**Sistema Híbrido Inteligente
para Suporte a um
Ambiente de Treinamento e Ensino ¹**

Jaqueline Brigladori Pugliesi

Orientação:

Profa. Dra. Solange Oliveira Rezende

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – USP, como parte dos requisitos para a obtenção do título de Mestre em Ciências – Área de Ciências de Computação e Matemática Computacional.

USP - São Carlos

Novembro de 1998

¹Trabalho realizado com auxílio do CNPq e PROTEM-CC.

Este documento foi preparado com o formatador de textos $\text{\LaTeX}$. O sistema de citações de referências bibliográficas utiliza o padrão *Chicago* do sistema $\text{\BibTeX}$.

© Copyright 1998 por Jaqueline Brigladori Pugliesi
Todos os Direitos Reservados

Dedicatória

Aos meus pais, Denizar e Maria Ilzi,
meu irmão Denizar e minha irmã Helena.

Agradecimentos

À Profa. Dra. Solange Oliveira Rezende, pela competente orientação, incentivo, dedicação e, principalmente, pela sua amizade, propiciando-me um grande crescimento profissional e pessoal.

À Maria Carolina Monard e Dóris Ferraz Aragon, pelo apoio, carinho, dedicação e amizade.

À Profa. Dra. Maria das Graça Volpe Nunes, pela atenção e valiosas contribuições para a realização deste trabalho.

Ao Marcos Ferreira de Paula e Ricardo Dalcorsso Fodra, pelo auxílio e colaboração que deram a este trabalho.

Aos meus amigos (Cris, Carine, Ernesto, Luis Carlos, Marisa, Chandler, Claudia, Gustavo, Thereza, Alex, Paulo, Robson, Marco, Garga, Rudinei, Fabiano, Sandra, Luciano, Mirla, Júlio, Silmara, Márcia, Daniel, Walter, Huei, Fábio, Gedson, Renato, Naiara, Antônio, Andrea, entre tantos outros), pelos bons momentos compartilhados. Em especial, ao Alex, Augusto e Chandler pelo precioso tempo dispensado no auxílio e revisão desta dissertação.

À todos os amigos do LABIC, pelo companherismo, amizade e união.

À Ana Elisa Tozetto Piekarski, pelo apoio, carinho e alegria em todos os momentos, os quais nunca vou esquecer e, acima de tudo, por ser uma grande amiga.

Ao José Augusto Baranauskas pelo carinho, incentivo e paciência.

À toda minha família, pelo amor, apoio e compreensão pelo tempo que estive ausente.

Às funcionárias da Pós-Graduação (Beth, Laura e Marília), da biblioteca, ao Paulinho, ao Jacques e à todos os funcionários do ICMC, pela maneira prestativa e educada que sempre me atenderam.

Ao CNPq pelo apoio financeiro.

Àqueles que, de um modo ou de outro, contribuíram para a realização deste trabalho.

À Deus, pela permissão e por estar sempre ao meu lado.

Todas as ciências, todas as artes e todas as
disciplinas e tecnologias delas derivadas
foram geradas pelas aptidões do mais antigo
e sofisticado engenho colocado à disposição
do homem: seu cérebro.
As ciências, tecnologias e disciplinas só podem
resolver problemas setoriais. A inteligência
é a única panacéia universal.

Roberto Lira Miranda

Sumário

1	Introdução	1
2	Abordagens para Apoio a Treinamento e Ensino	5
2.1	Considerações Iniciais	5
2.2	Sistema Baseado em Conhecimento	6
2.2.1	Estrutura Geral de um Sistema Baseado em Conhecimento	7
2.2.2	Processo de Desenvolvimento de um Sistema Baseado em Conhecimento	12
2.2.3	Principais Erros Considerados Durante as Etapas de Desenvolvimento do Sistema	14
2.2.4	Sumário da Seção	16
2.3	Aquisição de Conhecimento	16
2.3.1	Engenharia de Conhecimento	17
2.3.2	Técnicas de Aquisição de Conhecimento Explícito ou Manual	23
2.3.3	Técnica de Aquisição de Conhecimento Implícito ou Automática	27
2.3.4	Administração da Aquisição de Conhecimento	32
2.3.5	Sumário da Seção	33
2.4	Raciocínio Baseado em Casos	34
2.4.1	Os Sistemas de Raciocínio Baseado em Casos	34
2.4.2	Processo Básico de um Sistema de Raciocínio Baseado em Casos	35
2.4.3	Problemas Relacionados com Raciocínio Baseado em Casos	37
2.4.4	Sumário da Seção	38
2.5	Sistemas Tutores	39
2.5.1	Inteligência Artificial e Educação	39
2.5.2	Introdução a Sistemas Tutores Inteligentes	40

2.5.3	Estrutura Geral de um Sistema Tutor Inteligente	42
2.5.4	Sumário da Seção	44
2.6	Simulação	44
2.6.1	Conceitos Básicos Envolvendo Simulação	45
2.6.2	Uso de Simulação para Treinamento e Ensino	46
2.6.3	Sumário da Seção	48
2.7	Considerações Finais	48
3	Ambiente de Apoio a Treinamento e Ensino	50
3.1	Considerações Iniciais	50
3.2	Visão Geral do Ambiente de Apoio a Treinamento e Ensino	51
3.3	Arquitetura do Sistema Inteligente de Apoio a Treinamento e Ensino	54
3.3.1	Interface	54
3.3.2	Material Didático	56
3.3.3	Roteiros	58
3.3.4	Help	60
3.3.5	Prática/Treinamento	61
3.3.6	Sistema Tutor	62
3.3.7	Sistema Híbrido de Conhecimento	63
3.4	Considerações Finais	63
4	Sistema Híbrido de Conhecimento	65
4.1	Considerações Iniciais	65
4.2	Estrutura do Sistema Híbrido de Conhecimento	66
4.3	Processo de Desenvolvimento do Sistema Híbrido de Conhecimento	69
4.3.1	Principais Etapas no Desenvolvimento do Sistema Híbrido de Conhecimento	69
4.3.2	Modelo Utilizado no Desenvolvimento do Sistema	70
4.4	O Processo Utilizado para Aquisição de Conhecimento Explícito	72
4.5	Considerações Gerais sobre a Implementação do Sistema Híbrido de Conhecimento	75
4.6	Considerações Finais	76

5	Módulo Roteiros	
5.1	Considerações Iniciais	77
5.2	Modelagem do Domínio para Geração de Roteiros	77
5.2.1	Considerações Gerais sobre Modelagem Semântica	78
5.2.2	Modelagem do Domínio Utilizando o Generic Semantic Model	81
5.3	Gerador de Roteiros	82
5.4	Considerações Finais	87
6	Módulo Prática/Treinamento	88
6.1	Considerações Iniciais	88
6.2	A Biblioteca MLC++	89
6.3	Os Algoritmos CN2 e C4.5	95
6.4	Simulação Usando a Biblioteca MLC++	96
6.5	Considerações Finais	98
7	Exemplos de Execução do SIATE	99
8	Conclusão	107

Lista de Figuras

2.1	Sistema Baseado em Conhecimento <i>versus</i> Sistema Especialista.	7
2.2	Estrutura de um Sistema Baseado em Conhecimento.	8
2.3	Modelo Espiral para Desenvolvimento de um Sistema Baseado em Conhecimento.	14
2.4	Engenharia de Conhecimento.	17
2.5	Aquisição de Conhecimento Explícito.	24
2.6	Aquisição de Conhecimento Implícito.	28
2.7	Etapas do Processo de Extração de Conhecimento de Bases de Dados.	31
2.8	Formulário para Documentar a Aquisição de Conhecimento.	33
2.9	Ciclo de Raciocínio Baseado em Casos (Aamodt, 1994).	36
2.10	Arquitetura Básica de um Sistema Tutor Inteligente.	42
3.1	Visão Geral do Ambiente SIATE.	53
3.2	Arquitetura do Ambiente SIATE.	54
3.3	Página de Identificação Ambiente SIATE.	56
3.4	Página da Tela Principal do Ambiente SIATE.	57
3.5	Exemplo de uma Página do Material Didático.	58
3.6	Manipulação de Roteiros.	59
3.7	Exemplo de uma Página do Protótipo do SIATE com um Roteiro Ativado.	60
3.8	Projeto das Funcionalidades do Módulo Prática/Treinamento.	61
4.1	Estrutura do Sistema Híbrido de Conhecimento.	67
4.2	Processo para Aquisição de Conhecimento Explícito - IPAIA.	73
5.1	Uma Visão da Modelagem do Domínio.	82

5.2	Componentes do Sistema Híbrido de Conhecimento Utilizados pelo Gerador de Roteiros.	83
6.1	Interação entre indutor, categorizador, a etapa de teste e wrapper.	91
6.2	Classificação dos Algoritmos de Aprendizado de Máquina (Félix et al., 1998).	92
7.1	Página Inicial do Ambiente SIATE.	99
7.2	Página de Identificação do Ambiente SIATE.	100
7.3	Página de Cadastro do Usuário do SIATE.	100
7.4	Página da Tela Principal do Ambiente SIATE.	101
7.5	Página Inicial do Módulo Roteiros.	102
7.6	Página Inicial para Geração de Roteiros.	102
7.7	Página do Hiperdocumento com Roteiro Ativado.	103
7.8	Página da Tela Principal do Ambiente SIATE.	103
7.9	Página com uma Pergunta Acionada.	104
7.10	Página com a Pergunta sobre a Apresentação de Explicação.	105
7.11	Página com Janela de Regras.	105
7.12	Página com as Regras Geradas.	106

Lista de Tabelas

2.1	Tarefas e Fases de Aquisição de Conhecimento.	23
5.1	Características e Grupos das Páginas do Material Didático do SIATE.	85
6.1	Classificação de Alguns dos Indutores usados pela MLC++.	94

Lista de Abreviaturas

AC	Aquisição de Conhecimento
ACE	Aquisição de Conhecimento Explícito
ACI	Aquisição de Conhecimento Implícito
AM	Aprendizado de Máquina
BC	Base de Conhecimento
BD	Base de Dados
CAI	<i>Computer Aided Instruction</i>
CGI	<i>Common Gateway Interface</i>
DM	Data Mining
EC	Engenheiro de Conhecimento
FAQ	<i>Frequently Asked Questions</i>
FDM	<i>Functional Data Model</i>
GSM	<i>Generic Semantic Model</i>
HCI	<i>Human-Computer Interaction</i>
HDM	<i>Hipertext Design Model</i>
HTML	<i>Hipertext Markup Language</i>
IA	Inteligência Artificial
ICAI	<i>Intelligent Computer-Assisted Instruction</i>
ISLE	<i>Intelligent Simulation Learning Environment</i>
KDD	<i>Knowledge Discovery in Databases</i>
MCD	Módulo Coletor de Dados
ME	Módulo de Explicação
MER	Modelo Entidade-Relacionamento
MI	Motor de Inferência
<i>MLC++</i>	<i>Machine Learning Library in C++</i>

MT	Memória de Trabalho
NSBC	Núcleo do Sistema Baseado em Conhecimento
RC	Representação de Conhecimento
SBC	Sistema Baseado em Conhecimento
SDM	<i>Semantic Data Model</i>
SE	Sistema Especialista
SGBD	Sistema Gerenciador de Base de Dados
SHC	Sistema Híbrido de Conhecimento
SI	Sistema Inteligente
SIATE	Sistema Inteligente de Apoio a Treinamento e Ensino
ST	Sistema Tutor
STI	Sistema Tutor Inteligente
RBC	Raciocínio Baseado em Casos
WWW	<i>World Wide Web</i>

Resumo

A utilização de recursos Hipermissão e técnicas de Inteligência Artificial em ambientes de ensino e aprendizado oferecem uma melhor apresentação das informações aos usuários e proporcionam melhores resultados por habilitar o sistema a “raciocinar” sobre o que e como apresentar ensinamentos efetivos, estimulando o estudante a aprender.

Assim sendo, propõe-se, em um projeto de maior abrangência, a arquitetura de um ambiente denominado **SIATE** — Sistema Inteligente de Apoio a Treinamento e Ensino — que integra características de Sistemas Baseados em Conhecimento, Sistemas Tutores, Raciocínio Baseado em Casos, Hipermissão e Simulação, com o objetivo de enriquecer um ambiente exploratório de ensino com conhecimento especialista do domínio e de melhorar o aprendizado do estudante.

O presente trabalho, parte integrante do SIATE no domínio de Aquisição de Conhecimento, corresponde ao projeto e desenvolvimento de um Sistema Híbrido de Conhecimento, o qual contém o conhecimento especializado do domínio de aplicação e fornece apoio à geração de roteiros de páginas de um hiperdocumento e à ferramenta de treinamento do SIATE.

Abstract

The use of Hypermedia resources and Artificial Intelligence techniques in teaching and learning environments allows information to be presented in a clear manner to users and offers better results by allowing the system to “reason” about what and how effective teaching should be presented, stimulating the student to learn.

An architecture called **SIATE** — Intelligent System for Training and Teaching — has been proposed, as part of a much larger project. This architecture integrates characteristics from Knowledge Based Systems, Tutoring Systems, Case Based Reasoning, Hypermedia and Simulation, enriching an exploratory environment for teaching with expert knowledge about the domain, while also improving the student learning experience.

The present work, which is a part of the domain of Knowledge Acquisition in SIATE, corresponds to the design and implementation of a Hybrid Knowledge System. This system contains specialized knowledge about the application domain which is used to generate scripts for pages in a hyperdocument and support the training tool in SIATE.

Capítulo 1

Introdução

A visão de que o uso de computadores na educação promove o aumento da produtividade, ensinando mais pessoas em menos tempo, é o que baseava inicialmente as investigações sobre o uso de informática nessa área. Hoje em dia, o que se enfatiza na utilização de computadores na área educacional é o provável potencial quanto ao desenvolvimento de habilidades cognitivas, as quais proporcionam melhores meios de ensino, aprendizado e treinamento.

Atualmente, vários trabalhos têm sido desenvolvidos com o intuito de construir sistemas de treinamento e ensino onde o treinando/estudante possa interagir com as ferramentas que apoiem o processo de aprendizado, como é o caso do *Intelligent Simulation Learning Environment* (ISLE), que integra a tecnologia de Simulação e Sistema Tutor Inteligente (STI) (Schank, 1990; Schank, 1994a; Schank, 1994b).

Dentre as técnicas de aprendizado atuais destaca-se o treinamento, onde a teoria aprendida é colocada em prática fazendo com que a assimilação se torne mais rápida e eficiente. Uma forma de treinamento, da qual tem-se obtido bons resultados, é a simulação, onde os mais variados ambientes podem ter situações hipotéticas simuladas, de modo que o treinando possa, através de fatos reais, aprender praticando. Quando o treinando já possui conhecimento do problema a ser simulado, geralmente, ele assimila mais facilmente os recursos e as características que o simulador tem em comum com o mundo real. Porém, quando não existe este conhecimento por parte do treinando, o simulador pode tornar-se “ineficiente” (de assimilação mais difícil pelo aprendiz) por não fornecer nenhum tipo de monitoramento ou explicação durante a simulação.

Um ponto negativo nos ambientes que utilizam simulação é o fato de não oferecerem

todos os recursos necessários para que o ensino se torne inteiramente adequado. Então, para se obter um ambiente de aprendizado adequado, é necessário que existam recursos como: controle de características pessoais dos estudantes, recursos multimídia (por exemplo, áudio e vídeo), facilidades de navegação, formas de enviar mensagens especializadas ao usuário, facilidades de explicação, possibilidade de utilizar casos passados bem sucedidos, etc. Uma fonte de recursos para enriquecer tais ambientes é a Inteligência Artificial.

A Inteligência Artificial tem se mostrado um elo vital no movimento em direção ao ensino e treinamento, pois habilita o computador a “raciocinar” sobre uma grande quantidade de informações. Dentro dessa área de estudo, algumas tecnologias utilizadas são os Sistemas Baseados em Conhecimento (SBC), Raciocínio Baseado em Casos (RBC) e Sistemas Tutores (ST), que visam fornecer ao estudante oportunidade de questionamento, ilustrar situações ocorridas com casos semelhantes e armazenar informações individualizadas a respeito dos estudantes.

A utilização de recursos Hipermídia em ambientes de ensino oferece uma melhor apresentação das informações aos usuários e também os estimulam a aprender devido a recursos áudio-visuais disponíveis. O uso de técnicas de Inteligência Artificial proporciona melhores resultados para simulação e Hipermídia, por habilitar o sistema a “raciocinar” sobre o que apresentar a seguir e como melhorar esta apresentação em ensinamentos efetivos. Assim sendo, propõe-se, em um projeto de maior abrangência, a arquitetura de um ambiente denominado SIATE — Sistema Inteligente de Apoio a Treinamento e Ensino. Este ambiente integra características de Sistemas Tutores, Sistemas Baseados em Conhecimento, Raciocínio Baseado em Casos, Hipermídia e Simulação, visando um melhor aproveitamento desse ambiente de ensino pelo treinando.

Uma característica que deve ser ressaltada nesse ambiente é que o estudante possui liberdade na exploração do domínio. Ele pode recorrer, quando desejar ou considerar necessário, ao auxílio de sistemas com propriedades tutoriais e conhecimento especializado do domínio.

Esse projeto engloba dois trabalhos de mestrado. Um deles, proposto nesta dissertação, corresponde ao projeto e desenvolvimento de um Sistema Híbrido Inteligente. Este sistema contém o conhecimento especialista do domínio e casos passados, além de fornecer apoio à geração de roteiros e à ferramenta de treinamento do SIATE.

O outro sistema, proposto, desenvolvido e descrito em (Freire, 1998), corresponde ao

projeto e desenvolvimento de um Sistema Tutor sobre o mesmo domínio. Este sistema visa auxiliar o estudante na “navegação” pelo hiperdocumento e contém as informações referentes ao domínio de aplicação, bem como recursos Hipermissão existentes no SIATE.

Para validar a arquitetura proposta ao ambiente SIATE, foi construído um protótipo utilizando como domínio o tema Aquisição de Conhecimento (AC), abrangendo tanto a Aquisição de Conhecimento Explícito (ACE) quanto a Aquisição de Conhecimento Implícito (ACI). Esse domínio foi escolhido por se tratar de um estágio difícil — pois envolve a extração, interpretação e representação de conhecimento — e ser considerado o “gargalo” no desenvolvimento de Sistemas Inteligentes (SI).

A função principal dessa aplicação do SIATE é auxiliar engenheiros de conhecimento, público alvo do ambiente, a aprenderem como efetuar uma Aquisição de Conhecimento de forma correta e eficiente. Sendo assim, os exemplos que forem apresentados neste trabalho se referem ao domínio de conhecimento implementado nesse protótipo.

Dessa forma, os principais objetivos do presente trabalho são a proposta de uma arquitetura para apoio a treinamento e ensino e o desenvolvimento de um Sistema Híbrido de Conhecimento (SHC), que faz parte de um Sistema Inteligente de Apoio a Treinamento e Ensino no domínio de Aquisição de Conhecimento. O SHC viabiliza a realização de prática e treinamento e a geração de roteiros dinâmicos para percorrer um determinado conjunto de páginas do hiperdocumento contido no ambiente.

Visando atingir esses objetivos, este trabalho está organizado em oito capítulos.

No Capítulo 2 são descritas as abordagens para apoio a treinamento e ensino utilizadas neste trabalho, que são: Sistema Baseado em Conhecimento, Aquisição de Conhecimento, Raciocínio Baseado em Casos, Sistemas Tutores e Simulação. Dentro de cada abordagem são descritos os conceitos básicos e as características principais.

No Capítulo 3 é apresentado o ambiente proposto para apoio a treinamento e ensino SIATE, ressaltando principalmente sua arquitetura, bem como os módulos que o compõe.

No Capítulo 4 é mostrada a estrutura do Sistema Híbrido de Conhecimento, que representa um dos componentes do SIATE e é o principal objeto de estudo deste trabalho. Além disso, é apresentado o processo de desenvolvimento desse sistema.

No Capítulo 5 é apresentado o Gerador de Roteiros, o qual fornece apoio ao Módulo de Roteiros do SIATE. Também é mostrada a modelagem semântica do domínio de aplicação, utilizada na geração de roteiros dinâmicos.

No Capítulo 6 é mostrado o Módulo Prática/Treinamento do SIATE, bem como a biblioteca $\mathcal{MLC}++$, que é a ferramenta para treinamento utilizada no protótipo do ambiente.

No Capítulo 7 são apresentados alguns exemplos de execução do SIATE, mostrando as seqüências de passos necessários para a geração de um roteiro, bem como a utilização do Módulo Prática/Treinamento.

No Capítulo 8 são expostas as conclusões deste trabalho e sugestões de trabalhos futuros envolvendo o ambiente de prática e treinamento.

Finalmente, é apresentada a bibliografia que foi utilizada para que a realização deste trabalho fosse possível.

Capítulo 2

Abordagens para Apoio a Treinamento e Ensino

2.1 Considerações Iniciais

um número crescente de pesquisadores estão utilizando RBC para projeto de Sistemas Inteligentes (Milaré, 1997). A idéia básica de RBC é que a solução de problemas utilizando casos passados relevantes é uma abordagem muito natural para seres humanos. Este é o processo normalmente utilizado por especialistas para resolver um novo problema de uma maneira rápida e segura. Assim, o poder dessa técnica vem de sua habilidade para recuperar casos relevantes da memória de forma rápida e precisa.

A Inteligência Artificial tem se mostrado um elo vital no movimento em direção ao ensino e treinamento, pois habilita o computador a “raciocinar” sobre uma grande quantidade de informações.

Por outro lado, a simulação é uma forte ferramenta de apoio ao ensino e treinamento e gera resultados mais eficazes quando utilizada em conjunto com outras ferramentas como, por exemplo, Sistemas Baseados em Conhecimento, Sistemas Tutores Inteligentes, Raciocínio Baseado em Casos e recursos hipermídia.

Visando um ambiente de apoio a treinamento e ensino com este enfoque, neste capítulo são apresentados alguns tópicos de Inteligência Artificial considerados importantes e relevantes para o desenvolvimento deste trabalho. Na Seção 2.2 são apresentados os Sistemas Baseados em Conhecimento, na Seção 2.3 é descrita a etapa de Aquisição de Conhecimento, na

Seção 2.4 é mostrado o tópico referente a Raciocínio Baseado em Casos e na Seção 2.5 os Sistemas Tutores. São apresentadas também algumas características de simulação (Seção 2.6), uma vez que o ambiente de treinamento e ensino, proposto neste trabalho, usa técnicas de simulação para obter um melhor desempenho e aproveitamento do treinando.

2.2 Sistema Baseado em Conhecimento

Os Sistemas Baseados em Conhecimento são programas de computador que usam conhecimento representado explicitamente para resolver problemas. Eles manipulam conhecimento e informação de forma inteligente e são desenvolvidos para serem usados em problemas que requerem uma quantidade considerável de conhecimento humano e de “esperteza”.

Para que um SBC aproxime-se do desempenho de um especialista humano, o sistema deve possuir uma grande quantidade de conhecimento especializado disponível, ter acesso rápido a esse conhecimento e ser capaz de raciocinar adequadamente. A subárea de SBC é uma aproximação bem sucedida da solução de problemas clássicos de programação inteligente em IA (Giarratano & Riley, 1994).

Os SBCs¹ podem ser classificados como Sistemas Especialistas (SEs) quando o desenvolvimento do mesmo é voltado para aplicações onde o conhecimento a ser manipulado restringe-se a um domínio específico e conta com um alto grau de especialidade, possibilitando assim resultados mais precisos (Figura 2.1) (Arantes & Rodrigues, 1994).

Uma prova do sucesso de SBCs são suas várias aplicações hoje em dia em negócios, medicina, ciência e engenharia, assim como os vários livros, jornais, conferências e produtos voltados aos SBCs. Essa tecnologia pode incluir linguagens de SBCs, programas e hardware projetados para ajudar no desenvolvimento e execução de SBCs.

Por se assemelhar a especialistas humanos na resolução de problemas pertencentes a uma área de conhecimento, os SEs devem ser capazes de:

- interagir com o usuário, fazendo-lhe perguntas e, desta forma, reunir as informações de que necessita;
- desenvolver uma linha de raciocínio a partir dessas informações e do conhecimento nele

¹Os termos Sistema Baseado em Conhecimento e Sistema Especialista serão utilizados indistintamente nesse trabalho.

Figura 2.1: Sistema Baseado em Conhecimento *versus* Sistema Especialista.

embutido de forma a encontrar soluções satisfatórias;

- explicar, caso seja questionado pelo usuário, porque faz determinadas perguntas e como chega às suas conclusões;
- lidar com regras e informações imprecisas. Esta é uma característica que, entre outras, o diferencia dos sistemas convencionais, os quais são capazes de manipular somente dados e regras determinísticas.

A seguir será apresentada uma estrutura geral de um SBC, os principais módulos que o compõe, bem como um processo de desenvolvimento de um SBC.

2.2.1 Estrutura Geral de um Sistema Baseado em Conhecimento

Na Figura 2.2 pode ser observada a estrutura geral e básica de um SBC, que constitui-se de:

- Núcleo do Sistema Baseado em Conhecimento (NSBC) ou *Shell*, o qual desempenha as principais funções do SBC;
- Base de Conhecimento (BC), onde está representado todo o conhecimento do especialista sobre um determinado domínio; a forma de representação desse conhecimento

Figura 2.2: Estrutura de um Sistema Baseado em Conhecimento.

deve ser compatível com o modo de manipulação do NSBC, usando algum formalismo de Representação de Conhecimento;

- Memória de Trabalho (MT), consiste na área de trabalho do sistema;
- Base de Dados (BD), o sistema pode estar interagindo com uma BD para obtenção ou armazenamento de dados e/ou informações, ou haver Sistema Gerenciador de Base de Dados (SGBD), que interaja com o Motor de Inferência, para o gerenciamento da BD.

