REPRESENTAÇÃO DE CONHECIMENTO EM ENGENHARIA DO CONHECIMENTO

por

Nina Edelweiss

RP-163 Setembro/1991

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
Av. Osvaldo Aranha, 99
90210 - Porto Alegre - RS - BRASIL
Telefone: (0512) 281633
Telex: (051) 2680 - CCUF BR
FAX: (0512) 244164
E-MAIL: PGCC@sbu.ufrgs.anrs.br

Correspondência: UFRGS-CPGCC
Caixa Postal 1501
90001 - Porto Alegre - RS - BRASIL

Editor: Ricardo Augusto da Luz Reis (interino)

SÚMULA

LISTA DE FIGURAS

RESUMO

Este relatório apresenta um estudo das diferentes formas existentes para representação de conhecimento em Bases de Conhecimento de Sistemas de Inteligência Artificial. Inicialmente são vistos os diferentes paradigmas utilizados para representar conhecimentos, sendo cada um deles exemplificado através da modelagem de uma mesma aplicação - algumas das atividades executadas em uma agência de locação de fitas de vídeo. Em seguida são apresentadas algumas ferramentas utilizadas para a construção de bases de conhecimentos e de sistemas baseados em conhecimentos. Para finalizar são apresentados alguns sistemas baseados em conhecimento utilizados na área específica de Sistemas de Informação de Escritórios, focalizando sempre a forma como é efetuada a representação do conhecimento.

PALAVRAS-CHAVE: Inteligência Artificial, Bases de Conhecimento, Representação de Conhecimento, Sistemas Baseados em Conhecimento.

ABSTRACT

The result of a study about the different knowledge representation forms used in the Knowledge Bases of existent Artificial Intelligence Systems is presented here. The different paradigms used for knowledge representation are examined. The same example is used to present an example of each one of the paradigms - the modeling of some of the activities executed in a video rental shop. Some existing tools used in the construction of knowledge bases and knowledge based systems are presented. Finnaly, some knowledge based systems used in Office Information Systems are presented, showing how the knowledge representation is done.

KEYWORDS: Artificial Intelligence, Knowledge Bases, Knowledge Representation, Knowledge Based Systems.

1. INTRODUÇÃO

A Inteligência Artificial se preocupa com a solução de problemas através de máquinas, utilizando nesta solução conhecimentos sobre o domínio da aplicação que está sendo desenvolvida. Bases de conhecimento armazenam formalmente este conhecimento.

Um dos principais problemas da Inteligência Artificial consiste na representação e na utilização dos conhecimentos a respeito do mundo real. Sobre este conhecimento se baseiam todos os processos executados por "programas inteligentes". Dependendo da fase de desenvolvimento ou de utilização de um sistema, a manipulação de conhecimento apresenta objetivos diferentes - aquisição de conhecimentos, sua representação nas bases de conhecimento, sua recuperação e os raciocínios efetuados baseados nele. Entretanto, devido à sua própria natureza, o conhecimento apresenta propriedades que dificultam sua utilização, tais como ser muito volumoso, difícil de caracterizar e estar em constante mutação. Muitas pesquisas têm sido desenvolvidas para contornar estas dificuldades.

A forma utilizada para a representação do conhecimento influi diretamente nos métodos que serão utilizados tanto na aquisição deste conhecimento como na busca de soluções para os problemas propostos. A importância da representação do conhecimento nestes sistemas faz com que sejam conhecidos como sistemas baseados em conhecimento.

Dos sistemas baseados em conhecimento, os que mais se destacam são os sistemas especialistas, nos quais são armazenados conhecimentos de uma área específica, conhecimentos estes normalmente só disponíveis através de uma pessoa especialista da área. Os sistemas especialistas são utilizados para simular a atuação de um especialista, fornecendo diagnósticos, respondendo perguntas, etc.

A construção de sistemas baseados em conhecimento evoluiu muito nos últimos anos - de construções puramente manuais à geração automática através de ferramentas de apoio. A importância deste assunto gerou uma nova área - a Engenharia de Conhecimento.

Este trabalho tem por objetivo fazer um estudo de métodos de representação de conhecimento, no âmbito da Engenharia do Conhecimento. Inicialmente, no capítulo 2, é feita uma revisão

dos conceitos básicos - conhecimento, metaconhecimento, bases de conhecimento, sistemas especialistas, Engenharia do Conhecimento. Em seguida, no capítulo 3, são apresentados os diferentes paradigmas utilizados para representação do conhecimento. Cada um deles é exemplificado através de uma mesma aplicação, a modelagem de um escritório, uma agência de locação de fitas de vídeo. No capítulo 4 são feitas algumas considerações a respeito de construção de bases de conhecimento, englobando a aquisição, a modelagem e o refinamento do conhecimento. O capítulo 5 apresenta ferramentas que podem ser utilizadas para a construção de sistemas baseados em conhecimento, dando ênfase à construção da base de conhecimentos. O capítulo 6 apresenta algumas ferramentas baseadas em conhecimento, utilizadas na área de Sistemas de Informação de Escritórios. Em cada uma procura-se mostrar como é feita a representação do conhecimento. Para finalizar, o capítulo 7 apresenta algumas conclusões do presente trabalho.

2. CONCEITOS GERAIS

2.1 CONHECIMENTO E METACONHECIMENTO

Conhecimento é a soma de percepções de um indivíduo a respeito de aspectos de um determinado universo de discurso (domínio) em um determinado momento (tempo) [MAT89]. Um conhecimento particular deve sempre ser associado a (1) um indivíduo, (2) um domínio, e (3) um momento.

Segundo [VIC90], o conhecimento pode ser classificado em quatro níveis inter-relacionados: (1) conhecimento de domínio, no nível de aplicação; (2) conhecimento genérico, no nível de tarefa; (3) conhecimento básico, no nível da capacidade; e (4) conhecimento formal, no nível da tecnologia. Os elementos básicos de cada um destes níveis são, respectivamente: (1) conceitos, representação declarativa de objetos de um domínio de aplicação; (2) regras, através das quais podem ser estabelecidas, por exemplo, relações entre causa e efeitos, graus de crenças e probabilidades de conclusão; este conhecimento deve ser fornecido por um especialista no domínio de aplicação; (3) modelos, definidos por uma coleção de regras inter-relacionadas, usualmente associadas à hipóteses particular ou a uma conclusão global de diagnósticos; e (4) estratégias, definidas através de regras e de procedimentos, utilizadas na exploração do conhecimento.

O conhecimento pode também ser classificado segundo os seguintes aspectos: (1) conhecimento descritivo, com informações que descrevem características do conhecimento; (2) conhecimento operacional, que define a utilização do conhecimento; e (3) conhecimento organizacional, que representa relacionamentos entre conhecimentos.

Metaconhecimento significa conhecimento a respeito do conhecimento. Considerando uma determinada teoria - uma linguagem, um dispositivo dedutivo e um conjunto de axiomas - tudo o que puder ser expresso através desta teoria constitui conhecimento; os fatos que puderem ser definidos ou provados com respeito à teoria propriamente dita constituem o metaconhecimento [AIE86].

O metaconhecimento tem sido usado, entre outras áreas, em linguagens de representação do conhecimento. A representação do metaconhecimento junto ao conhecimento permite que se possa

não somente utilizar o conhecimento armazenado, mas também examiná-lo, fazer abstrações e dirigir sua aplicação, através da definição do conhecimento sobre o controle [DAV79]. Além disso, êle aumenta o poder de expressão das linguagens de representação de conhecimentos, uma vez que permite a formalização de crenças, raciocínio "default", raciocínio a respeito de visões múltiplas de um objeto, inferências em situações mutantes, etc. [AIE86].

Uma representação de conhecimentos apresenta, portanto, dois níveis diferentes [BEN88, DAV79]: (1) o nível objeto, que descreve o mundo exterior, através de entidades, relacionamentes e ações correspondentes a uma tarefa; e (2) o nível meta, que descreve o mundo interior da representação, caracterizando o nível objeto. As expressões simbólicas do nível objeto são as entidades primitivas do nível meta.

2.2 BASES DE CONHECIMENTO

Todo sistema utilizado em Inteligência Artificial possui uma base de conhecimentos. Nesta são armazenados tanto o conhecimento como o metaconhecimento, segundo uma representação adequada.

Uma base de conhecimentos é constituída de um conjunto de fatos e de um conjunto de regras e heurísticas (expressas através de estratégias) através das quais podem ser derivados novos fatos [VIC90].

Segundo [DEL86], uma base de conhecimentos KB é definida por um par $< KB_o , \vdash_L >$ onde KB_o é uma coleção de comandos (axiomas) em uma determinada linguagem de lógica L, e $\vdash_L$ é a relação de derivabilidade em L, isto é, especifica o que pode ser derivado a partir dos axiomas, dadas as regras de inferência em L. Então:

$$x \text{ pertence a } KB \quad s.s.s. \quad KB_o \vdash_L x$$

Deste modo, a base de conhecimentos KB contém não somente os comandos de KB_o, mas também outros que podem ser derivados deles em L.

O conteúdo de uma base de conhecimentos inclui conhecimento sobre três diferentes aspectos [MAT89]: (1) conhecimento a

respeito do domínio de aplicação; (2) informações sobre o processo que está sendo desenvolvido - dados específicos, resultados parciais, novos conhecimentos inferidos, etc; e (3) conhecimento heurístico sobre o domínio de aplicação e sobre o problema particular que está sendo resolvido.

Três diferentes níveis de percepção podem ser destacados na análise de uma base de conhecimentos: simbólico, de engenharia e de conhecimento.

O nível simbólico corresponde à representação simbólica do conhecimento na base de conhecimentos (também conhecido como nível de programa). O conhecimento é representado estruturalmente, como por exemplo, em termos de objetos e de relações entre instâncias destes objetos. São considerados, neste nível, as estruturas dos dados armazenados e os algoritmos necessários para a sua manipulação.

No nível de engenharia [BRA86] é considerado o ponto de vista do projetista do sistema. São considerados os aspectos organizacionais da base de conhecimentos, tais como especialização por refinamento, gerenciamento de versões e manipulação de exceções. Este nível se preocupa em como as pessoas encaram e processam as informações contidas na base de conhecimentos.

O conceito de nível de conhecimento [BRA86, NEW82, LEV84,86] foi introduzido por Newell. Neste nível, o conhecimento é caracterizado funcionalmente, através do que é capaz de realizar. Uma base de conhecimentos, neste nível, é tratada como um Tipo Abstrato de Dados, interagindo com o sistema e com o usuário somente através de um pequeno conjunto de operações. As respostas a estas operações caracterizam o domínio que está sendo modelado. No nível de conhecimento é considerado o conteúdo de informação da base de conhecimentos, sem se preocupar em como esta informação está armazenada ou como será manipulada.

Newell define um sistema, no nível de conhecimento, como um agente. Um agente é composto de um conjunto de metas, um conjunto de ações e de um corpo. O elemento básico deste nível é o conhecimento. O agente processa o seu conhecimento para determinar as ações que irá realizar. A lei de comportamento é o princípio da racionalidade: as ações são selecionadas para atingir as metas do agente.

A principal característica do nível de conhecimento é que nele o conhecimento é encarado sob ponto de vista de sua noção de competência, de sua potencialidade de gerar ações. Desta maneira se separa o conteúdo epistemológico de um sistema das ferramentas e dos sistemas utilizados na sua implementação.

O nível meta está contido no nível simbólico, pois nele está representada a sintaxe da representação - trata do conhecimento a respeito das estruturas simbólicas e de como manipulá-las.

Em [FOX86a] é proposta a idéia de um nível organizacional, que define como diversos agentes, cada um deles com limitações de conhecimento, podem cooperar uns com os outros para resolver seus problemas. Neste nível seriam, portanto, definidas características de segurança, sincronização, conhecimento incompleto e inconsistências.

2.3 SISTEMAS ESPECIALISTAS

Um sistema especialista é um sistema técnico que fornece respostas a perguntas de um determinado domínio [BUS89]. Resolve problemas em um domínio delimitado, problemas estes que normalmente só poderiam ser resolvidos por pessoas especialistas no conhecimento daquele domínio. Representa explicitamente grandes porções de conhecimento, sendo capaz de efetuar raciocínios sobre este conhecimento.

Um sistema especialista ideal deveria apresentar as seguintes características [WAL86]: (1) resolver problemas que somente poderiam ser resolvidos por pessoas especialistas na área considerada; (2) apresentar flexibilidade para incorporar novos conhecimentos; (3) fornecer apoio para elicitação, estruturação e transferência de conhecimentos; (4) permitir que o conhecimento armazenado seja facilmente visto; (5) fornecer explicações a respeito de seus resultados; (6) poder argumentar a respeito da natureza e da forma de execução de uma tarefa, baseado em conhecimentos inexatos; (7) poder se comunicar através de sentenças simples, semelhantes às da linguagem falada.

A arquitetura básica de um sistema especialista apresenta os seguintes componentes:

a) uma base de conhecimentos que contém todo o conhecimento necessário ao sistema para que possa desempenhar o papel de "especialista" no domínio de aplicação deste sistema, obtido de um especialista humano no assunto;

b) um motor de inferências, constituído por um conjunto de métodos capazes de manipular as informações armazenadas;

c) interfaces de diálogos, que asseguram os diálogos entre o sistema e um especialista, na fase de aquisição e validação de conhecimentos, e entre o sistema e um usuário, para resolução de problemas;

d) componente de aquisição de conhecimentos, através do qual o engenheiro de conhecimentos obtém o conhecimento de especialistas e o insere na base de conhecimentos; e

e) um gerador de explicações, dirigidas ao engenheiro de conhecimento, ao especialista ou ao usuário final; este gerador assegura ao usuário a possibilidade de acompanhar o raciocínio desenvolvido na solução de um problema (ex: sistemas CENTAUR [AIK83,85], INTEXP [GEO85b]).

Somente os três primeiros componentes estão presentes em todos os sistemas especialistas. Os últimos dois - ferramenta para apoio à aquisição do conhecimento e gerador de explicações - são uma nova tendência. A figura 2.1, tirada de [MAT89], apresenta um esquema da arquitetura básica de um sistema especialista.

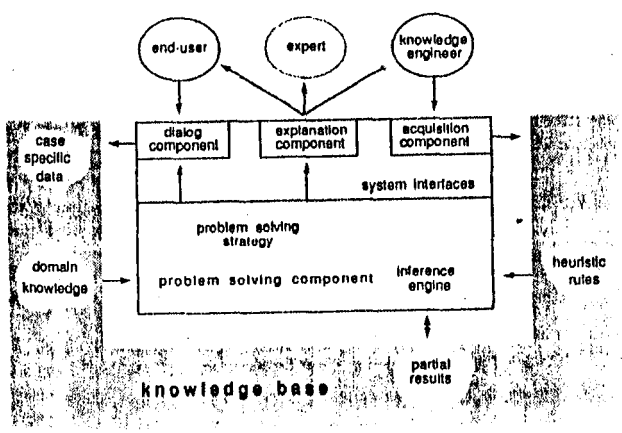

Figura 2.1 - ARQUITETURA DE UM SISTEMA ESPECIALISTA

2.4 ENGENHARIA DE CONHECIMENTO

No desenvolvimento de sistemas baseados em conhecimentos, o conjunto dos processos de aquisição, análise e representação do conhecimento é denominado de Engenharia do Conhecimento.

Diversas pessoas estão envolvidas nestes processos: (1) o especialista, que fornece os conhecimentos específicos do domínio que está sendo modelado; (2) o engenheiro do conhecimento, que efetua a modelagem deste conhecimento para que seja armazenado em uma base de conhecimentos; (3) o usuário, que é a pessoa que utiliza o sistema; e (4) o gerente, que administra todo o processo.

A Engenharia do Conhecimento baseia-se em técnicas de Psicologia, de Sociologia e de modelagem estruturada. Os processos podem ser apoiados por ferramentas de software disponíveis, que devem ser escolhidos de acordo com o domínio de aplicação.

O desenvolvimento de sistemas baseados em conhecimento não difere muito do desenvolvimento de outros tipos de sistemas de informação. Alguns aspectos são similares àqueles encontrados no desenvolvimento de sistemas de informação convencionais, tais como análise da informação, seleção de aplicações, gerência de projetos, captura dos requisitos do usuário, projeto modular e reusabilidade [WIE89]. Outros aspectos, entretanto, são fundamentalmente diferentes, envolvendo geralmente a construção da base de conhecimentos - desde aos problemas encontrados na aquisição do conhecimento, como na eficiência da modelagem utilizada para representar o conhecimento, que irá influenciar diretamente a recuperação deste conhecimento.

A contrução de sistemas baseados em conhecimento evoluiu muito nos últimos anos. As bases de conhecimento dos primeiros sistemas especialistas eram construídas manualmente, geralmente apresentando a forma de regras. A tendência atual é de geração automática de sistemas, através de ferramentas de apoio (ex: ambientes KEE [GEV87], KADS [WIE89], INTEXP [GEO85b], linguagem OPS5 [GEV87]). No capítulo 5 deste trabalho serão apresentadas algumas destas ferramentas.

Em [WEI89] é apresentada uma metodologia para desenvolvimento de sistemas baseados em conhecimento. Identifica diferentes processos de desenvolvimento, processos estes que podem ser

ativados, desativados e reativados independentemente, podendo inclusive executar em paralelo. Os processos são: (1) identificar o problema a ser resolvido pelo sistema; (2) examinar as possibilidades de solução do problema; (3) identificar subproblemas; (4) identificar e definir a estrutura conceitual da base de conhecimentos; (5) fazer o projeto conceitual; (6) definir projeto de detalhes; (7) descrever o conhecimento através de uma linguagem (codificar); (8) testar a lógica da implementação; (9) testar o conhecimento modelado, procurando identificar incorreções e ambiguidades; e (10) validar o sistema junto a especialistas/usuários. A metodologia apresentada utiliza um "shell" para implementar um protótipo do sistema. Este protótipo pode ser gerado de maneira incremental, resultando no sistema definitivo; ou ser criado através de uma prototipação rápida, sendo o protótipo utilizado como especificação do sistema a ser implementado.

3. PARADIGMAS DE REPRESENTAÇÃO DO CONHECIMENTO

A forma como o conhecimento é representado na base de conhecimentos é um dos pontos fundamentais de um sistema baseado em conhecimento. Ela determina como será efetuada a aquisição do conhecimento e quais os métodos a serem utilizados nos processos de inferência.

Nos sistemas implementados são encontradas formas bastante diferentes de representação. Podem ser identificadas três basicamente diferentes [VIC90]: (1) sistema lógico, onde o conhecimento é descrito através de asserções, postuladas como verdadeiras (um conjunto de axiomas) acerca do que é modelado; (2) sistema estrutural, que descreve o conhecimento através de objetos e de relações entre entidades a serem modeladas; e (3) sistema procedimental, onde o conhecimento é descrito através de procedimentos. As diferentes formas utilizadas para representação de conhecimentos se enquadram sempre em um ou mais destes sistemas acima.

