Instituto de Ciências Matemáticas de São Carlos

ISSN-0109-2569

Um sistema tutorial para o ambiente que auxilia a construção de núcleos específicos de sistemas especialistas (ANSEsp) (versão 1.0)

ANA CLÁUDIA NOIVO ARANTES

SOLANGE REZENDE RODRIGUES

Nº 19

RELATÓRIOS TÉCNICOS DO ICMSC

SÃO CARLOS

jun. / 1994

SYSNO 868593
DATA / /
ICMC - SBAB

Um Sistema Tutorial para o Ambiente que Auxilia a Construção de Núcleos Específicos de Sistemas Especialistas (ANSEsp) [1]

(versão 1.0)

Ana Cláudia Noivo Arantes
Solange Rezende Rodrigues

Universidade de São Paulo
Instituto de Ciências Matemáticas de São Carlos
Departamento de Computação de Estatística

São Carlos, junho de 1994

[1] **Trabalho desenvolvido com o auxílio da FAPESP e ILTC**

Conteúdo

1 Introdução **3**

2 Conceitos Gerais **5**

2.1 A Inteligência Artificial 5

2.2 Os Sistemas Baseados em Conhecimento 6

3 O Ambiente para Construção de Núcleos Específicos de Sistemas Especialistas - ANSEsp **10**

3.1 A Base de Conhecimento (BC) 11

3.2 O Núcleo do Sistema Especialista (NSE) 13

3.2.1 O Módulo Coletor de Dados (MCD) 14

3.2.2 O Motor de Inferência (MI) 24

3.2.3 O Módulo de Explicação (ME) 28

3.3 A Base de Dados (BD) 31

4 A Idéia do Ambiente **32**

4.1 A Passagem do Nível 0 para o Nível 1 34

4.2 A Passagem do Nível 1 para o Nível 2 35

5 Base de Conhecimento sobre Seleção de Vinhos **36**

6 Conclusões **47**

Lista de Figuras

1 Áreas Afins e Subáreas de IA 5

2 A IA, os SBCs e os SE 6

3 Os Sistemas Especialistas e os Usuários 7

4 Entradas e Saída de um Sistema Especialista 8

5 A Estrutura de um Sistema Especialista 9

6 Componentes da BC 12

7 Estrutura Modular do NSE 14

8 O Fluxo de Informações no MCD 15

9 A Base de Conhecimento e a Base Algorítmica 16

10 Grupos da Base Algorítmica 17

11 O Fluxo de Informações no MI 25

12 O Raciocínio Backward Chaining 26

13 O Raciocínio Forward Chaining 27

14 A Explicação '*Por_Que*' 29

15 A Explicação '*Como*' 30

16 A Explicação '*What_If* 31

17 O Ambiente para Construção de Núcleos Específicos de Sistemas Especialistas-ANSEsp 33

1 Introdução

O desenvolvimento tecnológico dos últimos anos tem possibilitado o surgimento de importantes instrumentos de apoio à educação. Assim, atividades como ensino e treinamento de pessoas tem contado com um aliado eficiente e moderno: a Informática. Esta contribui com três grupos de sistemas:

- os **Sistemas Tutoriais**: aqui o conhecimento a ser repassado apresenta-se de forma linear ou não linear em textos que podem ou não conter recursos gráficos. Os usuários de sistemas deste tipo caminham livremente pelos textos, buscando as informações de seu interesse. Estes sistemas geralmente substituem manuais de treinamento;
- os **Sistemas Tutores**: diferem dos Sistemas Tutoriais no que diz respeito aos objetivos propostos e à forma como as informações são repassadas. Destinados basicamente ao ensino, estes sistemas classificam seus usuários, restringindo e selecionando as informações a serem acessadas, mesmo que de forma não inteligente;
- os **Sistemas Tutores Inteligentes**: aqui a utilização de técnicas de Inteligência Artificial possibilitam que a atividade de ensino seja realizada de forma inteligente, levando-se em conta as particularidades de cada usuário. Tudo isso é possível graças ao conhecimento manipulado pelo sistema, o qual é devidamente representado e estruturado. Cada usuário - aluno - é avaliado e classificado pelo próprio sistema, que a partir disso decide a linha de evolução do ensino e quais informações poderão ser repassadas. Capazes de justificar suas ações, desenvolver raciocínios e tomar decisões, os STI em muitos se assemelham aos seres humanos na atividade de ensino.