Núcleo do Sistema Baseado em Conhecimento

O NSBC, que pode ser uma *Shell*, é responsável pelas principais atividades do sistema, como:

- interação com o usuário ou equipamentos externos;
- processamento do conhecimento utilizando alguma linha de raciocínio;
- justificação ou explicação das conclusões obtidas a partir deste raciocínio.

A *Shell* pode ser vista como um SBC com uma base de conhecimento vazia. Segundo (Jackson, 1986) as *Shells* são destinadas a permitir que não-programadores usufruam dos esforços de programadores que já resolveram um problema similar. Ferramentas de apoio para o desenvolvimento de um SBC também podem ser utilizadas.

O módulo NSBC é composto basicamente de três submódulos interdependentes, cada qual com funções específicas (Rodrigues, 1990). São eles:

- O Módulo Coletor de Dados (MCD), que é o responsável pela interação com o usuário, obtendo informações do problema em questão, através da formulação de sucessivas perguntas ao usuário. Quando ativado pelo Motor de Inferência, o MCD faz as perguntas necessárias e valida as respostas do usuário baseando-se em funções de crítica pré-estabelecidas na BC. Tais funções verificam a validade das respostas;
- O Motor de Inferência (MI), que é o responsável pelo desenvolvimento do raciocínio baseado nas informações obtidas pelo MCD e no conhecimento do(s) especialista(s). Isso é realizado através de mecanismos de inferência que manipulam o conhecimento representado na BC;

Para um MI que manipula regras, a linha de raciocínio pode ser desenvolvida por meio de dois processos:

- Encadeamento Regressivo ou *Backward Chaining*: neste processo parte-se da suposição de que cada provável solução é verdadeira. Feito isso, tenta-se reunir evidências que comprovem ser correta a solução previamente considerada. Tais evidências são procuradas nas informações fornecidas pelo usuário.

Resumindo, essa estratégia corresponde a perguntar:

“É possível provar as hipóteses a partir dos dados que temos?”

- Encadeamento Progressivo ou *Forward Chaining*: neste processo chega-se a uma solução para o problema a partir das informações fornecidas pelo usuário. Esse processo consiste em analisar esses dados, baseando-se nas regras de conhecimento, até se chegar a uma conclusão.

Resumindo, essa estratégia corresponde a perguntar:

“O que é possível concluir a partir dos dados que temos?”

- O Módulo de Explicações (ME), que é o responsável pela explicação, ou justificação, das conclusões obtidas e dos motivos pelos quais fez determinadas perguntas.

Os principais tipos de explicações são:

- **porque:** enquanto busca as soluções para o problema em questão, o MI necessita de algumas informações. O MCD é então ativado de modo a questionar o usuário. Se o usuário questionar o porquê da pergunta, o ME será ativado pelo MCD, e justificará a necessidade da mesma.
- **como:** ao final do processo de inferências, o MI chega às suas conclusões e as expõe ao usuário, se for requisitado. Se, além disso, o usuário deseja saber como as conclusões foram obtidas, o ME será ativado, e fornecerá toda a árvore de prova do processo. Essa árvore contém todas as provas intermediárias e relevantes realizadas durante o desenvolvimento do raciocínio. O ME fornece as conclusões tiradas a partir dessas provas até chegar a solução final.
- **what_if:** caso o usuário deseje saber qual seria o resultado obtido se uma ou mais informações fossem modificadas, ele deverá perguntar *what_if* ao final da execução do sistema. Se assim acontecer, o ME perguntará quais as condições a serem substituídas, apagará da MT antigas informações referentes a tais condições e ativará o MI, que reiniciará o processo de inferência. Desse modo, novas conclusões serão obtidas.

A maioria das *Shells* disponibilizam funções adicionais para facilitar o processo de desenvolvimento do sistema. Entre essas funções destacam-se os editores de Base de Conhecimento e *browser* para que o desenvolvedor possa acompanhar a linha de raciocínio do sistema durante o processo de desenvolvimento do SBC.

Enquanto executa suas funções, o NSBC interage com a MT, onde grava e acessa os dados temporários obtidos no processo. Além disso, manipula a Base de Conhecimento, a qual contém representado todo o conhecimento e, em alguns casos, heurísticas para manipulá-lo.

Base de Conhecimento

Informalmente, uma BC é um conjunto de representações de ações e acontecimentos do mundo. Cada representação individual é chamada de sentença. As sentenças são expressas numa linguagem específica chamada linguagem de representação de conhecimento (Russell & Norvig, 1995).

O conhecimento é representado na base de conhecimento usando uma das linguagens de Representação de Conhecimento (RC), tais como: regras de produção, redes semânticas, frames, scripts, etc. Os SBCs também podem usar uma combinação de diferentes técnicas de RC: estes tipos de sistemas são chamados de **Sistemas Híbridos** com relação à representação de conhecimento.

O conhecimento também pode incluir probabilidades ou fatores de incerteza, os quais podem ser usados para melhorar a validade das tomadas de decisões, ajudar a resolver conflitos e melhorar recursos de explicação. Técnicas para lidar com incerteza incluem Método Bayesiano, Teoria de Evidência de Dempster-Shafer, Teoria da Certeza (Fatores de Certeza) e Lógica Fuzzy (Mitchell, 1997).

Memória de Trabalho

A Memória de Trabalho representa a área de trabalho de um SBC, na qual são gravadas todas as respostas fornecidas pelo usuário durante as interações realizadas com o sistema, evitando que o usuário responda à mesma questão mais de uma vez. Nela podem ser gravadas todas as conclusões intermediárias e provas realizadas durante a execução dos programas. Pode-se dizer que a MT funciona como a “memória” do sistema, e portanto sua utilização traz algumas vantagens:

- permite fornecer ao usuário toda a árvore de prova correspondente às conclusões obtidas;
- não torna necessária a repetição da mesma pergunta ou de uma mesma prova;
- evita a realização repetida de provas de conclusões intermediárias.

A memória de trabalho do sistema armazena condições iniciais, hipóteses intermediárias e decisões, além de soluções finais.

Interface

A interface é um processador de linguagem projetado para produzir uma comunicação orientada a problemas entre o usuário e o sistema.

Ela é responsável pela interação entre o SBC e o usuário, proporcionando a comunicação em ambas as direções e realizando a tradução entre a representação interna do sistema e a linguagem de interface de maneira compreensível ao usuário.

Muitos princípios baseados nas teorias cognitivas têm sido propostos para projetos de interface, como resultado de pesquisas na área de Interação Usuário-Computador (HCI – *Human-Computer Interaction*) (Preece et al., 1994). Entretanto, a meta da maioria destas pesquisas é, de um modo geral, proporcionar ao usuário um alto grau de eficiência e eficácia com relação a utilização de um sistema. O usuário não deve ter que se adaptar à interface do sistema. Ao contrário, a interface deve ser projetada para que seja intuitiva e natural de modo que ele possa facilmente aprender a utilizá-la.

A utilização da Hipermídia também é considerado um fator importante, pois viabiliza a construção de uma interface com vários tipos de mídia, como texto, vídeo, imagens, gráficos, áudio, animação, entre outros. Isso proporciona ao usuário um ambiente mais amigável e propício ao aprendizado.

2.2.2 Processo de Desenvolvimento de um Sistema Baseado em Conhecimento

Várias pesquisas relatadas na literatura se referem ao projeto e desenvolvimento de Sistemas Baseados em Conhecimento. O desenvolvimento desses sistemas usam hoje métodos empíricos, e normalmente não têm suporte de metodologias que sejam amplamente aceitos. Um dos motivos disso é a falta de várias das características desejáveis de um processo de engenharia de software (confiabilidade, estimativa de custos, garantia de qualidade, etc.) (Liebowitz & Salvo, 1989).

O desenvolvimento de um SBC é um processo que depende em muito dos recursos disponíveis e de como este é organizado e gerenciado. O gerenciamento de projeto compreende:

- gerenciamento de atividades: planejamento, escalonamento, registro e análise;
- gerenciamento de configuração: gerenciamento do produto e de mudanças;
- gerenciamento de recursos: previsão, aquisição e otimização do uso dos recursos.

São vários os estágios de um processo de desenvolvimento e podem, resumidamente, ser apresentados como: estudo de possibilidades, prototipação rápida, refinamento do sistema, teste preliminar do sistema, qualidade comercial do sistema, manutenção e evolução.

Os erros que podem ocorrer no processo de desenvolvimento estão relacionados aos estágios desse processo e podem ser classificados genericamente como: erros no conhecimento, na semântica, na sintaxe, na inferência ou na limitação desse conhecimento.

Genericamente, um processo de desenvolvimento de software refere-se ao período de tempo que inicia com a especificação inicial do software e termina com sua disponibilidade para uso. Antes de pensar no desenvolvimento e manutenção separadamente, o conceito de processo prevê uma continuidade que conecta todos os estágios. O planejamento prévio da manutenção e evolução do sistema diminui o seu custo posterior.

Vários modelos de processo de desenvolvimento têm sido utilizados para softwares convencionais, entre eles destacam-se cascata, espiral e incremental, que serviram como suporte para propostas de alguns modelos de construção de SBCs (Pressman, 1994). As características dos SBCs fazem com que alguns modelos de processo de desenvolvimento tenham difícil aplicação. O modelo cascata é um exemplo, uma vez que a natureza iterativa da obtenção de conhecimento, a complexidade da validação e testes e a complexidade de obtenção completa do comportamento desses sistemas no início do projeto são fatores determinantes da dificuldade de sua utilização. Várias propostas de modelos podem ser vistas na literatura, podendo-se verificar que a maioria utiliza a prototipação e/ou espiral (Boehm, 1988; Rocha et al., 1997; Rezende & Pugliesi, 1998).

Na Figura 2.3 é mostrada uma adaptação do modelo espiral proposta por Giarratano para desenvolvimento de SBC (Giarratano & Riley, 1994). Cada iteração ao redor da espiral (do centro para fora) adiciona alguma nova capacidade funcional ao sistema. O ponto final da espiral não determina de fato o término do sistema, ao invés disso, uma nova espiral é iniciada com a manutenção e a evolução do sistema. A espiral pode ser refinada para especificar mais precisamente os estágios gerais de cada um dos quadrantes - planejamento, aquisição de conhecimento, codificação e avaliação.

Um dos estágios mais complexos e difíceis no desenvolvimento de um Sistema Baseado em Conhecimento é a Aquisição de Conhecimento, que visa a construção de uma Base de Conhecimento, sendo um dos componentes básicos de um SBC.

Informações mais detalhadas sobre processos de desenvolvimento podem ser encontra-

Figura 2.3: Modelo Espiral para Desenvolvimento de um Sistema Baseado em Conhecimento.

das em (Jubileu et al., 1998), o qual trata tanto de processos de desenvolvimento de software tradicional e de SBC, como também do processo para a Aquisição de Conhecimento.

2.2.3 Principais Erros Considerados Durante as Etapas de Desenvolvimento do Sistema

Os potenciais erros de desenvolvimento de um SBC podem ser classificados pelas etapas mais prováveis nas quais eles ocorrem (Giarratano & Riley, 1994). Esses erros incluem:

- **erros de conhecimento do especialista:** quando o especialista é a fonte de conhecimento do SBC, se seu conhecimento é errôneo, os resultados podem ser propagados através de todo o processo de desenvolvimento. Um ponto positivo de se construir um SBC é a potencial detecção de conhecimento errôneo quando o conhecimento do especialista torna-se explícito;
- **erros semânticos:** um erro de semântica ocorre quando o significado do conhecimento não é apropriadamente informado. Como por exemplo, suponha que um especialista diga “Você pode extinguir um fogo com água” e o engenheiro de conhecimento (EC) interprete isso como “Todo fogo pode ser extinguido com água”. Erros semânticos

ocorrem quando o engenheiro de conhecimento interpreta mal as respostas do especialista, ou o especialista interpreta mal a pergunta do engenheiro de conhecimento, ou ainda quando as duas situações ocorrem;

- **erros sintáticos:** esses são erros simples que ocorrem quando a entrada de uma regra ou fato é feita de maneira incorreta. Caso se utilize uma ferramenta para auxílio na construção do SBC, ela deve comunicar esses erros e uma mensagem apropriada deve ser mostrada. Outros erros que ocorrem na fase de construção da Base de Conhecimento são devido a erros de origem do conhecimento que não foram detectados em etapas anteriores;
- **erros do motor de inferência:** assim como qualquer espécie de software, o motor de inferência ou o NSBC pode possuir erros. Na medida que uma ferramenta de SBC é licenciada para uso geral, todos os erros comuns devem ser revisados e solucionados. Entretanto, podem existir erros que apenas aparecem em circunstâncias raras, tal como ter um número elevado de regras na memória de trabalho. Alguns erros podem ser bastante sutis e aparecer apenas em certos padrões de operações;
- **erros da cadeia de inferência:** esses erros podem ser causados por conhecimento errôneo, erros semânticos, erros no motor de inferência, especificação incorreta das prioridades das regras e interações inesperadas entre regras. Erros da cadeia de inferência mais complexos podem ocorrer devido à incerteza de regras e evidências, bem como em função da propagação de incerteza na cadeia de inferência. A escolha de um método para lidar com incerteza não resolve automaticamente todas as questões relacionadas à incerteza. Por exemplo, antes de escolher inferência Bayesiana simples, deve-se verificar se a premissa de independência condicional é garantida;
- **erros dos limites ignorantes:** um problema que é comum a todos os estágios de desenvolvimento é a especificação dos limites de ignorância do sistema. Os especialistas humanos conhecem a extensão de seu conhecimento e seu desempenho diminui gradativamente dada as barreiras de suas ignorâncias. Especialistas humanos devem ser honestos o suficiente para admitir mais incerteza em suas conclusões próximas a essas barreiras. Entretanto, a não ser que um SBC seja especificamente programado para admitir incerteza, ele pode continuar a suprir respostas mesmo que a cadeia de

inferência e evidência seja muito fraca.

2.2.4 Sumário da Seção

Nessa seção foi apresentada uma visão geral dos Sistemas Baseados em Conhecimento, com ênfase na sua estrutura e no seu processo de desenvolvimento.

Os Sistemas Baseados em Conhecimento consistem em um importante avanço tecnológico, no que diz respeito à automatização, ainda que limitada, do conhecimento e do raciocínio humano. As ferramentas que auxiliam a construção de Sistemas Baseados em Conhecimento permitem a otimização de esforços no desenvolvimento de tais sistemas, enfatizando as etapas de aquisição, estruturação, representação e validação do conhecimento.

O próximo item abordará alguns tópicos relacionados à Aquisição de Conhecimento, tais como: Engenharia de Conhecimento, técnicas para AC e alguns dos principais problemas encontrados.

2.3 Aquisição de Conhecimento

A Aquisição de Conhecimento é um estágio crucial no desenvolvimento de Sistemas Inteligentes. Como um processo, a AC envolve a extração, interpretação e representação do conhecimento de um dado domínio e é considerado como sendo um dos estágios mais difíceis e precários. Além disso, a AC não utiliza apenas uma disciplina, é uma área que inclui principalmente psicologia cognitiva e Inteligência Artificial (Goonatilake & Khebbal, 1995a).

Adquirir conhecimento especialista envolve a obtenção de: informação dos especialistas e/ou de fontes documentadas, classificação dessa informação em declarativa ou procedural, codificação dessa informação num formato utilizado pelo sistema, e checagem de consistência do conhecimento codificado com o conhecimento existente no sistema.

Como visto na Seção 2.2, extrair conhecimento de um especialista do domínio é um dos vários componentes no processo de construção de um Sistema Inteligente (Rodríguez, 1997).

Nos próximos itens serão abordados os conceitos de Engenharia de Conhecimento, técnicas de Aquisição de Conhecimento Explícito ou Manual, técnicas de Aquisição de Conhecimento Implícito ou Automática e algumas considerações finais sobre o processo de

2.3.1 Engenharia de Conhecimento

Engenharia de Conhecimento trabalha com o processo de transferência de conhecimento de alguma fonte (especialista, livros, etc.) para o computador conforme mostra a Figura 2.4.

Figura 2.4: Engenharia de Conhecimento.

O profissional que extrai o conhecimento do(s) especialista(s) — ou de outra fonte — interpreta e representa-o em tipos e estruturas de conhecimento na Base de Conhecimento é conhecido como engenheiro de conhecimento.

O engenheiro de conhecimento realiza as tarefas de estruturar a área do problema; interpretar, traduzir e integrar conhecimento especialista ao sistema; traçar analogias; apresentar contra exemplos; trazer à luz conceitos difíceis; e verificar a consistência do conhecimento.

Alguns pontos devem ser considerados na escolha da equipe de engenheiros de conhecimento, tais como: habilidade para aprendizado, comunicação, organização, conceitualização, diplomacia, documentação e conhecimento de técnicas de desenvolvimento de Sistema Base-

ado em Conhecimento.

Dentre as várias fontes de conhecimento destacam-se os especialistas, que são pessoas com alto grau de conhecimento e experiência no domínio. Outra fonte seriam os usuários finais, que normalmente têm uma visão geral do domínio do problema, e podem fornecer informações valiosas durante as investigações iniciais. Especialistas auxiliares podem fornecer conhecimento especializado de subproblemas, porém, em alguns casos, podem levar a conselhos conflitantes. Também se pode obter conhecimento na literatura existente sobre o domínio a partir de relatórios, *guidelines*, livros, manuais, etc, que oferecem *background* e informações nos estágios iniciais; ou extrai-lo de dados pois, em alguns domínios e situações, a quantidade de dados armazenados em bases de dados pode ser muito relevante e interessante.

Para que a Aquisição de Conhecimento seja feita adequadamente e de uma maneira organizada, é necessária a participação de algumas pessoas durante esse processo. Dentre elas podem ser citadas: de um lado o especialista do domínio e o usuário que irão fornecer o conhecimento que possuem e, de outro, a equipe de AC que irá extrair o conhecimento e é composta pelo gerente, engenheiro de conhecimento, anotador/assistente, observador e psicólogo.

Componentes do Processo de Aquisição de Conhecimento

A seguir serão vistos com mais detalhes os três componentes principais para o processo de AC, que são: os especialistas do domínio, a equipe de AC e o usuário final.

• Especialistas do Domínio

Os Sistemas Inteligentes fazem um uso intenso do conhecimento especializado para resolver problemas em nível de um especialista humano. Um especialista é uma pessoa que é especializada em uma determinada área, isso é, possui conhecimento ou habilidades especiais que não são conhecidas nem disponíveis para muitas pessoas. Um especialista pode resolver problemas que muitas pessoas não podem resolver por inteiro ou os resolve muito mais eficientemente (mas não necessariamente mais barato).

Conhecimento é a chave para a funcionalidade de Sistemas Inteligentes. Na maioria dos casos a origem para esse conhecimento é o especialista do domínio. O especialista do domínio é um indivíduo selecionado por seu alto grau de conhecimento em um dado campo e pela sua habilidade para comunicar conhecimento. Muitas variáveis

determinam as qualificações do especialista do domínio, incluindo se ele é “prático” (comumente ativo no domínio) ou “experiente” (não comumente ativo no domínio). Especialistas do domínio são, muitas vezes, as únicas fontes de informação preservadas pelas companhias especializadas. Em outros casos, vários especialistas do domínio podem ser consultados para oferecer especialidade, que podem ser combinadas e compartilhadas entre os trabalhadores menos especializados (McGraw & Briggs, 1989).

Qualquer que seja o papel profissional inicial, a pessoa, para atuar como especialista do domínio, dependerá de características como: habilidade para explicar conhecimentos ou heurísticas importantes, ser introspectivo, paciente, honesto, organizado e dedicado ao trabalho, comunicar-se eficientemente, bem como possuir disponibilidade de tempo (Rezende & Pugliesi, 1998).

Para selecionar especialistas apropriados, engenheiros de conhecimento devem identificar experiências, características e atributos que os permitirão alcançar os objetivos de desenvolvimento da Base de Conhecimento. A identificação dos requisitos dos especialistas do domínio é apenas o primeiro passo. Esses especialistas devem ter experiência no domínio. Pode acontecer que os especialistas disponíveis não possuam conhecimento atualizado, embora tenham vasta experiência sobre como exercitar habilidades cognitivas profundas dentro do domínio. Especialistas que tiveram que treinar outros no domínio podem saber expressar melhor seus conhecimentos.

Vale ressaltar que a participação de especialistas no processo de ACI também é importante, pois ele poderá esclarecer pontos que ficaram duvidosos ou obscuros, bem como fornecer informações para que o processo seja mais eficaz e eficiente.

• Equipe de Aquisição de Conhecimento

Uma característica fundamental para a formação da equipe de AC² é a interação entre seus membros. A interação entre os componentes de qualquer grupo de trabalho é indispensável — de certa maneira, isso é óbvio — principalmente na situação aqui tratada, que envolve um processo dependente da motivação de todos os participantes. Uma equipe de AC normalmente é composta pelos seguintes membros: Gerente,

²Neste trabalho, o termo Engenheiro de Conhecimento está sendo utilizado para designar a equipe de Aquisição de Conhecimento.

Engenheiro de Conhecimento, Anotador/Assistente, Observador e Psicólogo. A seguir serão apresentadas algumas características e funções de cada um desses membros.

1. Gerente

A coordenação de um projeto que envolva AC requer os mesmos procedimentos adotados em projetos convencionais de semelhante amplitude.

Compõem as atribuições de um gerente: assumir e delegar responsabilidades; estabelecer metas; alocar os recursos humanos e materiais necessários; monitorar e integrar os elementos envolvidos no processo; ser o interlocutor da equipe com a empresa e os usuários e atuar como moderador dos debates. Por outro lado, o gerente deve apresentar as seguintes qualidades: capacidade administrativa, espírito de liderança, objetividade, equilíbrio e senso de oportunidade.

Diversas falhas cometidas pelos gerentes podem inviabilizar a conclusão bem sucedida do projeto. Entre elas, destacam-se: a avaliação incorreta dos objetivos, ou falta de concordância do grupo quanto a eles; planejamento ineficiente para a alocação de tempo e recursos, que podem acarretar mudanças desastrosas no projeto; não estabelecimento de critérios de avaliação do desenvolvimento do projeto; e finalmente, um sistema de documentação inadequado, que não provê informações em tempo hábil.

2. Engenheiro de Conhecimento

O EC é o principal componente da equipe de Aquisição de Conhecimento. O EC é o responsável por absorver, em curto espaço de tempo, informações suficientes sobre domínio de conhecimento, geralmente novo. Ele deve analisar e compreender o raciocínio do especialista, estabelecendo um diálogo produtivo com esse, de modo a permitir a construção do conhecimento em conjunto com o especialista.

São requisitos indispensáveis ao EC: grande capacidade de aprendizado; conhecimentos em Ciência da Cognição, Psicologia, Sociologia e, principalmente, técnicas de programação em IA; bom relacionamento interpessoal; boa comunicação; criatividade; paciência; e, por fim, disposição para reformular posições pessoais, caso seja necessário.

Assim como o gerente, o EC também pode comprometer drasticamente todo o processo, desde que cometa falhas, tais como: não estabelecer um bom relaciona-

mento com o especialista que detém o conhecimento e pode se recusar a cooperar; não conseguir analisar o conhecimento do domínio, não podendo estabelecer a melhor maneira de representá-lo na máquina; e, por último, não aplicar as técnicas adequadas ao especialista e ao ambiente de trabalho, podendo inserir-se em um ciclo ininterrupto de AC no qual não consegue absorver nenhum conhecimento significativo.

3. Anotador/Assistente

O anotador/assistente é o responsável pelas anotações durante as sessões de AC. Constam das anotações comentários importantes, explicações e descrições do especialista, que devem ser detalhadamente anotadas, seguindo com fidelidade o raciocínio do especialista, para que a equipe possa posteriormente analisá-lo. Devem também fazer parte dessas anotações as questões levantadas pelo EC e os pontos não esclarecidos para serem questionados ao especialista nas sessões seguintes.

4. Observador

O observador não é considerado um elemento permanente da equipe de AC, mas pode fazer parte das sessões de AC quando conveniente. A esse componente cumpre apenas observar a sessão, não devendo intervir. Sua função principal é analisar as sessões, para sugerir posteriormente à equipe novas técnicas que conduzam a um melhor aproveitamento do tempo e do conhecimento do especialista.

O observador deve ser uma pessoa experiente ou na tarefa de AC ou no domínio do conhecimento tratado nas sessões.

5. Psicólogo

A presença de um psicólogo na equipe não é muito difundida. Na literatura, a sua participação se restringe a estudos e pesquisas, desenvolvidos à parte, para auxiliar o processo de AC (Pereira & Aragon, 1997). Considera-se, entretanto, que sua contribuição pode ser mais efetiva. Esse elemento pode auxiliar o EC na preparação das técnicas de AC a serem utilizadas nas sessões, e na elaboração de fichas de avaliação das mesmas, de acordo com os objetivos preestabelecidos no planejamento da AC.

A presença de um psicólogo na equipe possibilita uma melhor elaboração dos momentos de desânimo durante as sessões, uma vez que o processo de AC depende do relacionamento humano.

- **Usuário**

O usuário é o responsável por apresentar o problema. É ele quem identifica a necessidade de um Sistema Inteligente e, portanto, de um processo de AC para armazenar e operacionalizar, de algum modo, o conhecimento de um indivíduo especializado em uma determinada área.