Diferentes tipos de conhecimento requerem diferentes formas de representação de conhecimento [ALT90]. Segundo o conteúdo das informações armazenadas, as formas de representação de conhecimento podem ser classificadas em:

a) declarativas, quando são armazenados fatos verdadeiros do domínio, além de pequenos procedimentos que permitem manipular estes fatos com o objetivo de derivar novos fatos; apresentam vantagens tais como representação única das informações e facilidade para acréscimo de novas informações à base de conhecimentos;

b) procedimentais, quando além das informações declarativas são representados procedimentos que indicam como as informações podem ser utilizadas; as vantagens de sua utilização incluem facilidade de representar conhecimentos a respeito da maneira de realizar determinadas ações e de heurísticas para realizá-las eficientemente.

Algumas formas de representação do conhecimento se apropriam mais à representação do conhecimento declarativo, outras à do conhecimento procedimental. Isto deve ser levado em consideração ao procurar identificar a forma mais adequada à representação do conhecimento de um determinado domínio de aplica-

ção. Um sistema de representação de conhecimento, para que seja eficiente, deve apresentar as seguintes propriedades [RIC88]: (1) adequação de representação do conhecimento ao domínio da aplicação; (2) adequação inferencial; (3) eficiência inferencial; e (4) eficiência na aquisição de conhecimentos.

Embora muito se tenha pesquisado na área de representação do conhecimento, não se encontrou ainda uma indicação de qual o tipo de representação que deve ser utilizado para modelar uma determinada tarefa. A escolha de qual o método a adotar geralmente é feita de acordo com os conhecimentos a modelar e com as ferramentas disponíveis para implementar a base de conhecimentos.

Apresentamos, a seguir, as características fundamentais dos principais paradigmas de representação de conhecimentos [BEN87, MYL84, RIC88, VIC90, WIN84]. Para ilustrar sua utilização foi definida uma aplicação comum, a modelagem de algumas atividades desenvolvidas em uma agência de locação de fitas de vídeo. Procura-se, desta forma, comparar os diferentes paradigmas, identificando quais as facilidades e quais as falhas de cada um deles. Iniciamos descrevendo a aplicação que servirá de exemplo, seguindo com a apresentação dos paradigmas.

3.1 APLICAÇÃO: MODELAGEM DE UMA LOCADORA DE FITAS DE VÍDEO

Através deste exemplo procuramos identificar como seria feita a modelagem de um escritório, uma locadora de fitas de vídeo, através dos diferentes paradigmas de representação de conhecimento. Para não tornar o trabalho por demais extenso serão modeladas somente algumas das muitas tarefas desenvolvidas neste tipo de escritório: (1) o cadastramento de clientes; (2) o cadastramento de fitas; e (3) o registro de início de locações. Serão considerados somente os aspectos mais representativos de cada uma destas tarefas. Outras tarefas deste escritório seriam, por exemplo, aquelas relacionadas com os funcionários da locadora, com publicidade e de controle financeiro.

O cadastramento de um cliente é efetuado em mais de um cadastro. Um cadastro geral armazena as informações de todos os clientes da locadora. Dois outros cadastros trazem informações relacionadas a bons e maus clientes - o primeiro armazenando as informações a respeito de clientes que podem alocar fitas e o segundo, informações sobre clientes indesejáveis, aos quais será

vetada a locação de fitas. Todo cliente deve estar registrado em um destes cadastros, além do cadastro geral.

Cada cliente pode alugar no máximo 4 (quatro) fitas. O tempo máximo de locação foi considerado de 60 (sessenta) dias.

As informações manipuladas no exemplo são, portanto: (1) fita, com as propriedades código, nome do filme e tipo do filme; (2) catálogo de fitas, formado por objetos fita; (3) clientes, com as propriedades código, nome e endereço; (4) cadastro de bons clientes, armazenando códigos de clientes; (5) cadastro de maus clientes, formado por códigos de clientes; e (6) cadastro de locações, que contém informações a respeito das locações, incluindo código do cliente, código da fita alugada e data de início da locação.

3.2 REGRAS DE PRODUÇÃO

A representação do conhecimento através de regras de produção é bastante popular na codifição de conhecimento heurístico em programas para resolução de problemas científicos e, sobretudo, médicos. O conhecimento é representado através de um conjunto de fatos e assertivas, e de um conjunto de regras de produção. Os fatos podem ser representados através de simples sequências de caracteres, ou de objetos estruturados tais como listas, tuplas, registros, etc. As regras se apresentam da seguinte forma:

SE <condição> ENTÃO <consequência>

Quando a condição (premissa) for verdadeira, a consequência também o será. Esta consequência geralmente é a execução de uma ação. A ação pode ser uma interação com o usuário, pode ativar outra regra, ou até mesmo incluir novos fatos na base de conhecimentos.

O controle percorre as regras de algum modo sistemático, verificando as condições, à procura de alguma que seja verdadeira naquele momento. Deve existir algum mecanismo de solução de conflitos para os casos em que mais de uma regra seja satisfeita no mesmo instante.

Através desta forma de representação podem ser armazenados os seguintes aspectos de conhecimento [HAY85]: (1) inferências decorrentes de observações específicas; (2) abstrações, generalizações e classificações de determinados dados; (3) condições necessárias e suficientes para alcançar algum objetivo determinado; (4) indicações de localização de informações relevantes; (5) estratégias para eliminar incertezas e minimizar riscos; (6) consequências esperadas para situações hipotéticas; e (7) causas prováveis de sintomas. Existe, atualmente, um grande interesse na introdução de incertezas em bases de conhecimento que utilizam esta representação [EDD86].

As principais vantagens das regras de produção são [ALT90] a facilidade de tradução em regras de inferência, a simplicidade da notação, a facilidade de compreensão e a representação única de cada informação.

A principal desvantagem deste tipo de representação é a inerente falta de organização entre as regras, o que pode tornar muito demorada a recuperação de informações em grandes bases de conhecimento. Quando o número de regras de uma base de conhecimentos se torna muito grande, torna-se difícil também o entendimento dos interrelacionamentos entre as regras. Para diminuir esta dificuldade as regras devem ser organizadas em módulos pequenos, fáceis de serem manipulados e compreendidos.

Outras limitações desta forma de representação são: dificuldade na escrita das regras, na modificação posterior das regras por pessoas diferentes daquela que escreveu as regras originais e dificuldade de utilização das regras em aplicações diferentes daquela para a qual o sistema foi projetado. Uma análise detalhada do sistema MYCIN [CLA83], pr exemplo, detectou que regras individuais podem desempenhar papéis diferentes, possuem tipos diversos de justificativas e são construídas através de diferentes raciocínios para a escolha e a ordem de premissas. Existe, portanto, um conhecimento de projeto de regras, o qual consiste de conceitos estruturais e estratégicos. Se este conhecimento também fosse representado na base de conhecimentos, a tarefa de construção e de modificação das regras seria bastante facilitada.

Exemplos de sistemas especialistas que utilizam esta representação: MYCIN [BUC85, DAV77], PLANNER [RIC88] e R1 [FOX86a].

A modelagem das tarefas da agência de locação de fitas de vídeo através de regras de produção poderia ser efetuada através das regras abaixo. Não existe nenhuma ordem entre as regras, tendo a numeração apresentada por único objetivo a sua referência posterior neste texto.

(1) SE nome e endereço do cliente não estão no cadastro de clientes
E código do cliente não está no cadastro de clientes
ENTÃO cadastrar o cliente com este código, nome e endereço
E incluir o código deste cliente no cadastro de bons clientes.

(2) SE código está no cadastro de bons clientes
E existe pedido para incluir este código no cadastro de maus clientes
ENTÃO retirar o código do cadastro de bons clientes
E incluir o código no cadastro de maus clientes.

(3) SE código está no cadastro de maus clientes
E existe pedido para incluir este código no cadastro de bons clientes
ENTÃO retirar o código do cadastro de maus clientes
E incluir no cadastro de bons clientes.

(4) SE o código da fita não está no cadastro de fitas
ENTÃO cadastrar a fita (código, nome do filme, tipo do filme).

(5) SE o cliente é bom cliente (seu código está no cadastro de bons clientes)
E o cliente possui no máximo três locações
E o cliente não está de posse de alguma fita a mais de 60 dias
ENTÃO o cliente pode alugar uma fita.

(6) SE a fita não está alugada
E o cliente pode alugar uma fita
ENTÃO (alugar a fita)
esta fita está alugada
E cliente aluga esta fita
E data de locação desta fita é a data atual.

A regra (1) trata do cadastro de um novo cliente, incluindo-o simultaneamente no cadastro de bons clientes. As regras (2) e (3) fazem alterações nos cadastros de bons e maus clientes: quando for solicitada a inclusão de um cliente no cadastro de maus clientes, seu código deve ser retirado do cadastro de bons clientes e incluído no de maus clientes (e vice-versa). Deste modo, nunca ocorrerá o caso de um cliente estar registrado simultaneamente nos dois cadastros. O cadastramento de uma fita é realizado através da regras (4), devendo ser registrados seu código, nome do filme e tipo do filme. A regra (5) testa se um cliente tem condições de alugar uma fita no momento considerado e a regra (6) verifica se todas as condições de locação de uma determinada fita a um determinado cliente são satisfeitas e, em caso positivo, procede ao registro dos dados desta locação.

Não estamos apresentando aqui a forma utilizada para armazenar os fatos, forma esta que influi na maneira de efetuar as pesquisas na base de conhecimentos, de modo a permitir a avaliação das diferentes regras. A estruturação do conhecimento não pode ser representada através das regras. Esta forma de representação se apropria bem à modelagem de conhecimentos a respeito de trocas de estado.

3.3 LÓGICA DE PREDICADOS

Este paradigma de representação resultou das pesquisas em prova automática de teoremas. É utilizado para a representação de conhecimento declarativo. O conhecimento é representado através de um conjunto de axiomas que descrevem fatos do mundo real. Os axiomas são escritos segundo alguma linguagem de lógica de predicados, como fórmulas lógicas compostas de constantes, variáveis, funções, predicados, conectivos lógicos e quantificadores. A mais utilizada á a lógica de predicados de primeira ordem. A inferência é feita procurando provar algum dos axiomas.

As principais vantagens da utilização de lógica de predicados de primeira ordem são [BRO86, RE184]: (1) a representação de uma semântica formal bem definida e de fácil compreensão; (2) a simplicidade da notação; (3) a unicidade da definição das informações; (4) a uniformidade de representação (um esquema de representação padronizado), sendo todas as informações representadas através da mesma linguagem de lógica; (5) a uniformidade operacional, pois a teoria de prova de teoremas de primeira ordem é

o único mecanismo de manipulação das informações em todas as operações (recuperação de informações, dedução de novos fatos, solução de problemas, resposta a perguntas e prova de teoremas); e (6) a possibilidade de definição de novas lógicas a partir da lógica de primeira ordem. É de suma importância a presença de regras de inferência, através das quais podem ser definidos procedimentos para recuperação de informações, verificação de restrições semânticas e solução de problemas.

As principais dificuldades que apresenta são a falta de organização da base de conhecimentos, pois todas as fórmulas são independentes, e a dificuldade de representação de conhecimento procedimental e heurístico. Além disso, apresenta limitações na representação de alguns tipos básicos de conhecimento - tais como crenças, "defaults", conhecimento incompleto e autoconhecimento [BRO86]. Também apresenta dificuldades para manipular conhecimentos que possuam efeitos colaterais, o que precisa ser feito através de regras de produção [SMI86].

A principal aplicação deste paradigma é a linguagem PROLOG [STE86] através da qual ele ficou muito popular. PROLOG é um eficiente provador de teoremas para um subconjunto da lógica de primeira ordem. É a linguagem mais utilizada em aplicações de Inteligência Artificial na Europa. Nela, o conhecimento é armazenado através de fatos, representando as informações verdadeiras, e de regras que permitem inferir novas informações. Algumas das dificuldades apresentadas acima são superadas nesta linguagem, uma vez que apresenta uma ordem na definição das informações e combina representação lógica com procedimental, permitindo, portanto, a representação simultânea de conhecimento procedimental.

Outra aplicação deste paradigma é encontrada no sistema FOL [WEY80], através do qual é estabelecido um diálogo com o usuário a respeito de algum assunto. Este diálogo apresenta três momentos: (1) inicialmente é definida a linguagem que será utilizada, definição esta feita em conjunto com o usuário; (2) já na linguagem estabelecida, são discutidos os fatos (axiomas) que serão considerados verdadeiros; e (3) é estabelecido o diálogo a respeito das consequências destes fatos. FOL não é um provador de teoremas, mas um construtor interativo de provas.

A modelagem das tarefas da agência de locação de fitas de vídeo através de lógica de predicados, utilizando a linguagem de lógica de 1ª ordem apresentada em [CAS87], seria a seguinte:

(1) VxVyVz(fita(x,y,z) -> (codigo_fita(x) filme(y) tipo(z)))
(2) Vx(fita(x,_,_) -> y(fita(y,_,_) y=x))
(3) Vx(tipo(x) -> x (drama, comédia, desenho, musical))
(4) VxVyVz(cliente(x,y,z) ->
(codigo_cliente(x) nome(y) endereço(z)))
(5) Vx(cliente(x,_,_) -> y(cliente(y,_,_) y=x))
(6) Vx(cliente(x,_,_) -> (bom_cliente(x) mau_cliente(x)))
(7) Vx(bom_cliente(x) -> y(mau_cliente(y) y=x))
(8) Vx(mau_cliente(x) -> y(bom_cliente(y) y=x))
(9) VxVyVz(alugado(x,y,z) ->
(codigo_fita(x) codigo_cliente(y) data(z)))
(10) Vx(alugado(x,_,_) -> y(alugado(y,_,_) y=x))
(11) VxVy(alugado(x,y,_) -> (x_1 x_2 x_3 x_4 x_5 y_1(alugado(x_1,y_1,_)
alugado(x_2,y_1,_) alugado (x_3,y_1,_)
alugado (x_4,y_1,_) alugado (x_5,y_1,_)) y_1=y))
(12) VxVyVz(alugado(x,y,z) ->
x_1 y_1 z_1(alugado(x_1,y_1,z_1) (y_1=y ((z_1-z)>60 dias))))

As três primeiras fórmulas definem o cadastramento de fitas - a primeira apresenta as características de cada fita, a segunda determina que cada fita deve ter um código único e a terceira define os valores possíveis para o atributo tipo de filme. As fórmulas 4 a 8 tratam dos cadastros de clientes. A quarta fórmula define os atributos de um cliente, a quinta determina que cada cliente deve ter um código único e a sexta, que cada cliente deve estar registrado em um dos cadastros de maus e bons clientes. As fórmulas 7 e 8 definem que este último registro ocorra em somente um dos cadastros, ou seja, nenhum cliente pode estar registrado simultaneamente no cadastro de bons e de maus clientes. As últimas fórmulas definem o registro de início de uma locação. A fórmula 9 determina os atributos a serem registrados quando é efetuada uma locação. Através da fórmula 10 se evita que uma mesma fita seja alugada ao mesmo tempo a mais de um cliente. A fórmula 11 determina que nenhum cliente alugue, simultaneamente, cinco fitas e a última, que não possua nenhuma locação por período superior a 60 dias.

A definição das informações na base de conhecimentos seria efetuada através de fatos, tais como:

```
fita(123, Rainman, drama).
cliente(14, Marcos, av. do Baliza 20).
bom_cliente(14).
alugado(123,14,010191).
```

Vemos portanto que através de fórmulas lógicas podemos modelar tanto a estrutura como as restrições das informações que podem ser definidas. Não podemos, entretanto, determinar a seqüência em que uma determinada operação deve ser efetuada, para o que é necessário conhecimento procedimental. A utilização da linguagem PROLOG inclui algum conhecimento procedimental, embutido na própria linguagem, que pode ser utilizado para contornar em parte este problema.

3.4 REDES SEMANTICAS

As redes semânticas permitem representar o conhecimento através de modelos, os quais apresentam uma rede de objetos (objeto físico ou entidade conceitual) de um domínio de conhecimentos, associados entre si através de relacionamentos binários. As alterações na base de conhecimentos são efetuadas através da inserção e da remoção de objetos, e da manipulação das relações. São utilizadas para representação de conhecimento declarativo.

Estas redes são usualmente representadas graficamente, através de grafos orientados formados por nodos ligados através de arcos rotulados. Os nodos representam os objetos e os arcos, as associações entre pares de objetos. Procedimentos de inferência especializados operam estas redes.

Os arcos podem ser de diversos tipos, dependendo do relacionamento existente entre os objetos. O significado do rótulo do arco é que introduz a "semântica" na rede - uma semântica intuitiva, informal. Por exemplo, o arco "tem_um" representa uma propriedade do nodo. Outros arcos permitem classificar o conhecimento segundo hierarquias, com herança de propriedades entre classes, subclasses e instâncias. Através desta classificação a base de dados pode ser organizada por [MAT89]: (1) classificação, caracterizada pelos arcos "membro_de" e "instância_de"; (2) generalização, representada pelo arco "é_um"; e (3) agregação, caracterizada por "parte_de".

Uma das principais vantagens apresentada pelas redes semânticas é a concentração de todas as informações a respeito de um determinado objeto em torno do nodo que o representa, facilitando muito a recuperação de informações. Desta maneira é possível verificar facilmente se informações a respeito de um objeto são novas, redundantes, inconsistentes ou deriváveis a partir de relações prévias.

Além desta concentração de informações, as redes semânticas apresentam como principais vantagens, uma forte organização da base de conhecimentos e facilidade de compreensão devido à existência de notação gráfica.

Suas principais desvantagens são: (1) a falta de uma terminologia padronizada; (2) a falta de uma semântica formal, o que dificulta a verificação da corretude dos processos de inferência; e (3) poucas linguagens de reprentação implementadas.

Esta forma de representação de conhecimentos não é muito utilizada atualmente, a não ser em processamento de linguagem natural. Como exemplo de sistemas especialistas que a utilizam podemos citar PROSPECTOR [ABE86] e DIGS [SGU85]. Em DIGS, utilizado para diagnósticos, independente do domínio, os nodos correspondem a sintomas e os arcos, aos relacionamentos entre os sintomas.

Uma possível modelagem das tarefas da agência de locação de fitas de vídeo através de uma rede semântica é apresentada na figura 3.1.