Este trabalho descreve um Sistema Tutorial desenvolvido com o objetivo de auxiliar os profissionais e usuários do Ambiente que Auxilia a Construção de Núcleos Específicos de Sistemas Especialistas (ANSEsp). Este consiste em um sistema que, utilizado na construção de um Sistema Especialista, dispensa a implementação de programas que manipulem o conhecimento e realizem outras funções de um Núcleo de Sistema Especialista, as quais serão descritas nos próximos capítulos deste documento.

O Sistema Tutorial foi construído sob a forma de um hipertexto, que corresponde a um conjunto de telas interligadas, formando uma árvore. Esta configuração possibilita o acesso a uma ou mais telas a partir da tela corrente. Desse modo, o usuário pode acessar de maneira fácil e rápida somente as telas de seu interesse, além de poder atingir o nível de profundidade desejado.

Os usuários em potencial do Sistema Tutorial consistem basicamente de:

- *Engenheiros de Conhecimento:* responsáveis pela representação do conhecimento de um ou mais especialistas no domínio selecionado;
- *Engenheiros de Sistema:* responsáveis pela depuração em níveis do ANSEsp, de forma a construir um Núcleo de Sistema Especialista específico para uma aplicação em particular;
- *Usuários do ANSEsp.*

Grande é a heterogeneidade destes usuários no que diz respeito às informações de interesse. Por isso, o Sistema Tutorial foi desenvolvido de forma a ser abrangente e também específico, expondo desde conceitos de Inteligência Artificial e Sistemas Especialistas até particularidades do ANSEsp.

Palavras e expressões frequentemente utilizadas neste trabalho serão substituídas pelas seguintes abreviações:

IA - **I**nteligência **A**rtificial
SE - **S**istemas **E**specialistas
NSE - **N**úcleo de **S**istemas **E**specialistas
MCD - **M**ódulo **C**oletor de **D**ados
MI - **M**otor de **I**nferência
ME - **M**ódulo de **E**xplicação
BC - **B**ase de **C**onhecimento
BD - **B**ase de **D**ados

O trabalho está organizado da seguinte maneira: no Capítulo 2 serão apresentados os conceitos básicos de Inteligência Artificial relacionados ao desenvolvimento do ANSEsp. No Capítulo 3 serão descritos os aspectos relacionados à sua estrutura e implementação, e o Capítulo 4 tem como objetivo apresentar a idéia do Ambiente, e a forma como o mesmo pode ser manipulado na construção de um Sistema Especialista específico. O Capítulo 5 consiste na apresentação de uma Base de Conhecimento sobre seleção de vinhos, implementada de forma a ser manipulada pelo ANSEsp.

2 Conceitos Gerais

2.1 A Inteligência Artificial

Segundo Elaine Rich [7],

"A Inteligência Artificial se preocupa com o estudo de como fazer os computadores realizarem tarefas nas quais, até agora, o ser humano tem um melhor desempenho que a máquina."

A IA tem sido um dos grandes expoentes da Informática em termos de pesquisa e desenvolvimento nos últimos anos. Dotar máquinas de capacidades similares àquelas exclusivas dos seres humanos é um dos grandes desafios tecnológicos do momento. A evolução nesta área depende não somente do potencial das máquinas em si, mas também de conhecimentos do homem sobre si mesmo. Por isso a IA conta com contribuições de outras ciências como pode ser visualizado na figura 1 [1].

