowledge-Based Tutoring: The GUIDON Program*. The MIT Press. 1987.

[27] BRUSILOVSKY, P.; SCHWARZ, E.; WEBER, G. ELM-ART: An Intelligent Tutoring System on World Wide Web. *Lecture Notes in Computer Science*, Berlin: Springer Verlag, Vol. 1086, 261-269, 1996.

[28] *CLISP (Common Lisp)*. http://clisp.sourceforge.net/. 2001.

[29] BITTENCOURT, G. *Inteligência Artificial Distribuída*. Universidade Federal de Santa Catarina. http://www.das.ufsc.br/gia/artigos/iad.ps.gz. 2001.

[30] MARIETTO, M. G. B. *Definição Dinâmica de Estratégias Instrucionais em Sistemas de Tutoria Inteligente: Uma Abordagem Multiagentes na WWW*. São José dos Campos, SP, 2000. Tese de Doutorado em Engenharia Eletrônica e Computação. Instituto Tecnológico de Aeronáutica.

[31] FERBER, J.; GASSER, L. Intelligence Artificielle Distribuée. Proceedings of International Workshop on Expert Systems and Their Applications. Avignon, França, 1991.

[32] FREITAS, F. L. G.; BITTENCOURT, G. Cognitive Multi-Agent Systems for Integrated Information Retrieval and Extraction over the Internet. In: Simposio Brasileiro de Inteligência Artificial (Novembro de 2000: Atibaia, SP). *Anais do Brasileiro de Inteligência Artificial*. 310-319.

[33] STEELE JR.,G. L. *Common Lisp, the Language*. Burlington: Digital Press, 1984.

[34] ABELSON, H.; SUSSMAN, G. J. *Structure and Interpretation of Computer Programs*. The Mit Press, 1996.

[35] BITTENCOURT, G. *Fundamentos da Estrutura da Informação*. Universidade Federal de Santa Catarina. http://www.das.ufsc.br/~gb/fei/. 2001.

[36] FRIEDMAN-HILL, E. *Jess, the Java Expert System Shell*. Livermore, CA: Sandia National Laboratories report SAND98-8206, 1997.

[37] *JaCoWeb Security*. Universidade Federal de Santa Catarina. http://www.das.ufsc.br/jacoweb/. 2001.

[38] *Java(TM) Servlet technology*. http://java.sun.com/products/servlet/. 2001.

[39] *Java(TM) technology*. http://java.sun.com/. 2001.

[40] *CLIPS*. http://www.ghg.net/clips/CLIPS.html. 2001.

[41] JEON, H. *JATLite - Java Agent Template, Lite*. Stanford University. http://java.stanford.edu/. 2001.

[42] ARPA. *Knowledge Query and Manipulation Language - KQML*. http://www.cs.umbc.edu/kqml/. 2001.

[43] LABIDI, S.; SILVA, J. C.; COUTINHO, L. R. *et al*. Agent Based Architecture for Cooperative Learning Environment. In: Simposio Brasileiro de Informática na Educação (XI. Novembro de 2000: Maceió, AL). *Anais do XI Simposio Brasileiro de Informática na Educação*. Maceió, AL: Gráfica e Editora UFAL, 2000. 32-39.

[44] COUTINHO, L. R.; LABIDI, S.; JUNIOR, G. S. *et al*. A Learner Modeling Agent for Cooperative Learning. In: Simposio Brasileiro de Informática na Educação (XI. Novembro de 2000: Maceió, AL). *Anais do XI Simposio Brasileiro de Informática na Educação*. Maceió, AL: Gráfica e Editora UFAL, 2000. 17-23.

[45] JUNIOR, H. M. N.; COSTA, E. B.; SILVA, L. D. *et al*. Um Sistema Tutor Inteligente na Web de Apoio ao Ensino de Cálculo Diferencial. In: Simposio Brasileiro de Informática na Educação (XI. Novembro de 2000: Maceió, AL). *Anais do XI Simposio Brasileiro de Informática na Educação*. Maceió, AL: Gráfica e Editora UFAL, 2000. 435-436.