Devemos observar que o negligenciamento deste componente poderá acarretar na não utilização do sistema, independentemente da sua eficiência, bastando para isso que o usuário não se adapte ao sistema.

Tarefas e Fases de Aquisição de Conhecimento Explícito

Para cada ciclo de desenvolvimento, existem algumas fases discerníveis com relação ao nível de especialidade da informação desejada (McGraw & Briggs, 1989). Segundo McGraw a Aquisição de Conhecimento Explícito pode ser dividida em três grandes fases. A Tabela 2.1 mostra uma visão geral dessas fases de AC, quem são os maiores responsáveis por cada fase, tarefas específicas a serem conduzidas durante cada fase e estimativa da porcentagem total de esforço humano necessário. Essa porcentagem representa a divisão de tempo exigida para cada grupo — Equipe de Aquisição de Conhecimento e especialista do domínio — completar suas tarefas.

Os esforços de AC da Fase 1, em um dado ciclo de desenvolvimento, são mais gerais e, quando finalizadas, proporcionam um alicerce para o conhecimento que será extraído. Tarefas importantes a serem completadas durante essa fase incluem desenvolver um dicionário de conhecimento do vocabulário e conceitos do domínio, identificar relações entre esses conceitos, bem como suas principais funções, entradas e saídas. Nessa fase, técnicas como entrevistas desestruturadas podem ser apropriadas, sendo que o maior esforço será por parte do engenheiro de conhecimento. À medida que as informações se tornam mais específicas, o desenvolvimento passa para a Fase 2 e técnicas como entrevista e análise de protocolos se tornam mais adequadas. Nessa fase, ocorre a elaboração de exemplos e grafos de conhecimento, o desenvolvimento de um pseudocódigo e a revisão de notas de sessões

Fase 1 80% Equipe de AC 20% Especialista do Domínio	Definir o Problema Desenvolver um Mapa Mental Fazer uma Análise Funcional Fazer uma Análise de Tarefas
Fase 2 50% Equipe de AC 50% Especialista do Domínio	Realizar Sessões Iniciais de AC <i>Feedback</i> Realizar Sessões Aprofundadas de AC
Fase 3 80% Equipe de AC 20% Especialista do Domínio	Implementar, Testar e Refinar a Base de Conhecimento

Tabela 2.1: Tarefas e Fases de Aquisição de Conhecimento.

anteriores. A Fase 3 inclui a representação do conhecimento, seu refinamento, bem como teste de cenários.

2.3.2 Técnicas de Aquisição de Conhecimento Explícito ou Manual

Nesse item serão abordadas as principais técnicas de aquisição de conhecimento explícito ou manual (Figura 2.5), onde é necessária a presença de um ou mais especialistas do domínio, a partir dos quais os engenheiros de conhecimento farão toda a extração do conhecimento necessário para o desenvolvimento do sistema (Jones et al., 1996b; Jones et al., 1996a).

Uma descrição mais detalhada sobre Aquisição de Conhecimento Explícito é apresentada em (Rezende & Pugliesi, 1998) e maiores informações sobre as técnicas de ACE podem ser encontrados em (McGraw & Briggs, 1989).

Entrevista

Segundo McGraw, Gammack and Young (1985) a entrevista é a técnica mais utilizada por engenheiros de conhecimento para a eliciação de conhecimento a partir de um especialista.

Essa técnica de Aquisição de Conhecimento permite ao engenheiro de conhecimento compreender rapidamente conceitos e vocabulários importantes do domínio. Elas são muitas

Figura 2.5: Aquisição de Conhecimento Explícito.

vezes utilizadas nos estágios iniciais da AC.

Engenheiros de conhecimento não especializados podem considerar a entrevista como a única técnica de eliciação e podem ter uma percepção imprecisa de seu propósito ou estrutura. Entretanto, a entrevista pode não ser sempre a técnica mais adequada, e além disso, muitas vezes ela é incorretamente aplicada.

As entrevistas podem ser classificadas em dois grandes grupos:

- entrevista desestruturada: apropriada quando o engenheiro de conhecimento deseja explorar o problema (geralmente mais utilizada na fase inicial). É uma entrevista mais informal;
- entrevista estruturada ou focada: apropriada quando o engenheiro de conhecimento deseja informações específicas do conteúdo e do problema, e resulta em dados mais úteis para a Base de Conhecimento. É uma entrevista organizada e orientada a objetivos.

Brainstorming

Brainstorming originou-se da preocupação dos altos executivos de negócios para com os gerentes que eles supervisionavam. Estes sempre repetiam e imitavam a sabedoria de seus superiores. Embora isso fosse uma maneira politicamente segura de agir, acabava por excluir as idéias deles que pudessem ser úteis. Assim, criou-se essa técnica para encorajar a

criatividade em grupo. Dessa forma, *brainstorming* é um método formal usado para ajudar um grupo a gerar tantas idéias quanto forem possíveis num pequeno espaço de tempo. O máximo de benefício pode ser tirado ao se adotar a regra formal de sempre definir as idéias centrais e assegurar-se de que todos concordem com elas.

Brainstorming é uma técnica geralmente empregada para trabalho em equipe com múltiplos especialistas, e é bastante útil principalmente para as sessões iniciais de Aquisição de Conhecimento, quando deseja-se gerar soluções, mas não necessariamente avaliá-las. Também ajuda tanto o especialista quanto o engenheiro de conhecimento a “fugir” do lugar comum e estimular soluções mais criativas e muitas vezes melhores, além de reduzir críticas imediatas e comentários inibidores numa discussão.

Brainstorming também pode ser usado para auxiliar especialistas e engenheiros de conhecimento a libertarem-se de soluções óbvias ou convencionais para problemas complexos.

Análise de Protocolos

Análise de protocolos constitui uma técnica para analisar os resultados de uma sessão de Aquisição de Conhecimento. Em geral, pode ser antecedido por uma sessão de entrevista ou rastreamento de processo (descrito a seguir).

Análise de protocolos é um método usado para entender a maneira como uma pessoa resolve um problema e as operações específicas utilizadas para mover-se de um estado de conhecimento para outro. Protocolos são registros gravados ou transcritos, a partir de uma dada sessão, para análise posterior. Essas informações podem ser formadas tanto por sinais verbais, por exemplo, tom de voz, como por não-verbais, como coçar a cabeça.

Vale ressaltar que o processo de transcrição leva em geral de duas a três vezes mais tempo que a sessão de Aquisição de Conhecimento. Por isso, deve-se decidir entre seguir um procedimento rígido para criar protocolos formais ou simplesmente tomar notas informais a partir de uma fita gravada.

Rastreamento de Processo

Rastreamento de Processo é uma série de técnicas que permitem a determinação do processo (modo) de pensar de um indivíduo enquanto ele realiza uma tarefa ou chega a uma conclusão.

Em geral, o rastreamento de processo pode ser realizado de uma das seguintes ma-

neiras:

- Verbalização Concorrente: o especialista verbaliza seu processo de raciocínio enquanto realiza uma tarefa que está sendo investigada; ele deve “pensar em voz alta” ao mesmo tempo em que resolve um problema.
- Verbalização Retrospectiva: o especialista verbaliza seu processo de raciocínio depois de realizar uma tarefa que está sendo investigada; o engenheiro de conhecimento registra ou grava o procedimento utilizado para resolver o problema, para mais tarde poder revisar a sessão junto com o especialista e produzir um protocolo.

Os métodos mais utilizados para rastreamento são: observação do ambiente, informações limitadas, soluções limitadas, cenários simulados, analogias episódicas e análise de casos difíceis.

Estudo de Casos

No estudo de casos, elicit-se o conhecimento do especialista através de casos já documentados. Os casos devem cobrir várias possibilidades dentro do domínio, para eliciar conhecimentos específicos. O estudo de casos é útil para identificar sutilezas do especialista na tomada de decisões.

O engenheiro de conhecimento solicita ao especialista que fale sobre casos reais que tenha solucionado. Ou ainda, que o especialista descreva o caso mais difícil que solucionou, explicitando os passos desenvolvidos, assim como por que o considerou mais difícil. Um fator importante nessa técnica é que o caso não é dado ao especialista para que ele resolva. Ele próprio escolhe um ou mais casos resolvidos por ele, o que pode deixá-lo mais a vontade.

Introspecção

Na introspecção, o especialista verbaliza sensações, memórias, sentimentos e imagens que percebe enquanto resolve um problema. Essa técnica é útil para obter uma estratégia global da forma de resolução do problema.

As formas mais comuns de introspecção são: descrição retrospectiva (casos passados); simulação de um cenário (casos hipotéticos); e episódio crítico (casos difíceis).

Repertório Grid

Repertório Grid é uma técnica para eliciação e análise de um modelo do domínio do especialista. A teoria estabelece que cada pessoa atua como um “cientista” que classifica e organiza o seu mundo. Baseado nessas classificações, o indivíduo está apto a construir teorias de como um domínio particular funciona e agir no domínio baseado em suas teorias pessoais.

A técnica de repertório grid pode também ser usada para representar uma organização dos conceitos básicos do especialista num domínio específico. Para projetar um *grid*, o engenheiro de conhecimento deve primeiro elicitar um conjunto de construtores (por exemplo, características bipolares, como estável/não-estável).

O repertório grid é uma técnica multidimensional que tem como finalidade mostrar a estrutura de um conjunto particular de conceitos, identificar similaridades entre objetos, e agrupá-los conceitualmente. É uma técnica focada e normalmente usada para obter informações sobre opiniões ou outros conhecimentos imprecisos.

Essa técnica é apropriada quando existem vários conceitos relacionados, normalmente não identificados por novatos e é mais utilizada nas últimas fases de Aquisição de Conhecimento.

2.3.3 Técnica de Aquisição de Conhecimento Implícito ou Automática

Aqui serão apresentadas técnicas de aquisição de conhecimento implícito ou automática (Figura 2.6), que visam extrair conhecimento de dados e/ou informações referentes ao domínio, sem a necessidade de grande envolvimento de especialistas humanos.

Aprendizado de Máquina

Aprendizado de Máquina (AM) é uma subárea de Inteligência Artificial que pesquisa métodos computacionais relacionados à aquisição de novos conhecimentos, novas habilidades, e novas formas de organizar o conhecimento já existente, tendo por objetivo um melhor entendimento dos mecanismos de aprendizado dos seres humanos e a automação da Aquisição do Conhecimento (Weiss & Kulikowski, 1991; Monard et al., 1997).

Pesquisas em Aprendizado de Máquina concentram-se no desenvolvimento de algoritmos autônomos para adquirir conhecimento a partir de dados e na compilação e organização

Figura 2.6: Aquisição de Conhecimento Implícito.

desse conhecimento. Entretanto, devido a complexidade desse problema, a aplicação de AM tende a ser limitada a domínios simples. Deve-se, também, em AM, testar experimentalmente o conhecimento aprendido em conjuntos independentes de dados (Tecuci & Kodratoff, 1995).

Sistema de aprendizado é um programa de computador que toma decisões baseado em experiências acumuladas contidas em casos resolvidos com sucesso (Batista, 1997). Mais especificamente, ele se concentra em uma das tarefas de aprendizado mais básica e proeminente: a *classificação*. O objetivo fundamental de um sistema de aprendizado está em extrair conhecimento, por exemplo, na forma de regras de decisão, de um conjunto de dados tal que essas regras possam ser aplicadas a novos dados. “Aprender” consiste em escolher ou adaptar parâmetros da representação do modelo, no paradigma escolhido, de forma que o modelo classifique corretamente os exemplos conhecidos bem como novos exemplos.

Um sistema de aprendizado é projetado para trabalhar sob algum paradigma de aprendizado, tal como paradigma simbólico, estatístico, conexionista, genético, etc.

Extração de Conhecimento de Bases de Dados

O processo de Extração de Conhecimento de Bases de Dados (*Knowledge Discovery in Databases* - KDD), segundo Fayyad (Fayyad et al., 1996a), está definido como um processo não trivial de identificação válida, recente e potencialmente útil de padrões compreensíveis

embutidos nos dados.

A extração de conhecimento a partir de grande massas de dados deve ser vista como um processo interativo e iterativo, e não como um sistema de análise automática. Dessa forma não se pode esperar a extração de conhecimento útil simplesmente submetendo um conjunto de dados a uma “caixa preta” (Mannila, 1997a).

A interatividade do processo de KDD é pautada no amplo entendimento, por parte dos usuários desse processo, sobre o domínio da aplicação. Esse entendimento envolve a seleção de um subconjunto representativo dos dados, classes de padrões adequadas e bons critérios para avaliar os padrões dos conhecimentos extraídos.

O sucesso do processo de KDD depende, em parte, da interação entre esses usuários. É pouco provável que o analista encontre conhecimento útil a partir dos dados sem o aval do especialista sobre o que seria útil para um domínio específico. Além disso, a interatividade do processo requer a participação efetiva do usuário final e do especialista nas escolhas e decisões ao longo do processo (Fayyad et al., 1996a).

Na Figura 2.7 é mostrada uma aproximação do fluxo de dados em um processo KDD. O controle de fluxos em uma sessão de Data Mining (DM) é geralmente um processo iterativo de consultas, análises de tarefas, coleção de dados das bases de dados, padrões de busca e apresentação das decisões.

Os passos que representam o processo KDD, partindo dos dados brutos e terminando na avaliação do conhecimento que foi adquirido, são:

1. **Compreensão do domínio da aplicação:** nesta etapa, procura-se o completo entendimento do domínio da aplicação, considerando aspectos como: estudo de viabilidade e custos da aplicação do processo, verificação da quantidade e tipo do conhecimento disponível antes do início do processo de KDD e condições e metas do usuário final.
2. **Seleção e preparação dos dados:** uma amostra representativa (por exemplo, utilizando técnicas estatísticas) é retirada da base de dados, pré-processada e submetida aos métodos e ferramentas da etapa seguinte. A extração direta de conhecimento a partir de grandes bases de dados pode se tornar uma tarefa inviável. Grandes volumes de dados podem gerar um espaço de busca de padrões combinatorialmente explosivo. Além disso, a maioria dos algoritmos para extração de padrões possuem limitações quanto ao volume de dados que podem manipular. A busca de conhecimento em grandes bases

de dados pode ocasionar, ainda, o aumento das chances de encontrar-se padrões pouco significativos e até mesmo espúrios. Uma possível solução para esse problema envolve a tentativa de selecionar uma amostra significativa da base de dados da aplicação.

3. **Data Mining:** tem o objetivo de encontrar padrões/modelos (conhecimento) a partir dos dados. DM envolve a criação de modelos apropriados de representação dos padrões e relações identificados a partir dos dados. O resultado desses modelos, depois de avaliados pelo especialista ou usuário final, são empregados para prever os valores de atributos definidos pelo usuário final baseados em novos dados (Kerber et al., 1995; Fayyad et al., 1996b).
4. **Avaliação do conhecimento extraído:** o conhecimento é avaliado quanto a sua qualidade ou utilidade para que, em caso positivo, seja utilizado para apoio a algum processo de tomada de decisão. O processo de KDD não termina quando os padrões nos dados de entrada são extraídos. É preciso que o usuário entenda e possa julgar a utilidade do conhecimento extraído, contrastando-o com o conhecimento do especialista do domínio. Essa interação pode facilitar a busca das causas de possíveis erros ocorridos ao longo de todo esse processo.
5. **Consolidação e utilização do conhecimento extraído:** a consolidação do conhecimento extraído pressupõe a verificação e a solução de potenciais conflitos com o conhecimento previamente extraído antes do processo ser iniciado. O conhecimento pode, então, ser organizado pelo analista dentro de um modelo, usado para refinar o modelo existente na aplicação ou simplesmente documentado e informado ao usuário.

Vale acentuar que, apesar das ferramentas de visualização serem mais utilizadas na etapa de avaliação do conhecimento extraído, elas são de grande relevância para facilitar o entendimento e avaliação, principalmente do usuário final, dos resultados de cada etapa (como pode ser observado na parte inferior da Figura 2.7).

A definição de KDD dá uma idéia das características básicas do processo de busca do conhecimento e alguns pontos podem ser destacados. Primeiro, KDD trabalha com uma grande quantidade de dados, que precisam ser delimitados, e o processo de DM necessita de algoritmos eficientes, rápidos e eficazes. Segundo, o dado é geralmente armazenado em bases de dados, que usam tecnologias de gerenciamento de bases de dados (como armazenamento,

Figura 2.7: Etapas do Processo de Extração de Conhecimento de Bases de Dados.

indexação, otimização de consultas, etc.). Finalmente, os usuários dos sistemas KDD são geralmente pesquisadores de conhecimento, que podem auxiliar a tomada de decisão, com respeito a novas descobertas de conhecimento.

Data Mining é um dos processos centrais na extração de conhecimento de bases de dados, e visa extrair padrões interessantes embutidos em dados. Esses padrões podem ser descobertos através de regras que descrevem propriedades dos dados, padrões que ocorrem com frequência, classificação de objetos, etc. As tecnologias atuais fazem uma coleta simples de dados, porém as análises estão sendo cada vez mais custosas (Mannila, 1997b).

Data Mining envolve montar modelos, ou determinar padrões, a partir dos dados observados. Os modelos montados desempenham o papel de todo o conhecimento inferido: se os modelos refletem ou não conhecimento útil e interessante depende do processo KDD como um todo, onde o julgamento subjetivo humano é normalmente necessário.

A maioria dos métodos de Data Mining são baseados em conceitos de Aprendizado de Máquina, reconhecimento de padrões e estatística: classificação, *clustering*, etc.

2.3.4 Administração da Aquisição de Conhecimento

Vários fatores influenciam a escolha de uma técnica de AC que depende, entre outros, da escala do problema e do tipo de metas. As entrevistas desestruturadas e análise de conceitos, por exemplo, são mais úteis nos estágios iniciais, enquanto entrevistas estruturadas, rastreamento do problema e simulações tendem a ser mais úteis depois que o domínio estiver claramente definido. Outro ponto que vale destacar na determinação das técnicas a serem utilizadas refere-se às fontes de conhecimento, levando em consideração a disponibilidade de especialistas, o número e experiência desses especialistas, bem como disponibilidade de exemplos e bases de dados.

Um ponto chave na AC é o formulário que documenta os planos para, ou os resultados de, sessões de AC (McGraw & Briggs, 1989). Inicialmente, esse formulário pode ser usado para planejar a sessão de AC e alertar as pessoas envolvidas no processo de Aquisição de Conhecimento. Posteriormente, ele pode ser expandido para documentar as principais conclusões das etapas, planos futuros ou uma representação intermediária de possíveis regras para a Base de Conhecimento.

Os formulários para documentar a Aquisição de Conhecimento Explícito devem possuir o número, a data e a hora em que foi realizada a sessão; o nome do engenheiro de conhecimento, do especialista ou de materiais consultados; fatos, regras ou conceitos principais que foram derivados; e pontos levantados e áreas que requerem maior refinamento em sessões futuras. Na Figura 2.8 um exemplo de um formulário para AC é apresentado.

Normalmente, independentemente das habilidades do engenheiro de conhecimento, a natureza interpessoal da sessão de ACE, juntamente com a dificuldade da tarefa, assegura que problemas devam surgir. As seguintes dificuldades serão, muito provavelmente, evidentes por elas mesmas: negativismo e apatia, falta de compromisso, blocos de comunicações verbais e não verbais, hostilidade e reações defensivas, conflitos entre expectativas e realidades, conhecimento irrelevante, incorreto, incompleto ou inconsistente, incapacidade do especialista de verbalizar o seu conhecimento ou raciocínio e discordância entre múltiplos especialistas.

Já a Aquisição de Conhecimento Implícito possui outros tipos de dificuldades, tais como: definição dos objetivos, incerteza nos dados, dados incompletos (*missing values*), atualizações constantes na base de dados, *overfitting* e a escolha do algoritmo apropriado

Formulário de Aquisição de Conhecimento		
Sessão de AC# FP101		
Tópico da Sessão: Heurísticas CD	Engenheiro de Conhecimento: Solange	
Local da Sessão: SIM	Data da Sessão de AC: 01/06/96	
ED/Fonte de Conhecimento: Lee Ho	Tempo Total: 1 ½ Hrs	
Tipo de Sessão:	☐ Entrevista Aberta ☒ Rastreamento de Processos ☐ Simulação ☐ Revisão ☐ Análise de Constructs ☐ Outros.....	
Principais Metas da Sessão:		
<i>Revisar terminologia do dicionário. Usar análise de protocolos para observar. Gravar o Lee trabalhando com o cliente para determinar se os CDs estão bons. Usar verbalização retrospectiva com a fita para ele explicar o rumo que sua discussão tomou.</i>		
Resumo da Sessão:		
<i>(ver diagrama de fluxo em anexo do rumo da discussão e seus atributos.)</i>		
Regras Derivadas da Sessão:		
<i>(ver diagrama n.15 anexo.)</i>		

Figura 2.8: Formulário para Documentar a Aquisição de Conhecimento.

para a etapa de DM.

2.3.5 Sumário da Seção

Nesta seção foram apresentados alguns dos principais pontos relacionados à Aquisição de Conhecimento, que é uma das etapas mais importantes e difíceis no desenvolvimento de um Sistema Inteligente.

Para tanto, a AC pode ser dividida em Aquisição de Conhecimento Explícito e Aquisição de Conhecimento Implícito, e foram apresentadas suas principais características e técnicas relacionadas a cada um desses dois grandes grupos.

Foram vistos também alguns problemas encontrados na etapa de AC, bem como uma forma de administrar e documentar a AC.

No próximo item será apresentada a técnica de Raciocínio Baseado em Casos que é utilizada quando o domínio a ser manipulado pode ser expresso através de experiências passadas (casos).

2.4 Raciocínio Baseado em Casos

Raciocínio Baseado em Casos é uma recente abordagem na resolução de problemas de aprendizado que tem obtido muita atenção nos últimos anos e em muitos sentidos é fundamentalmente diferente de outras abordagens de Inteligência Artificial. Ao invés de contar apenas com conhecimento geral do domínio do problema, ou fazer associações junto a relações generalizadas entre descritores de problemas e conclusões, RBC utiliza o conhecimento específico de experiências anteriores, ou seja, situações concretas do problema (casos). Um problema novo é solucionado procurando-se por um caso similar e reusando-o na nova situação problema. Uma segunda diferença importante é que RBC é também uma abordagem para aprendizado incremental, desde que uma nova experiência possa ser guardada toda vez que um problema tenha sido resolvido, tornando-o imediatamente disponível para problemas futuros (Aamodt & Plaza, 1994).

Dessa forma, RBC consiste, basicamente, na resolução de um problema novo relembrando uma situação similar ocorrida anteriormente e reusando a informação e o conhecimento daquela situação. O aprendizado em RBC requer um conjunto de métodos bem elaborados a fim de extrair conhecimento relevante de experiências, integrar um caso em uma estrutura de conhecimento existente e indexar o caso para posterior *matching* com casos similares.

Examinando resultados teóricos e experimentais da psicologia cognitiva, parece claro que, em geral, a aprendizagem e a resolução de problemas por humanos são processos que envolvem a representação e utilização de vários tipos de conhecimento e a combinação de vários métodos de raciocínio.

2.4.1 Os Sistemas de Raciocínio Baseado em Casos

Raciocínio reutilizando casos passados é uma maneira poderosa e freqüentemente utilizada por humanos para resolver problemas. Essa pretensão é também apoiada por resultados de pesquisas da psicologia cognitiva. Parte do fundamento da abordagem baseada em casos é sua plausibilidade psicológica. Muitos estudos têm mostrado evidência empírica para o papel dominante de situações específicas previamente experimentadas (chamados casos) na resolução de problemas por humanos. Conforme descrito em (Aamodt & Plaza, 1994) Schank desenvolveu uma teoria de aprendizado e lembrança baseada na retenção de experiência numa estrutura de memória dinâmica e evolutiva; e Anderson mostrou que pessoas usam casos

passados como modelos quando aprendem a resolver problemas, especialmente no começo do aprendizado.

Na terminologia de RBC, um caso normalmente denota uma situação do problema (Lopes, 1995). Uma situação experimentada anteriormente, que tenha sido capturada e aprendida de tal maneira que ela possa ser reusada na solução de problemas futuros, é referenciada como sendo um caso passado, caso anterior, caso armazenado ou caso retido. Correspondentemente, um caso novo ou caso não resolvido é a descrição de um problema novo a ser resolvido. RBC é, de fato, um processo cíclico e integrado de solucionar um problema, aprender a partir dessa experiência, solucionar um novo problema, etc.

As tarefas centrais que todos métodos de RBC devem tratar são: identificar a situação do problema corrente, encontrar um caso anterior similar ao novo, usar aquele caso para sugerir uma solução para o problema corrente, avaliar a solução proposta, e, em um caso ideal, atualizar o sistema aprendendo a partir dessa nova experiência.

2.4.2 Processo Básico de um Sistema de Raciocínio Baseado em Casos

Em um nível mais alto de generalidade, um ciclo geral de RBC pode ser descrito pelos seguintes processos, como pode ser observado na Figura 2.9:

- RECUPERAR o(s) caso(s) mais similar(es);
- REUSAR a informação e o conhecimento do caso para solucionar o problema;
- REVISAR a solução proposta;
- RETER as partes dessa experiência que provavelmente serão úteis para solução de problemas futuros.

Um problema novo é solucionado **recuperando** um ou mais casos experimentados anteriormente, **reusando** o caso de uma maneira ou de outra, **revisando** a solução baseado no reuso de um caso anterior e **retendo** a nova experiência, incorporando-a na base de conhecimento (base de casos) existente. Cada um dos quatro processos envolvem um número de passos mais específicos.