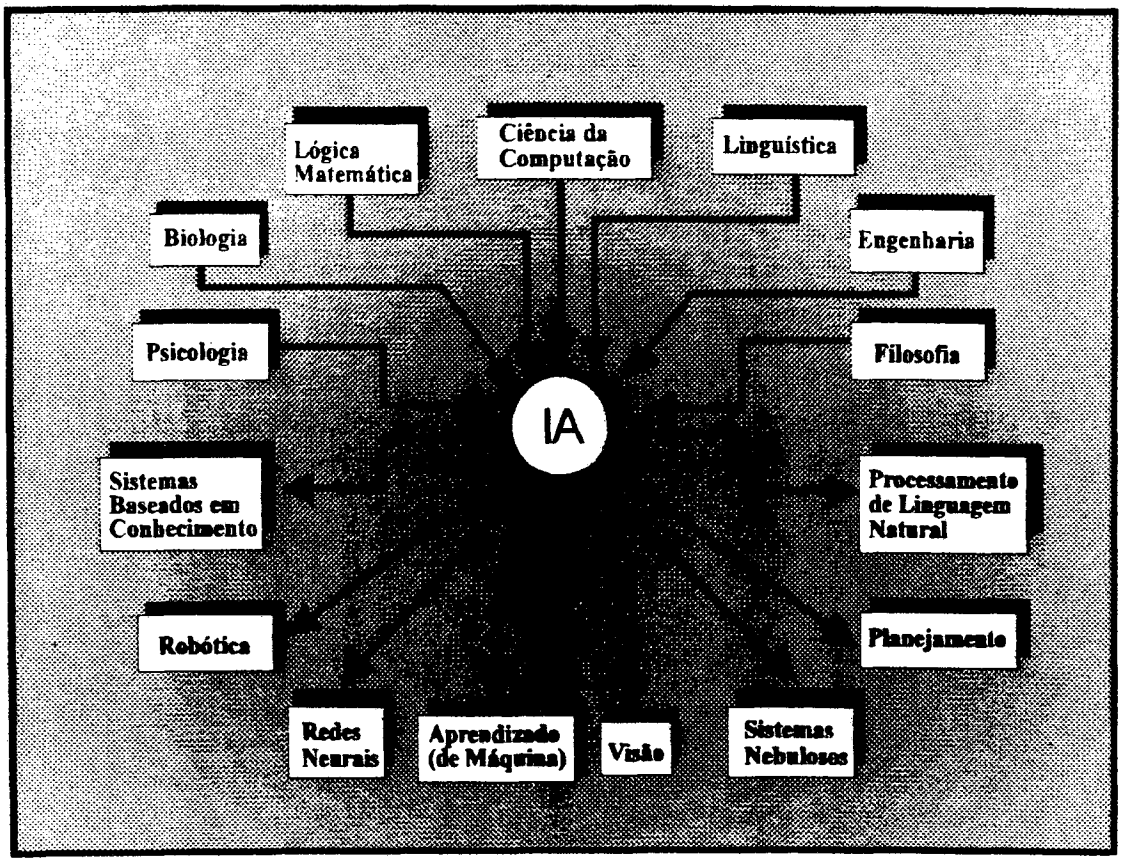

Figura 1: Áreas Afins e Subáreas de IA

Inúmeras são suas aplicações: Processamento de Linguagem Natural, Redes Neurais, Robótica, Sistemas Baseados em Conhecimento, entre outras. Neste trabalho nos concentraremos especificamente nos Sistemas Especialistas, que consistem em um dos grupos que compõem os Sistemas Baseados em Conhecimento.

2.2 Os Sistemas Baseados em Conhecimento

Os Sistemas Baseados em Conhecimento (SBCs) são programas que manipulam conhecimento e informações de forma inteligente. Quando tais sistemas são voltados ao ensino, são chamados Sistemas Tutores Inteligentes (STIs).

Quando desenvolvidos para aplicações onde o conhecimento a ser manipulado restringe-se a um domínio específico e conta com um alto grau de especialidade, possibilitando assim resultados mais precisos, os SBCs podem ser classificados como **Sistemas Especialistas (SEs)**. A figura 2 mostra a localização dos SBCs e dos SEs dentro da área de IA [1].