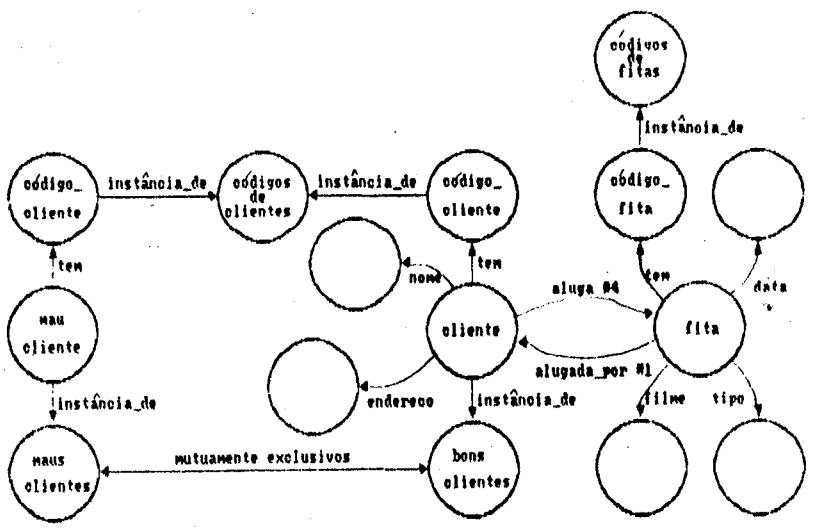

Figura 3.1 - REDE SEMANTICA DA AGENCIA DE FITAS DE VíDEO

Nesta rede um cliente é definido como uma instância de bons ou de maus clientes, sendo estes dois últimos conjuntos mutuamente exclusivos. Os bons clientes apresentam atributos código, nome e endereço, enquanto que dos maus clientes se armazena somente o código. Os códigos de todos os clientes são únicos. Uma fita possui atributos código (único), filme, tipo do filme e data de início de sua locação. A locação de uma fita por um cliente é representada através de dois arcos. O arco "alugada_por" possui cardinalidade 1, definindo que uma fita só pode ser alugada por um cliente; o arco "aluga", de cardinalidade 4, permite que um cliente alugue simultaneamente até 4 fitas.

Não foi possível modelar, utilizando somente a rede semântica, a restrição de locação de uma fita a um cliente que possua alguma outra fita por um período superior a 60 dias.

3.5 "FRAMES"

A noção de "frames" (enquadramentos) foi elaborada por Minsky no início dos anos 70, a partir de estudos em redes semânticas e organização de memória. Os "frames" são estruturas modulares através das quais se pode modelar o conhecimento. Um "frame" descreve um objeto específico ou um conjunto de objetos. É composto por "slots" (inserções), onde são armazenadas informações associadas a este objeto - relacionamentos deste objeto com outros objetos, atributos fixos e/ou variáveis, valores "default", apontadores para outros "frames", além de conjuntos de regras e de procedimentos que são executados cada vez que o "slot" é acessado. Este paradigma é utilizado para representação de conhecimento declarativo.

As hierarquias de classificação das redes semânticas também podem ser modeladas através de "frames", sendo ressaltada a herança de atributos. Podem ser encarados como uma coleção de nodos de redes semânticas, incluindo nodos vazios, os quais descrevem um objeto estereotipado, um conceito ou uma situação [VIC90].

Os "frames" são os antecessores dos sistemas orientados a objetos, sendo que estes últimos apresentam adicionalmente comunicação entre os objetos.

A principal vantagem da utilização de "frames" é a possibilidade de representação de relacionamentos complexos dentro do domínio.

Apresenta algumas dificuldades para a representação de conhecimento procedimental, pois se apropria para conhecimento declarativo. O conhecimento procedimental pode ser incluído nos "slots", através de chamadas a procedimentos que seriam executados no momento de acesso a este "slot".

Como exemplo de sistemas especialistas que apresentam esta forma de representação podemos citar CALLISTO [FOX86a,b] e WHEZZE [SMI85]. As linguagens de representação de conhecimentos FRL, KRL, OWL e KL-ONE [MYL84] utilizam "frames" na representação do conhecimento.

A modelagem das tarefas da agência de locação de fitas de vídeo através de "frames" é a seguinte:

```
frame cliente
   slots: código_cliente : INTEIRO
          nome           : STRING
          endereço       : STRING
          aluga          : fita.código_fita
             cardinalidade máxima: 4

frame bom_cliente
   superclasse: cliente
   slot:  conceito       : bom

frame mau_cliente
   superclasse: cliente
   slot:  conceito       : mau

frame fita
   slots: código_fita    : INTEIRO
          filme          : STRING
          tipo           : (drama, comédia, desenho, musical)
          locação        : "executar locação de fita"
          alugada_por    : ( bom_cliente U    )
          data_de_locação: ( DATA U    )
```

O "frame" cliente define as características de um cliente. O "slot" cliente.aluga possui uma faceta que define a

cardinalidade máxima deste "slot", limitando a locação de fitas por cliente a quatro. Os "frames" bom_cliente e mau_cliente possuem cliente como superclasse, herdando todos os seus atributos. Apresentam um novo "slot" que define o conceito deste cliente.

O "frame" fita define as características de uma fita da locadora. Devem ser definidos seu código, nome do filme e tipo do filme, sendo que este último é um dos tipos da lista apresentada. O "slot" fita.locação contém uma chamada a um procedimento que efetua a locação, verificando se o cliente não possui nenhuma fita a mais de 60 dias. Este procedimento inclui os dados do início da locação nos "slots" fita.alugada_por, fita.data_de_locação e cliente.aluga.

3.6 PROCEDIMENTOS

Nesta forma de representação o conhecimento é modelado através de procedimentos que descrevem o comportamento do domínio de interesse. A base de conhecimentos é formada por uma coleção de agentes ou processos. O comportamento é representado por trocas de mensagens entre estes agentes, descritas através de procedimentos. Para isto são utilizadas linguagens de programação de alto nível. Duas classes de linguagens, seguindo diferentes paradigmas, são muito utilizadas [MAT89]: linguagens orientadas a objetos e linguagens orientadas a dados.

Nas linguagens que seguem o paradigma de orientação a objetos são definidos objetos, os quais encapsulam definição de dados e de métodos. Os métodos são definidos através de procedimentos. A modelagem do comportamento de um domínio é feita através de trocas de mensagens entre os objetos. Na mensagem enviada é especificada a operação que deve ser executada pelo objeto que a receber, operação esta que deve corresponder a um dos métodos definidos para este objeto. Os procedimentos são ativados pela passagem de mensagens. Um exemplo de linguagem que segue este paradigma é SMALLTALK [GOL81].

Já nas linguagens orientadas a dados, a ativação dos procedimentos é efetuada através do acesso a um determinado dado ao qual o procedimento está ligado. Os procedimentos que são ativados em situações particulares são usualmente denominados de "demons". A associação de procedimentos a dados é útil na representação de conhecimento intencional, para expressar restrições

de integridade complexas e para provocar modificações na base de conhecimentos, baseadas em acessos particulares. Um exemplo deste tipo de representação é apresentado pela linguagem PLANNER [RIC88] de geração de planos de alto nível. Nesta linguagem, uma base de conhecimentos é encarada como uma coleção de asserções e de teoremas (os "demons").

Em última análise, a representação de um comportamento é feita através da definição de um plano para a utilização de uma informação.

Em [GEO86] é apresentado um formalismo para a representação do conhecimento procedimental que utiliza a noção de "processo". O formalismo possui semântica declarativa, através da qual podem ser definidos fatos sobre os efeitos de ações executadas no domínio de interesse, e semântica operacional, a qual define como o conhecimento armazenado pode ser utilizado.

As vantagens da representação procedimental sobre outras formas de representação são as interações diretas entre objetos, a boa organização, estruturação e ausência de tempo de procura de informações.

Suas principais desvantagens são a dificuldade de compreensão, de modificação, o sequenciamento e a existência de efeitos colaterais.

A modelagem das tarefas da agência de locação de fitas de vídeo através de procedimentos, utilizando uma pseudo-linguagem genérica, apresenta a seguinte forma:

```
Cadastrar cliente_novo (nome, endereço)
   SE nome, endereço não pertencem ao cadastro clientes
   ENTÃO obter código_cliente
         inicializar locações em zero
         incluir código_cliente, nome, endereço, locações no
            cadastro clientes
         incluir código_cliente no cadastro bons_clientes.

Cadastrar mau_cliente (código_cliente)
   retirar código_cliente do cadastro bons_clientes
   incluir código_cliente no cadastro maus_clientes
```

```
Cadastrar fita (filme, tipo)
   obter código_fita
   incluir código_filme, filme, tipo no cadastro fitas

Registrar locação (código_filme, código_fita, data)
   SE código_cliente não pertence ao cadastro maus_clientes
      E locações de código_cliente < 4
      E não existe locação deste código_cliente com período su-
           perior a 60 dias
   ENTÃO incrementar  locações deste código_cliente em uma uni-
           dade
         incluir  código_cliente, código_fita, data no cadastro
           locações.
```

Os procedimentos atuam sobre cinco cadastros: clientes, bons_clientes, maus_clients, fitas e locações. Este modo de modelar torna-se complexo à medida em que se detalham mais os procedimentos. O controle do número de fitas máximo por cliente, por exemplo, foi modelado através de uma variável "locações" associada a cada cliente. Esta deve ser inicializada, testada e incrementada a cada nova locação. Este controle está distribuído nos procedimentos, dificultando sua compreensão. O controle de locação somente para clientes que não possuam nenhuma fita em seu poder a mais de sessenta dias não foi detalhado no exemplo.

3.7 REPRESENTAÇÕES MIXTAS

Os sistemas especialistas hoje desenvolvidos geralmente utilizam mais de um destes paradigmas ao mesmo tempo. Deste modo, as deficiências de um destes modelos de representação são cobertas pelo outro.

A representação mais utilizada atualmente é uma combinação de "frames" e de regras. Os "frames" representam a estrutura das entidades do domínio, e as regras modelam os relacionamentos entre as entidades e sua evolução com o tempo. O metaconhecimento a respeito dos mecanismos utilizados pelo sistema para avaliação das regras também pode ser representado através de regras [MIS86]. Exemplos desta combinação são LOOPS [CHA87], KEE [FIK85] e CENTAUR [AIK83,85].

A associação de redes semânticas com outros paradigmas aumenta o seu poder de representação. Através da combinação de

redes semânticas com lógica, associando regras aos nodos, podem ser representados modelos orientados a objetos não-determinísticos. O modelo ERAE [DUB86], por exemplo, utilizado para representar o conhecimento adquirido na fase de coleta de requisitos de um sistema, utiliza redes semânticas e lógica. Derivou do modelo de Entidades-Relacionamentos de Chen [CHE76], representando o conhecimento através de triplas (Relacionamento de entidades, Atributo, Evento). Em [MYL83] é apresentada uma integração de redes semânticas com noções procedimentais, através de PSN.

Uma combinação de redes semânticas com "frames" se constitui na representação denominada de redes conceituais. Em uma rede conceitual os nodos representam conceitos. Um conceito pode apresentar atributos que descrevem suas propriedades (declarativas, procedurais ou estruturais). Os nodos são definidos, portanto, como "frames". As redes conceituais suportam o paradigma de orientação a objetos. Através desta notação pode ser representado tanto o conhecimento como o metaconhecimento. Estas redes são utilizadas no sistema KRYSIS [MAT89] e na linguagem RECON [OLI90].

Representação lógica pura não é praticamente utilizada - aparece combinada pelo menos com representação procedimental. É o caso das linguagens PROLOG [STE86] e FOL [WEY80].

3.8 OUTROS TIPOS DE REPRESENTAÇÃO

Além dos tipos clássicos de representação de conhecimentos apresentados, vários outros têm sido desenvolvidos. Vamos citar alguns, a título de ilustração.

Em [ATT86] é apresentado um sistema lógico para representação do conhecimento, baseado em descrições em lugar de predicados. Este sistema apresenta propriedades de herança e de representação de atributos, propriedades estas geralmente só encontradas em "frames" e redes semânticas. A linguagem utilizada neste sistema se denomina OMEGA.

Outra forma de representação é definida por uma forma especial de rede semântica, denominada Objeto-Atributo-Valor [LEU90]. Manipula triplas (O,A,V) onde O é um conjunto de objetos, A é um conjunto de atributos que representam características ou propriedades dos objetos e V é um conjunto de valores que es-

pecificam os atributos. O relacionamento entre um objeto e um atributo é da forma "tem_um" e o relacionamento entre um atributo e um valor, da forma "é_um". Os objetos e seus atributos correspondem aos nodos de uma rede semântica, representando o conhecimento estático. Os valores representam o conhecimento dinâmico, podendo ser modificados sem alterar os estáticos. É uma forma de representação muito estruturada, mas torna-se muito complexa quando cresce o número de objetos a serem representados.

Outro método utiliza hiper-redes ("Hypernet Method") [GEO85a] para a representação do conhecimento. Uma hiper-rede é uma estrutura definida pela tripla H = (N, R, p), onde "N" é um conjunto de objetos denominados hiper-nodos, "R" é um conjunto de entidades relacionais denominadas hiper-relações e "p" é uma lista de propriedades. Um hiper-nodo representa um objeto abstrato, sendo definido por $N = (h_n, S_n, I_n, E_n, p_n)$, onde "$h_n$" é o identificador do hiper-nodo, "S_n" define o seu tipo, "I_n" é uma hiper-rede de menor nível, na qual está a descrição da estrutura interna deste hiper-nodo em termos de hiper-nodos e de hiper-relações que o compõe, "E_n" é um conjunto de fórmulas-bem-formadas ("linkage patterns") que descrevem as possíveis ligações deste hiper-nodo com outros e "p_n" é uma lista de propriedades. Vemos, portanto, que a estrutura da hiper-rede é recursivamente definida, pois cada hiper-nodo é também constituído por uma hiper-rede. A linguagem de representação do conhecimentos que manipula estas estruturas denomina-se HYPER. Este método foi utilizado no Sistema Especialista INTEXP [GEO85b], um sistema independente do domínio de aplicação.

Uma forma de representação baseada em gramáticas de atributos é apresentada em [VOU89]. É estabelecida uma correspondência entre as primitivas de estruturação de conceitos e as primitivas utilizadas para estruturar uma gramática de atributos. As gramáticas de atributo permitem a especificação de informações estruturais, sendo possível ainda a associação de informações condicionais e procedimentais aos conceitos.

Em [MAY85] é apresentada uma teoria unificada para representação do conhecimento, através da qual se pode: (1) provar resultados gerais que se aplicam a várias linguagens de representação do conhecimento; (2) transferir representações de conhecimentos particulares de uma linguagem de representação para outra; e (3) combinar conhecimentos representados em uma determinada linguagem com conhecimentos representados em outras linguagens. A

teoria se baseia em conceitos de "signature", "frames", "structures" e "galleries". O artigo citado demonstra que esta teoria inclui outras linguagns de representação do conhecimento, inclusive o formalismo de Montague [DOW81] para representação de linguagem natural.

4. CONSTRUÇÃO DE BASES DE CONHECIMENTO

A construção de bases de conhecimento apresenta três etapas diferentes - aquisição, modelagem e refinamento do conhecimento.

4.1 AQUISIÇÃO DO CONHECIMENTO

A fase de aquisição do conhecimento, também conhecida como elicitação do conhecimento, consiste na transferência do conhecimento de uma fonte adequada para a base de conhecimentos. São procurados os fatos e dados relacionados com o domínio da aplicação (conhecimento declarativo) e as normas de utilização destes fatos (conhecimento procedimental). A técnica que será utilizada na representação deste conhecimento desempenha um papel importante na sua aquisição.

O processo de aquisição do conhecimento deve ser efetuado com apoio de um engenheiro do conhecimento. Seu objetivo é buscar o conhecimento necessário ao sistema que está sendo construído, de uma maneira ordenada e objetiva. O conhecimento pode ser retirado de livros, de banco de dados, de resultados de programas de simulação e, principalmente, de especialistas humanos no domínio de aplicação do sistema.

A transferência do conhecimento de um especialista no domínio de aplicação para a base de conhecimentos de um sistema especialista é um do pontos críticos do desenvolvimento destes sistemas. É um processo demorado, tedioso, não bem definido, que envolve uma interação entre o engenheiro do conhecimento e o especialista no domínio da aplicação. Se não for utilizado algum procedimento de apoio, informações incorretas podem ser inseridas na base de conhecimentos.

O processo de aquisição de conhecimento não deve ser confundido com aprendizagem [OLI90]. Na aquisição do conhecimento o sistema desempenha um papel relativamente passivo, sendo as modificações na base de conhecimentos efetuadas através da engenheiro de conhecimento. Na aprendizagem, o sistema participa ativamente, efetuando ele próprio as modificações na base de conhecimentos. Aprender significa adquirir automaticamente novos conhecimentos, reestruturar conhecimentos anteriores ou melhorar o desempenho de tarefas [VIC90].

Diversas técnicas foram desenvolvidas para o processo de aquisição do conhecimento de um especialista, sendo as mais utilizadas as entrevistas e a utilização de ferramentas automatizadas.

4.1.1 ENTREVISTAS

Em uma entrevista o engenheiro de conhecimento faz perguntas a um especialista do domínio da aplicação. Tanto as perguntas como as respostas devem ser relevantes, indicando processos de solução de problemas no domínio considerado - fatos, regras e heurísticas. As entrevista podem ser de vários tipos.

Uma entrevista aberta consiste na interação direta entre o ongenheiro de conhecimentos e o especialista, através de perguntas e de respostas. As perguntas são colocadas pelo engenheiro, de acordo com o andamento da entrevista. Este método é de suma importância quando aplicado no início do processo de aquisição do conhecimentos, pois ajuda o engenheiro de conhecimentos a compreender a natureza, a extensão e a complexidade do conhecimento a ser adquirido. Existem diversas técnicas de entrevistas abertas: técnicas para estabelecer o contato, técnicas para estabelecer o entendimento, técnicas para expandir o raciocínio.

Em uma entrevista dirigida o engenheiro de conhecimento coloca perguntas ao especialista com o objetivo de detalhar conhecimentos específicos do domínio. Este tipo de entrevista geralmente é feito após uma entrevista aberta.

Entrevistas estruturadas (também conhecidas como análise de protocolos) [COO86] consiste na observação, por parte de um engenheiro de conhecimentos, de um especialista no seu domínio de atuação, enquanto soluciona um problema. Nestas entrevistas é importante a utilização de equipamentos de áudio ou de vídeo, que possibilitam a posterior observação do trabalho desenvolvido pelo especialista.

Aspectos importantes de uma entrevista são [BLI87]: (1) o nível social do encontro, devendo os participantes simpatizar um com o outro, de modo a permitir sinceridade na colocação das respostas; (2) a comunicação entre as duas pessoas envolvidas no processo - o especialista e o engenheiro de conhecimento, uma vez que geralmente um não conhece a área de conhecimento do outro;

(3) a habilidade do entrevistador de colocar perguntas relevantes.