2.4.3 Problemas Relacionados com Raciocínio Baseado em Casos

Como em Inteligência Artificial em geral, não existem métodos de RBC universais aplicáveis a todo domínio de aplicação, o desafio de RBC, tal como em outros assuntos, é propor métodos que sejam convenientes para resolução de problema de aprendizado em domínios de assunto específicos e ambientes de aplicação específica. Alinhado com os modelos de tarefas apresentados, os problemas centrais de pesquisas em RBC podem ser agrupados em cinco áreas. Um conjunto de soluções coerentes para esses problemas constitui um método de RBC (Aamodt & Plaza, 1994; Aamodt, 1994):

- Representação dos casos

Um sistema de RBC é fortemente dependente da estrutura e do conteúdo de sua coleção de casos — sempre referenciado como sua memória de casos. Visto que um problema é resolvido pela chamada de um caso anterior adequado para a solução do novo problema, o processo de busca e *matching* do caso deve ser efetivo e com baixo tempo de execução. Além disso, visto que a experiência do problema resolvido deve ser armazenada de alguma maneira, esses requisitos também se aplicam ao método de integração de um novo caso na memória. O problema de representação em RBC é, fundamentalmente, o problema da decisão do que armazenar em um caso, encontrando a estrutura apropriada para descrever o conteúdo do caso e decidindo como a memória de casos deve ser organizada e indexada para recuperação e utilizada de maneira eficiente.

- Métodos de recuperação

A tarefa de recuperação começa com um problema (parcial) de descrição e termina quando o melhor caso anteriormente armazenado tenha sido encontrado. Suas sub-tarefas se referem à identificação de características, *match* inicial, pesquisa e seleção executados nessa ordem. A tarefa de identificação, basicamente, sugere um conjunto de descritores de problemas relevantes. O objetivo da tarefa de *match* é retornar um conjunto de casos que são suficientemente similares ao novo caso — dando um limiar de similaridade de alguma qualidade, enquanto a tarefa de seleção trabalha nesse conjunto de casos a fim de escolher o melhor *match* (ou pelo menos o primeiro caso para tentativa).

- Métodos de reuso

A reutilização de uma solução recuperada de um caso no contexto do novo caso focaliza dois aspectos: a diferença entre o caso passado e o caso corrente e qual parte do caso recuperado pode ser transferida para o novo caso. As subtarefas possíveis para reuso são copiar e adaptar.

Copiar: em tarefas simples de classificação, as diferenças são consideradas não relevantes, enquanto similaridades são relevantes, e a classe do caso recuperado é transferida para o novo caso como sua classe. Esse é um tipo trivial de reuso.

Adaptar: existem duas maneiras principais de reusar casos passados: reuso da solução do caso anterior (reuso transformacional) e reuso do método anterior que construiu a solução (reuso derivacional).

- Métodos de revisão

Quando uma solução do caso gerada pela fase de reuso não estiver correta, surge uma oportunidade de aprendizado por falhas. Essa fase é chamada de revisão do caso e consiste de duas tarefas: (1) avaliar a solução do caso gerada pelo reuso. Se bem-sucedido, aprender a partir do sucesso (retenção de caso) e (2) caso contrário, reformular a solução do caso usando conhecimento específico do domínio ou entrada do usuário.

- Métodos de retenção

Esse é o processo de incorporação do que é útil recuperar, a partir do episódio de solução do novo problema, dentro do conhecimento existente. O aprendizado por sucesso ou falha da solução proposta é *triggered* pelo resultado da avaliação e possível reparo. Isso envolve selecionar qual informação reter do caso, de que forma retê-la, como indexar o caso para posterior recuperação dos problemas similares e como integrar o novo caso à estrutura de memória.

2.4.4 Sumário da Seção

Essa seção descreveu uma visão geral de Raciocínio Baseado em Casos enfatizando o processo básico de um sistema de RBC juntamente com alguns dos principais problemas, em grande parte não resolvidos, encontrados na área.

Vale observar que esta técnica é importante para representar conhecimento específico e que muitos ganhos podem ser obtidos quando o conhecimento geral do domínio é combinado com conhecimento específico.

Na próxima seção serão apresentados os Sistemas Tutores, levando-se em consideração a sua estrutura básica, sua importância e sua utilização.

2.5 Sistemas Tutores

As modalidades mais utilizadas de software educacionais não têm apresentado resultados conforme o esperado e a distância entre os elementos envolvidos no processo de ensino-aprendizagem com tutores humanos e sistemas tutores tem sido considerável. Isso pode ser observado nos softwares educacionais, modalidade CAI (*Computer Aided Instruction*), nos quais o conteúdo programático, apesar de ser colocado de forma atrativa e atual (recursos multimídia), não permite uma utilização mais personalizada e adequada ao perfil do aluno que utiliza o software, uma vez que o comportamento do ambiente é sempre o mesmo, sessão após sessão. As causas são diversas e vão desde a falta de uma teoria de ensino-aprendizagem sistematizada que permita o projeto de sistemas educacionais mais robustos e eficazes, até as limitações de hardware e software ainda existentes nos dias de hoje (Viccari & Giraffa, 1996).

A idéia geral consiste em aplicar técnicas e métodos de Inteligência Artificial em sistemas CAI, surgindo assim os sistemas ICAI (*Intelligent Computer-Assisted Instruction*), onde o nome mais utilizado na literatura é Sistemas Tutores Inteligentes e cuja tendência é fazer com que o software se torne um elemento mais ativo no processo de interação com o aluno.

2.5.1 Inteligência Artificial e Educação

A incorporação de técnicas de IA na educação, a fim de produzir artefatos de computadores educacionalmente úteis, data do início dos anos de 1970. No início dos anos de 1980, pesquisas nesse campo foram claramente divididas em dois grupos diferentes, com o aparecimento de duas escolas de pensamento (Nwana, 1990). O primeiro, e menor dos dois grupos, advoga *ambientes de exploração*, que são ambientes que encorajam o descobrimento do aprendizado,

isto é, aprender fazendo. O segundo e maior grupo é o de *tutor inteligente*, que se referencia a ele mesmo como reformista, pois prefere um aperfeiçoamento gradual (isto é, evolução) na presença de qualidade de educação usando técnicas de IA. Eles advogam um paradigma onde o computador atua como um tutor.

2.5.2 Introdução a Sistemas Tutores Inteligentes

Sistemas Tutores Inteligentes são sistemas de ensino através de computador que procuram modelar o aprendizado humano na medida em que sabem o que ensinar (conteúdo), a quem ensinar (modelagem do estudante), como ensinar (estratégias pedagógicas) e de que forma ensinar (estratégias didáticas) (Marietto, 1996). Em STI são utilizadas técnicas de Inteligência Artificial que fazem o sistema agir com características inteligentes, como, por exemplo, detectar o momento de aprendizado do estudante e, a partir dessa informação, mudar as estratégias de ensino.

Vários trabalhos questionam a filosofia de construção de STI, discutindo a interrelação entre algumas de suas estruturas, como por exemplo, pedagógica, didática, psicológica, computacional e cognitiva. Entretanto, os STIs não têm representado de forma adequada a riqueza do processo de aprendizagem e várias pesquisas estão sendo desenvolvidas com o objetivo de propor arquiteturas que os melhor representem. O projeto e desenvolvimento de tais tutores reside na intersecção entre ciência da computação, psicologia cognitiva e pesquisas educacionais. Essa área de intersecção é normalmente referenciada como ciência cognitiva.

O problema central com os primeiros sistemas consiste que eles eram incapacitados de proporcionar um *feedback* rico ou individualizado, uma vez que não eram projetados para saber o que eles estavam ensinando, a quem eles estavam ensinando e como eles estavam ensinando.

Com relação ao reconhecimento dos estados mentais do estudante, os atuais STIs têm limites pois, diferentemente do tutor humano, ainda não podem reconhecer características tais como o estado de ânimo de um estudante, sua expressão facial e suas motivações para o estudo. Tem-se, então, que o rastreamento que os STIs devem realizar dos estados de conhecimento e comportamento do estudante atualmente é incompleto, devido à ausência de uma teoria consistente de representação desses estados mentais e de técnicas eficazes para

implementar o diálogo homem-máquina.

O entendimento do conteúdo depende também da natureza do conhecimento abordado. Essa natureza relaciona-se à forma como o conhecimento é interpretado e manipulado pelo cérebro humano, sendo que a maioria dos STIs trabalham com o conhecimento do tipo procedural e/ou declarativo.

O conhecimento procedural está relacionado a tarefas possíveis de serem algoritmizadas, ou seja, de terem sua execução dividida em comandos básicos e uma estrutura de controle do fluxo desses comandos. Na representação do conhecimento procedural usam-se, geralmente, regras de produção. O uso de regras de produção permite representar tanto regras de inferência específicas do conteúdo abordado quanto especificações de comportamento genérico. Já o conhecimento declarativo está relacionado com a manipulação de conjuntos de informações, sendo que em sua representação os *frames* e as redes semânticas têm maior aplicabilidade.

Esta predominância do uso de regras de produção, *frames* e redes semânticas na representação do conhecimento evidencia a ligação existente entre STI e SBC. Isso porque, na maioria das vezes, tenta-se aproveitar a teoria utilizada na construção da base de conhecimento de um SBC para a construção do módulo do domínio do STI. Entretanto, essa adaptação falha, pois a forma de representação do conhecimento deve atender a finalidade do aplicativo. No caso dos STI, pretende-se representar o conhecimento humano de forma mais abrangente possível, e a natureza do mesmo não se restringe a procedural ou declarativa. Por isto, conceitos como redes neurais, algoritmos de vida artificial, entre outros, poderiam ser usados como diferenciais no momento da estruturação da modelagem do estudante (Marietto, 1996).

Nos STIs, o ensino é apoiado em uma grande base de conhecimento, construída pelos especialistas, a respeito do tema a ser ensinado. A partir dessa base, o sistema interage com o aluno, como um tutor computadorizado, com o poder indutivo similar a de um professor humano, onde a modelagem cognitiva é constante e progressiva. Para ser “inteligente”, um tutor deve ser flexível, isto é, ter capacidade para aprender com o meio ambiente e atualizar seu conhecimento (Viccari & Giraffa, 1996).

2.5.3 Estrutura Geral de um Sistema Tutor Inteligente

Os Sistemas Tutores Inteligentes existentes variam muito quanto ao tipo de arquitetura. A Figura 2.10 ilustra uma arquitetura básica de um STI, que é composta pelo módulo do especialista, módulo do estudante, módulo do tutor e módulo da interface.

Figura 2.10: Arquitetura Básica de um Sistema Tutor Inteligente.

Módulo do Domínio

O módulo do domínio ou módulo do especialista compreende os fatos e as regras de um domínio específico a ser transmitido para o estudante, isto é, o conhecimento dos especialistas. Ele é o componente especialista do tutor, constituído pelo material instrucional, por uma geração sistemática de exemplos, pela formulação de diagnósticos e pelos processos de simulação.

Esse módulo cumpre uma função dupla. Primeiramente, ele serve como origem do conhecimento a ser apresentado ao estudante, que inclui geração de questões, explicações e respostas. Em segundo lugar, ele fornece um padrão para avaliação do desempenho do estudante. Nesse último caso, ele deve ser capaz de gerar soluções para problemas no mesmo contexto do estudante, de modo que respostas respectivas possam ser comparadas.

Módulo do Estudante

O módulo do estudante refere-se a representação dinâmica do conhecimento emergente e habilidades do estudante. Nenhum STI pode ser desenvolvido sem uma compreensão do estudante.

Esse módulo representa o conhecimento e as habilidades cognitivas do aluno em um dado momento. É constituído por dados estáticos e dados dinâmicos (desempenho do aluno), que serão de fundamental importância para o tutor poder comprovar hipóteses a respeito do aluno.

Esse módulo deve incluir todos os aspectos do comportamento e conhecimento do estudante que têm possíveis repercussões no seu desempenho e aprendizado.

Esse módulo desempenha duas funções: age como fonte de informação sobre o estudante, e serve como uma representação do conhecimento do estudante. Completando essas funções, ele atua em papéis incluindo correção, elaboração, estratégia, diagnóstico e avaliação.

Módulo do Tutor

O módulo do tutor é a parte do STI que projeta e regula as interações instrucionais com o estudante. Em outras arquiteturas, esse módulo é referenciado como estratégias didáticas ou módulo pedagógico. Ele está intimamente ligado ao módulo do estudante, utilizando o conhecimento sobre o estudante e sua própria estrutura tutorial para decidir quais atividades pedagógicas serão apresentadas. A ordem e maneira no qual os tópicos são tratados podem produzir diferentes experiências de aprendizado.

Uma estratégia é um plano, ou seja, uma estrutura de ação visando atingir determinados objetivos. As estratégias de ensino podem ser vistas como “esquemas de planos”, que definem formas de apresentar o material instrucional ao aluno.

Módulo da Interface

O módulo de interface com usuário é o componente de comunicação do STI que controla a interação entre o estudante e o sistema. Em ambas as direções, ele faz a tradução entre a representação interna do sistema e uma linguagem de interface que é compreensível pelo estudante.

Sabe-se que uma boa interface é vital para o sucesso de qualquer sistema interativo, e sistemas tutores inteligentes não constituem uma exceção (Nunes et al., 1993). Pelo contrário, pode-se dizer que a questão da interação cresce de importância nesta classe de sistemas, pois é na interação que o sistema tutor exerce duas de suas principais funções: a apresentação do material instrucional e a monitoração do progresso do estudante através da recepção da resposta do aluno.

2.5.4 Sumário da Seção

Os Sistemas Tutores Inteligentes transcendem os enfoques de treinamento e prática, pois estão projetados para assumir a responsabilidade de ensinar, utilizando um diálogo de iniciativa mista e os seres humanos e a máquina são tratados como processadores de conhecimento.

Nessa seção foram apresentados alguns dos pontos principais relacionados à ST, como seus conceitos básicos e sua estrutura geral.

Na próxima seção serão apresentadas noções básicas sobre simulação.

2.6 Simulação

Um simulador pode ser definido como “Um dispositivo em um laboratório que possibilita o operador produzir, sob condições de teste, fenômenos como os que ocorrem no desenvolvimento real” (Webster’s Third New International Dictionary).

Essa definição estabelece que simulador é um “dispositivo” que funciona realmente como algo real. Quando essa definição foi desenvolvida, provavelmente era verdade que um modelo de um dispositivo em funcionamento poderia ser imaginado como um “simulador”. Hoje em dia, é mais comum um simulador ser um programa de computador, mas o objetivo do programa é o mesmo do modelo na frase anterior — produzir “... fenômenos como os que ocorrem no desenvolvimento real” (Sylvester & Pham, 1997).

A simulação é uma das mais poderosas ferramentas de análise disponível para projeto e operação de processos ou sistemas. O processo de simulação antes da implantação do sistema real é muito importante uma vez que permite a aceleração do funcionamento do sistema no tempo, possibilita prever os quase inevitáveis acidentes que ocorrem na implan-

tação de um sistema real e permite poupar recursos econômicos, pois dispensa a construção de protótipos físicos para testes.

Devido a sua grande versatilidade e flexibilidade, a simulação é largamente utilizada em operações técnicas e atividades de pesquisa. Algumas das muitas áreas de aplicação da simulação são: sistemas de computação, sistemas de manufatura, negócios, entidades governamentais, ecologia e meio-ambiente, sociedade e comportamento, biociências, etc.

As ferramentas de simulação desenvolvidas por engenheiros e projetistas para estudo dos mais diversos sistemas apresentam uma evolução diretamente ligada com a tecnologia de suporte (hardware e software) disponíveis no momento de sua implementação, tendo evoluído desde os modelos físicos em escala reduzida e os modelos matemáticos até a tendência de simuladores inteligentes e interativos com interface gráfica. Esses sistemas devem simular o mundo real de forma mais simples. Eles permitem que o aprendiz seja inserido em um espaço, onde ele pode explorar e aprender vários princípios. O aprendiz é colocado diante de várias situações, onde ele pode desenvolver hipóteses, testá-las e analisar os resultados. O aprendiz percebe o resultado de sua ação e a modifica, caso o resultado observado não tenha sido o esperado. Existem vários tipos de simulações, como: manipulação de materiais e substâncias onde existem riscos de explosão, de contaminação, etc.; situações onde o aprendiz possui uma tarefa, como: pilotar um avião, gerenciar uma cidade, etc.; situações de desastres, como: enchentes, tornados, terremotos, etc.; e situações de alto custo, como: tomar decisões de custos em uma viagem ao redor do mundo.

2.6.1 Conceitos Básicos Envolvendo Simulação

Muitas pessoas referem-se a simulações como “caixas pretas”. Pode-se fornecer uma entrada à “caixa preta”, e ela retorna uma saída ou resultado. A simulação pode consistir no envio contínuo de entradas para a “caixa preta” e obtenção das saídas.

Simulação pode ser exercitada de forma a mimetizar o comportamento do sistema. Através do experimento (simulação), várias observações são realizadas para dar subsídio às várias conclusões sobre o sistema (Soares, 1992).

Esses experimentos, ou *simulações*, vão permitir inferências sobre os sistemas: sem construí-los, se são apenas sistemas propostos; sem perturbá-los, se são sistemas de alto custo operacional ou sem segurança para que se façam experimentos; e sem destruí-los, se o

objetivo da experimentação é determinar os limites de funcionamento do sistema.

Desta forma, simulação pode ser usada tanto para projeto, quanto para análise de procedimento ou até para determinação de desempenho. Em outras palavras, simulação pode ser empregada como ferramenta: exploratória para definição de um sistema ou problema; de análise para a detecção de elementos críticos; para síntese e avaliação de soluções propostas e de planejamento para desenvolvimentos futuros.

2.6.2 Uso de Simulação para Treinamento e Ensino

Simulações por computador são usadas em muitos contextos, como análises *“what-if”*, experimentação e instrução.

O uso instrucional de simulação em meio computacional possui quatro características básicas (Jong, 1991):

- presença de modelos subordinados formalizados e manipuláveis: em simulação por computador, isso significa que um fenômeno, um processo, ou um sistema é formalizado em um modelo e implementado como um programa de computador. O modelo pode ter características qualitativas e/ou quantitativas;
- presença de objetivos de aprendizado: a simulação deve ser usada para pesquisar um certo objetivo de aprendizado. Esses objetivos podem ser de diferentes tipos, como: aquisição de conhecimento sobre um modelo subordinado; obtenção de procedimentos específicos ou desenvolvimento do conhecimento sobre o processo que está sendo simulado (experimentação). É essencial que a saída do programa seja calculada ou inferida pela implementação do modelo em resposta à entrada do estudante;
- elicitación de processos específicos de aprendizado: a simulação deve ser usada com a finalidade de invocar características específicas de processos de aprendizado (como geração de hipóteses). O estudante pode gerar suas próprias hipóteses e testá-las logo em seguida. Essa característica exclui as situações em que o estudante é forçado a uma manipulação randômica, por exemplo, por causa de uma lacuna de um conceito anterior necessário;
- presença de atividade do estudante: deve existir algum nível de atividade do estudante, ou seja, o estudante deve realmente manipular algo na simulação, por exemplo,

setando variáveis e parâmetros de entrada, escolhendo variáveis de saída, etc. Essa característica exclui as situações em que a simulação é usada meramente como dispositivo de demonstração.

Segundo Hebenstreit (1987) *apud* (Jong, 1991) a simulação por computador cria um “mundo” entre a realidade e algum modelo abstrato; essa camada intermediária pode ajudar o estudante a transpor a distância entre a realidade e o modelo.

Empregar simulações em instrução é importante por vários motivos. Provavelmente, o mais importante deles é que os estudantes estarão comprometidos numa atividade de exploração e aprendizado — uma abordagem que é defendida em teorias modernas de instrução/aprendizado. Criando realidades hipotéticas ou modificando escalas de tempo em simulações, pode-se manter essa maneira de aprendizado. Razões adicionais para a utilização de simulação são os aspectos de motivações e a possibilidade de se criar situações inaceitáveis no mundo real, por razões de perigo, custo ou tempo.

Existe ainda um grande número de outras razões, entre elas, tem-se o aumento da motivação, intensificação do aprendizado cognitivo de informações factuais, melhor entendimento do processo, melhoria das habilidades de pensamento crítico, aperfeiçoamento da capacidade de transferência de aprendizado para outras situações, desenvolvimento de uma atitude mais positiva frente ao aprendizado e ao processo instrucional, desenvolvimento de habilidades sociais e de comunicação, melhora do ambiente em sala de aula, e aumento da flexibilidade do professor. Outro ponto que deve ser considerado, no uso de simulação em instrução, relaciona-se ao econômico e outras objeções ou dificuldades em aprendizado em situações da vida real.

Entretanto, estudantes podem ser muito passivos, não usando as oportunidades que o ambiente de simulação oferece. Por isso, um suporte profissional é necessário, caso se deseje que o aprendizado por simulação se torne efetivo. Em alguns casos, esse suporte é fornecido por instrutores humanos, podendo também ser fornecido por um ambiente de aprendizado por computador. Esse tipo de ambiente de simulação inteligente é conhecido como *Intelligent Simulation Learning Environment* (Jong, 1991).

ISLE utiliza simulação juntamente com *helps* avançados, sugestões, facilidades de explicações e ensino, etc. Tal ambiente deve oferecer ao estudante um roteiro através da simulação, guiado pelas suas necessidades instrucionais específicas.

Porém, apesar do recente desenvolvimento e do fato de que numerosas simulações são utilizadas em instrução, há ainda pouca pesquisa em como oferecer suporte aos estudantes em aprendizado exploratório.

Existem outras características do estudante que possivelmente podem estar relacionadas com o aprendizado por simulação por computador, como, por exemplo, conhecimento anterior, estilo cognitivo e enfoque de aprendizado. Uma análise dessas características sugere que pode ser coerente não dar aos estudantes uma simulação pura, com liberdade máxima, mas ter essa liberdade reduzida adicionando-se direção explícita à simulação.

2.6.3 Sumário da Seção

O desenvolvimento das ferramentas de simulação é uma consequência da globalização econômica e o decorrente aumento da competitividade foi bastante favorecido pela veloz evolução tecnológica da eletrônica e da informática. O aumento da competitividade exige das empresas cortes substanciais tanto no desenvolvimento de novos produtos quanto na operação das plantas e treinamento de recursos humanos sendo a simulação uma ferramenta que possibilita a realização de tais tarefas com custos efetivamente reduzidos quando comparados com os custos da realização das mesmas utilizando sistemas físicos reais. A evolução tecnológica vivenciada permite que o aumento do poder de análise dos sistemas simuladores ocorra paralelamente com a diminuição dos preços dos mesmos, realimentando positivamente o processo e criando um ciclo virtuoso que traz incentivos tanto para os usuários quanto para os desenvolvedores destes sistemas simuladores.

Nessa seção foram abordados temas gerais sobre o processo de simulação, bem como algumas considerações sobre a sua utilização em treinamento e ensino.

2.7 Considerações Finais

Neste Capítulo foram mostrados pontos relevantes e importantes que devem ser considerados ao se construir um ambiente que vise o treinamento e ensino de usuários. A simulação foi também apresentada por se tratar de uma ferramenta para treinamento, que pode ter suas deficiências supridas através da utilização de um Sistema Inteligente que dê o apoio necessário ao usuário no caso de dúvidas ou dificuldades.

Outro ponto que merece destaque é o fato da Aquisição de Conhecimento ser o domínio de aplicação neste primeiro protótipo do SLATE, sendo esse um dos motivos da ênfase dada a esse tema.

O capítulo seguinte apresenta algumas considerações gerais sobre ambiente de apoio a treinamento e ensino, uma visão geral e a arquitetura do Sistema Inteligente de Apoio a Treinamento e Ensino, bem como uma explicação sobre cada um dos módulos e componentes desse ambiente.

Capítulo 3

Ambiente de Apoio a Treinamento e Ensino

3.1 Considerações Iniciais

Para um Ambiente de Apoio a Treinamento e Ensino é interessante utilizar as vantagens e facilidades de diferentes tipos de recursos, que juntos propiciem um ambiente de fácil utilização e acesso por parte do estudante/treinando¹, visando um aprendizado rápido e objetivo.

É amplamente reconhecido que a utilização de recursos hipermídia em ambientes de ensino oferece uma maior flexibilidade de uso e uma melhor apresentação das informações aos usuários, estimulando-os a aprender. Por outro lado, técnicas de Inteligência Artificial podem contribuir em ambientes dessa natureza capacitando o sistema a “decidir” sobre que informações apresentar e como e quando apresentá-las, baseando-se em informações sobre o usuário, em conhecimento especialista do domínio da aplicação, em inferências obtidas a partir desse conhecimento e em casos passados, entre outras fontes.

Este trabalho propõe um ambiente voltado a treinamento e ensino que integre características de Sistemas Baseados em Conhecimento, Raciocínio Baseado em Casos, Sistema Tutor e Simulação, utilizando ainda recursos de hipermídia, e é definido de acordo com os seguintes objetivos básicos:

¹Neste capítulo os termos treinando, estudante, aluno, treinando/estudante serão utilizados indistintamente, significando o usuário do ambiente aqui proposto.