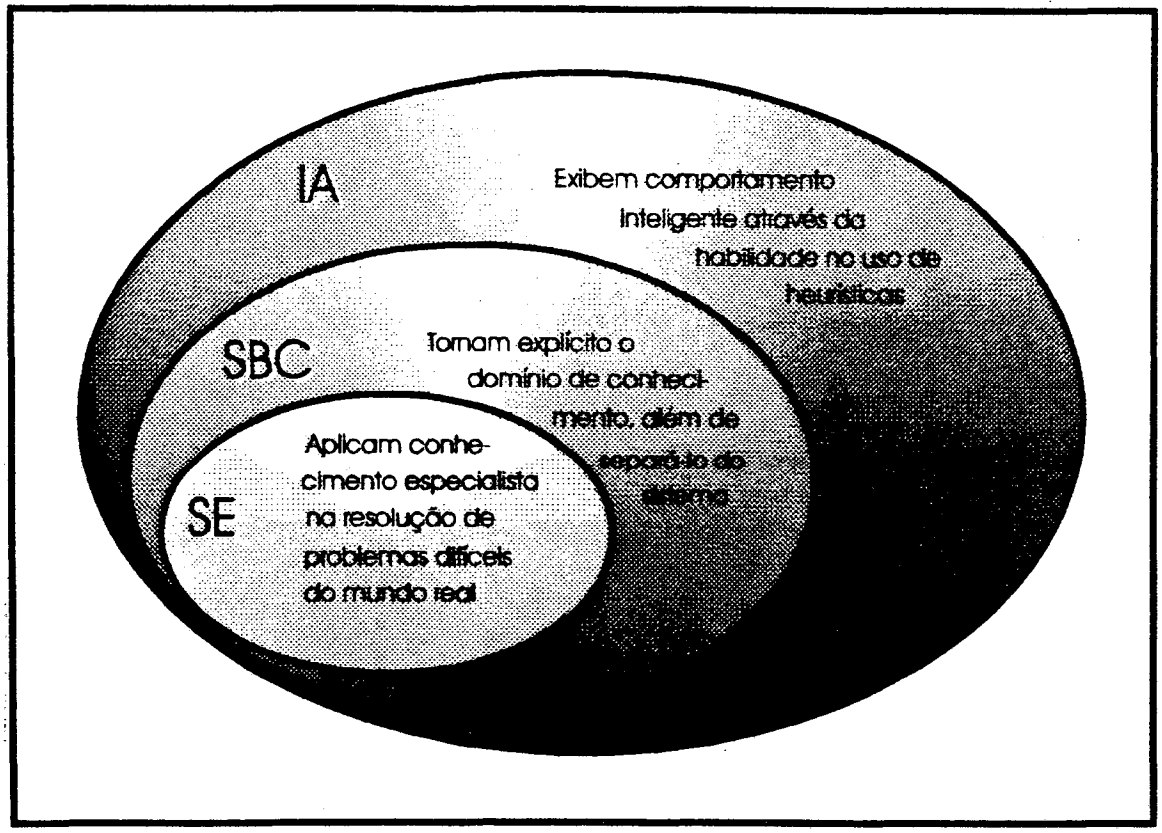

Figura 2: A IA, os SBCs e os SE

Este trabalho relaciona-se à construção de SEs a partir do ANSEsp. Os SEs possuem comportamentos e objetivos similares aos de profissionais humanos conhecidos como especialistas. Um especialista humano pode ser definido como um profissional capaz de compreender, raciocionar e solucionar problemas em seu domínio de conhecimento. Tais profissionais são em geral essenciais em seu ambiente de trabalho, não somente pelos conhecimentos que possuem mas também pela capacidade de manipulá-los de forma eficiente, fornecendo desde diagnósticos confiáveis até tomadas de decisão inteligentes.

Problemas especialmente complicados demandam soluções corretas e ações muitas vezes imediatas, ou seja, demandam especialistas. Mas tais profissionais são raros e nem sempre estão disponíveis. O fato é que no mundo moderno a quantidade de problemas que surgem supera em muito a de bons especialistas disponíveis. Os Sistemas Especialistas surgiram como consequência deste fato (figura 3).