Um dos mais graves problemas apresentados nas técnicas de entrevistas é que envolvem introspecção e expressão verbal do conhecimento. As dificuldades de exprimir o conhecimento podem gerar informações incompletas, inadequadas e inexatas.

O produto de uma entrevista são dados verbais. As informações que serão representadas na base de conhecimentos serão construídas e modeladas a partir de uma análise destes dados verbais, pelo engenheiro de conhecimentos. Estas informações devem ser apresentadas novamente ao especialista para validação dos conceitos, uma vez que podem ser feitas deduções incorretas do material coletado [FRE87].

4.1.2 FERRAMENTAS AUTOMATIZADAS PARA AQUISIÇÃO DE CONHECIMENTO

As ferramentas automatizadas que têm por objetivo a aquisição do conhecimento apresentam editores de texto e interfaces baseados em conhecimento, que facilitam a interação do especialista com o sistema. Os sistemas RLL e KAS [BAR83, WAT83], por exemplo, apresentam editores muito sofisticados, que fazem verificação não só da sintaxe mas também da semântica das informações fornecidas.

Estas ferramentas podem ser classificadas segundo o grau de cooperação humana que apresentam - desde aquelas que somente captam os conhecimentos de um especialista, até sistemas de aprendizagem automática, sem interferência humana.

Como exemplo de uma ferramenta que apresenta uma alto grau de interação com o especialista podemos citar TEIRESIAS [DAV79].

Quando o problema a ser resolvido pelo sistema especialista envolve um grande número de dados, é conveniente a utilização de ferramentas que possuam pouca interação com o especialista, retirando as informações de um banco de dados. Sistemas especialistas deste tipo podem ser gerados pelo "shell" SYSTEM X-I [LEU90].

O sistema especialista DENDRAL apresenta um programa que escreve automaticamente as regras - o META-DENDRAL [BUC78].

Deste modo é contornada a dificuldade de escrita de regras na transferência iterativa de conhecimento a partir de um especialista. O sistema DENDRAL colhe o conhecimento em tabelas de regras de produção, construídas com o auxílio do META-DENDRAL.

Várias ferramentas baseiam a representação do conhecimento adquirido na estratégia de solução de problemas que será utilizada sobre a base de conhecimentos. Para isto, deve ser bem definido o papel que o conhecimento irá desempenhar na solução de problemas - ou seja, a base de conhecimentos deverá ser definida no nível de conhecimento. A definição funcional da base de dados, isto é, da função que será desempenhada pelo conhecimento, apresenta várias vantagens [CLA85, MAR89, SWA83]: (1) indica quais as informações relevantes que devem ser solicitadas aos especialistas na fase de aquisição; (2) permite que seja examinada a adequação da utilização do sistema especialista na solução de um determinado problema; (3) permite que o sistema de solução de problemas já saiba como aplicar este conhecimento; (4) facilita a implementação de explicações do sistema sobre suas decisões; (5) permite a escolha de um conjunto de problemas apropriados para testar a validade do sistema; e (6) permite um mapeamento da descrição de um problema do domínio de aplicação para a estratégia de sua solução, possibilitando um melhor entendimento deste processo. Como exemplo de ferramenta que utiliza este enfoque podemos citar SALT [MAR89].

O sistema ODYSSEY [FIK81] apresenta uma ferramenta para aquisição de dados de Sistemas de Informação de Escritórios, a qual utiliza conhecimentos do domínio de aplicação para guiar o processo de aquisição.

4.2 MODELAGEM DO CONHECIMENTO

A modelagem do conhecimento significa a sua representação através de algum formalismo eficiente, o qual possibilita a sua inserção na base de conhecimentos. Esta modelagem é efetuada pelo engenheiro do conhecimento ou pela ferramenta que estiver sendo utilizada. Em alguns casos, o próprio especialista deve fornecer as informações no formalismo empregado, o que dificulta a sua atuação, podendo inclusive dificultar o processo de aquisição do conhecimento.

4.3 REFINAMENTO DO CONHECIMENTO

Nas fases de aquisição e de modelagem do conhecimento não é levado em consideração o desempenho da base de conhecimentos. A preocupação destas fases iniciais é na aquisição de todo o conhecimento necessário ao sistema. Em uma fase posterior é feito o refinamento progressivo do conhecimento, com o objetivo de procurar um melhor desempenho. No caso de sistemas baseados em regras, por exemplo, a fase de aquisição consiste na obtenção de conjuntos de regras inteiras, necessárias à prova de determinada hipótese. Na fase de refinamento serão analisadas os componentes das regras individuais.

O refinamento do conhecimento geralmente é efetuado através de ferramentas automatizadas. Em [POL84], por exemplo, é apresentado o sistema SEEK, o qual fornece, interativamente, conselhos a respeito de refinamento de regras durante o projeto de um sistema especialista. Estes conselhos consistem em fornecer sugestões de generalizações ou especializações de regras. O sistema utiliza estudos de casos armazenados, com conclusões conhecidas, para guiar o engenheiro do conhecimento ou o especialista no refinamento das regras. Em [GIN88] é apresentado o sistema SEEK2, o qual estende as capacidades do sistema SEEK, o qual tem a capacidade de efetuar refinamento automático sem a interferência humana. Utiliza, para isto, análise empírica do comportamento das regras.

5. FERRAMENTAS DE DESENVOLVIMENTO DE SISTEMAS BASEADOS EM CONHECIMENTO

Os primeiros sistemas especialistas foram construídos manualmente, sendo quase sempre baseados em regras de produção. Os especialistas precisavam escrever as regras diretamente na linguagem de representação do conhecimento, o que dificultava a aquisição do conhecimento. Os implementadores dos sistemas precisavam construir tanto as estruturas de representação (regras, "frames"), como os mecanismos de inferências e de recuperação das informações. A medida que os sistemas se tornaram maiores, armazenando grande volume de conhecimento, esta prática se tornou inviável. Diversas ferramentas têm sido desenvolvidas para auxiliar o desenvolvimento de sistemas baseados em conhecimento, apoiando desde a fase de aquisição do conhecimento, como também a sua modelagem, depuração e manutenção. Algumas destas ferramentas decorreram de sistemas implementados, utilizando a experiência adquirida durante a sua construção (por ex: EMYCIN [BAR83, WAT83], cujo desenvolvimento foi uma consequência do sistema MYCIN).

As ferramentas de desenvolvimento de sistemas baseados em conhecimento podem ser classificadas em [BAR83, DIB87]:

a) linguagens de programação, sendo a mais utilizada para este fim LISP. Não é uma ferramenta muito apropriada, sendo necessário conhecer programação para poder desenvolver sistemas.

b) linguagens de representação de conhecimento, que são linguagens especialmente criadas para serem utilizadas em Engenharia de Conhecimentos. Entre elas podemos citar RLL [BAR83,WAT83], OPS5 [BAR83, GEV87] e PROLOG [STE86].

c) "shells", que são espécies de "cascas" que fornecem uma maneira padronizada para a representação do conhecimento, além de possuirem uma máquina de inferência própria. São mais simples e mais fáceis de utilizar do que as linguagens de representação de conhecimento, embora geralmente sejam projetadas para solucionar só um determinado tipo de problema. Nesta categoria se enquadram os sistemas EMYCIN, KAS e EXPERT [BAR83, WAT83].

d) sistemas híbridos, que integram diversas ferramentas em um único ambiente. Através destes sistemas se alcança maior flexibilidade e eficiência na geração de sistemas do que com os anteriores. A integração de diferentes ferramentas, entretan-

to, aumenta o grau de dificuldade de utilização da ferramenta. O sistema KEE [FIK85] se inclui nesta categoria.

e) "shells" de aplicação, que utilizam sistemas híbridos e possuem informações a respeito de objetos genéricos e de métodos de inferência capazes de resolver uma determinada classe de problemas.

Neste capítulo serão apresentados alguns exemplos de ferramentas para construção de sistemas baseados em conhecimento, procurando enfatizar como é representado o conhecimento em suas bases de conhecimento. Inicia-se por algumas linguagens de representação do conhecimento. Em seguida são vistas algumas linguagens de modelos semânticos de dados, que podem ser encaradas como uma classe de linguagens de representação de conhecimento. E, para finalizar, é feita a descrição resumida de alguns sistemas ("shells" e sistemas híbrido). Em [GEV87] podem ser encontradas descrições de outras ferramentas para desenvolvimento de sistemas especialistas.

5.1 LINGUAGENS DE REPRESENTAÇÃO DO CONHECIMENTO

As linguagens de representação do conhecimento são linguagens especialmente criadas para serem utilizadas na modelagem do conhecimento. A seguir serão vistas as principais características de algumas destas linguagens. As três primeiras modelam o conhecimento através de "frames", uma utiliza "frames" associados em uma rede semântica e a última, regras de produção.

5.1.1 SRL

SRL [FOX86a,b] é uma linguagem de representação de conhecimentos que utiliza a estrutura de "frames". Seu desenvolvimento iniciou em torno de 1977, tendo sido utilizada inclusive em aplicações industriais. A unidade primitiva de informação é o esquema, que consiste na representação simbólica de um conceito. Um esquema é definido por um nome e por um conjunto de "slots" e de valores. Os "slots" são utilizados para representar atributos e informações estruturais e relacionais. Os valores podem ser expressos por qualquer expressão LISP e por esquemas de referência quando estes são definidos por "strings". O exemplo a seguir foi retirado de [FOX86b]:

```
(( h1-spec
   IS-A:"engineering-activity"
   SUB-ACTIVITY-OF:"develop-board-h1"
   INITIAL-ACTIVITY-OF:"develop-board-h1"
   ENABLED-BY:"TRUE"
   CAUSE:"h1-spec-complete"
   DESCRIPTION:"Develop specification for the cpu board"))
```

SRL incorpora conceitos de metainformação, "demons", restrições e valores legais para "slots" e facilidades de contexto. Metainformações podem ser associadas a um esquema, a um "slot" ou a um valor através de outro esquema, chamado metaesquema, associado a este esquema, "frame" ou valor.

Podem ser associadas facetas a "slots". Existem três tipos de facetas definidas: DEMON, DOMAIN, RANGE e CARDINALITY. Além disso, possui um conjunto de relações padronizadas para formar as taxonomias e hierarquias. Um aspecto importante é que podem ser definidas relações de herança pelo usuário, definindo assim suas próprias regras de hierarquia.

Para suportar aplicações com grande número de dados, SRL permite a integração a um banco de dados convencional.

Várias extensões desta linguagem foram implementadas. O ambiente de engenharia de conhecimento ISLISP, que é baseado em SRL, possui diversas ferramentas de inferência, que operam sobre os esquemas: (1) HSRL, que utiliza um interpretador lógico que aceita modelos em SRL como axiomas, resultando em uma representação de "frames" associada com lógica; (2) PSRL, um interpretador de regras de produção que opera os esquemas; (3) OSRL, que possibilita programação orientada a objetos baseada nos esquemas; (4) ESRL, que oferece um mecanismo que manipula eventos; e (5) KBS, que consiste em um sistema de simulação baseado em conhecimento. Além destas, este ambiente apresenta também ferramentas para construção de sistemas: (1) RETINAS, que é um sistema baseado em janelas, operando os esquemas; (2) KBCI é um sistema baseado em comandos; e (3) CPAK, que é um pacote gráfico.

Vários sistema especialistas foram construídos através da utilização desta linguagem. Entre êles podemos citar:

a) CALLISTO [FOX86a,b] é um sistema especialista para gerência de grandes projetos, controlando atividades, a configuração de

produtos e recursos. Proporciona suporte à decisão e facilidades para tomadas de decisão. A ênfase principal deste sistema é na representação semântica de produtos e de atividades. Através da linguagem SRL são representados não só os produtos e as atividades, mas também metainformações, relações definidas por usuários e especificações de buscas. Utiliza PSRL para representar heurísticas de gerência de projetos e ESRL para "scheduling" de projetos. Está em desenvolvimento um sistema CALLISTO distribuído, o qual apresentará uma base de conhecimentos distribuída.

b) ISIS [FOX86a,b] é um sistema especialista que tem por objetivo fornecer suporte inteligente para gerência e controle de produção, modelando, selecionando, sequenciando, dividindo recursos e tempo de execução, e monitorando atividades. Uma das aplicações feitas através de ISIS foi na produção de lâminas de turbinas a vapor ("steam turbine blades"). Esta aplicação possui uma planta com 30.000 partes, 200 máquinas e aproximadamente 800 ordens ativas a cada instante. A base de conhecimentos contém informações sobre a operação e a produção de cada uma destas partes, que deve ser representada por um esquema em SRL separado. A base de conhecimentos aceita acesso paralelo, através de terminais.

c) PDS [FOX86b] é um sistema especialista que tem por objetivo fazer diagnósticos baseados em sensores de processos físicos.

5.1.2 RLL

RLL (Representation-Language Language) [BAR83, WAT83] é um conjunto estruturado de ferramentas através das quais um engenheiro do conhecimento pode construir, utilizar e modificar sistemas especialistas. A base de conhecimentos do sistema RLL contém, além de suas rotinas particulares, informações a respeito de programação em geral.

O sistema contém um conjunto de estruturas primitivas, tais como tipos de "slots", mecanismos de controle, esquemas de herança e métodos para associar ações que permitem acessar fatos particulares. Estas estruturas são armazenadas em uma biblioteca, estando disponíveis ferramentas para acessá-las, modificá-las e combiná-las. Além disso, podem ser definidas novas estruturas primitivas. O engenheiro do conhecimento tem grande liberdade pa-

ra definir tipos de controle e formas de dados.

A representação do conhecimento é baseada em "frames", sendo cada "frame" composto de "slots", onde estão definidos os atributos. Todas as informações são definidas através de "frames", sejam elas entidades físicas ou abstratas, tipos de entidades, regras, etc.

O desenvolvimento de RLL se deu com o objetivo de construir uma linguagem de representação para o sistema EURISKO [LEN83]. Aspectos adicionais foram desenvolvidos, resultando em uma linguagem de uso geral. Como exemplo de sistema implementado através de RLL podemos citar o sistema WHEZZE [SMI85]. Seu domínio de aplicação são diagnósticos de funções pulmonares.

5.1.3 KRL

A linguagem de representação de conhecimentos KRL (Knowledge Representation Language) [MYL84] é uma linguagem muito sofisticada e versátil, que suporta valores numéricos e "strings", incluindo fatos, números, categorias e frases. As unidades básicas de representação do conhecimento são estruturas complexas semelhantes a "frames". Através dos "slots" podem ser definidos diversos tipos diferentes de informações, inclusive procedimentos. Possui suporte para contextos de crenças. Aceita metainformações a respeito das descrições.

O sistema especialista ESP ADVISOR [GEV87] utiliza esta linguagem de representação. Este sistema, baseado em PROLOG, serve para projetar e construir protótipos de sistemas especialistas que guiem usuários na execução de uma operação detalhada.

5.1.4 KL-ONE

A linguagem de representação do conhecimento KL-ONE [MOR86, MYL84, SCH82, WOO86] é uma linguagem centrada em objetos. Originou-se nas redes semânticas tradicionais. Suas estruturas principais, entretanto, não se baseiam em conjuntos ou proposições (como nas redes semânticas tradicionais), mas em um objeto conceitual estruturado, denominado conceito. O modelo de representação do conhecimento é, portanto, uma combinação de "frames" e de redes semânticas.

Apresenta mecanismos de herança múltipla. Uma rede em KL-ONE não é a representação de todo um domínio, mas somente a visão que um indivíduo tem deste domínio.

KL-ONE apresenta duas sublinguagens: (1) a linguagem de descrição, através da qual são descritos os termos que vão formar o vocabuário conceitual; e (2) a linguagem de asserções, que utiliza o vocabulário anteriormente definido, através da qual são feitas declarações sobre o domínio.

O elemento principal de uma descrição nesta linguagem é, portanto, o conceito - correspondendo aos nodos da rede. Um conceito corresponde a um objeto ou a uma entidade do modelo. É estruturado como um "frame", apresentando "slots" aos quais se pode associar diversas tipos de informações ("defaults", modalidades, valores, restrições, informações procedimentais, etc). Existem dois tipos principais de conceitos: (1) conceitos genéricos, através dos quais diversos indivíduos de diferentes domínios podem ser descritos; e (2) conceitos individuais, que descreve um só indivíduo, em um contexto particular.

Conceitos genéricos são organizados em taxonomias, segundo seus relacionamentos. Estas taxonomias apresentam mecanismos de herança multipla. Uma determinada taxonomia apresenta sempre um conceito raiz, do qual todos os outros conceitos são descendentes. A ligação de um conceito com um conceito mais geral é feita através da relação SUPERC. O conceito mais geral é denominado superconceito.

A estrutura interna de um conceito é expressa em termos de: (1) um conjunto de funções ("roles"), as quais descrevem relacionamentos potenciais entre instâncias do conceito e outros conceitos intimamente associados a êle (tais como suas propriedades, partes, etc.); e (2) um conjunto de condições estruturais, que representam os inter-relacionamentos entre estas funções. As funções e condições estruturais podem ser associadas diretamente a um conceito, ou serem herdadas de conceitos mais gerais.

A definição de um conceito genérico deve apresentar, portanto: (1) sua localização na taxonomia, através da definição de seus superconceitos; e (2) sua estrutura interna.

Na figura 5.1, tirada de [WOOD86], é apresentada a representação gráfica utilizada para definições em KL-ONE. As elip-

ses representam os conceitos; uma função é representada por um quadrado dentro de um círculo. O rótulo V/R significa "value-restriction" - representando que este relacionamento poderia ser preenchido ou por um valor, ou por uma restrição.

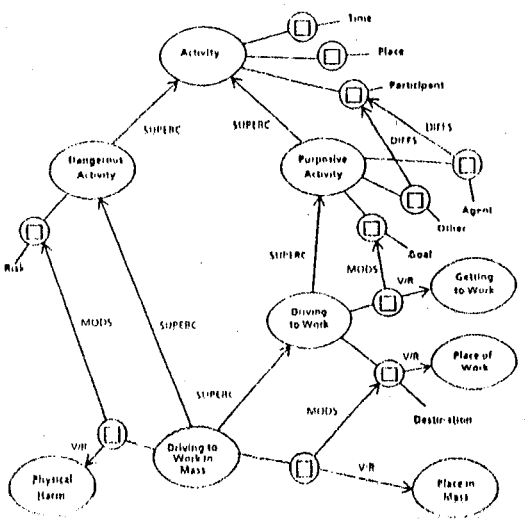

Figura 5.1 - REPRESENTAÇÃO GRÁFICA DE UMA DEFINIÇÃO DE KL-ONE

Um dos principais aspectos de KL-ONE é que proporciona uma classificação automática de um novo conceito na rede. Esta classificação é efetuada com base no tipo do conceito, no tipo dos conceitos que definem suas funções, etc.