- estender a gama de conhecimento a ser explorado, de modo que o treinando possa tirar dúvidas conceituais sobre o domínio;
- fornecer ao treinando a oportunidade de questionar sobre o que ele está observando ou observou na simulação;
- ilustrar, com casos análogos, situações ocorridas durante a simulação. Esses casos são aqueles considerados representativos pelos especialistas do domínio. Os parâmetros que indexarão os casos e que servirão de base para a função de similaridade a ser usada deverão ser definidos com tais especialistas;
- armazenar informações a respeito de cada treinando, de modo que seja possível fazer inferências a respeito do progresso do treinando, enquanto este estuda o domínio da aplicação.

Neste capítulo é apresentada uma visão geral do ambiente de apoio a treinamento e ensino, bem como sua arquitetura com uma explicação e especificação de seus módulos e componentes.

3.2 Visão Geral do Ambiente de Apoio a Treinamento e Ensino

Dentro dessa abordagem mostrada está a arquitetura de um ambiente denominado Sistema Inteligente de Apoio a Treinamento e Ensino, que integra características de Hipermídia, Simulação, Sistema Baseado em Conhecimento, Raciocínio Baseado em Casos e Sistema Tutor, propiciando a construção de um ambiente adequado para treinamento e ensino utilizando as vantagens pertencentes a cada uma dessas tecnologias. Nos moldes dos *Intelligent Simulation Learning Environments* (Jong, 1991), este ambiente combina a estratégia de exploração do domínio pelo estudante, através de um hiperdocumento, com recursos de perguntas e respostas e simulação baseada no conhecimento especialista.

Um dos objetivos deste trabalho está relacionado com a proposta de uma arquitetura para um ambiente voltado ao treinamento e ensino, que promova acesso ao conhecimento através da WWW (*World Wide Web*), na tentativa de que o mesmo venha a ser fonte de

informação e de instrução local e/ou remota, além da utilização de hiperdocumentos, a fim de proporcionar uma interação amigável entre o estudante e o sistema.

Uma das características visadas pelo SIATE é a liberdade de movimento no hiperdocumento que o treinando/estudante possui para a investigação do domínio a ser estudado, como por exemplo: Aquisição de Conhecimento, Arquitetura de Computadores, Linguagens de Programação, Recursão, etc. Porém, o estudante pode também recorrer ao auxílio do sistema, quando necessário.

Além disso, por se tratar de um ambiente exploratório e de reflexão, evita-se a interrupção involuntária do estudante/treinando. Mesmo quando questionado, o sistema tende a priorizar respostas que estimulem o estudante a fazer sua própria análise da situação, através, por exemplo, do fornecimento de casos análogos, sugestão de atividades, proposta de resolução de exercício, releitura de informações contidas no hiperdocumento, percurso de um roteiro pré-definido ou geração de um roteiro dinâmico específico sobre o tópico em questão.

Para validar a arquitetura proposta foi construído um protótipo utilizando como domínio o tema Aquisição de Conhecimento. Por ser um processo, a AC envolve a extração, interpretação e representação do conhecimento de um dado domínio e é considerada como sendo o estágio mais difícil e precário no processo de construção de Sistemas Inteligentes. Portanto, o SIATE tem por objetivo o aprendizado e treinamento de equipes de Aquisição de Conhecimento, de maneira a torná-los aptos a efetuarem uma boa e eficiente Aquisição de Conhecimento em qualquer que seja o domínio desejado. Então, sua função principal é auxiliar os membros de uma equipe de Aquisição de Conhecimento, principalmente os Engenheiros de Conhecimento, público alvo do SIATE, a aprenderem como efetuar uma Aquisição de Conhecimento de forma correta e eficiente (Rezende & Pugliesi, 1998). Sendo assim, os exemplos apresentados a seguir se referem ao domínio de conhecimento escolhido para este protótipo de aplicação.

Esta arquitetura é proposta por dois trabalhos de mestrado do ICMC-USP (Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo), um correspondente à parte do Sistema Tutor que dispõe do apoio pedagógico do SIATE e do Módulo referente ao Material Didático (Freire, 1998), e outro referente ao desenvolvimento do Sistema Híbrido de Conhecimento, do Módulo Gerador de Roteiros e do Módulo Prática/Treinamento, objeto de estudo deste trabalho.

Na Figura 3.1 é mostrada uma visão geral do ambiente proposto para apoio a treinamento e ensino. Nessa figura é ilustrada uma visão geral do SIATE, como pode ser observada nas três camadas e dois níveis.

Figura 3.1: Visão Geral do Ambiente SIATE.

O primeiro nível corresponde à interface inicial do SIATE. Nele podem ser observadas duas camadas de interface distintas: tela de *login* e tela principal do sistema.

O segundo nível, que possui apenas a terceira camada, mostra os módulos e os componentes do sistema e corresponde à estrutura do SIATE, apresentada na Figura 3.2.

3.3 Arquitetura do Sistema Inteligente de Apoio a Treinamento e Ensino

O Sistema Inteligente de Apoio a Treinamento e Ensino é composto por quatro módulos (Material Didático, Roteiros, Help e Prática/Treinamento) com o apoio de dois componentes (Sistema Tutor e Sistema Híbrido de Conhecimento), bem como uma interface para interação com o usuário. Os elementos que compõem o SIATE serão descritos a seguir, e na Figura 3.2 podem ser observados tanto os módulos e componentes (que estão sendo implementados em dois projetos de mestrado) quanto a comunicação e a transmissão de dados existente entre esses elementos.

Figura 3.2: Arquitetura do Ambiente SIATE.

3.3.1 Interface

Um computador multimídia certamente se distingue quando utilizado para treinamento ou educação numa situação um-a-um com o estudante. A capacidade da apresentação multimídia pode mostrar ao estudante material de vários tipos de origem: texto, gráfico, áudio,

vídeo, animações, simulações ou fotografias. Quando esses recursos são combinados com interatividade, normalmente, é criado um ambiente de aprendizado mais efetivo. O estudante tem a oportunidade de selecionar o material de aprendizado que ele desejar, e o computador pode traçar o caminho do progresso do estudante e oferecer uma apresentação de acordo com as necessidades desse estudante. A criação de aplicações de treinamento precisa de um tipo de software diferente daqueles utilizados para apresentações comerciais, pois é necessário um rico ambiente interativo (Luther, 1994).

O módulo de interface com o usuário é o componente de comunicação do sistema que controla a interação entre o estudante/treinando e o sistema. Em ambas as direções, traduz a representação interna do sistema para uma linguagem de interface que é compreensível para esse estudante/treinando. Deve acomodar os objetivos do sistema e do estudante, podendo enriquecer esta interação com outros meios de comunicação (som, imagem, etc.), além de fornecer meios para que seja possível ao professor e/ou especialista alterar ou atualizar o sistema (Nunes et al., 1993).

A interface tem que ser construída de uma maneira que estimule o estudante, tornando-a mais amigável. Quanto mais um estudante se sentir inserido no seu ambiente de aprendizado, mais rica será a sua experiência, e esta inserção pode ser facilitada através da interação com uma interface de múltiplos meios. A escolha do meio de comunicação mais adequado ao tópico a ser ensinado e a eficácia da comunicação são problemas relevantes.

A partir do módulo interface torna-se possível a interação entre o estudante e o sistema. Para a construção, tanto da interface quanto do sistema como um todo, foram utilizadas regras e técnicas de desenvolvimento de interface, tais como as propostas por (Preece et al., 1994) em estudos feitos sobre HCI. Tais regras abrangem características como: disposição, tamanho e quão representativos são os ícones, utilização de cores, número e equilíbrio dos objetos dispostos em uma tela, e alguns outros pontos importantes para a obtenção de aplicações com recursos visuais e uma interação amigável e bem projetada.

Ao entrar no SIATE, é apresentado ao usuário uma tela inicial e, em seguida, uma página com dois campos, *login* e *password*, para que o mesmo possa ter acesso ao sistema (Figura 3.3). Caso o usuário esteja acessando o sistema pela primeira vez, ele deve confirmar a senha e preencher um formulário contendo alguns dados pessoais, que serão armazenados em uma base de dados. Essas informações incluem: nome completo, sexo, profissão, nível de conhecimento sobre o domínio e data de nascimento, que serão fornecidas apenas na primeira

vez, quando o usuário se cadastra no sistema. Essas informações fazem parte da Base do Estudante do SIATE, que está contida no Sistema Tutor.

Figura 3.3: Página de Identificação Ambiente SIATE.

A página seguinte a essa identificação e validação da senha é a tela principal do sistema (ver Figura 3.4), que contém como opções os quatro módulos do SIATE (Material Didático, Roteiros, Help e Prática/Treinamento) e a opção de saída do sistema. Vale lembrar, que essa tela corresponde à segunda camada do primeiro nível da Figura 3.1 (Visão Geral do Ambiente SIATE), e que, além disso, esses módulos contam com o apoio do Sistema Tutor e do Sistema Híbrido de Conhecimento.

Para implementar o cadastro dos usuários foi utilizado o software miniSQL, que é um gerenciador de bancos de dados para Unix, ou seja, é um programa que manipula conjuntos de informações armazenadas.

Maiores detalhes e informações podem ser encontradas em (Freire, 1998).

3.3.2 Material Didático

O Material Didático é o módulo do sistema que possui parte do conhecimento sobre o domínio que está sendo ensinado e/ou treinado, com o objetivo de transmitir (interagindo com os outros módulos quando necessário) ao estudante/treinando todo esse conhecimento da melhor

Figura 3.4: Página da Tela Principal do Ambiente SIATE.

forma possível. Esse módulo pode ser implementado utilizando qualquer recurso para construção de sistemas, porém a decisão de qual ferramenta utilizar dependerá tanto do domínio da aplicação, quanto da plataforma disponível e público alvo ao qual o sistema se destina. Uma forma interessante e atual de se implementar esse módulo é através da utilização de um hiperdocumento, onde podem ser utilizados diversos recursos multimídia, visando auxiliar e acelerar o aprendizado do treinando, bem como aumentar seu interesse pelo assunto estudado.

O Módulo Material Didático consiste de um hiperdocumento para WWW que possui informações sobre o domínio da aplicação (no caso do protótipo implementado, a Aquisição de Conhecimento) e sobre o ambiente de ensino SIATE, como pode ser observado na Figura 3.5. É nele também que estão presentes as características e descrições do ambiente SIATE, tais como sua arquitetura geral e os módulos que a compõe, fornecendo ao estudante um visão geral do ambiente que está utilizando. Outras informações também contidas neste módulo são referentes tanto ao glossário com alguns termos técnicos utilizados, quanto à bibliografia utilizada para o estudo do domínio especialista e para a confecção do hiperdocumento.

O Material Didático tem seu domínio modelado utilizando as regras do HDM (*Hipertext Design Model*), que é uma forma de modelagem utilizada para representar graficamente aplicações hipermídia. A implementação deste módulo utiliza HTML (*Hipertext Markup*

Figura 3.5: Exemplo de uma Página do Material Didático.

Language), bem como Applets Java, JavaScript, CGIs (*Common Gateway Interface*) e MySQL. Maiores detalhes quanto a implementação encontram-se descritos em (Freire, 1998), e referentes ao conhecimento disponível no Material Didático em (Rezende & Pugliesi, 1998; Jubileu et al., 1998; Félix et al., 1998; Rezende et al., 1998; Horst et al., 1998).

3.3.3 Roteiros

O Módulo Roteiros é responsável pela disponibilização de roteiros (guia direcionado) sobre o material didático existente. Esses roteiros podem tanto ser sugeridos a partir de inferências efetuadas pelo SHC (dinâmicos), quanto selecionados a partir de um menu de opções que contém os roteiros pré-definidos. Portanto, os roteiros podem ser pré-definidos pelo autor, ou gerados dinamicamente pelo SHC, levando-se em consideração os objetivos, necessidades e o histórico do treinando. Os roteiros se referem ao conteúdo do domínio de conhecimento contido no sistema.

Portanto, este módulo pode ser visto como contendo duas funções básicas e principais, que são a seleção e geração de roteiros, como pode ser visto na Figura 3.6.

Na seleção de roteiros, o estudante poderá selecionar diretamente um dos roteiros pré-definidos, por meio de um menu de opções. Esse tipo de roteiro deve ser definido por um professor ou conhecedor do domínio de conhecimento e tem o intuito de auxiliar e gui-

Figura 3.6: Manipulação de Roteiros.

ar estudantes que possuam pouco ou nenhum conhecimento do ambiente ou domínio da aplicação.

Já na geração de roteiros, através da interação SHC-estudante, o estudante fornece ao SHC informações sobre as quais serão feitas inferências, a fim de detectar o ponto que estudante deseja atingir (por dificuldade ou interesse em aprender) ou que o sistema concluiu que o estudante precisa praticar mais.

Ambos os tipos de roteiros ocorrem por iniciativa do estudante ou por uma sugestão do ST, uma vez que nesse sistema o estudante possui liberdade de ação e tomada de decisão.

Quando se está utilizando um roteiro, surge na parte inferior da página um frame com botões que conduzem o estudante à página anterior ou posterior pertencente ao roteiro em questão, bem como esse roteiro pode ser visualizado através de um grafo, onde cada nó do grafo corresponde a uma página do hiperdocumento pertencente ao roteiro. Essa visualização é baseada no roteiro que está sendo utilizado atualmente e tem como finalidade oferecer ao estudante uma visão de onde ele se encontra dentro do roteiro como um todo e quais são as páginas que pertencem a esse roteiro. E como quarta opção tem-se o botão para sair do roteiro, que retorna para a página com os roteiros pré-definidos. A Figura 3.7 mostra uma página da WWW com alguns dos recursos disponíveis no SIATE, entre eles um roteiro pré-definido com grafo ativado em uma tela principal do material didático.

Figura 3.7: Exemplo de uma Página do Protótipo do SIATE com um Roteiro Ativado.

Uma observação relevante é que caso o estudante saia do roteiro por meio de algum outro *link* contido na página, ele pode retornar ao ponto em que parou no roteiro, através da utilização dos botões do roteiro **Anterior** e **Próximo**. Isso possibilita que o estudante possa ter acesso a uma página diferente que lhe despertou o interesse, mas não perder o roteiro que estava percorrendo (voltando aonde estava).

Detalhes referentes a esse módulo serão mais explorados no Capítulo 5.

3.3.4 Help

O Módulo Help disponibiliza informações sobre a utilização de cada um dos módulos do SIATE, bem como explicações sobre a melhor maneira de se utilizar os recursos oferecidos pelos seus componentes do sistema.

Este módulo está disposto na forma de um hiperdocumento, e possui suas informações separadas em ordem alfabética de modo a facilitar a busca por parte do estudante/treinando. Cada item de informação corresponde a uma página do hiperdocumento, onde se pode encontrar detalhes sobre o item em questão.

3.3.5 Prática/Treinamento

O Módulo Prática/Treinamento permite que o estudante estude questões e dúvidas sobre o domínio de aplicação do sistema, com o apoio especializado do Sistema Híbrido de Conhecimento. Este módulo utiliza recursos que possibilitam ao estudante treinar um evento ou fato do mundo real, sendo que as tarefas ou a liberdade do estudante varia de acordo com o domínio, simulador ou ferramenta para treinamento que está sendo utilizado no sistema.

Para a construção da interface deste módulo foram estudadas teorias e regras contidas em HCI, de modo que se tornasse viável a efetiva utilização de um simulador em um ambiente de treinamento e ensino (ver Figura 3.8).

Figura 3.8: Projeto das Funcionalidades do Módulo Prática/Treinamento.

Para tanto foram estudados e projetados os possíveis ícones a serem utilizados na interface do módulo Prática/Treinamento de um sistema de apoio a treinamento e ensino, levando-se em consideração as técnicas de Interação Usuário-Computador, o *background* de possíveis usuários e as necessidades das pessoas ou empresa interessada pelo projeto.

O projeto de ícones, mesmo não abrangendo todos os conceitos apresentados, proporcionou um volume relevante de informações, sobre como projetar uma interface que realmente abranja aspectos importantes como: características físicas e psicológicas dos usuários, funcionalidade, facilidade de utilização, clareza e coerência de informações, alto nível de utilização e interação, uma aparência agradável, utilização de cores e representação gráfica em interfa-

ces, restrições de atenção e memória e que, além disso, sejam especificamente relacionados ao domínio dos usuários ou que mantenham padrões pré-existentes que sejam familiares aos usuários, seu *background* e seu perfil.

Informação quanto ao *background* dos usuários, utilização de ícones e interface gráfica foram obtidas através da aplicação de um questionário aos possíveis usuários do sistema.

Para isso, foram utilizadas algumas regras que envolvem características sobre perguntas (que não devem ser ambíguas e devem ser claramente explicitadas), características sobre nível de informações a serem levantadas (que não se deve ter escalas diferentes num mesmo teste) e sobre tamanho do questionário (que não se deve ultrapassar mais que dois lados de uma folha de papel, a não ser em algum caso muito específico).

No caso exemplificado no SIATE, a ferramenta para treinamento utilizada é a biblioteca *MCC++* (Kohavi et al., 1994) (apresentada com mais detalhes no Capítulo 6), constituída por algoritmos de Aprendizado de Máquina, possibilitando ao estudante realizar experimentos práticos com diferentes algoritmos. Esse treinamento será realizado, quando o sistema inteligente identificar sua necessidade, visando auxiliar o treinando/estudante, principalmente quanto à Aquisição de Conhecimento Implícito, de forma que seu aprendizado seja mais rápido e efetivo.

3.3.6 Sistema Tutor

O Sistema Tutor é o responsável por orientar o estudante em relação às suas dúvidas e dificuldades quanto ao domínio da aplicação, enquanto navega pelo Material Didático, além de avaliá-lo após a navegação por um roteiro (Nunes & Hasegawa, 1997). Neste ambiente, existem duas formas do estudante interagir com o ST. Na primeira delas, o ST extrai informações sobre o estudante, as quais armazena na Base do Estudante. Na segunda, o estudante pode acionar o ST através de botões disponíveis em páginas do Material Didático, a critério do autor.

Quando o ST é acionado, o estudante é conduzido a um menu de opções, onde estão disponíveis os tipos de auxílio que o ST tem a oferecer: Perguntas, Exercícios, Exemplos e Sugestões. As Perguntas são relativas ao conteúdo da página corrente, sendo que as respostas são pré-definidas, ou seja, as *Frequently Asked Questions* (FAQs). Por outro lado, os Exercícios, Exemplos e Sugestões são oferecidos e construídos dinamicamente a partir de

informações da página corrente e da Base do Estudante (que páginas visitou, que exemplos viu, que exercícios resolveu, etc.). Maiores explicações sobre o ST podem ser encontradas em (Freire, 1998), por ser este módulo desenvolvido em um outro trabalho de mestrado. No protótipo sobre Aquisição de Conhecimento, uma das Sugestões envolve a apresentação, pelo Sistema Híbrido de Conhecimento, de casos análogos.

3.3.7 Sistema Híbrido de Conhecimento

O Sistema Híbrido de Conhecimento contém o conhecimento especialista do domínio necessário ao auxílio do treinando, de forma a acelerar e complementar o processo de treinamento/aprendizado. Para tanto, são utilizadas tecnologias de Sistema Baseado em Conhecimento e Raciocínio Baseado em Casos (Aamodt & Plaza, 1994; Goonatilake & Khebbal, 1995a). Por ser importante a utilização de casos para treinamento, o SHC integra casos específicos e conhecimento geral do domínio, sendo seu motor de inferência capaz de manipular tanto o conhecimento geral (regras e frames) quanto o específico (casos reais do domínio de aplicação). O conhecimento sobre AC no protótipo está sendo extraído através de técnicas de Elicitação de Conhecimento.

O SHC será visto com mais detalhes no Capítulo 4, por ser este o ponto principal desta dissertação.

3.4 Considerações Finais

Os sistemas de Hipermídia podem levar ao desenvolvimento de novos produtos educacionais, integrando vários aspectos da tecnologia de informação e metodologias de ensino. Isso torna possível o treinamento à distância através da WWW, que atualmente é um meio rápido de se obter informações sobre vários assuntos, a partir de diferentes partes do mundo.

O ambiente SIATE tem o objetivo de integrar a tecnologia de Hipermídia com técnicas de Inteligência Artificial visando o enriquecimento de um ambiente exploratório de ensino com conhecimento especialista do domínio.

Vale ressaltar que as atividades de simulação em ambientes de instrução são largamente advogadas por todas modernas teorias de ensino e aprendizagem. Certamente, a principal razão é o fato de que o aprendiz envolve-se ativamente na exploração e, conseqüentemente,

no aprendizado do conceito ou atividade em questão. Outras razões relevantes incluem a maior motivação para o envolvimento e a possibilidade de criar situações inadmissíveis no mundo real por questões de perigo, custo ou tempo.

Outro ponto que merece destaque é a possibilidade de interação que o estudante possui em relação ao ambiente SIATE, tanto para contar com explicações providas pelo SHC, quanto para criação dinâmica de roteiros a fim de suprir necessidades e resolver eventuais dúvidas que venham a surgir durante seu aprendizado.

O protótipo construído está sendo validado com estudantes e professores de Inteligência Artificial e outros profissionais da área, a fim de se levantar subsídios para eventuais correções e aprimoramentos.

Capítulo 4

Sistema Híbrido de Conhecimento

4.1 Considerações Iniciais

É possível obter bons resultados adicionando as características de Sistemas Baseados em Conhecimento em ambientes de simulação para o desenvolvimento de um sistema de treinamento. Informações e conhecimento complementares podem ser adquiridos a partir da interação com bases de dados e de conhecimento, o que pode acelerar o processo de aprendizado. A vantagem de se usar um SBC (sistema de raciocínio qualitativo) juntamente com o simulador (sistema de raciocínio quantitativo) é fornecer ao treinando a opção de, por exemplo, entender a simulação através dos mecanismos de explicação do SBC. A qualidade do SBC é determinante no sucesso de tal integração. Nesse sentido, muito investimento foi feito na área de levantamento de requisitos, Aquisição de Conhecimento e desenvolvimento do SBC.

É importante lembrar que várias pesquisas têm sido desenvolvidas no sentido de se utilizar casos do domínio em Sistemas Inteligentes (Aamodt & Plaza, 1994). Por ser importante a utilização de casos para treinamento, o sistema que está sendo desenvolvido visa também a integração de casos específicos e conhecimento geral do domínio.

Durante a interação com o treinando, a partir da interface, são fornecidos dados ao sistema, que servem também para alimentar o Sistema Híbrido de Conhecimento que, quando questionado pelo treinando, será capaz de fornecer explicações. Assim, ao detectar uma pergunta do tipo *“Por que isso?”*, o sistema deve estimular o raciocínio do treinando ao oferecer-lhe uma resposta parcial ou ainda lhe apresentar um caso análogo já analisado

por um especialista.

Nesta estrutura de sistemas podem ser feitas perguntas diretamente via botões com semântica bem definida (*Por quê? E agora?*), como também via interface, em linguagem controlada por menus. Enquanto que esse tipo de interface facilita a formulação de diversas perguntas, também se caracteriza por ser um meio bastante controlado de processamento da língua (Leite, 1991). Dessa forma, ao conseguir formular uma questão através do menu, garante-se, antecipadamente, que o sistema terá capacidade de respondê-la. As respostas nem sempre são resultados de inferências realizadas pelo sistema, ou seja, elas podem resultar de simples consultas a uma base de dados ou de casos.

As demais seções deste capítulo tratam da estrutura e implementação do Sistema Híbrido de Conhecimento, bem como do processo que foi utilizado para seu desenvolvimento e para a aquisição do conhecimento nele contido.

4.2 Estrutura do Sistema Híbrido de Conhecimento

O Sistema Híbrido de Conhecimento foi estruturado de modo a englobar e manipular conhecimento geral, conhecimento específico (casos do domínio da aplicação) e o conhecimento sobre a modelagem do domínio (para a geração de roteiros).

A Figura 4.1 mostra a estrutura geral do SHC, na qual podem ser vistas três partes principais, que são: um Motor de Inferência que agrupa todas as formas de raciocínio contidas no SHC, uma Base de Conhecimento que está subdividida em três sub-bases e contém o conhecimento sobre o qual serão feitas as inferências e a Camada de Interação Usuário-SHC, também dividida em três sub-partes, que faz a comunicação entre o sistema e o meio externo. A seguir serão detalhadas cada uma dessas partes, bem como suas sub-partes.

- Base de Conhecimento (BC): armazena todo o conhecimento existente no SHC. Esse conhecimento foi obtido através de especialistas do domínio e está representado por meio de algumas das linguagens de representação existentes, como regras de produção, redes semânticas, casos, etc. Ela está subdividida em três sub-bases: base com conhecimento geral, base de casos e base da modelagem do domínio (redes semânticas).
 - Conhecimento Geral: nessa base está contida parte do conhecimento do SHC, que corresponde à prática/treinamento. Esse conhecimento está na forma de regras

Figura 4.1: Estrutura do Sistema Híbrido de Conhecimento.

de produção e fatos, extraídos de especialistas do domínio em questão, que no protótipo do SHC, é a Aquisição de Conhecimento Implícito.

- Base de Casos: nessa base estão contidos alguns casos relevantes ao SHC, que envolve exemplos de utilização de algoritmos de AM contidos na *MLC++* (ver Capítulo 6). Os casos estão representados usando estruturas Prolog que serão detalhadas no Capítulo 6.
- Base da Modelagem do Domínio: essa base possui o conhecimento para a definição de roteiros dinâmicos, sendo esse conhecimento modelado usando o *Generic Semantic Model* (GSM) e representado por uma rede semântica estendida, como será visto com mais detalhes no Capítulo 5.
- Motor de Inferência do SHC: responsável pela realização de todo o raciocínio do SHC, que é baseado tanto nas respostas e informações fornecidas pelo usuário através da camada de interação usuário-SHC, quanto no conhecimento contido na BC. O MI, ao

manipular regras, utiliza tanto a linha de raciocínio *Backward Chaining* ou Encadeamento Regressivo, quanto a *Forward Chaining* ou Encadeamento Progressivo. Para sua implementação, utilizou-se o LPA WIN-Prolog¹, bem como a ferramenta Flex que o acompanha (Pugliesi & Rezende, 1997).