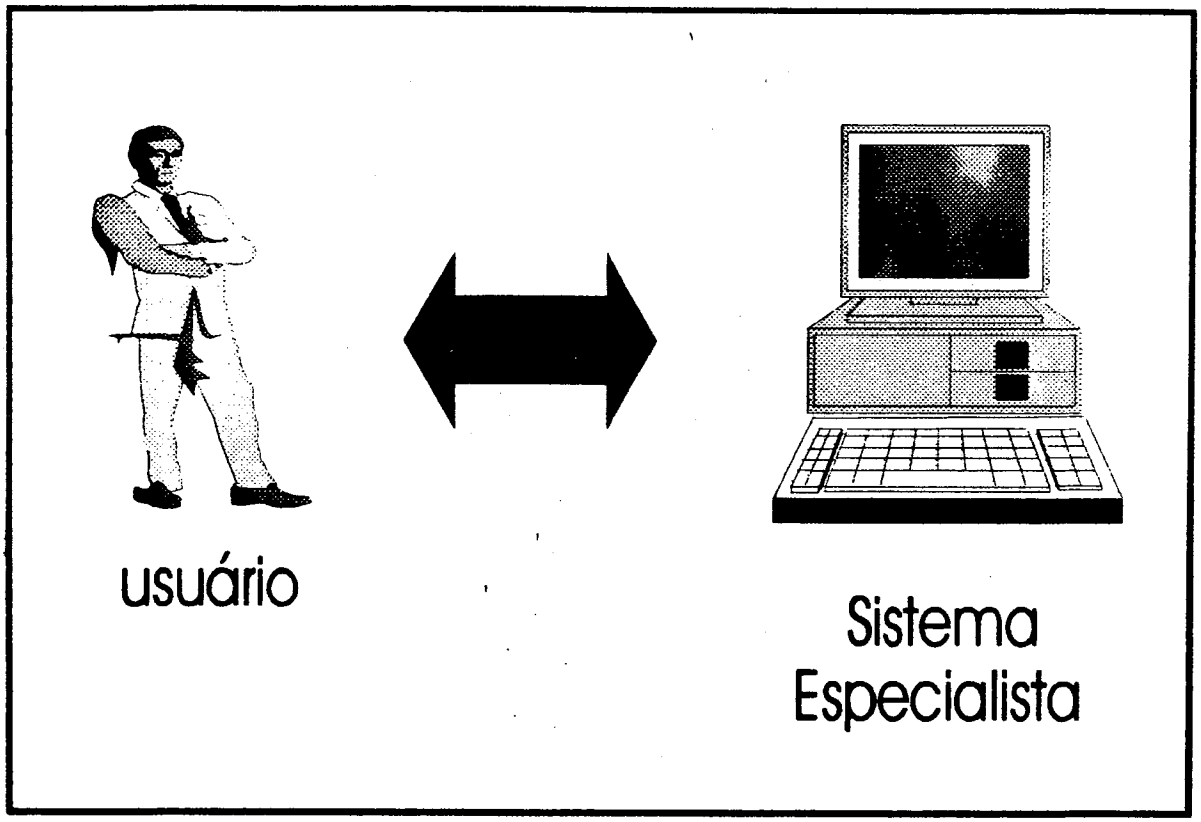

Figura 3: Os Sistemas Especialistas e os Usuários

Por se assemelhar a especialistas humanos na resolução de problemas pertencentes a uma determinada área de conhecimento, os SEs devem ser capazes de [5,1,8]:

- interagir com o usuário fazendo-lhe perguntas, e dessa forma reunir as informações de que necessita;
- desenvolver uma linha de raciocínio a partir destas informações e do conhecimento nele embutido, de forma a encontrar soluções satisfatórias;
- explicar ao usuário, no mínimo, *porque* faz determinadas perguntas e *como* chega às suas conclusões, caso seja questionado;
- lidar com regras e informações imprecisas. Esta é uma característica que, entre outras, diferencia-o dos sistemas convencionais, os quais são capazes de manipular somente dados e regras determinísticas.