As estruturas de conceitos e funções são similares à estrutura de "frames-slots". As principais diferenças são [WOO86]: (1) a definição e utilização da taxonomia de conceitos gerais; (2) a presença de condições estruturais associadas a um conceito; (3) os relacionamentos explícitos entre funções em níveis diferentes de generalização; e (4) o objetivo básico de KL-ONE de modelar a semântica e a estrutura conceitual de um espaço abstrato de conceitos, não representando simplesmente uma estrutura de dados a ser implementada.

Em [SCH82] são citadas duas implementações de KL-ONE: (1) INTERLISP KL-ONE, em Interlisp; e (2) KLONETALK, em Smalltalk. O sistema KRYPTON [LEV86] se baseia em KL-ONE.

5.1.5 OPS5

OPS5 [BAR83, GEV87] é uma linguagem de desenvolvimento de sistemas especialistas da família OPS, que possui várias versões. Incorpora mecanismos de controle e de representação. É uma linguagem baseada em regras de produção. A representação do conhecimento é feita em um banco de dados denominado de memória de trabalho, o qual consiste de um conjunto de estruturas de símbolos constantes. Podem ser utilizados dois tipos de estruturas de símbolos: vetores de símbolos e objetos com atributos. Os elementos podem variar de tamanho durante a execução e os objeto podem ganhar ou perder atributos.

As regras da linguagem são capazes de processar estas estruturas de dados. Se apresentam sob a forma:

(P <nome_da_regra> <antecedente> -> <consequente>)

sendo o antecedente formado por uma ou mais condições e o consequente por uma ou mais ações. Tanto nas condições como na especificação de uma ação podem aparecer referências a variáveis, números e símbolos.

Permite ao programador a utilização de símbolos cujo significado será definido por regras fornecidas pelo próprio programador.

O interpretador da linguagem OPS5 fornece ao programador um ambiente de programação interativo convencional, semelhante ao do interpretador LISP. Não apresenta facilidades para representação de objetos sofisticados, mas apresenta capacidade de depuração. Possui um esquema de indexação sofisticado (algoritmo Rete), que permite localizar regras de uma maneira muito rápida. Uma de suas principais desvantagens é não ser muito fácil de utilizar por pessoas não especialistas em programação.

A primeira utilização desta linguagem na área de engenharia do conhecimento foi na construção do sistema R1 [FOX86a, MCD82], posteriormente denominado de XCON [WIN84]. R1 é um sistema especialista destinado a configurar sistemas de computadores. Recebe como entrada uma lista ordenada de componentes e determina quais as modificações que devem ser efetuadas na ordem de componentes fornecida, de modo a incrementar a funcionalidade do sistema. Fornece ainda, como saída, um conjunto de diagramas que

apresentam os inter-relacionamentos entre os componentes. A base de conhecimentos do R1 é composta de aproximadamente 4000 regras de produção, escritas em OPS5. Um subconjunto destas regras implementa um interface para um gerenciador de banco de dados, no qual estão detalhes dos objetos e de seus atributos. Estas informações são utilizadas no processo de configuração. O banco de dados possui aproximadamente 9000 objetos descritos, possuindo cada um de 25 a 125 atributos.

5.2 LINGUAGENS DE MODELOS SEMÂNTICOS DE DADOS

Modelos semânticos de dados podem ser comparados aos modelos de representação de conhecimento através de redes semânticas. Podem ser considerados como uma primeira tentativa de importar conceitos de Inteligência Artificial para Bancos de Dados [NIX87]. Existem algumas implementações destes modelos de dados em bancos de dados relacionais (GEM [TSU84], IRIS [LYN87]). Vamos citar as características de alguns destes modelos, através das linguagens que os representam. Pode-se observar que, nas linguagens apresentadas, todas utilizam conceito de objetos organizados em hierarquias. Estes conceitos podem ser modelados através da associação de "frames" e de redes semânticas.

5.2.1 TAXIS

TAXIS [MYL80, NIX87] é uma linguagem utilizada para a construção interativa de Sistemas de Informação. Possui: (1) um compilador de especificações em TAXIS para código PASCAL [NIX87]; e (2) um ambiente de desenvolvimento de programas TAXIS, que utiliza a linguagem LISP.

Utiliza o conceito de redes semânticas para a modelagem de dados e de procedimentos, associado ao conceito de bancos de dados relacionais. Seu objetivo é construir o esquema conceitual de um banco de dados, através de um modelo de dados semântico.

É uma linguagem orientada a objetos, suportando mecanismos de abstração com conceitos de generalização, classificação e agregação. Apresenta herança múltipla de atributos, hierarquias "é_um" de transações, metaclasses, atributos tipados e mecanismos de manipulação de exceções.

O conceito fundamental do modelo de dados é a entidade. Entidades podem ser criadas e destruídas; tem identificador único, sendo diferentes de qualquer entidade passada, presente ou futura. Coleções de entidades que apresentam propriedades comuns são definidas como classes. Coleções de classes, por sua vez, são definidas como metaclasses. Todos os elementos da linguagem são encarados como entidades - constantes, "strings", exceções e classes. A definição de uma classe apresenta atributos, que são agrupados em categorias: atributos fixos, variáveis, computados e únicos.

A definição de transações também são tratadas como descrições de classes. Invocações de transações são instâncias destas classes. As transações também apresentam atributos, agrupados em categorias tais como parâmetros, pré-requisitos, ações, pós-requisitos e expressões de retorno. Não são permitidos atributos multivalorados.

As transações são organizadas em hierarquias de especialização. Estas podem ser representadas através de grafos acíclicos, apresentando conceitos de herança múltipla de atributos. Para evitar a ambiguidade de interpretação, todo atributo herdado através de herança múltipla deve ser declarado em uma única classe, mais geral.

A seguir são apresentadas a definição de uma classe e a definição de uma transação, definições estas tiradas do exemplo de [NIX87]:

```
define AnyDataClass Person with
     unchangeable
          name: AnyString
          SIN: SINValue
     changeable
          addr:Address
          age: (| 0::120 |)
     unique
          personID: (SIN)
endAnyDataClass
```

```
define AnyTransactionClass ReviewSalary
        (e: Employee, incr:Money) with
    prerequisits
        moneyAvailable ?:
            (e.dept.SalTotal + incr <= e. dept.maxSalTotal)
            elseRaiseException NoMoneyAvailable(e,icr)
    actions
        changeSal: e.sal <- e.sal + incr
        report: (e.sal).print
        changeTotal:
            e.dept.salTotal <- e.dept.salTotal + incr
    postrequisites
        toohigh ?: (e.sal > 75000)
        elseRaiseException SalaryTooHigh(e, incr)
endAnyTransactionClass
```

Os conceitos introduzidos na definição da linguagem TAXIS influiram em muitos outros modelos semânticos de dados, tais como IRIS [LYN87], apresentado a seguir.

5.2.2 IRIS

O modelo semântico de dados do sistema IRIS [FIS87, LYN86] suporta abstrações estruturais de alto nível, tais como classificação, generalização/especialização e agregação, bem como abstrações comportamentais. É um modelo orientado a objetos. Baseia-se em três construtores básicos: (1) objetos, os quais representam entidades físicas e conceitos do domínio da aplicação; (2) tipos, que são conjuntos de objetos que apresentam propriedades comuns; e (3) funções, que definem propriedades dos objetos. Os tipos e as funções também são representados como objetos. Os tipos são organizados em estruturas que suportam generalização e especialização, e herança múltipla.

Uma estrutura de tipos pode ser definida através de um grafo dirigido acíclico, associado a um conjunto de restrições. Em [LYN87] é apresentada uma representação gráfica para estes esquemas.

Em [LYN87] é apresentada também a implementação deste modelo de dados através da utilização de técnicas de bancos de dados relacionais.

5.2.3 GALILEO

A linguagem GALILEO [ALB85] se baseia em uma linguagem tradicional (ML), na qual foram incluindas características de modelagem de dados semânticos. O desenvolvimento de GALILEO é parte integrante do sistema DIÁLOGO, da Universidade de Pisa, Itália.

Suporta os seguintes mecanismos de abstração: classificação, agregação, generalização e modularização. Apresenta tipos, representando conjuntos de valores associados a conjuntos de operações primitivas sobre estes valores. Os tipos podem ser pré-definidos ou construídos. Estes últimos são constituídos por tuplas, sequências, valores alteráveis, funções, tipos abstratos e união de tipos. Os tipos podem ser organizados hierarquicamente, através do conceito de subtipo, com herança de atributos. Além dos tipos, em GALILEO podem ser definidas classes, caracterizadas por um nome e pelo tipo de seus elementos. As classes podem apresentar hierarquias "é_um", definindo subclasses, com herança de atributos.

As ações são representadas através de transações, expressas através de expressões de alto nível. As transações podem ser simples ou compostas. Estas últimas são compostas por transações simples. A linguagem apresenta, ainda, um tratamento para falhas.

A linguagem GALILEO, projetada para suportar características de dados semânticos, apresenta também características de orientação a objetos.

5.2.4 DAPLEX/PDM

O modelo semântico do DAPLEX [SHI81] é representado através de um esquema que apresenta tipos de entidades, funções (relacionamentos) entre estes tipos e taxonomias (hierarquias de generalização/especialização) entre tipos de entidades. Uma entidade representa um objeto do mundo real. Este modelo pode ser comparado a uma rede semântica onde os nodos são os tipos de entidades e as funções são os relacionamentos entre os nodos.

PDM (Probe Data Model) [MAN86], modelo de dados do sistema PROBE (um sistema gerenciador de banco de dados orientado ao conhecimento), é uma extensão da linguagem DAPLEX, com objetivo

de fornecer mecanismos gerais para suprir as necessidades das aplicações não convencionais. PDM também apresenta entidades, funções e tipos de entidades. Podem ser criadas hierarquias de generalização/especialização, com herança de funções entre os tipos. Além disso, novos conceitos são adicionados para representar os dados não-convencionais.

5.3 SISTEMAS DE DESENVOLVIMENTO

A seguir são apresentadas as principais características de alguns sistemas que podem ser utilizados no desenvolvimento de sistemas baseados em conhecimento, em especial de sistemas especialistas. Os sistemas apresentados variam desde simples "esqueletos" para preenchimento do conhecimento de um domínio específico, até ambientes completos envolvendo metodologias complexas, como é o caso de KADS. O objetivo é dar uma idéia da variedade de ferramentas que estão sendo desenvolvidas nesta área.

5.3.1 EMYCIN

MYCIN [CLA83, DAV77] é um sistema especialista baseado em regras. É usado para diagnósticos em doenças infecciosas (para consulta e seleção de antibióticos na terapia de bacterimia). EMYCIN [BAR83, WAT83] evoluiu a partir do MYCIN, sendo entretanto, independente do domínio de aplicação. Pode ser utilizado para solução de quaisquer problemas dedutivos que envolvam diagnósticos. EMYCIN não é uma linguagem de programação, mas uma espécie de "esqueleto" que pode ser utilizado para a construção de sistemas de diagnóstico baseados em regras. Apresenta um ambiente próprio para o desenvolvimento sistemas especialistas que forneçam diagnósticos.

EMYCYN utiliza regras de produção para a representação do conhecimento. Toda regra possui uma premissa e uma ação. A premissa é uma condição lógica, apresentando sempre os seguintes quatro componentes:

<função_predicado> <objeto> <atributo> <valor>

As funções_predicados são IGUAL, CONHECIDO, DEFINIDO, etc., e os atributos IDENTIDADE, TAMANHO, etc. O sistema foi implementado em LISP.

As entidades de um domínio são organizadas em hierarquias, através de árvores de contexto, apresentando algumas das características de herança dos mecanismos de "frames". As regras do EMYCIN podem ser encaradas como representações de redes semânticas. Esta rede apresenta elos sem nomes. Os tipos de relacionamentos não são rotulados.

O conhecimento estratégico é representado através de meta-regras, independentes do domínio. Estratégias de resolução de problemas podem ser incorporadas ao sistema através de regras de auto-referência. Numa regra deste tipo, o objetivo concluído pela ação também aparece na premissa. Esta regra somente será aplicada quando todas as regras que não contém auto-referências foram aplicadas.

O sistema EMYCIN permite tratamento de incertezas, incluindo um fator de certeza do diagnóstico nas regras, fator este que varia de -1 a 1. Quando a premissa é conhecida, mas não com certeza, o fator de certeza do diagnóstico é modificado.

EMYCIN apresenta facilidades de explicação - pode explicar o raciocínio desenvolvido ao usuário. Isto permite a depuração da base de conhecimentos.

Possui uma ferramenta de aquisição de conhecimento que permite uma muito rápida construção da base de conhecimentos, denominada TEIRESIAS [DAV79]. As regras podem ser fornecidas em uma linguagem bastante próxima da natural. TEIRESIAS se enquadra na classe de ferramentas de auxílio à transferência de conhecimentos de um especialista para um programa, não apresentando características de aprendizagem (de derivação de novos conhecimentos a partir dos já adquiridos). A transferência do conhecimento é feita através de um diálogo amigável do sistema com o especialista. Entretanto, ao mesmo tempo em que possibilita esta transferência de conhecimentos, TEIRESIAS é capaz de detectar erros no conhecimento coletado, permitindo que este seja corrigido pelo especialista. Executa, portanto, simultaneamente a aquisição e a depuração (localização e caracterização de erros) do conhecimento. A capacidade de depuração do TEIRESIAS é devido ao seu conhecimento do domínio da aplicação. O programa TEIRESIAS foi codificado em INTERLISP.

Assim como o EMYCIN, outros sistemas gerais evoluiram do desenvolvimento de sistemas especialistas específicos. É o ca-

so, por exemplo, do sistema KAS [BAR83, WAT83], que se originou no sistema PROSPECTOR. PROSPECTOR é um sistema especialista que tem por objetivo solucionar problemas de geologia econômica - aconselhamento sobre a localização de jazidas de ouro. KAS, por sua vez, é independente do domínio da aplicação.

5.3.2 INTEXP

INTEXP (INTeligent EXPert) [GEO85b] é um sistema especialista independente do domínio de aplicação. Sua base de conhecimentos armazena conhecimentos declarativos e regras. A unidade de conhecimento armazenado é denominada descrição. Uma descrição define uma entidade do domínio. As descrições são organizadas em dicionários, de acordo com categorias padronizadas de entidades: objetos, procedimentos e regras. A forma de representação é similar a um "frame". Para cada entidade definida devem ser fornecidas, entre outras, as seguintes informações: (1) características gerais, entre as quais o nome, sua categoria (objeto/procedimento), sua classe (encadeamento taxonômico da estrutura), e seu tipo, que identifica o conjunto de operadores que podem atuar nesta entidade; (2) a estrutura externa, definido as relações que associam esta entidade a outras entidades, e que inclui a definição de configurações válidas; (3) a estrutura interna, definindo relacionamentos entre os componentes da entidade; e (4) outras propriedades particulares desta entidade.

Este sistema foi implementado em DM-LISP. Possui mecanismo de explicações, possibilitando o acompanhamento do raciocínio efetuado pelo sistema.

5.3.3 XPLAIN

XPLAIN [SWA83] se destina a construção de sistemas especialista para consultas. O sistema gerado terá capacidades de explicações, sendo o conhecimento necessário a estas explicações captado durante a construção do sistema.

Possui um programador automático que cria o sistema. O programador automático busca e representa, simultaneamente, dois tipos de conhecimentos: (1) conhecimentos do domínio, traduzidos em fatos descritivos, tais como relacionamentos e hierarquias de classificação; e (2) conhecimentos a respeito de princípios do

domínio, compreendendo heurísticas e métodos, conhecimentos estes necessários ao processo de explicações.

A linguagem de representação de conhecimentos utilizada por XPLAIN se denomina XLMS (eXperimental Linguistic Memory System). Trata-se de uma extensão de LISP que permite utilizar nomes estruturados. O conhecimento é modelado através de "frames" e de "slots", podendo ser organizado em hierarquias.

5.3.4 LOOPS

LOOPS [CHA87] é um ambiente baseado em conhecimento, para desenvolvimento de programas orientados a objetos, que executa em máquinas LISP.

A linguagem LOOPS possui três tipos de objetos: (1) instâncias, semelhantes a objetos de dados do LISP; (2) classes, que definem conjuntos de instâncias; e (3) metaclasses, cujas instâncias são classes.

As classes contém informações a respeito de variáveis de instâncias (variáveis locais armazenadas junto a cada instância de uma classe), de classes de instâncias (variáveis armazenadas junto a cada objeto de classe, acessíveis a partir de cada instância da classe) e de métodos (procedimentos que são utilizados para executar operações de instâncias de classes). Além disso, cada classe possui uma lista de superclasses desta classe, possibilitando mecanismos de herança de instâncias de variáveis de classes de variáveis e de métodos.

Um método é invocado quando é enviada uma mensagem a um objeto.

O ambiente LOOPS possui uma função DefRSM (Define Rule Set as a Method), para facilitar a definição de regras a serem utilizadas como método. Esta função é utilizada no sistema de manipulação de mensagens baseado em conhecimento, apresentado no próximo capítulo.

5.3.5 KRISYS

KRISYS (Knowledge Representation and Inference SYStem) [MAT89] é um sistema para o gerenciamento de bases de conhecimento grandes e compartilhadas. Tem por objetivo apoiar os três níveis da arquitetura de um sistema baseado em conhecimento: de aplicação, de engenharia e de implementação.

O nível de aplicação é atendida pelo sistema KOALA, o qual é responsável pelo interface de aplicação do sistema, manipulado através da linguagem KOALA (KRISYS Object Abstraction LAnguage). Através da utilização deste interface, o usuário tem uma visão abstrata e funcional da base de conhecimentos. A linguagem apresenta duas operações: "TELL", utilizada para fornecer informações à base de conhecimentos; e "ASK", através da qual são colocadas perguntas ao sistema.

O nível de engenharia corresponde ao sistema ORES (Object-centered REpresentation System). O engenheiro de conhecimentos interage com o sistema através deste módulo.

O modelo de representação do conhecimento do KRISYS, utilizado no nível de engenharia, se denomina KOBRA (KRISYS OBject-centered RepresentAtion Model). É definido por uma combinação de várias técnicas de representação de conhecimento e de modelagem de dados, sendo uma representação orientada a objetos.