- Memória de Trabalho: corresponde à área de trabalho do sistema. É uma memória temporária, que armazena todas as conclusões intermediárias ou provas realizadas durante a execução dos programas e inferências contidas no MI, além de acompanhar o problema através de todas as suas fases. Ela contém os dados de entrada do problema, as respostas do usuário, as asserções que foram deduzidas da BC, etc. A MT carrega a sequência histórica do que se tem feito com o objetivo de poder justificar, se solicitado, os passos da linha de raciocínio.
- Camada de Interação Usuário-SHC: responsável pela comunicação com o usuário e obtenção de informações, sobre o problema em questão, através de perguntas feitas sucessivamente ao treinando/estudante e validação das mesmas por meio de funções de crítica baseadas no conhecimento fornecido pelos especialistas do domínio.
 - Módulo Coletor de Dados (MCD): faz perguntas ao estudante e armazena suas respostas, visando a escolha de um bom algoritmo de AM para ser executado (simulação), bem como a definição do roteiro apropriado.
 - Módulo de Explicação (ME): fornece ao estudante explicações sobre as conclusões que o SHC chegou, especialmente sobre a simulação executada utilizando a *MCC++* (prática/treinamento).
 - Módulo de Saída de Roteiro (MSR): a principal função deste módulo é a escrita, na Base de Roteiros, do roteiro gerado dinamicamente.

Vale ressaltar que, como pode ser observado através da descrição da estrutura do SHC, este sistema integra diferentes tecnologias, tais como: SBC e RBC, o que exigiu um cuidado especial no seu desenvolvimento.

Para a implementação do SHC (tanto conhecimentos quanto inferências e camada de comunicação) foi utilizado um processo de desenvolvimento de software específico para esse

¹*Logic Programming Associates* (LPA) é uma *software house* inglesa especializada no desenvolvimento e suporte a compiladores Prolog e softwares relacionados.

tipo de sistema, de modo a formalizar todos os passos e procedimentos necessários para tal, como descrito nas próximas seções.

4.3 Processo de Desenvolvimento do Sistema Híbrido de Conhecimento

No desenvolvimento de software é importante que se tenha uma metodologia que auxilie a sua construção, de modo a se obter um sistema com qualidade. Para isso, existem diversos modelos de ciclo de vida de desenvolvimento de software, no qual cada um se adapta mais a um determinado tipo de software.

O Sistema Híbrido de Conhecimento, assim como os Sistemas Baseados em Conhecimento, é um tipo específico de software. Esse sistema tem como ponto chave a aquisição e manipulação de conhecimento e, portanto, a aquisição desse conhecimento deve ser bem elaborada e realizada, para que se produza um sistema com qualidade e confiabilidade.

Esta seção tratará tanto das principais etapas do desenvolvimento do SHC, quanto do modelo utilizado para tal.

4.3.1 Principais Etapas no Desenvolvimento do Sistema Híbrido de Conhecimento

As principais etapas no desenvolvimento do Sistema Híbrido de Conhecimento foram:

- **Especificação dos Requisitos:** envolveu estudo de viabilidade, desenvolvimento de metas, refinamento do domínio, escolha da equipe de Aquisição de Conhecimento e de projeto, e identificação de fontes de conhecimento;
- **Aquisição de Conhecimento:** foi definido o processo de adquirir, organizar e estruturar o conhecimento, bem como realizada a documentação do sistema a qual envolve, entre outros, a definição de um dicionário de conhecimento;
- **Projeto:** envolveu a definição da estrutura e organização do conhecimento, a definição de métodos para processamento, a seleção de ferramentas adequadas ao desenvolvimento e a documentação;

- **Implementação:** realizou-se a codificação e documentação do sistema;
- **Teste:** envolveu a validação e verificação do sistema e é considerado um processo contínuo, pois há a necessidade de assegurar que o sistema funcione corretamente, forneça resultados verdadeiros (corretos) e satisfaça os requisitos do cliente. Neste caso, os testes se resumiram a dois tipos básicos: testes feitos pelos programadores durante e após a implementação de cada parte dos programas do ambiente e os testes a serem feitos por professores e estudantes (usuários do sistema) após a finalização da implementação (este último ainda não realizado);
- **Manutenção:** é a responsável por eventuais mudanças nos requisitos do sistema, enfatizando a Aquisição de Conhecimento contínua e a avaliação do sistema em andamento.

O desenvolvimento de Sistemas Inteligentes normalmente usam métodos empíricos, e não tem suporte de metodologias que sejam amplamente aceitas. Um dos motivos é a falta de várias características desejáveis de um processo de Engenharia de Software, por exemplo, confiabilidade, estimativa de custos, garantia de qualidade, etc. (Liebowitz & Salvo, 1989). O processo de desenvolvimento do SHC foi realizado utilizando o Modelo Espiral, tratado a seguir.

4.3.2 Modelo Utilizado no Desenvolvimento do Sistema

Este trabalho foi desenvolvido utilizando o Modelo Espiral, que é um modelo utilizado para desenvolvimento de softwares tradicionais com adaptações para desenvolvimento de SBC, e é representado por uma espiral que pode ser refinada em quatro quadrantes, que são: Planejamento, Aquisição de Conhecimento, Codificação e Avaliação do Sistema.

Cada iteração ao redor da espiral (do centro para fora) adiciona alguma nova capacidade funcional ao sistema. O ponto final da espiral não determina de fato o término do sistema, ao invés disso, uma nova espiral é iniciada com a manutenção e a evolução do sistema. Na Figura 2.3, página 14, foi mostrada uma adaptação do modelo espiral proposta por Giarratano para desenvolvimento de SBC (Giarratano & Riley, 1994). Esse modelo foi utilizado para o desenvolvimento do Sistema Híbrido de Conhecimento.

No sistema aqui implementado, tem-se, a seguir, a especificação e o detalhamento do que foi utilizado em cada fase:

1. Planejamento: engloba as etapas de especificação dos requisitos e projeto (descritas anteriormente) do sistema.

Para a determinação da viabilidade do uso de técnicas de IA para auxiliar o processo de aprendizado do ambiente de apoio ao treinamento e ensino foram levantadas e respondidas algumas questões:

- (a) A tarefa exige uso de conhecimento especializado?
- (b) Quão disponível é este conhecimento?
- (c) Quem executa esta tarefa?
- (d) Existe alguma razão para acreditar que a solução algorítmica não daria certo?
- (e) A tarefa envolve julgamentos ou incertezas?
- (f) É aceitável que o sistema nem sempre apresente a solução correta?

Desta forma, a seguir são apresentadas algumas razões pelas quais o ambiente proposto (SIATE) utilizou técnicas e características de Sistemas Inteligentes:

- um sistema que ensina/treina usuários em uma área específica, como a Aquisição de Conhecimento, requer um conhecimento especializado do domínio em questão, para geração de roteiros, fornecimento de respostas, aconselhamento sobre algoritmos de AM, etc.;
- o conhecimento necessário para construir a Base de Conhecimento pode ser encontrado na literatura, porém foi verificado que um especialista no domínio fornece métodos e técnicas mais atualizados e adequados aos tipos de problemas para a apresentação e estruturação desse conhecimento;
- a tarefa ensinada pelo ambiente proposto é normalmente realizada por engenheiros de conhecimento, sendo necessária a utilização de técnicas de ensino e aprendizado utilizadas por seres humanos;
- as inferências realizadas pelo sistema proposto envolvem julgamentos e incertezas e, além disso, é aceitável que o sistema nem sempre apresente a solução ideal ou correta, pois caso isso ocorresse não traria para o usuário nenhum grande prejuízo.

2. Aquisição de Conhecimento: engloba a etapa de AC propriamente dita do sistema. Nessa fase foi utilizado o processo de Aquisição de Conhecimento IPAIA, detalhado na Seção 4.4.
3. Codificação: engloba a etapa de implementação do sistema, que será detalhada na Seção 4.5.
4. Avaliação do SHC: engloba a etapa de teste e validação do sistema, que serão feitos por professores, alunos de IA e engenheiros de conhecimento após toda a codificação do sistema.

No desenvolvimento de um Sistema Inteligente podem ocorrer vários erros, principalmente durante a AC. Dessa forma, um cuidado especial foi tomado com relação a AC para o SHC.

Como a Aquisição de Conhecimento é considerada o estágio mais complexo do desenvolvimento de SI, dispendendo muito tempo e recursos, considerou-se viável a utilização do IPAIA, que é um processo bem definido para Aquisição de Conhecimento Explícito, por possuir uma natureza iterativa para obtenção do conhecimento, entre outras características. O processo IPAIA está descrito a seguir.

4.4 O Processo Utilizado para Aquisição de Conhecimento Explícito

O processo de Aquisição de Conhecimento aqui descrito denomina-se IPAIA (acrônimo para as iniciais de suas fases) e apresenta orientações que proporcionam uma abordagem de Aquisição de Conhecimento Explícito de uma maneira organizada (Jubileu et al., 1998). Esse processo pode ser direcionado tanto para grandes quanto para pequenos projetos de AC e possui em seu desenvolvimento cinco fases:

1. Início do Processo de Aquisição de Conhecimento;
2. Preparação da Sessão de Aquisição de Conhecimento;
3. Aquisição do Conhecimento;

4. Implementação do Conhecimento;

5. Avaliação do Conhecimento.

A fase 1 é realizada apenas uma vez, e as fases 2, 3, 4 e 5 compõem uma etapa contínua de melhoramento do conhecimento, como mostrado na Figura 4.2.

Fase 1 - Início do Processo de Aquisição de Conhecimento: nesta fase foram descritos o domínio de conhecimento, os termos chaves e as referências. Também foi identificado um resumo simplificado dos conceitos relacionados ao domínio de conhecimento, para que as pessoas participantes do processo de Aquisição de Conhecimento possam compreendê-lo melhor. Foi desenvolvido um mapa mental do domínio de conhecimento, realizada a análise funcional na qual módulos distintos, entradas e saídas atuais são identificados, selecionada a equipe de AC e escolhida a estrutura de pré-representação do conhecimento dentro do domínio específico. Os passos que fazem parte desta fase podem ser resumidos em: selecionar o domínio de conhecimento; analisar o problema; selecionar a equipe de Aquisição de Conhecimento e selecionar a estrutura de pré-representação do conhecimento.

Figura 4.2: Processo para Aquisição de Conhecimento Explícito - IPAIA.

Fase 2 - Preparação da Sessão de Aquisição de Conhecimento: nesta fase foram planejadas as sessões de Aquisição de Conhecimento, selecionadas as técnicas mais adequadas à aplicação e identificados os membros da equipe de AC envolvidos nessas sessões. De forma geral, esta fase contém os seguintes passos: selecionar as técnicas para a Aquisição de Conhecimento, considerando o tamanho e meta do programa de Aquisição de Conhecimento, a disponibilidade de especialista(s) do domínio, a quantidade e experiência da equipe de AC e o tipo de conhecimento; identificar membros da equipe de AC envolvidos na Aquisição de Conhecimento e planejar sessões de Aquisição de Conhecimento.

Fase 3 - Aquisição do Conhecimento: nesta fase adquiriu-se os conhecimentos referentes ao domínio, que foram armazenados na Base de Conhecimento, ou seja, executou-se o que foi planejado na fase anterior.

Fase 4 - Implementação do Conhecimento: nesta fase, o conhecimento adquirido foi representado formalmente. Para isso, utilizou-se a estrutura de representação do conhecimento selecionada na Fase 1 deste processo de Aquisição de Conhecimento.

Fase 5 - Avaliação do Conhecimento: nesta fase foi realizada a verificação e validação do conhecimento a ser refinado, averiguando se o conhecimento existente e as funcionalidades implementadas são suficientes, bem como a necessidade de um novo ciclo.

Primeiramente, para a Aquisição de Conhecimento referente a este protótipo do SI-ATE foi realizada a Fase 1, na qual foi selecionado que o domínio de aplicação seria a Aquisição de Conhecimento voltado ao treinamento de alunos da disciplina de Inteligência Artificial e de Engenheiros de Conhecimento. Esta fase inicial foi realizada apenas uma vez. Diferentemente da Fase 1, as fases seguintes — Fase 2, Fase 3, Fase 4 e Fase 5 — foram realizadas várias vezes, pois a realização do processo de AC envolveu a execução de diferentes sessões de AC, através da utilização de diferentes técnicas e fontes de conhecimento. Algumas técnicas de ACE utilizadas foram: entrevistas estruturadas e desestruturadas, estudo de casos, *brainstorming*, e rastreamento de processo (explicadas na Seção 2.3.2).

4.5 Considerações Gerais sobre a Implementação do Sistema Híbrido de Conhecimento

A implementação do SHC foi toda realizada utilizando a linguagem de programação lógica Prolog. Informalmente, Prolog é uma linguagem na qual os programas são descritos como regras (cláusulas) usadas para provar relações entre objetos, sendo que um programa consiste em um conjunto dessas regras. Essa linguagem foi escolhida por ser adequada à implementação de protótipos e sistemas que utilizam conhecimento, bem como a manipulação do mesmo. Para tanto, foi utilizado o software LPA WIN-Prolog versão 3.6 e as ferramentas Flex versão 1.2 e ProWeb versão 1.0 que o acompanham. O Flex é uma ferramenta projetada para a construção de Sistemas Baseados em Conhecimento, com suporte à programação baseada em regras de produção e, além disso, possui uma linguagem própria, a *Knowledge Specification Language* (KSL), que é baseada na língua inglesa e torna o programa mais legível, sendo fácil de ser entendido até mesmo por não programadores. O ProWeb é uma ferramenta que, basicamente, fornece um *link* entre uma página HTML mostrada no *browser* do cliente e sua aplicação (desenvolvida em LPA-Prolog ou ferramentas associadas) rodando no servidor.

As funções executadas pelo SHC dão apoio, principalmente, a dois módulos do SIATE: Módulo Roteiros (Capítulo 5) e Módulo Prática/Treinamento (Capítulo 6), dentre os quatro módulos existentes (conforme mostrado na Figura 3.2, página 54).

Uma forma de apoio ao SIATE está relacionada à geração de roteiros dinâmicos e a escrita dos mesmos na Base de Roteiros do ambiente, de modo a ser possível sua visualização por intermédio do hiperdocumento existente no ambiente. Uma outra opção é quanto ao apoio dado à Ferramenta para Treinamento visando um suporte e auxílio ao aprendizado do treinando, sendo essa ajuda realizada na forma de perguntas e respostas feitas ao usuário.

De acordo com a Figura 4.1, página 67, ao Módulo Roteiros estão relacionados o MCD, o MSR e a Base de Modelagem do Domínio e ao Módulo Prática/Treinamento o ME, o MCD, a Base de Conhecimento Geral e a Base de Casos. Além disso, tanto o Motor de Inferência quanto a Memória de Trabalho são compartilhados pelos dois módulos.

4.6 Considerações Finais

A construção de um Sistema Inteligente utilizando várias tecnologias produz bons resultados na medida em que se beneficia das vantagens pertencentes a cada uma delas, bem como de todas as facilidades e viabilidades geradas por essa interação.

No SHC utilizou-se a junção de diferentes tecnologias visando com isso construir um módulo para um ambiente voltado ao treinamento e ensino com características que auxiliem o estudante dando-lhe um *feedback* a respeito de suas dúvidas e questões.

O capítulo seguinte tratará com mais detalhes do Módulo Roteiros do SIATE. Para tanto, será mostrada a modelagem do domínio que foi utilizada para a representação do conhecimento envolvido na geração de roteiros, bem como o Gerador de Roteiros implementado.

Capítulo 5

Módulo Roteiros

5.1 Considerações Iniciais

O Módulo Roteiros é responsável, no SIATE, tanto pela seleção de um roteiro predefinido, quanto pela geração dinâmica de um roteiro em particular, como foi mostrado na Figura 3.6, página 59.

A idéia do desenvolvimento de um Gerador de Roteiros surgiu da necessidade de criar roteiros específicos para um determinado treinando, orientando-o na navegação pelo Material Didático. Neste caso, faz-se necessária a modelagem do domínio de aplicação, para que o sistema possa gerar um novo roteiro de acordo com as solicitações/necessidades do treinando.

Este capítulo trata da parte de geração de roteiros do Módulo Roteiros do SIATE. Esta geração é realizada de forma dinâmica pelo SHC, considerando-se algumas respostas do treinando (através de uma interação com o SHC), como também o domínio de conhecimento que está sendo ensinado e todo o conhecimento armazenado no sistema. Será descrita também a elaboração da modelagem do domínio utilizada para a geração de roteiros, assim como a maneira pela qual foi implementado o módulo gerador desses roteiros.

5.2 Modelagem do Domínio para Geração de Roteiros

O hiperdocumento referente ao Material Didático, descrito na Seção 3.3.2, página 56, foi construído agregando uma parte do conhecimento do domínio da aplicação, a ACE, com

propriedades e características do SIATE. Após isso, o mesmo foi modelado utilizando o modelo semântico *Generic Semantic Model* (Hull & King, 1987), que será descrito a seguir.

5.2.1 Considerações Gerais sobre Modelagem Semântica

Em termos gerais, complementos de modelagem semântica trabalham em representação de conhecimento e na nova geração de modelos de dados baseados no paradigma de programação orientada a objeto. Além disso, possuem a vantagem de fornecerem suporte ao projeto e evolução de bases de dados.

Modelos semânticos foram introduzidos primeiramente como esboços de ferramentas de projeto — um esboço pode primeiro ser projetado num modelo semântico de alto nível e depois traduzido para um dos modelos tradicionais para a implementação final. A ênfase dos modelos semânticos iniciais era em modelar corretamente relacionamentos de dados que surgiam freqüentemente em aplicações típicas de base de dados. Os modelos semânticos são mais complexos do que os modelos relacionais e encoraja uma visão mais navegacional dos relacionamentos dos dados (Hull & King, 1987).

Similarmente, os modelos semânticos tentam proporcionar abstrações mais poderosas de especificações de esboços de dados do que as suportadas por modelos relacionais, hierárquicos e de rede propiciando um nível mais alto de abstração para modelagem de dados.

Aqui serão apresentados detalhes de um modelo semântico em particular, o *Generic Semantic Model*, que está baseado em três dos mais proeminentes modelos encontrados na literatura: Modelo Entidade-Relacionamento (MER), Modelo Funcional de Dados (FDM – *Functional Data Model*), e Modelo Semântico de Dados (SDM – *Semantic Data Model*), sendo mais semelhante à do FDM.

O GSM é representativo de uma grande classe de modelos semânticos; como resultado de ser um pouco eclético, ele enevoa uma importante distinção filosófica crescente na literatura de modelagem semântica.

Os componentes primários do GSM, também utilizados em outros modelos semânticos, são as representações explícitas de objetos, atributos e relacionamentos entre objetos, construtores para tipos complexos, relacionamentos do tipo IS-A (é-um), e componentes derivados do esboço:

- objeto fundamental ou entidade: são representados por nós triangulares, indicando que eles correspondem a tipos abstratos de dados;
- relacionamento subtipo ou supertipo: também chamados de relacionamento IS-A. Os subtipos são indicados por nós circulares (indicando que o tipo subordinado é dado em outro lugar do esboço), junto com flechas duplas indicando relacionamentos IS-A. Note que, na ausência de qualquer restrição, os conjuntos correspondentes a esses subtipos podem se sobrepor;
- tipos construtores: que em um modelo semântico podem ser construídos a partir de tipos atômicos. Um exemplo de tipo construído é a agregação (isto é, produto cartesiano) de *printable types*, que é representada em um esboço por um nó circular com um $\times$ em seu interior, e que possui “filhos” que correspondem às coordenadas da agregação. A agregação é uma forma de abstração oferecida pela maioria dos modelos de dados semânticos. Um segundo tipo construtor é o agrupamento ou associação que é usado para construir conjuntos de elementos de um tipo existente. O agrupamento é representado por um nó circular com um $*$ em seu interior e é usado para formar conjuntos de elementos.

Como exposto anteriormente, os tipos objeto podem ser modelados em um esboço semântico como sendo abstratos, *printable* ou construído e podem ser definidos utilizando um relacionamento IS-A. Por meio dessa flexibilidade, o projetista do esboço pode escolher o construtor apropriado ao significado do tipo objeto no ambiente de aplicação particular.

Até agora, o foco foi em como os tipos objeto e subtipos podem ser representados em modelos de dados semânticos. Um outro componente fundamental da maioria dos modelos semânticos consiste em mecanismos para representação de atributos (isto é, funções) associados a esses tipos e subtipos. Deve ser observado que, diferentemente das funções tipicamente encontradas em linguagens de programação, vários atributos surgidos em esboços de dados semânticos não são computáveis, e eles são especificados explicitamente pelo usuário para corresponder a fatos do mundo real. Os atributos são representados utilizando flechas simples iniciando no domínio do atributo. Os atributos podem ser tanto monovalorados (representados por uma flecha com uma única seta na ponta) ou multivalorados (representado por uma flecha com ponta dupla). Os atributos do nó pai são herdados pelos nós filhos.

Essencialmente, modelos semânticos encapsulam aspectos estruturais dos objetos, ao passo que linguagens orientadas a objetos encapsulam aspectos comportamentais dos objetos.

Existem duas distinções centrais entre a abordagem orientada a objetos e a de modelos semânticos. Primeiro, modelos orientados a objeto tipicamente não personificam os tipos de construtores ricos dos modelos semânticos. Do ponto de vista estrutural, modelos orientados a objeto suportam apenas a habilidade de definir atributos simples e multivalorados. Segundo, a herança dos modelos é estritamente diferente da herança dos atributos (como em modelos semânticos). Em um modelo semântico, a herança de atributos é apenas entre tipos, os quais um é subconjunto do outro. A herança de um método, visto que ele é uma propriedade comportamental — e não estrutural — pode ocorrer entre tipos aparentemente distintos.

Modelos e sistemas de dados semânticos proporcionam as seguintes vantagens sobre os tradicionais sistemas orientados a registros:

1. separação ampliada de componentes conceituais e físicos;
2. diminuição de sobrecarga semântica de tipos relacionados;
3. disponibilidade de mecanismos de abstração convenientes.

A escolha da ênfase — baseada em atributo ou baseada em tipos de construtores — afeta os mecanismos de linguagem que parecem naturais para a manipulação de dados semânticos.

O modelo semântico GSM utiliza construtores locais para representar a estrutura dos dados. Para isso, serão divididos em três partes, que focalizam os tipos, atributos e relacionamentos IS-A:

- tipos atômicos e construtores: esse modelo provê mecanismos para representar tipos de objetos construtores atômicos e não atômicos. Os dois mais prominentes construtores de tipos são: agregação e agrupamento;
- atributos: um atributo de um argumento em um esboço GSM é um relacionamento binário direto entre dois tipos (retratado por uma seta), e um atributo n -argumentos é um relacionamento direto entre um conjunto de n tipos e um tipo (retratado por uma seta com n caudas). Os atributos podem ser monovalorados, retratado por uma seta

com um ponteiro em sua cabeça, ou multivalorado, retratado usando uma seta com dois ponteiros em sua cabeça.

No GSM, não existem restrições quanto aos tipos da origem ou do destino de um atributo.

Um outro tipo de atributo é o atributo tipo, que associa um valor com um tipo completo, ao invés de associar um valor com cada objeto no domínio ativo;

- relacionamentos IS-A: fornece a habilidade de representar relacionamentos IS-A ou supertipo/subtipo. Intuitivamente, um relacionamento IS-A de um tipo **sub** para um tipo **super** indica que cada objeto associado com o **sub** está associado com o tipo **super**.

Na maioria dos modelos semânticos cada atributo definido no tipo **super** está automaticamente definido no **sub**; ou seja, atributos de **super** são herdados por **sub**. Geralmente, também é verdade que um subtipo pode ter atributos não compartilhados pelo tipo pai.

Todos esses conceitos foram utilizados na modelagem do domínio descrita a seguir.

5.2.2 Modelagem do Domínio Utilizando o Generic Semantic Model

Como já foi exposto, o domínio de Aquisição de Conhecimento que está contido no Material Didático do SIATE foi modelado utilizando o modelo semântico GSM. Por se tratar de uma modelagem semântica, considerou-se o relacionamento existente entre os conceitos tratados no Material Didático e não as ligações existentes entre as páginas do hiperdocumento. Como se pode observar na Figura 5.1, a modelagem está concentrada no supertipo AC, por ser esse o domínio de aplicação do ambiente. Pode-se observar que todas as outras classes do modelo estão ligadas a ele. Vale lembrar que todas as informações do Material Didático, tais como domínio do conhecimento, arquitetura do SIATE, componentes, etc., são também representadas nessa figura.

Todas as pessoas que participam, de uma maneira ou de outra, do processo de Aquisição de Conhecimento foram agrupadas em uma única classe, por possuírem características em comum.

Figura 5.1: Uma Visão da Modelagem do Domínio.

Na próxima seção é apresentada como essa modelagem do conhecimento utilizando o GSM foi implementada a fim de gerar roteiros dinâmicos.

5.3 Gerador de Roteiros

Como já mencionado, o Gerador de Roteiros foi desenvolvido considerando-se a necessidade de se construir roteiros específicos e particulares a um determinado treinando, visando suprir seus interesses, bem como suas dificuldades. Na Figura 5.2, a parte hachurada em destaque mostra os módulos do SHC acionados para a geração dinâmica de um roteiro.