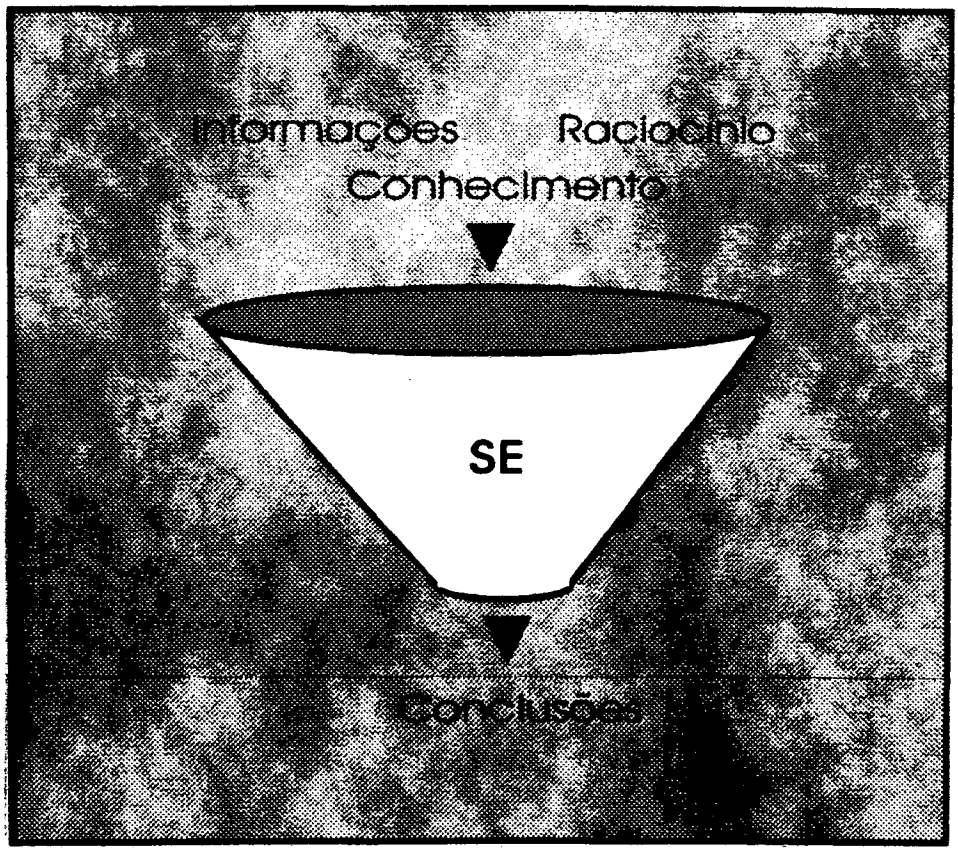

Figura 4: Entradas e Saída de um Sistema Especialista

A estrutura geral de um SE constitui-se basicamente de [5]:

- um **NSE** (Núcleo de Sistema Especialista), o qual desempenha suas principais funções;
- uma **BC** (Base de Conhecimento), onde está representado todo o conhecimento do especialista no domínio selecionado. A forma de representação deve ser compatível com a forma de manipulação do NSE;
- uma **BD** (Base de Dados), a qual consiste na área de trabalho do sistema.

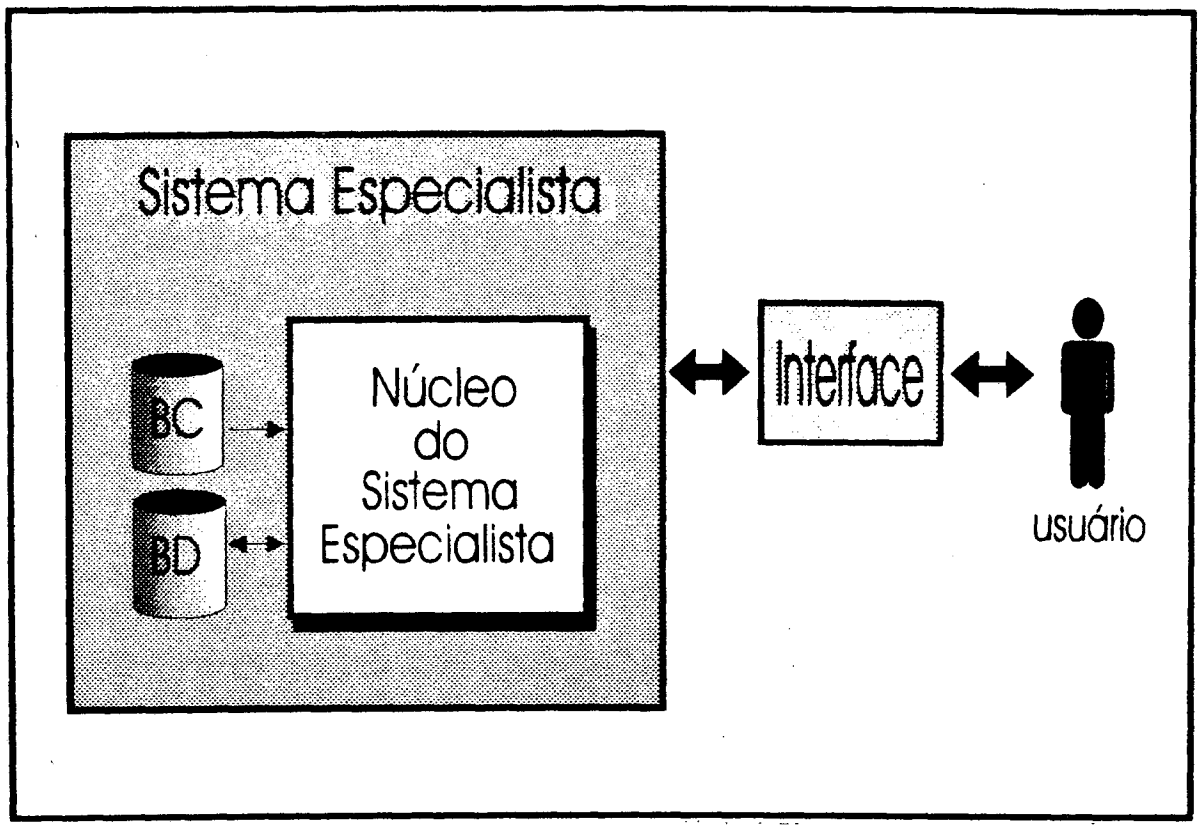

Figura 5: A Estrutura de um Sistema Especialista

O desenvolvimento do Sistema Tutorial tem como objetivo integrar o ANSEsp com os conceitos básicos envolvidos em sua idealização, assim como instruir seus usuários na manipulação de sua estrutura básica. A seguir temos a descrição do ANSEsp, dos módulos que o compõem e dos demais módulos do Sistema Especialista com os quais se comunica.