O conceito básico é o esquema, que é a representação simbólica de uma entidade do mundo real. Um esquema é definido por um nome (único no sistema) e por um conjunto de atributos que descrevem as propriedades desta entidade. Os atributos podem ser de dois tipos: "slots" utilizados para descrever os aspectos descritivos, e métodos, através dos quais são definidos os aspectos operacionais. As ações são representadas através de trocas de mensagens entre objetos. "Slots" e métodos apresentam a mesma estrutura: nome, valor, tipo e nome do esquema a que pertencem. Relacionamentos entre objetos, apresentados em "slots" pré-definidos, definem as abstrações, que podem ser de generalização, classificação, associação e agregação. Através destas abstrações podem ser criadas hierarquias entre objetos, com herança múltipla de atributos.

Além disso, regras podem ser declaradas como esquemas, possibilitando capacidade de raciocínio através desta técnica.

5.3.6 EXPERT

EXPERT [BAR83, WAT83] é um sistema de desenvolvimento de modelos de consulta. Os problemas que mais se adaptam a este sistema são aqueles baseados em classificação, isto é, quando existe uma lista pré-definida de conclusões potenciais, a partir da qual o programa escolhe a resposta adequada. Inicialmente era utilizado para resolver problemas na área de diagnósticos médicos sendo depois estendido para outras áreas.

O modelo de representação do conhecimento no EXPERT consiste de hipóteses, descobertas ("findings") e regras de decisão. As hipóteses são as conclusões a que o sistema chega. São organizadas em taxonomias de classificação, apresentando as categorias de diagnósticos e prognósticos. As descobertas são fatos ou observações elicitados durante consulta efetuadas pelo sistema ao usuário, podendo ser verdadeiras, falsas, numéricas ou não-disponíveis. As regras de decisão descrevem relacionamentos lógicos entre as descobertas e as hipóteses. Podem ser definidos três tipos de regras: (1) regras FF, que relacionam descobertas; (2) regras FH, relacionando descoberta com hipótese; e (3) regras HH, entre hipóteses. O sistema gerado apresenta interfaces amigáveis para interação com usuários.

O desenvolvimento de protótipos de sistemas através de EXPERT é bastante fácil e rápido. EXPERT foi desenvolvido em FORTRAN, apresentando características de portabilidade entre equipamentos que utilizam esta linguagem. A criação de um sistema através desta ferramenta é bastante similar à escrita e à execução de um programa de computação. Foram realizados diversos trabalhos de integração de modelos EXPERT com bancos de dados.

5.3.7 KEE

O sistema KEE [FIK85, GEV87] é um ambiente muito utilizado para a construção de sistemas especialistas sofisticados. Constitui-se de um conjunto bastante grande de ferramentas de engenharia de conhecimento, utilizando linguagens de representação do conhecimento baseadas em regras, "frames", descrições e/ou procedimentos. Apresenta diversos interfaces, os quais incorporam janelas, "menus" e gráficos. O ambiente possui ainda um compilador de regras e um depurador.

KEE utiliza uma linguagem de representação do conhecimento baseada em "frames", aos quais podem ser associados procedimentos. Um "frame", aqui denominado de unidade, descreve um objeto ou uma classe de objetos do domínio da aplicação. Os "frames" podem ser organizados em taxonomias de generalizaçaõ/especialização, através de relacionamentos entre "frames" ("member links" e "subclass links"). Existem dois tipos de "slots", os "own slots" e "member slots". "Own slots" podem ocorrer em qualquer "frame" sendo utilizados para descrever atributos do objeto ou da classe de objetos representado pelo "frame". Já os "member slots", estes só podem ocorrer em "frames" que descrevem classes, sendo utilizados para descrever atributos de cada membro da classe. Um "slot" pode ter mais de um valor. O número de valores de um atributo representado por um "slot" pode ser limitado através da utilização de duas facetas disponíveis, que definem a cardinalidade máxima e mínima do "slot".

O sistema permite a definição de procedimentos associados aos "frames". Isto pode ser feito através de duas maneiras: através da definição de métodos e de valores ativos. Os métodos são associados aos "frames", armazenados como valores de "slots" identificados como "message responders". Consistem de procedimentos escritos em LISP que são executados em consequência de mensagens enviadas ao "frame". Os valores ativos são procedimentos ou coleções de regras de produção associados a um "slot" e executados a cada vez que o "slot" é acessado (atuam como "demons").

O sistema KEE apresenta a possibilidade de utilização através de regras de produção. Internamente, cada regra é representada como um "frame". É utilizada uma linguagem de lógica de predicados simples para a representação da regra. Os predicados desta linguagem representam os relacionamentos que podem ser representados na linguagem dos "frames". Também podem ser utilizadas, como predicados, quaisquer funções LISP. KEE foi implementado em COMMON-LISP.

5.3.8 SYSTEM X-I

O SYSTEM X-I [LEU90] é um "shell" para construção de sistemas especialistas, que utiliza conhecimento estruturado.

A arquitetura do SYSTEM X-I é constituída de: (1) um módulo de aquisição de conhecimento bastante flexível; (2) uma

máquina de inferências; (3) um interface para extração automática de dados de um banco de dados, que também pode ser utilizado para a aquisição de conhecimentos, principalmente em se tratando de grande número de dados; (4) facilidades de recuperação de conhecimento difuso; (5) um interface para métodos externos.

O sistema possui suporte para dois modos diferentes de aquisição de conhecimentos: (1) editores tradicionais, para serem utilizados por engenheiros de conhecimento que estejam familiarizados com o sistema e com sua representação de conhecimentos; e (2) aquisição interativa, para quando não houver esta familiaridade com a representação do conhecimento.

A base de conhecimentos de um sistema especialista construído através do SYSTEM X-I é constituída de uma coleção de classes, objetos e métodos. Os conhecimentos são estruturados sob forma de regras e de procedimentos, apresentando características de herança de métodos (relações "é_um" e "tem_um") entre classes e superclasses. O enfoque de orientação a objetos da representação de conhecimentos suporta abstração de dados e de conhecimentos.

Este sistema proporciona a integração de especialistas, procedimentos e bancos de dados, aspectos relevantes nas aplicações atuais.

5.3.9 KADS

O projeto KADS [WIE89], um projeto ESPRIT em desenvolvimento na Universidade de Amsterdam, tem por objetivo o desenvolvimento de uma metodologia para ser utilizada no desenvolvimento de sistemas baseados em conhecimento. Esta metodologia deverá apresentar um conjunto de ferramentas que apoiem todo o ciclo de desenvolvimento de um sistema, desde a análise inicial até as fases de teste e manutenção.

Em KADS, um sistema baseado em conhecimento não é encarado como um simples depósito de conhecimentos extraídos de um especialista, mas como um modelo operacional que apresenta um comportamento específico. Deste modo, o processo de desenvolvimento de um sistema deve considerar não somente a modelagem do conhecimento obtido do especialista, mas também de outros aspectos relacionados com o mundo real, tais como estrutura e estraté-

gia de organização, requisitos do usuário e cooperação entre usuário e sistema, grau de perícia e ambiente tecnológico.

A modelagem inicial em KADS é efetuada no nível de conhecimento, sem levar em conta a posterior implementação. A independência dos detalhes técnicos do sistema podem levar à utilização de modelos genéricos para determinadas classes de tarefas.

Esta modelagem é efetuada "top-down", em três níveis consecutivos, respectivamente: (1) nível de tarefa, onde é realizada a análise das tarefas que o usuário e o sistema irão desempenhar, resultando em um modelo de tarefas; (2) nível semântico, onde um modelo de processamento do futuro sistema é obtido, apresentando os objetos semânticos, os procedimentos semânticos e o controle que deverá ser efetuado sobre eles; e (3) nível de comunicação, onde são especificados os interfaces de comunicação entre o usuário e o sistema.

A modelagem do conhecimento se baseia no papel que este desempenha no processo de raciocínio. Quatro categorias de conhecimento foram identificadas, categorias estas que podem ser encaradas como camadas, uma vez que cada camada interpreta a descrição da camada inferior:

a) camada de domínio, que corresponde ao conhecimento estático, englobando a teoria declarativa do domínio. É feita a descrição de conceitos e seus atributos, de relações hierárquicas entre conceitos e das regras do domínio. As descrições são geralmente feitas através de alguma linguagem estruturada e as relações hierárquicas, sob forma de grafos.

b) camada de inferência, que apresenta a sequência canônica de inferências necessárias à solução do problema. É composta de metaclasses que descrevem o papel que os conceitos do domínio podem desenpenhar em um processo de reciocínio, e de fontes de conhecimento, que são descrições funcionais de alguma inferência feita sobre metaclasses. As descrições nesta camada apresentam tipicamente duas partes: (1) a estrutura da inferência, que geralmente é uma versão adaptada de um dos modelos de interpretação contidos na biblioteca; e (2) a descrição das fontes de conhecimento, descritas geralmente sob forma de "frames".

c) camada da tarefa, que descreve como fontes de conhecimento podem ser combinadas para a obtenção de determinados objetivos. É composta de objetivos e de tarefas. É descrita geralmente sob forma de procedimentos em pseudo-código, nos quais os comandos se referem ou às fontes de conhecimento, ou a ações de interfaces externas.

d) camada de estratégia, que determina quais os objetivos que são relevantes para a solução de um determinado problema. Estes objetivos são fixados através de planos e de metaregras.

A fase de projeto deverá se basear no resultado desta fase preliminar de análise. Deverá ser procurada alguma ferramenta para mapear, da maneira a mais fiel possível, os conceitos definidos durante a análise. Nos experimentos apresentados em [WIE89], por exemplo, foi utilizada a linguagem KL-ONE (ou uma de suas derivadas) para representar os elementos do nível do domínio.

6. FERRAMENTAS BASEADAS EM CONHECIMENTO, UTILIZADAS PARA MODELAGEM DE SISTEMAS DE INFORMAÇÃO DE ESCRITÓRIOS

Sistemas de Informação de Escritórios (SIE) são tipos particulares de Sistemas de Informação. São sistemas sócio-técnicos, onde um número elevado de pessoas executa tarefas definidas, com auxílio de computadores. Geralmente é constituído de um conjunto de estações de trabalho e de estações funcionalmente especializadas (tais como servidores de arquivos e de impressoras), interconectados em uma rede de comunicação de dados. As mensagens que são passadas através desta rede podem ser compostas de textos, dados estruturados, imagem e voz, sendo usualmente encaradas como objetos multimidia.

Em [KAY87] é apresentada uma taxonomia para o conhecimento de escritórios: (1) conhecimento estrutural, relativo aos objetos que circulam no escritório; (2) conhecimento procedimental, constituído de informações a respeito de procedimentos e estratégias do escritório; e (3) modelos de usuários, que são conhecimentos relativos a funcionários de um escritório particular.

O conhecimento estrutural ainda pode ser subdividido em: (1) conhecimento da estrutura organizacional, a respeito de projetos, departamentos e posições dentro deles; (2) conhecimento da estrutura de pessoal, constituindo informações a respeito de membros individuais das equipes de trabalho; e (3) conhecimento da estrutura dos recursos, relativo a recursos não-humanos, tais como formulários, sistemas de correio (eletrônico e papel) e bancos de dados.

Atividades de escritórios e procedimentos administrativos requerem não somente a representação declarativa de conhecimentos, mas também a representação da sequência de execução das atividades (conhecimento procedimental).

Diversas ferramentas baseadas em conhecimento têm sido desenvolvidas para SIEs, apresentando diferentes formas de representação de conhecimento. Apresentaremos a seguir algumas destas ferramentas, indicando seus objetivos principais e dando uma idéia de como é feita a representação do conhecimento em suas bases de conhecimento. As ferramentas apresentadas variam de simples sistemas de apoio a alguma função executada no escritório, a ambientes de desenvolvimento de aplicações.

6.1 ODYSSEY

ODYSSEY [FIK81] é uma ferramenta de apoio à aquisição de dados de Sistemas de Informação de Escritórios. Esta ferramenta baseia o processo de aquisição em conhecimentos a respeito das tarefas a serem executadas no domínio específico. Foi desenvolvida com o objetivo de analisar a representação e a utilização de conhecimento a respeito de tarefas do domínio no apoio ao processo de aquisição de dados.

Se destina, especificamente, a auxiliar no planejamento de uma viagem, feito através de um conjunto de formulários eletrônicos específicos. Com as informações fornecidas através do preenchimento destes formulários é montada uma base de dados, a qual vai conter a descrição dos planos da viagem. O conhecimento do domínio é utilizado somente para auxiliar o usuário no preenchimento dos formulários.

ODYSSEY foi projetado e implementado em KRL1, que é uma implementação da linguagem KRL em INTERLISP. Esta linguagem permite uma representação de conhecimentos em estruturas de "frames" e uma programação orientada a objetos. Apresenta, ainda, procedimentos, invocados através de operações sobre os objetos.

Um formulário é representado como um objeto com estrutura de árvore, representando codo nodo uma área retangular do formulário. Cada subárvore representa todas as subáreas contidas na área representadas pelo nodo raiz desta subárvore. O nodo raiz da árvore representa o formulário completo; as folhas representam os campos. Cada nodo representa um objeto de dado ativo e possui funções a ele associadas, as quais comandam o comportamento do formulário.

Todas as informações relativas ao plano de viagem, obtidas a partir do preenchimento dos formulários, são armazenadas na base de dados do sistema. A base de dados, entretanto, não contém nenhuma referência aos formulários propriamente ditos, sendo portanto independente deles. Utiliza conhecimentos do domínio de aplicação (no caso, de planejamento de viagens) para elaborar o plano de viagem com base nos dados coletados a partir dos formulários. Este conhecimento do domínio é representado, na base de dados, através de "frames".

6.2 DRACON

Existem muitas ferramentas de apoio ao trabalho de escritórios que são baseadas em conhecimento. Um dos principais problemas destas ferramentas consiste na aquisição do conhecimento do domínio da aplicação, no caso, conhecimento a respeito das tarefas executadas no escritório. Diversas ferramentas têm sido desenvolvidas para este objetivo. DRACON é uma delas.

DRACON [MAH90] é um interface inteligente, para aquisição e exibição de conhecimento sobre planos em SIEs. Um plano consiste de informações a respeito de uma atividade, como por exemplo, seu objetivo, subobjetivos, pré-condições para sua execução e efeitos que provoca. Sua utilização é dirigida para o sistema de planejamento POLYMER.

Este interface se baseia no modelo do escritório do ponto de vista de seus usuários finais, que são os funcionários deste escritório. O especialista do domínio de aplicação é, neste caso, o usuário final do escritório. O objetivo do interface é ajudar estes especialistas do domínio a apresentarem seu conhecimento de uma maneira próxima à sua visão do conhecimento operacional. Além disso, fornece aconselhamento a este especialista, em uma linguagem que pode ser entendida por pessoas que não estão familiarizadas nem com o interface, nem com a tarefa que está sendo modelada.

O interface apresenta graficamente as entidades do domínio. A representação de cada item na base de conhecimentos é feita através de um ícone. Objetos, relacionamentos entre objetos e ordem de execução são todos representados através de ícones. Os conteúdos da base de conhecimentos podem ser visualizados através de janelas. Toda a manipulação destes ícones é feita através de janelas gráficas. O usário pode, inclusive, modificar a ordem dos ícones na base de conhecimentos.

6.3 SISTEMA GERENCIADOR DE MENSAGENS

Em [CHA87] é proposto um sistema gerenciador de mensagens, baseado em conhecimento. O sistema funciona como interface do usuário, que a êle submete suas mensagens (textos, formulários, gráficos, imagens , voz), as quais são armazenadas em um banco de dados relacional. A manipulação inicial destas mensagens

é feita por um gerenciador de mensagens. Em seguida são passadas para um filtro de mensagens baseado em linguística, que é um sistema de gerenciamento de mensagens baseado em conhecimento e que tem por finalidade "filtrar" as mensagens desnecessárias. As mensagens relevantes são então utilizadas por um sistema especialista que as passa para a rede de comunicação.

O sistema especialista contém uma base de conhecimentos composta de regras de alerta, definidas pelo usuário. A forma geral de uma regra de alerta é a seguinte:

IF <condição> THEN <ação>

Através destas regras é feita a monitoração das alterações efetuadas na base de dados. Uma alteração nesta base de dados pode ativar uma regra de alerta associada à uma classe de mensagens. No caso de serem ativadas mais de uma regra simultaneamente, elas serão tratadas de acordo com a prioridade da mensagem que monitoram.

A ativação de uma regra pode levar a execução de ações tais como efetuar uma alteração na base de dados, recuperar uma informação, executar uma atividade do escritório, etc. Deste modo, a introdução de uma base de conhecimentos no gerenciador de mensagens faz com que algumas das funções de trabalho humano de um escritório possam ser automaticamente processadas.

Foi feita uma implementação deste sistema através do ambiente LOOPS. Neste ambiente, uma regra de alerta é implementada como um método a ser invocado quando uma mensagem é enviada para um objeto. As mensagens, em LOOPS, são essencialmente chamadas a procedimentos. LOOPS apresenta a função DefRMS para simplificar a definição de regras a serem utilizadas como métodos.

6.4 EP-X

Uma das áreas de SIE em que ferramentas baseadas em conhecimento podem ser muito úteis é na recuperação de informações bibliográficas. As pessoas que procuram informações em uma biblioteca geralmente não são especialistas na área desta informações. A recuperação de um documento é feita através de "queries", compostos de palavras-chave do domínio de aplicação conectados através de operadores lógicos (AND, OR, NOT). A construção destes

"queries" constitui uma tarefa não muito fácil, principalmente porque as pessoas geralmente não sabem bem o que estão procurando. À medida em que a consulta vai se realizando, a noção do que é desejado vai sendo refinada. Uma ferramenta adequada pode, portanto, auxiliá-las a definir e refinar seus tópicos de interesse.

EP-X (Environment Pollution eXpert) [SMI89] é uma das ferramentas implementadas com este objetivo - auxílio à identificação de documentos publicados, cujas informações estão armazenadas em um banco de dados. É um sistema de busca baseado em semântica, que se destina à recuperação de documentos do campo de poluição ambiental.

Possui uma base de conhecimentos que descreve tópicos pertinentes ao campo da poluição e um banco de dados onde estão armazenadas as informações a respeito de documentos relevantes a tópicos particulares desta área. O conhecimento é representado através de primitivas semânticas hierarquicamente definidas e de "frames". Os "frames" indicam a utilização dos conceitos e os relacionamentos entre conceitos. As informações que preenchem os "slots" são organizadas em hierarquias que indicam os relacionamentos entre as classes. Apresentam um mapeamento de palavras para primitivas semânticas. Diferente classes de ambiguidades e de equivalência são representadas através de classes de "triggers".

Cada documento é representado como um uma instância de um "frame", possuindo associada a ela uma lista de identificadores, que indica autores, título, resumo, etc. Para auxiliar na recuperação de informações, EP-X apresenta ainda, associado a cada nodo da hierarquia de conceitos, informações a respeito dos documentos. Deste modo o sistema pode, a partir da hierarquia apropriada, coletar identificadores de documentos relevantes.