Quando é solicitado um roteiro específico ao Módulo de Roteiros do SIATE, o Motor

Figura 5.2: Componentes do Sistema Híbrido de Conhecimento Utilizados pelo Gerador de Roteiros.

de Inferência do SHC, utilizando o conhecimento contido na base Modelagem do Domínio, interage com o treinamento através do Módulo Coletor de Dados, fazendo perguntas e coletando respostas e gera um roteiro que é gravado na base de Roteiros pelo Módulo de Saída de Roteiros.

O Gerador de Roteiros foi implementado utilizando a linguagem de programação lógica Prolog. Para tanto, foi utilizado o software LPA WIN-Prolog versão 3.6. Esse Gerador de Roteiros manipula uma rede semântica estendida, baseada na modelagem GSM do domínio, bem como as informações sobre as páginas do Material Didático mostradas na Tabela 5.1. Como já mencionado, o domínio para esse protótipo é a Aquisição de Conhecimento, e sua modelagem foi mostrada na Figura 5.1.

Nome do Arquivo	Descrição	Grupo
AC.html	Visão geral sobre Aquisição de Conhecimento	ac
ac_explicita.html	Aquisição de Conhecimento Explícito	ace
ac_implicita.html	Aquisição de Conhecimento Implícito	aci
administracao_AC.html	Administração da AC	ac

continua na próxima página

continuação da página anterior

Nome do Arquivo	Descrição	Grupo
analise_prot.html	Análise de Protocolos	ace
aprendiz_maq.html	Aprendizado de Máquina	aci
arq_geral.html	Arquitetura Geral do SIATE	siate
bibliografia.html	Bibliografia Utilizada	siate
brainstorming.html	Brainstorming	ace
camada1.html	Primeira Camada da Arquitetura SIATE	siate
camada2.html	Segunda Camada da Arquitetura SIATE	siate
camada3.html	Terceira Camada da Arquitetura SIATE	siate
componentes.html	Módulos e Componentes do Ambiente SIATE	siate
descr_ferramenta.html	Descrição da Ferramenta para Treinamento	siate
descr_roteiros.html	Descrição do Módulo Roteiros	siate
dominio_SIATE.html	Descrição sobre o Domínio do Ambiente SIATE	siate
engenharia.html	Engenharia de Conhecimento	ac
engenheiro.html	Engenheiro de Conhecimento	pessoa
entrevista.html	Entrevista	ace
esc_especial.html	Escolha do Especialista	pessoa
estudo_casos.html	Estudo de Casos	ace
extra_CBD.html	Extração de Conhecimento de Bases de Dados	aci
glossario.html	Glossário de Termos Técnicos	siate
index.html	Tela Principal do Módulo Material Didático	siate
inducacao.html	Indução de Regras em ACI	aci
inf_tecnicas.html	Informações Técnicas	siate
interface.html	Interface do SIATE	siate
introducao.html	Introdução ao Módulo Material Didático	ace
introspeccao.html	Introspecção	siate
mat_didatico.html	Descrição do Módulo Material Didático	siate
modulo_tutor.html	Descrição do Sistema Tutor	siate
objetivos_princ.html	Principais Objetivos do Módulo Material Didático	siate
prat_treina.html	Descrição do Módulo Prática/Treinamento	siate
problemas.AC.html	Problemas Considerados na AC	ac

continua na próxima página

continuação da página anterior

Nome do Arquivo	Descrição	Grupo
rastreamento.html	Rastreamento de Processos	ace
repert_grid.html	Repertório Grid	ace
SBC.html	Sistema Baseado em Conhecimento	ac
shc.html	Descrição do Sistema Híbrido de Conhecimento	siate
SIATE_descr.html	Descrição do Ambiente SIATE	siate
tare_fases_AC.html	Tarefas e Fases da AC	ac

Tabela 5.1: Características e Grupos das Páginas do Material Didático do SIATE.

O conhecimento referente a modelagem do domínio na Base de Conhecimento é armazenado através de relações e estruturas Prolog, que foram extraídas da modelagem GSM do domínio.

Todos os relacionamentos existentes entre os objetos pertencentes à modelagem são especificados de forma a possibilitar “caminhar” pela modelagem, ou seja, utilizar e manipular todo o conhecimento que essas ligações oferecem. A implementação dos relacionamentos foi realizada de uma forma genérica, de modo a facilitar possíveis complementos, bem como a utilização da modelagem para um outro domínio. Um exemplo de relacionamentos é mostrado a seguir.

```
relacionamento(is_a,ac,ace).  
relacionamento(is_a,ac,aci).
```

A partir da modelagem apresentada na Figura 5.1 e das páginas WWW contidas no hiperdocumento referente ao Material Didático (como mostra a Tabela 5.1) foram construídos os grupos. Esses grupos foram mapeados como uma estrutura contendo características referentes a ele, tais como: abreviação do nome do grupo, *string* com o nome completo do grupo e uma lista com as páginas do hiperdocumento associadas a ele, na qual cada elemento é uma outra lista contendo o nome do arquivo HTML, bem como a informação sobre a obrigatoriedade ou não dessa página em um roteiro simplificado, sendo que *op* significa página opcional e *ob* significa página obrigatória na geração de um roteiro básico.

A seguir pode ser visualizado o exemplo de um grupo.


```
grupo(ac, 'Aquisicao de Conhecimento',
      [['AC.html', ob], ['administracao_AC.html', op],
       ['engenharia.html', op], ['problemas_AC.html', op],
       ['SBC.html', op], ['tare_fases_AC', op]]).
```

Uma outra estrutura que também faz parte da Base da Modelagem do Domínio são as perguntas, que possuem dois argumentos: a página do hiperdocumento e o texto da pergunta relacionada a essa página. Esse tipo de relação exige uma interação com o meio externo, que pode ser o treinando, usuário, estudante, etc. O MCD é o responsável por essa interação, ou seja, por fazer uma pergunta e coletar a resposta fornecida. As perguntas só serão feitas caso a informação necessária não possa ser inferida baseando-se no restante do conhecimento que o sistema possui.

Exemplos de perguntas podem vistos a seguir.

```
pergunta('problemas_AC.html',
         'Voce gostaria de conhecer os principais problemas de Aquisicao de
         Conhecimento?').
pergunta('administracao_AC.html',
         'Voce gostaria de obter detalhes sobre a administracao da Aquisicao
         de Conhecimento?').
```

O Módulo Coletor de Dados (visto na Figura 4.1, página 67) implementado está encarregado de formular perguntas ao usuário e armazenar as respostas na Memória de Trabalho, sendo essas respostas usadas pelo Motor de Inferência do SHC para chegar a conclusões quanto a um roteiro apropriado para a situação. Quando o MI do SHC tenta provar uma meta que não pode ser deduzida da Base de Conhecimento ou das perguntas já respondidas pelo usuário, ele ativa o MCD fornecendo o que deve ser perguntado.

Os tipos de perguntas implementadas são: afirmativas e negativas (perguntas que necessitam de resposta sim ou não) e várias opções (possibilitando a escolha de uma ou mais opções).

O Gerador de Roteiros começa perguntando ao usuário sobre os tópicos que são de seu interesse, por motivo de dúvidas, estudo, curiosidade, etc. A partir das primeiras respostas, caso seja necessário, o sistema faz ao usuário perguntas mais específicas, para que possa ser delimitado o caminho que será utilizado no roteiro que está sendo gerado.

A parte do Motor de Inferência do SHC referente ao Gerador de Roteiros foi implementada utilizando regras de produção e raciocínio *Backward Chaining*. Ele possui um programa que manipula toda a modelagem semântica apresentada na Figura 5.1, considerando também as respostas fornecidas pelo usuário via MCD.

Toda a inferência realizada conta com o auxílio da Memória de Trabalho, que armazena as respostas fornecidas pelo treinando na geração de um roteiro em particular.

A partir da cláusula apresentada a seguir é acionada a geração de roteiros.

A geração de roteiros é acionada pelo Motor de Inferência que, a partir de interações (perguntas e respostas) com o usuário, seleciona a princípio o grupo do domínio que o usuário deseja percorrer. Feito isso, são obtidas as páginas do hiperdocumento pertencente àquele grupo, bem como todas as páginas pertencentes a subgrupos relacionados ao grupo inicial. Para isso, é verificada toda a hierarquia abaixo do grupo em questão. Por fim, todas as páginas são concatenadas em uma única lista, gerando assim o roteiro dinâmico que será exibido ao usuário que o requisitou.

Após a geração do roteiro dinâmico, feita pelo MI através de inferências, o mesmo é enviado ao Módulo de Saída de Roteiros, que armazena o novo roteiro gerado na Base de Roteiros pertencentes ao ambiente SIATE, de modo que seja utilizado pelo treinando que o solicitou e que também possa vir a ser utilizado posteriormente por um outro usuário.

A estrutura armazenada na Base de Roteiros contém as páginas do hiperdocumento pertencentes ao roteiro em questão, linearmente ordenadas. A apresentação desse roteiro é a mesma utilizada com os roteiros predefinidos mostrada na Seção 3.3.3, página 58.

5.4 Considerações Finais

A geração de roteiros dinâmicos se torna importante pelo fato de proporcionar uma orientação personalizada a um determinado treinando.

Para que isso fosse possível foi implementado um sistema que, a partir da manipulação de uma rede semântica estendida, baseada na modelagem do domínio e da interação com o treinando, fornecesse a ele um roteiro que o auxiliasse e suprisse suas dúvidas.

No próximo capítulo será apresentado o Módulo Prática/Treinamento pertencente ao SIATE, bem como a biblioteca *MCC++* utilizada como ferramenta para treinamento do ambiente.

Capítulo 6

Módulo Prática/Treinamento

6.1 Considerações Iniciais

Ambientes de treinamento baseados em simulação computacional são largamente utilizados hoje em dia. Implementações de sucesso podem ser conferidas em (Schank, 1994a; Schank, 1994b). Nesse tipo de ambiente, um pressuposto comumente assumido por seus autores é o de que o treinando possui um razoável conhecimento sobre o modelo subjacente e, portanto, é capaz de analisar e aprender a partir dos resultados da simulação. Além disso, é comum que os softwares de simulação não ofereçam quaisquer outros recursos que não sejam aqueles estritamente relacionados com a tarefa/atividade em questão. A realidade pode, no entanto, ser diferente. Treinandos possuem diferentes graus de conhecimento prévio e possuem métodos idiossincráticos de aprendizagem. Enquanto a simulação se constitui, sem dúvida, num recurso positivo em ambiente de treinamento, a suficiência desse recurso ainda é questionável.

Parece bastante razoável acreditar que outros subsídios podem ser adicionados a um ambiente de simulação a fim de se acelerar o processo de aprendizagem e treinamento. Nesse sentido, tais recursos compõem um subsistema de raciocínio qualitativo, nos moldes de QMaPS (*Qualitative Reasoning for Simulation Learning Environments*) (Joolingen, 1994). Dentre os recursos adicionais vantajosos, pode-se citar:

- (a) a capacidade do ambiente oferecer informações adicionais sobre o modelo e os conceitos relacionados a ele, através de respostas a perguntas do treinando;
- (b) a capacidade do sistema mostrar situações análogas àquela experimentada pelo

treinando, a fim de acelerar o processo de análise dos resultados obtidos;

- (c) a capacidade do sistema tratar (responder, diagnosticar, explicar) questões relacionadas ao domínio, porém não tratadas diretamente pelo simulador.

Outro ponto importante, nesta última década, uma imensa quantidade de dados tem sido gerada pelas organizações, as quais estão perdendo competitividade por se acharem impossibilitadas de entender, interpretar e extrapolar esses dados. Métodos de aprendizado de máquina, métodos estatísticos e métodos de reconhecimento de padrões provêm algoritmos para extrair conhecimento de dados ajudando a analisar as informações, encontrando padrões e melhorando a precisão de uma predição. Um problema que os usuários e analistas enfrentam quando tentam descobrir padrões é que existem muitos algoritmos disponíveis e é difícil determinar qual utilizar em cada situação. A *MLC++* está sustentada, principalmente, no fato de não existir um único algoritmo de Aprendizado de Máquina que possa fornecer ótimos resultados de precisão para todas as tarefas. Isto foi teoricamente provado e demonstrado experimentalmente (Kohavi et al., 1997). Devido a este fato, recomenda-se testar vários algoritmos com o mesmo conjunto de dados, para comparar e determinar qual apresenta o melhor desempenho para cada caso.

O Módulo Prática/Treinamento do SIATE utiliza raciocínio qualitativo, através do SHC, bem como raciocínio quantitativo, através da simulação usando a *MLC++*.

Neste capítulo serão vistas algumas características da biblioteca *MLC++*, a descrição de dois de seus algoritmos (CN2 e C4.5) e aspectos da simulação usando essa biblioteca.

6.2 A Biblioteca *MLC++*

Os algoritmos de Aprendizado de Máquina apresentados neste trabalho estão contidos na *MLC++* (versão 2.1), que é uma biblioteca de classes em *C++* de Algoritmos de Aprendizado de Máquina. O projeto da *MLC++* começou em 1993, na Universidade de Stanford, e trata-se de um software de domínio público (inclusive os códigos fontes), sendo que desde 1995 encontra-se sob a responsabilidade da Silicon Graphics. A *MLC++* encontra-se disponível em <http://www.sgi.com/Technology/mlc/index.html>.

Essa biblioteca foi projetada para ajudar na seleção de algoritmos e/ou no desenvolvimento de novos algoritmos de Aprendizado de Máquina. Dessa forma, *MLC++* tem um

papel dual: serve tanto para um sistema com usuários finais quanto para desenvolvedores de algoritmos, e trabalha com mais de 30 algoritmos, alguns implementados internamente e outros externamente. Além de ser uma biblioteca de classes em *C++*, provê utilitários e ferramentas que podem ser usadas isoladamente, sendo o benefício principal do seu uso, a habilidade para reunir classes (que são objetos *C++*) e as instâncias para criar novos algoritmos ou variantes de algoritmos existentes. Uma das vantagens da biblioteca *MCC++* é a transformação automática do formato dos dados de entrada para os diversos algoritmos de Aprendizado de Máquina.

Considerando-se as diferentes funcionalidades (classes) da *MCC++* ela pode ser dividida em quatro tipos: classes de suporte geral, classes essenciais, categorizadores e algoritmos de indução, que são descritos a seguir:

- **Classes de Suporte Geral:** estas classes provêem apoio para operações gerais, não específicas aos algoritmos de Aprendizado de Máquina, suportando classes para matrizes, listas encadeadas, cadeias de caracteres, números aleatórios e gráficos.
- **Classes Essenciais:** estas são as ferramentas básicas que são compartilhadas por muitos algoritmos de Aprendizado de Máquina supervisionado. Elas estão divididas de acordo com as suas funcionalidades: Input/Output, Geração e Conversões, e Wrappers.
- **Categorizadores:** os categorizadores são utilizados para prever a categoria de uma instância (na literatura são comumente chamados de classificadores, mas na *MCC++* usa-se esse termo para evitar confusão entre classificador, classes e classes *C++*). Eles são as estruturas básicas geradas pelos algoritmos de indução.
- **Algoritmos de Indução:** são os algoritmos de Aprendizado de Máquina (chamados pelos autores da biblioteca de indutores) capazes de gerar regras de produção ou árvores de decisão a partir de níveis discretos da classe do conjunto de dados. Em termos da *MCC++* um indutor cria um categorizador.

Na Figura 6.1 é ilustrada a interação que existe entre indutores, categorizadores, a etapa de teste e wrappers. A parte superior ilustra a indução e subseqüentes testes de um categorizador. A parte inferior mostra um wrapper *cross-validation* que gera os *k-folds* e calcula a média da estimativa de precisão do indutor para um conjunto de dados. A

separação do indutor e do categorizador representa um papel importante na decomposição global da biblioteca $MCC++$ para construção de vários categorizadores.

Figura 6.1: Interação entre indutor, categorizador, a etapa de teste e wrapper.

Os usuários da biblioteca $MCC++$ devem conhecer o funcionamento básico dos indutores, uma vez que, como já citado, não existe um indutor ótimo para todos os conjuntos de dados. Existem indutores que têm um ótimo desempenho para alguns conjuntos de dados, mas para outros, apresentam um desempenho fraco. Isto depende de vários fatores, tais como: tamanho do arquivo de dados, número de registros, número de *features*, quantidade de valores contínuos e discretos, etc. (Kohavi et al., 1997).

Alguns dos indutores implementados dentro da $MCC++$ são: Const, Decision Tables, ID3, Lazy Decision Trees, Nearest-neighbor, Naive-bayes, IR, OODG (*Oblivious read-Once Decision Graph*), Option Decision Trees, Perceptron, Winnow, etc. Existem ainda outros indutores, os externos, que fazem interface com a $MCC++$ tais como: C4.5, C4.5-rules, C4.5-unpruned, CART, CN2, IB, OC1, PEBLS (*Parallel Exemplar-Based Learning System*), Ripper, T2, etc.

Classificar todos estes indutores não é uma tarefa fácil, pois existe uma infinidade de métodos para agrupá-los. O método de classificação dos indutores a ser escolhido parte da forma como os indutores aprendem, para depois ser dividido de acordo com seu paradigma de Aprendizado de Máquina e em seguida dividí-los de acordo com a linguagem utilizada para

descrever o aprendizado e finalmente classificá-los pelo modo em que podem ser integrados novos exemplos. Uma síntese desses métodos pode ser vista na Figura 6.2.

DIMENSÕES PARA CLASSIFICAÇÃO DE ALGORITMOS			
TIPO DE APRENDIZADO	PARADIGMAS DE APRENDIZADO	LINGUAGENS DE DESCRIÇÃO	MODO EM QUE NOVOS EXEMPLOS SÃO INTEGRADOS
<ul style="list-style-type: none"> ♦ SUPERVISIONADO ♦ NÃO SUPERVISIONADO 	<ul style="list-style-type: none"> ♦ SIMBÓLICO: <ul style="list-style-type: none"> • Proposicional • Relacional ♦ ESTATÍSTICO ♦ INSTANCE-BASED ♦ CONEXIONISTA ♦ GENÉTICO 	<ul style="list-style-type: none"> ♦ DE EXEMPLOS OU INSTÂNCIAS ♦ DE HIPÓTESES OU CONCEITOS APRENDIDOS <ul style="list-style-type: none"> • Regras de Produção • Árvores de Decisão • Redes Semânticas ♦ DE TEORIA DE DOMÍNIO OU CONHECIMENTO DE FUNDO 	<ul style="list-style-type: none"> ♦ INCREMENTAL ♦ NÃO INCREMENTAL

Figura 6.2: Classificação dos Algoritmos de Aprendizado de Máquina (Félix et al., 1998).

A forma básica de classificar os algoritmos de Aprendizado de Máquina, a partir da forma como eles aprendem, pode ser realizada utilizando um aprendizado supervisionado, ou um aprendizado não supervisionado. O aprendizado supervisionado pode ser representado como um processo sintetizado, ou seja, generalizando a partir de casos conhecidos, de tal forma que, dado um conjunto de observações no qual as classes são conhecidas, o objetivo está em encontrar regras capazes de classificar novas observações entre as classes já existentes. Já o aprendizado não supervisionado pode ser visto como um processo analítico, ou seja, expandindo alguma descrição implícita, de tal forma que, dado um conjunto de observações, o objetivo é tentar estabelecer a existência de classes ou clusters nos dados.

Com relação aos paradigmas de Aprendizado de Máquina, estes podem ser classificados em:

- Paradigma Simbólico: busca aprender construindo representações simbólicas de um conceito através da análise de exemplos e contra-exemplos desse conceito. Esses métodos de aprendizado simbólico podem ser tratados como: proposicionais, que é

um dos mais simples formalismos lógicos existentes e, lidam apenas com enunciados declarativos, chamados proposições (ou sentenças); e relacionais, que podem estar representados pela Programação Lógica Indutiva (PLI) onde se constroem programas lógicos — restritos às cláusulas de Horn — a partir de exemplos positivos e negativos da relação a ser aprendida, usando no aprendizado, informação fornecida pela teoria de domínio.

- Paradigma Estatístico: as técnicas estatísticas, em geral, tendem a focar tarefas em que todos os atributos têm valores contínuos ou ordinais. Muitos deles também são paramétricos, assumindo alguma forma de modelo e então encontrando valores apropriados para os parâmetros do modelo a partir de dados.
- Paradigma Instance-based: uma forma de classificar um caso é lembrar de um caso similar cuja classe é conhecida e assumir que o novo caso terá a mesma classe.
- Paradigma Conexionista: as redes neurais são construções matemáticas relativamente simples que utilizam o mecanismo de paralelismo, onde são conectados um grande número de pequenas unidades de processamento ligadas em rede.
- Paradigma Genético: este formalismo de classificação é derivado do modelo evolucionário de aprendizado. Um classificador genético consiste de uma população de elementos de classificação que competem para fazer uma predição.

Em um sistema de aprendizado é necessário utilizar linguagens para representar exemplos ou instâncias, conceitos induzidos e, dependendo do caso, a teoria de domínio previamente conhecida. Estas linguagens são comumente referenciadas como linguagens de descrição, e estão divididas por:

- Linguagem de descrição de exemplos ou instâncias: descreve os exemplos utilizados pelo programa para aprender conceitos, estabelecendo limites sobre os tipos de padrões que o sistema pode aprender, como, por exemplo, atributo-valor, etc.
- Linguagem de descrição de hipóteses ou conceitos aprendidos: descreve o estado interno de um programa de aprendizado, correspondente a teoria dos conceitos ou padrões que existem nos dados, estabelecendo limites sobre o que pode ou não pode ser aprendido.

Entre essas linguagens se encontram: árvore de decisão, regras de produção, redes semânticas, etc.

- Linguagem de descrição da teoria de domínio ou conhecimento de fundo: descreve todo o conhecimento prévio que o programa possui a respeito do domínio.

A Tabela 6.1 apresenta a classificação de alguns dos algoritmos utilizados pela biblioteca *MLC++* de acordo com a classificação mostrada anteriormente e resumida na Figura 6.2.

Indutor	Tipo	Paradigma	Linguagem de Descr. Hipótese
aha-IB	Supervisionado	Instance-based	Exemplos
C4.5	Supervisionado	Proposicional	Árvore de decisão
C4.5rules	Supervisionado	Proposicional	De árvore p/ regras
C4.5unpruned	Supervisionado	Proposicional	Árvore de decisão
CART	Supervisionado	Estatístico	Árvore de decisão
CN2	Supervisionado	Proposicional	Regras de produção
Decision Tables	Supervisionado	Instance-based	Exemplos
ID3	Supervisionado	Proposicional	Árvore de decisão
MC4	Supervisionado	Proposicional	Árvore de decisão
Naive Bayes	Não supervisionado	Estatístico	
Nearest-neighbor	Não supervisionado	Estatístico	
Neural Network	Supervisionado	Conexionista	Sistema de equação matemática
OC1	Supervisionado	Proposicional	Árvore de decisão
OneR	Supervisionado	Proposicional	Regras de produção
Perceptron	Supervisionado	Conexionista	Sistema de equação matemática
T2	Supervisionado	Proposicional	Árvore de decisão
Winnow	Supervisionado	Conexionista	Sistema de equação matemática

Tabela 6.1: Classificação de Alguns dos Indutores usados pela *MLC++*.

Uma descrição completa sobre a biblioteca *MLC++* encontra-se em (Kohavi et al., 1994; Kohavi et al., 1997; Félix et al., 1998).

A seguir serão mostrados alguns detalhes e pontos principais de dois algoritmos pertencentes à *MLC++* e utilizados no SIATE.

6.3 Os Algoritmos CN2 e C4.5

Para a prática e treinamento deste trabalho foram utilizados alguns algoritmos da biblioteca *MCC++* sendo os dois principais o CN2 e C4.5.

O algoritmo CN2 é um algoritmo supervisionado, pertencente ao paradigma simbólico, que utiliza regras de produção como linguagem de descrição de exemplos e é não incremental. Basicamente, o algoritmo toma um conjunto de instâncias como entrada e gera um conjunto de regras da forma “se...então” para classificar todas as instâncias.

O algoritmo CN2 gera uma lista de decisão de forma iterativa, construindo uma regra a cada iteração. Uma regra é construída procurando algum *complex* (condições) que cobre um grande número de instâncias para uma classe arbitrária C_i e poucas de outras classes. Tendo encontrado alguns bons *complex*, o algoritmo remove do conjunto de treinamento as instâncias cobertas, e a regra “se $\langle \text{complex} \rangle$ então prevê C_i ” é adicionada para o fim da lista de decisão. Para o conjunto restante, uma nova regra é construída, até que nenhum *complex* adicional de grande importância seja encontrado.

O algoritmo C4.5 foi introduzido por J. Ross Quinlan (Quinlan, 1993) para induzir modelos de classificação a partir de um conjunto de dados. C4.5 é uma extensão do algoritmo original ID3 e permite atributos não avaliados, atributos contínuos, atributos discretos, derivação de regras, entre outras. O sistema C4.5 consiste de quatro programas principais: gerador de Árvore de Decisão (c4.5), gerador de Regras de Produção (c4.5rules), interpretador de Árvore de Decisão (consult), e interpretador de Regras de Produção (consultr).