3 O Ambiente para Construção de Núcleos Específicos de Sistemas Especialistas - ANSEsp

A construção de um Sistema Especialista consiste em uma atividade árdua e custosa, por envolver as seguintes etapas de desenvolvimento [2,6]:

- **A aquisição de conhecimento**. Aqui o Engenheiro de Conhecimento extrai o conhecimento do(s) especialista(s) ou de outras fontes;
- **A estruturação e representação do conhecimento**. Após a identificação e contextualização do problema, o Engenheiro de Conhecimento deve selecionar um esquema apropriado para representá-lo. As opções para a representação incluem: representação lógica, redes semânticas, frames, regras de produção, representação procedimental, etc.
- **A implementação dos módulos do sistema**. Uma vez selecionadas a estrutura e forma de representação do conhecimento, inicia-se a programação. Aqui o conhecimento adquirido é transferido para uma Base de Conhecimento, que será manipulada pelos diversos módulos do sistema como um todo. Tais módulos são programas que incluem métodos de inferência, formas de interação com o usuários e outras entidades externas ao sistema, mecanismos de justificação e/ou explicação de seus procedimentos, etc.;
- **A validação do sistema**. É um processo que inclui a avaliação e refinamento do sistema, até que o Sistema Baseado em Conhecimento atinja um nível satisfatório de desempenho.

Tais atividades requerem tempo e envolvimento de diversos profissionais. O ANSEsp tem por objetivo minimizar estes esforços, eliminando a etapa de implementação do sistema. Para construir-se um Sistema Especialista Específico, será necessário:

- Elaborar uma Base de Conhecimento, onde será estruturado o conhecimento obtido de um ou mais especialistas na etapa de aquisição;
- Utilizar o Núcleo de Sistema Especialista do ANSEsp. Trata-se de um Núcleo completo, ou seja, capaz de manipular qualquer Base de Conhecimento, mas poderá ser depurado de modo a ser executado em uma aplicação específica.

A seguir serão apresentados os conceitos relativos à elaboração de uma Base de Conhecimento que possa ser manipulada pelo ANSEsp.

3.1 A Base de Conhecimento (BC)

A Base de Conhecimento é parte essencial de um Sistema Especialista, pois nela está armazenado todo o conhecimento do especialista, o qual consistirá no domínio de aplicação do sistema [2].

É baseado neste conhecimento e em dados fornecidos pelo usuário que se processará o raciocínio e se chegará às soluções do problema em questão. A representação é feita sob a forma de relações Prolog, as quais podem pertencer a cinco grupos distintos [5]:

- *perguntáveis:* são relações que exigem interação com um meio externo ao NSE, o qual pode ser o usuário, equipamentos ou outros sistemas. O Módulo Coletor de Dados, devido às suas funções, é o responsável pela interpretação de relações deste tipo.

  Uma relação perguntável é reconhecida por:

  ```
  perguntavel(<meta>).
  ```

  onde **meta** corresponde à meta perguntavel.

- *decisão:* são relações interpretadas pelo Núcleo do Sistema Especialista, e representam regras que fazem uso de fatores de certeza. Relações deste tipo são capazes de tirar conclusões provavelmente corretas, mas não seguramente corretas, e são reconhecidas por predicados da forma:

  ```
  <pred>(fc,C,F,<expr>)
  ```

  onde:

  - **C** representa o fator de certeza que será calculado, se necessário, durante uma consulta;
  - **F** representa o fator de certeza fornecido pelo especialista para esta regra;
  - **fc** é o functor que indica ser esta uma relação de decisão;
  - **expr** é a hipótese que faz uso de fatores de certeza.

- *sistema:* são relações declaradas na BC e que são pré-definidas do Prolog, e portanto diretamente executadas por ele. Uma relação de sistema é incapaz de explicar seu funcionamento ou de tirar conclusões que sejam provavelmente corretas, mas não seguramente corretas. Por serem pré-definidas, estas relações não precisam ser declaradas na BC. Como exemplo, podemos citar: *clause/2*, *findall/3*, *lenght/2*, expressões aritméticas, etc.