Outros sistemas que se aplicam na área de auxílio à recuperação de informações bibliográficas através de ferramentas automatizadas são RUBRIC, THOMAS, PLEXUS, CoalSEARCH e CANSEARCH. Em [CHE90] é apresentado um método baseado em conhecimento, para o projeto de sistemas de recuperação de documentos.

6.5 COKES

COKES (Carleton Office Knowledge Engineering System) [KAY87] é um sistema dedutivo baseado em conhecimento, que tem

por objetivo coordenar a atuação de diversos assistentes inteligentes automáticos que apoiam trabalhos em escritórios. Estes assistentes estão distribuídos ao longo de uma rede de comunicação, estando o conhecimento do domínio espalhado nesta rede.

COKES proporciona suporte a consultas concorrentes, de um modo tal que o usuário pode executar diversas tarefas simultanea e sincronizadamente. Foi implementado em MPROLOG, sobre um sistema operacional UNIX.

A representação do conhecimento do escritório é feita através de "frames" e de regras de produção. Os "frames" são utilizados para o conhecimento estrutural, que compreende conhecimento sobre funcionários, a empresa e os recursos. Podem ser representadas classes e instâncias das classes. Cada "frame" possui um nome único, que o identifica. Nos "slots" são armazenadas tuplas de três elementos, respectivamente nome do "slot", nome da faceta e valor do "slot". Estes valores podem ser sequências de caracteres, números, listas ou estruturas. Os nomes das facetas indicam informações a respeito do valor do "slot", podendo ser um valor explícito ou o nome de um predicado PROLOG a ser invocado ("daemon"). Um "frame" do exemplo apresentado em [KAY87] tem a seguinte forma:

```
frame : employee
slots : [
        [ako,value, office_object],
        [hand_mail_address,if_needed,determine_office_address],
        [email_address,if_needed,determine_unix_addres],
        [superior,if_needed,determine_employee_boss],
        works_on,appended,appended_works_on]
        ].
```

O conhecimento procedimental é armazenado na forma de regras de produção. Uma regra possui um nome, uma lista que representa as condições da regras e outra lista representando as ações a serem executadas quando as condições forem satisfeitas. Uma regra apresentada em [KAY87] tem a seguinte forma:

```
rule: distribute_report.
if:   [goal(identify_distribution),
      goal(send_out_reports)].
then: [goal(distribute_report)].
```

Apresenta um interface com o usuário muito simples, facilitando a sua utilização. Está, ainda, integrado com um sistema de mensagens baseado em computador, que permite a vários assistentes executarem seus procedimentos através de passagem de mensagens e de conhecimento entre si.

A arquitetura de COKES apresenta quatro subsistemas de alto nível, conforme pode ser visto na figura 6.1, tirada de [KAY87]: (1) o executor de consultas múltiplas, que controla as consultas e gerencia a filas; (2) o gerenciador do sistema baseado em conhecimento, que manipula as consultas do ambiente COKES, apresentando um gerenciador das regras de produção, um gerenciador dos "frames", um processador de acesso aos "frames" e um interpretador de regras; (3) o gerenciador de contexto, que controla acessos ao contexto das consultas e ao contexto permanente; e (4) o gerenciador de serviços de caixa-postal, que controla acessos e "buffering" de mensagens de outros assistentes do escritório.

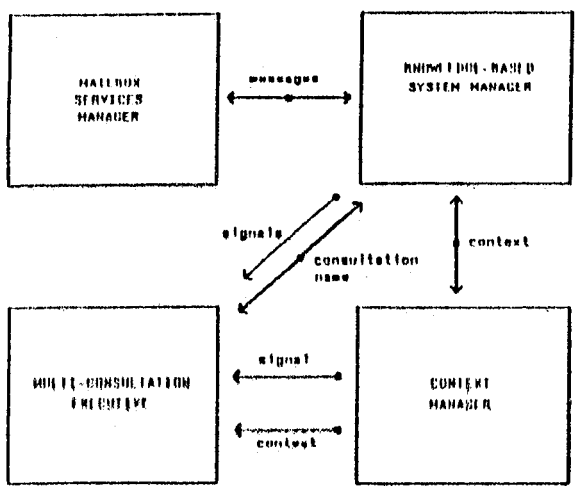

Figura 6.1 - ARQUITETURA DE ALTO NÍVEL DE COKES

6.6 AMS

O sistema AMS (Activity Manager System) [TUE88] consiste basicamente de um "shell" para aplicações de SIE - tanto para descrição como para execução de procedimentos nestes ambientes. Apresenta: (1) representação dos conceitos de escritórios em diferentes níveis de aplicação; (2) ferramentas para dedução e es-

truturas de controle próprias para estes ambientes; (3) um interface que permite uma interação apropriada com os usuários dos escritórios; e (4) conhecimentos gerais para aplicações. O AMS pode ser utilizado tanto pelos especialistas de organização como pelas pessoas que trabalham no escritório. É um produto do projeto Esprit, que estuda técnicas para auxiliar pessoas que trabalham em escritórios.

O sistema foi construído com base em LE-LISP. O conhecimento é armazenado em um sistema de representação de conhecimentos ("KRS"). Apresenta uma ferramenta para representação do conhecimento, que é orientada à representação através de "frames". Entretanto, outros formalismos de representação de conhecimentos podem também ser utilizados, existindo inclusive a possibilidade de integração de diversos formalismos, permitindo a representação de construções complexas. Estes outros formalismos (representação através de regras, lógica formal, etc.) são fornecidos pelo sistema em uma biblioteca. Apresenta, ainda, interfaces para preenchimento de formulários e ferramentas gráficas para manipulação dos objetos armazenados, onde podem ser monitoradas informações incompletas e redes de atividades.

A representação do conhecimento é feita através do conceito de atividades. Uma atividade representa qualquer tarefa executada no âmbito do SIE, devendo apresentar todas as informações que são importantes para esta tarefa. A definição de uma atividade deve encapsular os seguintes tipos de informação: (1) um estado inicial, que descreve as pré-condições para a execução desta atividade; (2) um estado final, que descreve o efeito causado pela execução da atividade - pós-condição; e (3) o corpo da atividade, que define como ela deverá ser executada.

Existem dois tipos de atividades: (1) atividade terminal, cujo corpo é composto de ações que não podem ser decompostas, como por exemplo, preencher um formulário; e (2) atividade complexa, quando o corpo contém descrições que podem ser decompostas. As atividades são organizadas em estruturas de árvore, representando diferentes níveis de abstração.

A sequência de execução das diferentes atividade de um escritório é representada através de redes de atividades. Uma rede de atividade é representada graficamente através de um grafo orientado. Cada nodo representa uma atividade, sendo a relação de precedência de execução entre as atividades representada através

das arestas do grafo. A sincronização de atividades é representada através da associação de operadores aos nodos. Na figura 6.2 está representada uma rede de atividades, tirada de [TUE88].

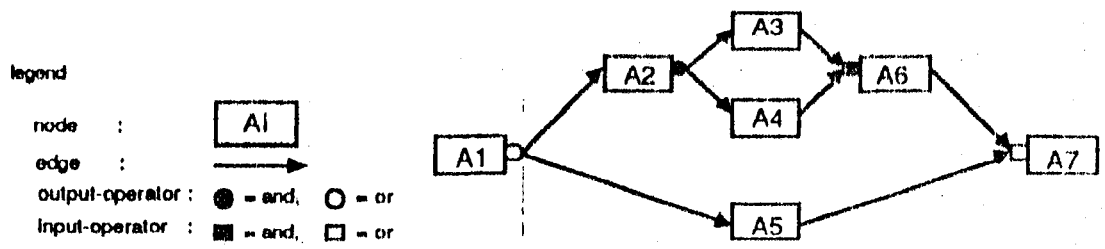

Figura 6.2 - REDE DE ATIVIDADES EM AMS

O corpo de uma atividade pode ser definido através de uma rede de atividades, possibilitando a representação de diferentes níveis de generalização. Uma rede de atividades pode ser o refinamento de uma atividade mais geral, e pode conter atividades que são refinadas através de outras redes de atividades. Isto leva a uma hierarquia de redes de atividades.

Como a base de conhecimentos é organizada em diferentes níveis de abstração, o conhecimento dos níveis mais abstratos pode ser compartilhado na construção de estruturas concretas diferentes (conceito de reusabilidade de conhecimento). Este compartilhamento é representado através do conceito denominado MOPA ("Memory Organization Packet for Activities"), que permite a organização da memória em pacotes de conhecimento abstrato. O conceito MOPA apresenta dois elos, o primeiro apontando para a rede de atividades abstrata ("A-AN") e o segundo apontando para uma lista de atividades. Esta lista representa o conhecimento de como deve ser executada a sequência no nível de abstração corrente, completando o conhecimento da rede "A-AN". MOPAs são interligados através de elos que representam casos concretos, constituindo uma hierarquia na base de conhecimentos.

6.7 RECAST

RECAST (REquirements Collection And Specification Tool) [FUG90] é uma ferramenta que tem por objetivo a reutilização de documentos do desenvolvimento de aplicações de Sistemas de Informação em geral. Está em desenvolvimento no projeto ITHACA (ESPRIT II), projeto no qual está sendo desenvolvido um ambiente de desenvolvimento de software baseado no paradigma de objetos e em reusabilidade. O ambiente de ferramentas ITHACA propicia a reutilização de vários componentes de aplicações previamente desenvolvidas: requisitos de aplicações, projetos de aplicações, resultados de execução, documentações.

O ambiente de ferramentas ITHACA está centrado em torno de uma base de conhecimentos, a Base de Informações de Software. Nesta base de conhecimentos estão armazenadas informações sobre componentes disponíveis para serem reutilizados. Ao ser desenvolvida uma nova aplicação, esta base de conhecimentos é consultada pelas ferramentas ITHACA, à procura de componentes que possam ser totalmente reutilizados, ou que possam ser parcialmente reaproveitados, sofrendo alguma modificação ou refinamento.

As ferramentas ITHACA se destinam a dois tipos de usuários: (1) os engenheiros da aplicação, responsáveis pelo desenvolvimento de componentes genéricos de aplicações para domínios particulares, (tais como requisitos de aplicações, especificações, objetos do projeto, documentos de projeto), que apresentariam características apropriadas para serem utilizados em mais de uma aplicação deste domínio; e (2) pessoas que desenvolvem aplicações, as quais procuram utilizar estes componentes em aplicações particulares. Os engenheiros de aplicação são responsáveis pela manutenção das ferramentas ITHACA, pois são êles que controlam a base de conhecimentos, acrescentando, modificando e retirando informações.

As ferramentas ITHACA apoiam as fases de desenvolvimento de um sistema, especificamente de coleta de requisitos e de especificação. Procura-se reutilizar especificações de requisitos e identificar objetos de projetos que satisfaçam estas especificações.

Na ferramenta RECAST procura-se a reutilização de documentos obtidos nas fases iniciais do desenvolvimento de um sistema - coleta de requisitos, especificação funcional e configuração

dos projetos. Cada uma destas fases tem como produto um documento de desenvolvimento. RECAST está baseado em conhecimento de desenvolvimento de Sistemas de Informação em diferentes domínios de aplicação. Pode ser visto como um gerador de sistemas, baseado no conhecimento de um determinado domínio de aplicação e em modelos de definição de requisitos. Nas duas primeiras fases do desenvolvimento, RECAST procura enfatizar a reutilização de documentos.

RECAST é formado por três ferramentas, uma para modelagem de requisitos (RMT), a segunda para aquisição dos requisitos (RCT) e a terceira para especificação dos requisitos (RST). Estas três ferramentas estão centradas em conhecimentos fornecidos pelo modelo de requisitos, o qual possui conhecimentos do domínio de aplicação e das fases de definição de requisitos.

O modelo dos requisitos é armazenado na base de conhecimentos através de uma rede semântica. Esta rede contém: (1) conhecimento sobre modelos de requisitos, domínios de aplicação e sobre o processo de definição, em RECAST, de novos domínios de aplicação e componentes reutilizáveis ("metaconhecimento"); e (2) conhecimento sobre a aquisição dos requisitos neste domínio e do processo de especificação utilizado em ITHACA ("conhecimento"). A figura 6.3, tirada de [FUG90], apresenta a arquitetura de RECAST.

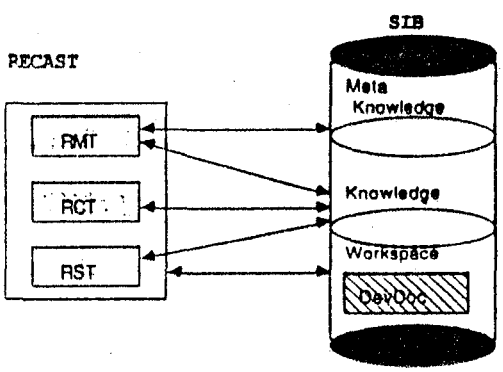

Figura 6.3 - ARQUITETURA DE RECAST

Esta rede semântica é manipulada pelo engenheiro da aplicação, que é quem define: (1) as características do modelo de requisitos, tais como seu nome, os domínios de aplicação para os quais sua utilização é apropriada, se o modelo é orientado a dados (por exemplo, um modelo E-R) ou a processos (por exemplo, uma rede de Pètri), e se existem ferramentas externas para processamento do modelo (por exemplo, um editor gráfico para entrada de entidades do modelo); (2) as entidades do modelo, que constituem os nodos da rede semântica; são idetificadas por seu nome e têm uma forma (por exemplo, representação gráfica) e alguns atributos; e (3) os elos do modelo, definido como um nodo, representando os relacionamentos entre as entidades; um elo é identificado por seu nome, tem algumas características, entre as quais as entidades de origem e de destino, e seu tipo, que pode ser "é_um", "parte_de", etc., ou pode ser definido pela aplicação.

7. CONCLUSÃO

Todos os sistemas utilizados em Inteligência Artificial apresentam uma base de conhecimentos. Esta contém conhecimentos do domínio da aplicação que está sendo desenvolvida e das heurísticas utilizadas para solucionar problemas. A representação do conhecimento na base de conhecimentos é um dos pontos fundamentais no desenvolvimento de sistemas baseados em conhecimento. Da forma de representação utilizada dependem os métodos de aquisição de conhecimento do domínio da aplicação e as técnicas utilizadas para a busca de respostas.

Este trabalho apresentou um estudo geral sobre paradigmas de representação de conhecimento, com vistas à sua utilização no âmbito de Engenharia do Conhecimento. Foram apresentados os principais paradigmas de representação de conhecimento utilizados: regras de produção, lógica de predicados, redes semânticas, "frames" e procedimentos. As vantagens e desvantagens de cada um deles foram analisadas.

Para exemplificar a utilização dos diferentes paradigmas foi escolhida uma aplicação da área de Sistemas de Informação de Escritórios - a modelagem de algumas das atividades desenvolvidas em uma agência de locação de fitas de vídeo. Definidas as atividades, sua modelagem foi efetuada através de cada uma das formas de representação de conhecimento apresentadas. Deste modo foi possível mostrar melhor as diferenças entre as formas de representação, além de poder compará-las.

As representações feitas, embora se apresentando de formas absolutamente diferentes, foram capazes de modelar as principais características desta aplicação. Cada uma delas, entretanto, mostrou deficiências em algum ponto, comprovando que a modelagem ideal deve ser realizada através de uma combinação dos paradigmas apresentados, de modo que as deficiências de um deles sejam cobertas pelo outro. Na modelagem através de regras, por exemplo, não foi mostrado como as informações são estruturadas na base de conhecimento, sendo definidas somente as ações executadas sobre elas. Já a modelagem através de redes semânticas, embora se aproprie à definição da estruturação das informações, apresenta dificuldades para representação de conhecimentos procedimentais. Na representação através de "frames", outra forma apropriada à representação da estrutura das informações, a associação de procedimentos a "slots" possibilita a solução deste problema. As re-

presentações através de linguagens procedimentais se apresentam de forma muito complexa, dificultando sua construção e posterior compreensão. A forma que melhor se apropriou ao exemplo apresentado foi aquela que utiliza a lógica de primeira ordem, embora ainda falte a representação de algum conhecimento procedimental. Além disso, a utilização de linguagens de lógica não é de todo trivial, dificultando a representação dos conhecimentos.

Dos estudos realizados pode-se concluir que não existe uma forma de representação que seja a ideal, própria para utilização em qualquer aplicação. A representação de entidades (físicas e/ou abstratas) tanto pode ser feita através de "frames", como de modelagem orientada a objetos, de lógica, ou de regras de produção. As ações a serem executadas durante a aplicação podem ser representadas através de regras, de lógica, de mensagens ou de procedimentos. Isto foi comprovado através do estudo de ferramentas existentes para o desenvolvimento de sistemas baseados em conhecimento, apresentado no capítulo 5. As ferramentas apresentadas apresentam as mais diversas formas de representação de conhecimentos, formadas muitas vezes de combinações dos paradigmas apresentados.

A construção de sistemas baseados em conhecimento é uma tarefa complexa e exaustiva. A necessidade de apoio nesta tarefa levou a um novo ramo de pesquisa, a Engenharia do Conhecimento. Os conceitos básicos de Engenharia do Conhecimento forma apresentados neste trabalho, sempre dando ênfase à modelagem do conhecimento.

Várias ferramentas foram desenvolvidas com o objetivo de apoiar a tarefa de desenvolvimento de sistemas baseados em conhecimento. Apresentamos algumas das ferramentas existentes, procurando verificar como é feita a representação do conhecimento em suas bases de conhecimento. Na construção de um sistema baseado em conhecimento, os fatores que definem a forma como o conhecimento deve ser representado são, portanto, o tipo da aplicação que será desenvolvida e as ferramentas disponíveis para o seu desenvolvimento.

Especial ênfase deve ser dada às técnicas utilizadas para a aquisição do conhecimento, que é uma tarefa fundamental para o bom desempenho de um sistema. Ferramentas especialmente destinadas para esta finalidade existem em grande número, utilizando inclusive interfaces gráficos. A forma como o conhecimento

é representado durante esta aquisição depende exclusivamente da ferramenta utilizada, podendo apresentar diversas formas diferentes.

Foram também apresentadas algumas ferramentas baseadas em conhecimento utilizadas em uma área específica - em Sistemas de Informação de Escritórios. Devido às características intrínsecas desta área, inúmeras são as ferramentas para ela desenvolvidas. Existe uma forte tendência de aumento da utilização de ferramentas deste tipo em escritórios. As ferramentas apresentadas são de aplicação bem diversificada, indo desde simples sistemas especialistas a ambientes de apoio complexos. Também nesta apresentação foi buscada a forma de representação do conhecimento, que nas aplicações de SIEs tende a ser através de objetos, utilizando formas variadas de "frames" associados em redes semânticas.