Na construção de uma árvore de decisão é necessário um conjunto de dados de treinamento, onde cada registro — exemplo — deve ter a mesma estrutura, consistindo de vários pares de atributo/valor. Um desses atributos representa a meta do registro — atributo-classe —, por exemplo: o atributo que seja mais relevante. O problema é determinar uma árvore de decisão com base nas respostas das questões sobre os atributos não-metas. O atributo-classe trabalha com valores discretos e representa a classe a qual o exemplo pertence.

O programa utiliza dois métodos (árvore de decisão ou conjunto de regras de produção) para permitir o uso da informação imprecisa sobre os valores dos atributos, e caracterizar o grau de certeza associado ao predicado mais semelhante entre a classe e suas alternativas.

Quando se pretende utilizar regras de produção como linguagem de representação de

hipóteses faz-se necessário usar o C4.5-rules que é um algoritmo supervisionado, pertencente ao paradigma simbólico, e não incremental.

Na próxima seção será mostrado como funciona a simulação no Módulo Prática/Treinamento do SIATE utilizando a biblioteca *MLC++*, bem como alguns detalhes sobre a forma de representação usada na implementação.

6.4 Simulação Usando a Biblioteca *MLC++*

O paradigma simbólico é usado por sistemas que buscam o aprendizado construindo representações simbólicas de um conceito por meio da análise de exemplos e contra-exemplos. As representações simbólicas geralmente estão na forma de expressão lógica, árvore de decisão, regras de produção, etc.

Como já foi descrito, neste protótipo, é utilizada como ferramenta para implementação o Flex, que acompanha o LPA-Prolog. Para o treinamento, é utilizada a biblioteca *MLC++*. Esse treinamento é realizado com o apoio do Sistema Híbrido de Conhecimento (Seção 4.2).

A solicitação de uma simulação no SIATE ocorre através do Módulo Prática/Treinamento, que é acionado a partir da página de menu do ambiente. Esta página foi mostrada na Figura 3.4, página 57.

Ao começar uma simulação, o Motor de Inferência do SHC, utilizando as informações contidas na Base de Conhecimento Geral, interage com o treinando através do Módulo Coletor de Dados, fazendo perguntas e coletando respostas. Essas perguntas são, basicamente, de dois tipos: as que aceitam mais de uma resposta, e as que aceitam apenas uma resposta. Elas foram construídas utilizando as perguntas predefinidas da ferramenta Flex. Um exemplo dessas perguntas encontra-se a seguir:

```
question cont_discr
    Os dados do atributo meta sao continuos? ;
    choose one of sim, 'nao'
    because Porque eh uma informacao muito importante .
```

```
question atrib_meta
```

```
Os valores do atributo-classe sao conhecidos? ;
```

```
choose one of sim, 'nao'
```

```
browse file q_explicacao .
```

Essas perguntas feitas ao usuário visam a obtenção de informações sobre o conjunto de dados que ele possui e deseja aplicar a um algoritmo da biblioteca, para que em seguida possa ser sugerido a ele um bom algoritmo a ser utilizado.

As próximas perguntas a serem feitas levam em consideração a resposta anterior, não ocorrendo também a execução da mesma pergunta mais de uma vez.

O treinamento tem a opção de questionar sobre uma explicação do porquê está sendo feita a ele determinada pergunta, sendo sua resposta uma simples frase ou informações contidas em um arquivo. Todas as explicações às perguntas estão contidas no Módulo de Explicação.

O conhecimento sobre os algoritmos de AM contidos nesse sistema estão representados na forma de regras, que usa raciocínio *forward chaining*, e relações, que usa raciocínio *backward chaining*, do Flex, e são ativadas por meio de uma cláusula de ação do sistema, que pode ser vista a seguir:

```
action treinamento ;
```

```
do restart
```

```
and forward_chain(fcfs,true,fail,fixed,g_todas_regras,RegrasUsadas)
```

```
and if the answer to solucao_apresentada is 'com explicacao'
```

```
then explain( RegrasUsadas )
```

```
else true
```

```
end if
```

```
and nl .
```

Essa ação executa um raciocínio *forward chaining* e armazena todas as regras que forem utilizadas. Depois disso, é perguntado ao usuário se ele deseja ou não obter essa informação, sendo, em caso positivo, fornecida ao mesmo uma janela com as opções das regras que podem ser selecionadas por ele para uma explicação. Essas explicações também estão contidas no ME.

Após inferido qual o algoritmo mais apropriado para o problema em questão, é selecionado na Base de Casos o caso que melhor se adapte a esse problema. Os casos estão representados na Base de Casos de maneira simplificada, tal como o exemplo abaixo:

```
caso(CN2, crx, crx.saida, [ "/usr/mlclib/mlc",  
    "/usr/mlclib/mlc:/lib:$MLCDIR", "::$MLCDIR/db",  
    7, "/usr/dot/sgi.mips2/lib/lefty", ~/.mlcoptions ],  
    [ crx.names, crx.data, crx.test, cn2 ],  
    [ 690, 15, 37, 2, [44.5%, 55.5%], 80%]).
```

Assim, é apresentado ao usuário os resultados obtidos a partir da simulação da execução do algoritmo de AM contido no caso sugerido pelo sistema.

6.5 Considerações Finais

Ultimamente, a utilização de simulação em ambientes que visam o treinamento e ensino estão se tornando cada vez mais freqüentes. Porém, a simulação não é uma ferramenta auto-suficiente, pois não oferece auxílio e apoio ao estudante durante o processo de treinamento.

Visando isso, foi desenvolvido um sistema que capacitasse o ambiente de treinamento a fornecer informações e conhecimentos adicionais sobre o que está sendo ensinado, sendo também possível ao estudante uma interação com o sistema através de perguntas e respostas. Esse sistema faz parte do Módulo Prática/Treinamento do ambiente SIATE.

No capítulo seguinte são mostrados exemplos de execução, na qual são apresentados os passos necessários para a geração de um roteiro dinâmico, bem como a utilização do Módulo Prática/Treinamento.

Capítulo 7

Exemplos de Execução do SIATE

Neste capítulo são apresentados exemplos de execução do SIATE. Para isso, são mostradas as telas iniciais do ambiente, bem como algumas telas do Módulo Roteiros e Prática/Treinamento. Nestes dois módulos existe o apoio fornecido pelo Sistema Híbrido de Conhecimento.

Para não tornar o exemplo muito extenso, apenas algumas janelas serão mostradas.

Figura 7.1: Página Inicial do Ambiente SIATE.

Ao clicar no desenho central da página (Figura 7.1), é fornecida ao usuário uma tela de identificação (Figura 7.2), que de acordo com o login e a senha do estudante, verifica se o mesmo já utilizou o sistema alguma outra vez. Para os novos estudantes, o sistema pede confirmação de senha e apresenta uma página com um formulário para a entrada de informações sobre os dados pessoais do estudante.

Figura 7.2: Página de Identificação do Ambiente SIATE.

Figura 7.3: Página de Cadastro do Usuário do SIATE.

Depois que o estudante estiver com seus dados validados, aparecerá a página que corresponde a Tela Principal do ambiente (Figura 7.4). As opções do menu estão dispostas sob a forma de cinco botões acompanhados por ícones que correspondem aos quatro módulos do ambiente (Material Didático, Roteiros, Help e Prática/Treinamento) e a opção Sair.

Na Figura 7.4 foi colocada uma seta vermelha apontando para o botão que deve ser acionado para se iniciar o Módulo Roteiros.

Após selecionada essa opção, é apresentada a tela inicial do Módulo Roteiros (Figura 7.5), onde pode ser vista uma lista de *links* que podem ser acionados a partir dessa página.

Ao se selecionar o *link* Geração de Roteiros, é acionado automaticamente o Gerador de Roteiros, apresentando ao usuário uma página com uma pergunta recebida através da comunicação com o Prolog (Figura 7.6).

Após serem feitas as perguntas e recebidas as informações necessárias, o roteiro dinâmico, apresentado como resposta pelo Gerador de Roteiros, é iniciado. Esse roteiro pode ser visualizado graficamente através de uma sequência de páginas selecionadas (Figura 7.7).

Figura 7.4: Página da Tela Principal do Ambiente SIATE.

Figura 7.5: Página Inicial do Módulo Roteiros.

Figura 7.6: Página Inicial para Geração de Roteiros.

Figura 7.7: Página do Hiperdocumento com Roteiro Ativado.

Após percorrido todo o roteiro, ou parte dele, retorna-se para a página principal do ambiente (Figura 7.8). Desta vez, a seta vermelha aponta para o botão que inicia o Módulo Prática/Treinamento.

Figura 7.8: Página da Tela Principal do Ambiente SIATE.

Acionando-se esse botão, é apresentada ao usuário páginas contendo as perguntas necessárias à execução da simulação de um algoritmos de AM. Um exemplo de pergunta é mostrado na Figura 7.9.

Após terem sido feitas todas as perguntas necessárias, o SHC apresenta outra janela questionando se o usuário deseja ou não observar as regras utilizadas no processo de “raciocínio” (Figura 7.10).

Caso se deseje uma explicação, todas as regras utilizadas são mostradas ao usuário, para que este possa selecionar a que lhe convier (Figura 7.11).

Finalmente, é apresentada ao usuário uma janela com algumas regras de produção geradas através da execução do algoritmo selecionado.

Figura 7.9: Página com uma Pergunta Acionada.

Figura 7.10: Página com a Pergunta sobre a Apresentação de Explicação.

Figura 7.11: Página com Janela de Regras.

Figura 7.12: Página com as Regras Geradas.

Como foi dito anteriormente, em ambos os conjuntos de exemplos apresentados, algumas telas intermediárias, contidas ao se realizar a execução como um todo, foram omitidas. Isso se deve ao fato do objetivo deste capítulo ser fornecer uma idéia geral do ambiente SIATE e, em particular, à geração de roteiros e à prática e treinamento.

Vale ressaltar que o primeiro conjunto de telas (Figuras 7.4, 7.5, 7.6 e 7.7) mostra alguns dos passos necessários para a geração de um roteiro dinâmico. Este tipo de roteiro é individualizado e baseado na respostas fornecidas pelo usuário. O roteiro gerado é então apresentado ao usuário que o solicitou. A maneira como é apresentado ao estudante não difere da apresentação dos roteiros predefinidos.

As Figuras 7.8, 7.9, 7.10, 7.11 e 7.12 se referem ao Módulo Prática/Treinamento. Nesse sentido, são feitas perguntas ao usuário sobre as características de seu conjunto de dados. As respostas fornecidas pelo usuário são utilizadas para a seleção de um dado algoritmo de Aprendizado de Máquina e recuperação de um caso referente àquele algoritmo. Depois disso, é mostrado ao estudante um subconjunto das regras de produção que foram obtidas a partir da simulação desse caso.

No próximo capítulo são apresentadas as conclusões obtidas sobre o presente trabalho.

Capítulo 8

Conclusão

Ambientes de treinamento baseados em simulação computacional vêm sendo bastante utilizados. Nesse tipo de ambiente, um pressuposto comumente assumido por seus autores é o de que o treinando possui um razoável conhecimento sobre o modelo subjacente e, portanto, é capaz de analisar e aprender a partir dos resultados da simulação. Além disso, é comum que os softwares de simulação não ofereçam quaisquer outros recursos que não sejam aqueles estritamente relacionados com a tarefa ou atividade em questão.

É interessante notar que outros recursos podem ser adicionados ao ambiente de simulação a fim de se acelerar o processo de aprendizagem e treinamento, tais como técnicas de Inteligência Artificial e Hipermedia.

Técnicas de Inteligência Artificial, como Sistemas Baseados em Conhecimento, Raciocínio Baseado em Casos e Sistemas Tutores podem contribuir em ambientes dessa natureza, aumentando o seu poder de ensino e capacitando-o a “decidir” sobre que conteúdo apresentar e como e quando apresentá-los. Isto pode ser realizado baseando-se nas informações sobre o usuário, no conhecimento especialista do domínio da aplicação, nos resultados de inferências obtidas a partir desse conhecimento e em casos passados, entre outras fontes.

Além disso, a utilização de recursos de Hipermedia em ambientes de ensino oferece uma maior flexibilidade de uso e uma melhor apresentação das informações aos usuários. Isto os estimula a aprender devido, principalmente, a recursos áudio-visuais disponíveis.

Dentro desse contexto, foi proposta a arquitetura de um ambiente para treinamento e ensino denominado **SIATE**, com a finalidade de auxiliar o estudante e incrementar esse tipo de ambiente com recursos que torne o treinamento mais interessante e efetivo. Para

isso, essa proposta integra características de Sistema Tutor, Sistema Baseado em Conhecimento, Raciocínio Baseado em Casos, Hipermissão e Simulação. É importante lembrar que a arquitetura proposta para esse ambiente congrega dois trabalhos de mestrado.

O objetivo do SIATE consiste em enriquecer um ambiente exploratório de ensino com conhecimento especialista do domínio, através da integração de tecnologias, como a Inteligência Artificial, Hipermissão e Simulação. Isto se mostrou satisfatório, considerando-se o protótipo implementado e os resultados nele obtidos.

A principal contribuição do presente trabalho foi o desenvolvimento de um Sistema Inteligente, chamado Sistema Híbrido de Conhecimento, para dar suporte ao SIATE. Este apoio é dado tanto com relação a utilização de conhecimento especializado do domínio de aplicação e casos passados, quanto referente a geração dinâmica de roteiros para ajuda individualizada do estudante, que é realizada a partir de uma rede semântica estendida.

Sendo assim, o SHC é o componente do SIATE que fornece o suporte necessário ao Módulo Roteiros e ao Módulo Prática/Treinamento a fim de executar as tarefas contidas nesses módulos.

Visando a validação do ambiente proposto, foi construído um protótipo tendo a Aquisição de Conhecimento para Sistemas Inteligentes como domínio de aplicação. O intuito foi de verificar a viabilidade da arquitetura proposta, assim como a integração dos módulos do SIATE.

Nesta versão do SIATE, o enfoque principal foi no desenvolvimento e na integração de todos os módulos e componentes existentes nessa arquitetura para treinamento e ensino. Já o conhecimento do domínio da aplicação foi representado utilizando-se exemplos simples.

Para esse tipo de problema, o sistema funciona adequadamente, inclusive sendo capaz de gerar um roteiro dinâmico, utilizando a linguagem Prolog, e mostrá-lo através do hiperdocumento pertencente ao ambiente. Além disso, permite a navegação por um ambiente exploratório e o suporte de conhecimento especializado.

Como medida futura, o protótipo construído deverá ser validado com estudantes e professores de Inteligência Artificial, além de outros profissionais da área, a fim de levantar subsídios para aprimoramentos e eventuais correções.

O protótipo mostra-se adequado para tratar as situações em que sejam interessante efetuar simulações e utilizar casos bem sucedidos que possam ser usados como experiência passada.

Com relação à mudança de domínio serão necessárias algumas alterações no SHC. Entre elas, a Base de Conhecimento do sistema deve ser modificada, pois deverá conter o conhecimento referente ao novo domínio. No Módulo Roteiro, o maior esforço será na construção do modelo semântico do novo domínio e das perguntas. Feito isso, a transcrição para linguagem Prolog (construção da Base de Modelagem do Domínio) é quase direta, já que o código foi definido visando essa facilidade. Todo o raciocínio, ou seja, o Motor de Inferência mantém-se inalterado, visto que ele manipula uma Base de Conhecimento qualquer que segue as formas definidas pela representação da modelagem GSM. A Camada de Interação Usuário-SHC, devido à forma genérica de implementação, também não necessita de alteração.

Outra possível mudança que merece destaque, refere-se à substituição da Base de Conhecimento Geral e da Base de Casos, que deverá conter o conhecimento geral e específico, além das perguntas, que devem retratar o novo domínio de aplicação.

Portanto, além de validar o ambiente de treinamento e ensino já existente, novos domínios podem ser testados a fim de se verificar a generalidade da arquitetura proposta.

Outras sugestões de trabalhos futuros são as seguintes:

- Completar a modelagem do domínio para a geração de roteiros dinâmicos que atuem em outros módulos do SIATE.
- Incrementar o conhecimento do domínio — Aquisição de Conhecimento — existente, bem como o número de casos. Com isso, haverá a necessidade da implementação de mecanismos mais eficientes de Raciocínio Baseado em Casos para busca e adaptação do caso mais semelhante contido na Base de Casos.
- Implementar outras formas de explicações fornecidas aos usuários, como, por exemplo, *what.if* (“O que aconteceria se”) que permite substituir apenas algumas condições e executar o processo de inferência novamente.
- Englobar todos os algoritmos disponíveis na *MCC++* no Módulo Prática/Treinamento.

Referências

- Aamodt, A. (1994). Explanation-driven case-based reasoning. *Topics in Case-based Reasoning*, Springer Verlag, pages 274–288.
- Aamodt, A. & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations and system approaches. *AI Communications*, 7(1):39–59.
- Arantes, A. C. N. & Rodrigues, S. R. (1994). Um sistema tutorial para o ambiente que auxilia a construção de núcleos específicos de sistemas especialistas (ANSEsp). Technical Report 19, ICMC-USP¹
- Batista, G. E. A. P. A. (1997). Um ambiente de avaliação de algoritmos de aprendizado de máquina utilizando exemplos. Master's thesis, ICMC-USP. Dissertação de Mestrado.
- Boehm, B. W. (1988). A spiral model of software development and enhancement. *IEEE Computer*.
- Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996a). The KDD process for extracting useful knowledge from volumes of data. *Communication of the ACM*, 39(11):27–34.
- Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (1996b). *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, Menlo Park, CA.
- Félix, L. C. M., Rezende, S. O., Doi, C. Y., de Paula, M. F., & Romanato, M. J. (1998). *MCC++* biblioteca de aprendizado de máquina em C++. Technical Report 72, ICMC-USP.
- Freire, M. E. P. (1998). O sistema tutor de um ambiente inteligente para treinamento e ensino. Master's thesis, ICMC-USP. Dissertação de Mestrado.
- Giarratano, J. C. & Riley, G. (1994). *Expert Systems: Principles and Programming*. PWS Publishing Company - 2nd Edition.

¹A partir de 03/03/95 a autora Rodrigues, S. R. passou a usar o nome Rezende, S. O.

- Goonatilake, S. & Khebbal, S., editors (1995b). *Intelligent Hybrid Systems*. John Wiley & Sons Ltd.
- Goonatilake, S. & Khebbal, S. (1995a). *Intelligent Hybrid Systems: Issues, Classifications and Future Directions*, chapter 1. In (Goonatilake & Khebbal, 1995b).
- Horst, P. S., Padilha, T. P. P., Rocha, C. A. J., Rezende, S. O., & Carvalho, A. C. P. L. (1998). Knowledge acquisition using symbolic and connectionist algorithms for credit evaluation. *Anais do IEEE World Congress on Computational Intelligence - WCCI'98 - ANchore - Estados Unidos*, pages 277–282.
- Hull, R. & King, R. (1987). Semantic database modeling: Survey, applications, and research issues. *ACM Computing Surveys*, 19(3):201–260.
- ICDT 97 (1997). *Proceedings International Conference on Database Theory*, Delphi, Greece. Springer-Verlag.
- Jackson, P. (1986). *Introduction to Expert Systems*. Addison-Wesley.
- Jones, S. R., Miles, J. C., & Read, M. W. (1996a). A comparison of knowledge elicitation methods. *Expert Systems*, 13(4):277–295.
- Jones, S. R., Miles, J. C., & Read, M. W. (1996b). Standardising the elimination of unnecessary data from interview transcripts for the purposes of knowledge-based systems - a case study. *Expert Systems*, 13(1):3–14.
- Jong, T. (1991). Learning and instruction with computer simulations. *Education & Computing* 6, pages 217–229.
- Joolingen, W. V. (1994). Qmaps: Qualitative reasoning for simulation learning environments. *Artificial Intelligence in Education*, 5(2):177–197.
- Jubileu, A. P., Pugliesi, J. B., Rezende, S. O., & Sanches, R. (1998). Proposta de um processo para a aquisição de conhecimento explícito. Technical Report 77, ICMC-USP.
- Kerber, R., Livezey, B., & Simoudis, E. (1995). *A Hybrid System for Data Mining*, chapter 7, pages 121–141. In (Goonatilake & Khebbal, 1995b).
- Kohavi, R., John, G., Long, R., Manley, D., & Pfleger, K. (1994). *MCC++: A machine learning library in C++*. *Tools with Artificial Intelligence, IEEE Computer Society Press*, pages 740–743. <http://www.sgi.com/Technology/mlc>.

- Kohavi, R., Sommerfield, D., & Dougherty, J. (1997). Data mining using *MCC++*: A machine learning library in C++. *International Journal on Artificial Intelligence Tools*, 0(0).
- Leite, J. C. (1991). A interação entre usuários e sistemas de computadores em linguagem natural orientada por menus. Master's thesis, PUC-Rio. Dissertação de Mestrado.
- Liebowitz, J. & Salvo, D. A. D. (1989). *Structuring Expert Systems: Domain, Design and Development*. Yourdon Press.
- Lopes, A. A. (1995). Raciocínio baseado em casos. Master's thesis, ICMC-USP. Dissertação de Mestrado.
- Luther, A. C. (1994). *Authoring Interactive Multimedia*. Academic Press, Inc. AP Professional.
- Mannila, H. (1997a). Data mining: Machine learning, statistic and databases. <http://www.cs.helsinki.fi/~mannila/>.
- Mannila, H. (1997b). Methods and problems in data mining. In (ICDT 97, 1997).
- Marietto, M. G. B. (1996). Tendências nas áreas de sistemas tutores inteligentes e modelagem do estudante. <http://www.ita.cta.br/~marietto/tipaper.html>.
- McGraw, K. L. & Briggs, K. H. (1989). *Knowledge Acquisition: Principles and Guidelines*. Prentice Hall.
- Milaré, C. R. (1997). Sistema híbrido: Raciocínio baseado em casos e redes neurais. Master's thesis, ICMC-USP. Dissertação de Mestrado.
- Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Series in Computer Science.
- Monard, M. C., Batista, G. E. A. P. A., Kawamoto, S., & Pugliesi, J. B. (1997). Uma introdução ao aprendizado de máquina por exemplos. Technical Report 29, ICMC-USP.
- Nunes, M. G. V. & Hasegawa, R. (1997). Developing intelligent tutoring systems for mathematics. *Computers and Artificial Intelligence*, 16(6):561–581.
- Nunes, M. G. V., Turine, M. A. S., Maltempi, M. V., & Hasegawa, R. (1993). Uso de hipertexto/hipermídia em sistemas tutores inteligentes. Technical Report 009, ICMC-USP.

- Nwana, H. S. (1990). Intelligent tutoring systems: An overview. *Artificial Intelligence Review*, 4:251-277.
- Pereira, L. F. S. M. & Aragon, D. F. (1997). A aquisição de conhecimento e o processo de modelagem. Technical Report 02/97, ILTC - Niterói-RJ.
- Preece, J., Rogers, Y., Sharp, H., & Benyon, D. (1994). *Human-Computer Interaction*. Addison-Wesley.
- Pressman, R. S. (1994). *Software Engineering: A Practitioner's Approach*. Edição Européia, McGraw-Hill.
- Pugliesi, J. B. & Rezende, S. O. (1997). Ferramenta flex para desenvolvimento de sistemas baseados em conhecimento. Technical Report 27, ICMC-USP.
- Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc., Los Altos, California, USA.
- Rezende, S. O., Oliveira, R. B. T., Félix, L. C. M., & Rocha, C. A. J. (1998). Visualization for knowledge discovery in database. *Anais da International Conference on Data Mining - Rio de Janeiro*, pages 81-95.
- Rezende, S. O. & Pugliesi, J. B. (1998). Aquisição de conhecimento explícito ou manual. Technical Report 37, ICMC-USP.
- Rocha, C. A. J., Padilha, T. P. P., Horst, P. S., & Rezende, S. O. (1997). Processo de desenvolvimento de sistemas baseados em conhecimento considerando aspectos de qualidade. *Anais da SEPAI - Semana Paraense de Informática - Belém*.
- Rodrigues, S. R. (1990). Um ambiente para auxiliar a construção de núcleos específicos de sistemas especialistas. Master's thesis, ICMC-USP. Dissertação de Mestrado.
- Rodríguez, C. H. (1997). Knowledge acquisition: Techniques in and barriers to knowledge acquisition. <http://osiris.sund.ac.uk/~cs0che/chweb/paper2.htm>.
- Russell, S. J. & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*. Prentice Hall.
- Schank, R. C. (1990). Case-based teaching: Four experiences in education software design. *Interactive Learning Environments*, 1(4):231-253.

- Schank, R. C. (1994a). Active learning through multimedia. *IEEE-Multimedia*, pages 69–78.
- Schank, R. C. (1994b). What we learn when we learn by doing. *Technical Report #60, Northwestern University*.
- Soares, L. F. G. (1992). *Modelagem e Simulação Discreta de Sistemas*. Editora Campus Ltda.
- Sylvester, A. & Pham, T. (1997). Basics of computer simulation. <http://tommy.jsc.nasa.gov/~sylveste/basics.html>.
- Tecuci, G. & Kodratoff, Y. (1995). *Machine Learning and Knowledge Acquisition: Integrated Approaches*. Academic Press.
- Viccari, R. M. & Giraffa, L. M. M. (1996). Sistemas tutores inteligentes: Abordagem tradicional x abordagem de agentes. *Tutorial do XIII Simpósio Brasileiro de Inteligência Artificial - Curitiba - PR*.
- Weiss, S. M. & Kulikowski, C. A. (1991). *Computer Systems That Learn. Classification and Prediction Methods From Statistics, Neural Nets, Machine Learning and Expert Systems*. Morgan Kaufmann Publishers, CA.