Do trabalho apresentado podemos concluir, portanto, que a representação do conhecimento é fundamental na atuação global de um sistema - na aquisição do conhecimento, na sua modelagem, na sua manipulação e nas possibilidades de inferir novos conhecimentos. Não existe, entretanto, uma forma que possa ser rotulada como a melhor. As diversas ferramentas vistas, tanto de apoio à construção de sistemas como para utilização em um domínio específico, apresentam as formas mais variadas possíveis de representação do conhecimento.

REFERÊNCIAS BIBLIOGRÁFICAS

[ABE86] ABEL,M. Introdução aos sistemas especialistas - descrição dos sistemas MYCIN, PROSPECTOR, DIPMETER, ADVISOR e MUPROSPECTOR. Porto Alegre, CPGCCC/UFRGS, 1986. 100p. (TI).

[AIE86] AIELLO,L; CECCHI,C.; SARTINI,D. Representation and use of metaknowledge. Proceedings of the IEEE, New York, 74(10):1304-21, Oct.1986.

[AIK83] AIKINS,J.S. Prototypical knowledge for expert systems. Artificial Intelligence, Amsterdam, 20(2):163-210, Feb.1983.

[AIK85] AIKINS,J.S. A Representation scheme using both frames and rules. In: BUCHANAN,B.G. & SHORTLIFFE,E.H. Rule-based expert systems. Readings, Addison-Wesley, 1985. p.424-40.

[ALB85] ALBANO,A.; CARDELLI,L.; ORSINI,R. GALILEO: a Strongly-typed interactive conceptual language. ACM Transactions on Database Systems, New York, 10(2), 1985.

[ALT90] ALTENKRUEGER,D.E. KBMS: Aspects, theory, and implementation. Information Systems, Oxford, 15(1):1-8, 1990.

[ATT86] ATTARDI,G. & SIMI,M. A Description-oriented logic for building knowledge bases. Proceedings of the IEEE, New York, 74(10):1335-44, Oct. 1986.

[BAR83] BARSTOW,D.R. et al. Languages and tools for knowledge engineering. In: HAYES-ROTH,F.; WATERMAN,D.A.; LENAT,D.B. Building expert systems. Reading, Addison-Wesley, 1983. p.283-348.

[BEN87] BENCH-CAPON,T.J.M. Knowledge representation. In: Artificial intelligence state-of-art repport, Exeter Devonshire, 1987. p.29-37.

[BEN88] BENJAMIN,D.P. A Metalevel manifesto. In: INTERNATIONAL MACHINE LEARNING, META-REASONING AND LOGICS. Sesimbra Feb. 15-17, 1988. Proceedings. Sesimbra, 1988. p.19-26.

[BLI87] BLISS,J. & OGBORN,J.M. Knowledge elicitation. In: Artificial Intelligence state-of-art repport, Exeter Devonshire, 1987. p.39-49.

[BRA83] BRACHMAN,R.J. What IS-A is and isn't: an analysis of taxonomic links in semantic networks. Computer, Los Alamitos, 16(10):30-6, Oct. 1983.

[BRA86] BRACHMAN,R. & LEVESQUE,H. The Knowledge level. In: BRODIE,M.L. & MYLOPOULOS,J. (eds.) On Knowledge management systems. New York, Springer-Verlag, 1986. p.9-12.

[BRO86] BRODIE,M.L. & JARKE,M. On Integrating logic programming and databases. In: KERSCHBERG,L. (ed.) Expert database systems. Menlo Park, Benjamin/Cummings, 1986. p.191-207.

[BUC78] BUCHANAN,B.G. & FEIGENBAUM,E.A. Dendral and Meta-Dendral: their applications dimension. Artificial Intelligence, Amsterdam, 11(1):5-24, Aug. 1978.

[BUC85] BUCHANAN,B.G. & SHORTLIFFE,E.H. Rule-base expert systems: the MYCIN experiments of the Stanford Heuristic Programming Project. Reading, Addison-Wesley, 1985. 748p.

[BUS89] BUSCHMANN,H. Entwurf eines kreativen Expertensystems. Angewandte Informatik, Braunschweig, (2):63-75, 1989.

[CHA87] CHANG,S-K & LEUNG,L. A Knowledge-based management system. ACM Transactions on Office Information Systems, New York, 5(3):213-36, Jul. 1987.

[CHE76] CHEN,P.P. The Entity-relationship model: towards a unified view of data. ACM Transactions on Database Systems, New York, 1(1):9-36, 1976.

[CHE90] CHEN,H. & DHAR,V. A Knowledge-based approach to the design of document-based retrieval systems. SIGOIS Bulletin, New York, 11(2,3):281-90, Apr. 1990.

[CLA83] CLANCEY,W.J. The Epistemology of a rule-based expert-system - a framework for explanation. Artificial Intelligence, Amsterdam, 20:215-51, 1983.

[CLA85] CLANCEY,W.J. Heuristic classification. Artificial Intelligence, Amsterdam, 27(3):289-350, Dec. 1985.

[COO86] COOKE,N.M. & McDONNALD,J.E. A Formal methodology for acquiring and representing expert knowledge. Proceedings of the IEEE, New York, 74(10):1422-30, Oct. 1986.

[DAV77] DAVIS,R.; BUCHANAN,B.G.; SHORTLIFFE,E.H. Production rules as a representation for a knowledge-based consultation program. Artificial Intelligence, Amsterdam, 8(1):15-46, Feb. 1977.

[DAV79] DAVIS,R. Interactive transfer of expertise: acquisition of new inference rules. Artificial Intelligence, Amsterdam, 12(2):121-58, Aug.1979.

[DEL86] DELGRANDE,J.P. & MYLOPOULOS,J. Knowledge representation: features of knowledge. In: BIBEL,W. & JORRAND,P. (eds.) Fundamentals of Artificial Intelligence - an advanced course. Berlin, Springer-Verlag, 1986. p.3-36.

[DIB87] DIBBLE,D. & BOSTROM,R.P. Managing expert systems projects: factors critical for successful implementation. In: ACM SIGBDP-SIGCPR CONFERENCE, Coral Gables, Mar. 5-6, 1987. Proceedings. New York, ACM, 1987. p.96-128.

[DOW81] DOWTY,D.; WALL,R.E.; PETERS,S. Introduction to Monteque Semantics. Dordrecht, D. Reidell, 1981. 313p.

[DUB86] DUBOIS,E. et al. A Knowledge representation language for requirements engineering. Proceedings of the IEEE, New York, 74(10):1431-44, Oct. 1986.

[EDD86] EDDY,W. & PEI,G.P. Structures of rule-based belief functions. IBM Journal of Research and Development, New York, 30(1):93-101, Jan. 1986.

[FIK81] FIKES,R.E. Odyssey: a Knowledge-based assistant. Artificial Intelligence, Amsterdam, 16(3): 331-61, July 1981.

[FIK85] FIKES,R. & KEHLER,T. The Role of frame-based representation in reasoning. Communications of the ACM, New York, 28(9):904-20, Sept. 1985.

[FIS87] FISHMAN,D.H. et al. IRIS: An Object-oriented database management system. ACM Transactions on Office Information Systems, New York, 5(1):48-69, Jan. 1987.

[FOX86a] FOX,M.S. & McDERMOTT,J. The Role of databases in knowledge-based systems. In: BRODIE,M.L. & MYLOPOULOS,J. (eds.) On Knowledge-based management systems: integrating Artificial Intelligence and database technologies. New York, Springer-Verlag, 1986. p.407-30.

[FOX86b] FOX,M.S.; WRIGHT,J.M.; ADAM,D. Experiences with SRL: an analysis and frame-based knowledge representation. In: KERSCHBERG,L. (ed.) Expert database systems. Menlo Park, Benjamin/Cummings, 1986. p.161-72.

[FRE87] FREEMAN,P.R.W. Knowledge-based management systems and data processing. In: Artificial intelligence state-of-art repport, Exeter Devonshire, 1987. p.51-63.

[FUG90] FUGINI, M.G. & PERNICI,B. RECAST: a tool for reusing requirements. In: STEINHOLTZ,B.; SOLVBERG,A.; BERGMAN,L. (eds.) Advanced information systems engineering. Heidelberg, Springer-Verlag, 1990. p.339-64.

[GEO85a] GEORGESCU,I. The Hypernets method for representing knowledge. In: BIBEL,W. & PETKOFF,B. (eds.) Artificial Intelligence: methodology, systems, applications. Amsterdam, North-Holland, 1985. p.47-58.

[GEO85b] GEORGESCU,I. et al. INTEXP: A Domain independent expert system. In: BIBEL,W. & PETKOFF,B. (eds.) Artificial Intelligence: methodology, systems, applications. Amsterdam, North-Holland, 1985. p.129-35.

[GEO86] GEORGEFF,M.P. & LANSKY,A.L. Procedural knowledge. Proceedings of the IEEE, New York, 74(10):1383-98, Oct. 1986.

[GEV87] GEVARTER,W.B. The Nature and evaluation of commercial expert system building tools. Computer, New York, 20(5):24-42, May 1987.

[GIN88] GINGSBERG,A.; WEISS,S.M.; POLITAKIS,P. Automatic knowledge base refinement for classification systems. Artificial Intelligence, Amsterdam, 35(2):197-226, Jun. 1988.

[GOL81] GOLDBERG,A. Introducing the Smalltalk-80 system. Byte, 6(8):14-26, Aug. 1981.

[HAY85] HAYES-ROTH,F. Rule-based systems. Communications of the ACM, New York, 28(9):921-32, Sept. 1985.

[KAY87] KAYE,A.R. & KARAM,G.M. Cooperative knowledge-based assistants for the office. ACM Transactions on Office Information Systems, New York, 5(4):297-326, Oct. 1987.

[LAF86] LAFUE,G.M.E. & SMITH,R.G. Implementation of a semantic integrity manager with a knowledge representation system. In: KERSCHBERG,L. (ed.) Expert database systems. Menlo Park, Benjamin/Cummings, 1986. p.333-50.

[LEN83] LENAT,D.B. EURISKO: A Program that learns new heuristics and domain concepts. Artificial Intelligence, Amsterdam, Mar. 1983.

[LEU90] LEUNG,K.S. & WONG,M.H. An Expert-system shell using structured knowledge. Computer, Los Alamitos, 23(3):38-47, Mar. 1990.

[LEV84] LEVESQUE,H.J. The Logic of incomplete knowledge bases. In: BRODIE,M. L.; MYLOPOULOS,J; SCHMIDT,J.W. On Conceptual modelling. New York, Springer-Verlag, 1984. p.165-89.

[LEV86] LEVESQUE,H.J. & BRACHMAN,R.J. Knowledge level interfaces to information systems. In: BRODIE,M.L. & MYLOPOULOS,J. (eds.) On Knowledge management systems. New York, Springer-Verlag, 1986. p.13-34.

[LYN86] LYNGBAEK,P. & KENT,W. A Data modeling methodology for the design and implementation of Information Systems. In: INTERNATIONAL WORKSHOP ON OBJECT-ORIENTED DATABASE SYSTEMS, Pacific Grove, California, Sept. 23-26, 1986. Proceedings. K.Dittrich & U.Dayal (eds), 1986. p.6-17.

[LYN87] LYNGBAEK,P. & VIANU,V. Mapping a semantic database model to the relational model. SIGMOD Record, New York, 16(3):132-42, Dec. 1987.

[MAH90] MAHLING,D.E. & CROFT,W.B. An Interface for the acquisition and display of office procedures.

[MAN86] MANOLA,F. & DAYAL,U. PDM: an Object-oriented data model. INTERNATIONAL WORKSHOP ON OBJECT ORIENTED DATABASE SYSTEMS, Proceedings, 1986. p.18-25.

[MAR89] MARCUS,S. & McDERMOTT,J. SALT: A Knowledge acquisition language for propose-and-revise systems. Artificial Intelligence,Amsterdam, 39(1):1-38, May. 1989.

[MAT89] MATTOS,N.M. An Approach to Knowledge Base Management - requirements, knowledge representation and design issues. Kaiserslautern, Univ. Kaiserslautern, 1989. (Ph.D. thesis).

[MAY85] MAYOH,B.H. Unified theory of knowledge representation. In: BIBEL,W. & PETKOFF,B. (eds.) Artificial Intelligence: methodology, systems, applications. Amsterdam, North-Holland, 1985. p.35-46.

[MCD82] McDERMOTT,J. R1: A Rule-based configurer of computer systems. Artificial Intelligence, Amsterdam, 19(1):39-80, Sept. 1982.

[MIS86] MISSIKOF,M. & WIEDERHOLD,G. Towards a unified approach for expert and database systems. In: KERSCHBERG,L. (ed.) Expert database systems. Menlo Park, Benjamin/Cummings, 1986. p.383-400.

[MOR86] MORGENSTERN,M. The Role of constraints in databases, expert systems, and knowledge representation. In: KERSCHBERG,L. (ed.) Expert database systems. Menlo Park, Benjamin/Cummings, 1986. p.351-68.

[MYL80] MYLOPOULOS,J.; BERNSTEIN,P.A.; WONG,H.K.T. A Language facility for designing database-intensive applications. ACM Transactions on Database Systems, New York, 5(2):185-207, Jun. 1980.

[MYL83] MYLOPOULOS,J.; SHIBAHARA,T.; TSOTSOS,J.K. Building knowledge-based systems: the PSN experience. Computer, Los Alamitos, 16(10):12-18, Oct. 1983.

[MYL84] MYLOPOULOS,J. & LEVESQUE,H. An Overview of knowledge representation. In: BRODIE,M.L.; MYLOPOULOS,J; SCHMIDT,J.W. On Conceptual modelling. New York, Springer-Verlag, 1984. p.3-17.

[NEW82] NEWELL,A. The Knowledge level. Artificial Intelligence, Amsterdam, 23(2):155-212, July 1984.

[NIX87] NIXON,B. et al. Implementation of a compiler for a semantic data model: experiences with TAXIS. SIGMOD Record, New York, 16(3):118-31, Dec. 1987.

[OLI90] OLIVEIRA,F.M. Controle de aprendizagem no nível do meta-conhecimento. Porto Alegre, CPGCC/UFRGS, 1990. 53p. (RP nº 124).

[POL84] POLITAKIS,P. & WEISS,S.M. Using empirical analysis to refine expert system knowledge bases. Artificial Intelligence, Amsterdam, 22(1):23-48, Jan. 1984.

[REI84] REITER,R. Towards a logical reconstruction of the relational database theory. In: BRODIE,M.L.; MYLOPOULOS, J.; SCHMIDT,J.W. (eds.) On Conceptual modelling. New York, Springer-Verlag, 1984. p.191-233.

[RIC88] RICH,E. Inteligência Artificial. São Paulo, McGraw hill, 1988. 503p.

[SHI81] SHIPMAN,D. The Funcional data model and the data language DAPLEX. ACM Transactions on Database Systems, New York, 6(1):140-73, Mar. 1981.

[SCH82] SCHMOLTZE,J.G. & BRACHMAN,R.J. Proceedings of the 1981 KL-ONE Workshop. Cambridge, Bolt Beranek and Neuman Inc., 1982. 269p. (Report Nº 4842).

[SGU85] SGUREV,V. et al. DIGS: a domain independent expert system for technical diagnostics. In: BIBEL,W. & PETKOFF,B. (eds.) Artificial Intelligence: methodology, systems, applications. Amsterdam, North-Holland, 1985. p.137-44.

[SMI85] SMITH,D.E. & CLAYTON,J.E. Another look at frames. In: BUCHANAN,B.G. & SHORTLIFFE,E.H. Rule-based expert systems. Reading, Addison-Wesley, 1985. p.441-54.

[SMI86] SMITH,J.M. Expert database systems: a database perspective. In: KERSCHBERG,L. (ed.) Expert database systems. Menlo Park, Benjamin/Cummings, 1986. p.3-15.

[SMI89] SMITH,P.J. et al. Knowledge-based search tactics for an intelligent intermediary. ACM Transactions on Office Information System, New York, 7(3):246-70, July 1989.

[STE86] STERLING,L. & SHAPIRO,E. The Art of PROLOG - advanced programming techniques. Harvard, The MIT Press, 1986.

[SWA83] SWARTOUT,W.R. XPLAIN: A System for creating and explaining expert consulting programs. Artificial Inteligence, Amsterdam, 21(3):285-325, Sept. 1983.

[SZO86] SZOLOVITS,P. Knowledge-based systems: a survey. In: BRODIE,M.L.& MYLOPOULOS,J. (eds.) On Knowledge-based management systems: integrating Artificial Intelligence and database technologies. New York, Springer-Verlag, 1986. p.339-52.

[TSU84] TSUR,S. & ZANIOLO,C. An Implementation of GEM - supporting a semantic data model on a relational back-end. SIGMOD Record, New York, 14(2):286-95, 1984.

[TUE88] TUENI,M.; LI,J.; FARES,P. AMS: A Knowledge-based approach to tasks representation, organization and coordination. In: CONFFERENCE ON OFFICE INFORMATION SYSTEMS, Palo Alto, Mar. 23-25, 1988. Proceedings. New York, ACM, 1988. p.78-87.

[VIC90] VICCARI, R.M. Inteligência Artificial: representação do conhecimento. Pôrto Alegre, CPGCC/UFRGS, 1990. 71p.

[VOU89] VOUROS,G.A & SPYROPOULOS,C.D. A Methodology for conceptual representation of knowledge using attribute grammars. Angewandte Informatik, Braunschweig, 7:287-93, 1989.

[WAL86] WALKER,A. Knowledge systems: principles and practice. IBM Journal of Research and Development, New York, 30(1):2-13, Jan 1986.

[WAT83] WATERMAN,D.A. & HAYES-ROTH,F. An Investigation of tools for building expert systems. In: HAYES-ROTH,F.; WATERMAN,D.A.; LENAT,D.B. Building expert systems. Reading, Addison-Wesley, 1983. p.168-218.

[WEI89] WEITZEL,J.R. & KERSCHBERG,L. Developing knowledge-based systems: reorganizing the systemdevelopment life cycle. Communications of the ACM, New York, 32(4):482-89, Apr. 1989.

[WEY80] WEYHRAUCH,R.W. Prolegomena to a theory of mechanized formal reasoning. Artificial Intelligence, Amsterdam, 13(1):133-70, Apr. 1980.

[WIE89] WIELINGA,B.; SCHREIBER,G.; de GREEF,P. KADS Synthesis. Amsterdam, Univ. of Amsterdam, 1989. 62p.

[WIN84] WINSTON,P.H. Artificial Intelligence. Reading, Addison-Wesley, 1984. 524p.

[WOO86] WOODS,W.A. Important issues in knowledge representation. Proceedings of the IEEE, New York, 74(10):1322-34, Oct. 1986.