- *primitivas:* são relações que auxiliam a execução de uma prova, e que devem ser executadas diretamente pelo Motor de Inferência do Prolog. Vale observar que este Motor de Inferência é diferente daquele de um Núcleo de Sistema Especialista, pois não é capaz de lidar com informações imprecisas nem de armazenar as conclusões intermediárias obtidas durante sua execução. As relações primitivas são programas Prolog que efetuam cálculos e procedimentos algorítmicos.

- *definidas:* são semelhantes às relações primitivas mas, por serem interpretadas pelo Núcleo do Sistema Especialista e não pelo Prolog, tornam-se capazes de ativar tanto as relações de decisão quanto as perguntáveis. Relações deste tipo são todas aquelas que não se encaixam nos tipos citados anteriormente. Entre elas, existem relações especiais, as quais serão verdadeiras se e somente se pelo menos N de suas submetas forem válidas. Estas são representadas por:

```
<pred>(verifica(N),<lista_expr>):-
      atomo_1 or atomo_2 or ... or atomo_m.
```

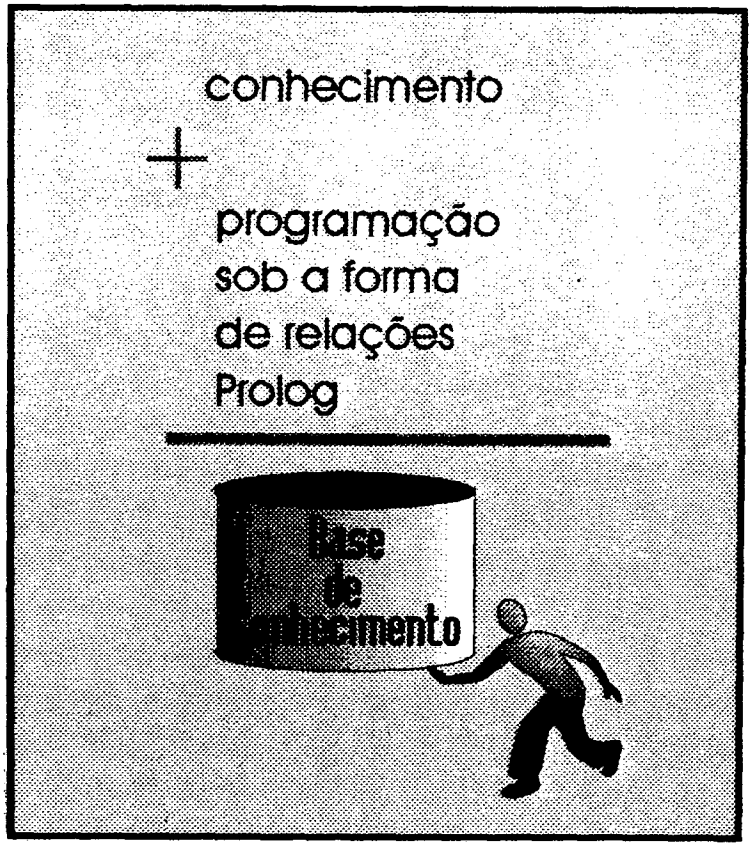

Figura 6: Componentes da BC

3.2 O Núcleo do Sistema Especialista (NSE)

O NSE é o módulo do ANSEsp onde são realizadas suas principais as atividades. São elas [4,8]:

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

Ele é composto basicamente de três subsistemas interdependentes, cada qual com funções específicas. São eles:

- o **MCD** (Módulo Coletor de Dados), responsável pela obtenção de informações do problema em questão. Esta atividade, em geral, é realizada através de perguntas feitas sucessivamente ao usuário;
- o **MI** (Motor de Inferência), responsável pelo desenvolvimento do raciocínio, baseado nas informações obtidas pelo MCD e no conhecimento do(s) especialista(s);
- o **ME** (Módulo de Explicação), o qual explica, ou justifica, as conclusões obtidas e os motivos pelos quais faz determinadas perguntas.

Enquanto executa suas funções, o NSE interage como uma **BD** (Base de Dados), onde grava e acessa os dados temporários obtidos no processo. Além disso, utiliza uma **BC** (Base de Conhecimento), a qual contém representado todo o conhecimento extraído de um ou mais especialistas e, em alguns casos, heurísticas para manipulá-lo. A figura 7 apresenta a estrutura geral do NSE do ponto de vista do usuário do sistema, ao qual não é permitido alterar a BC [8].

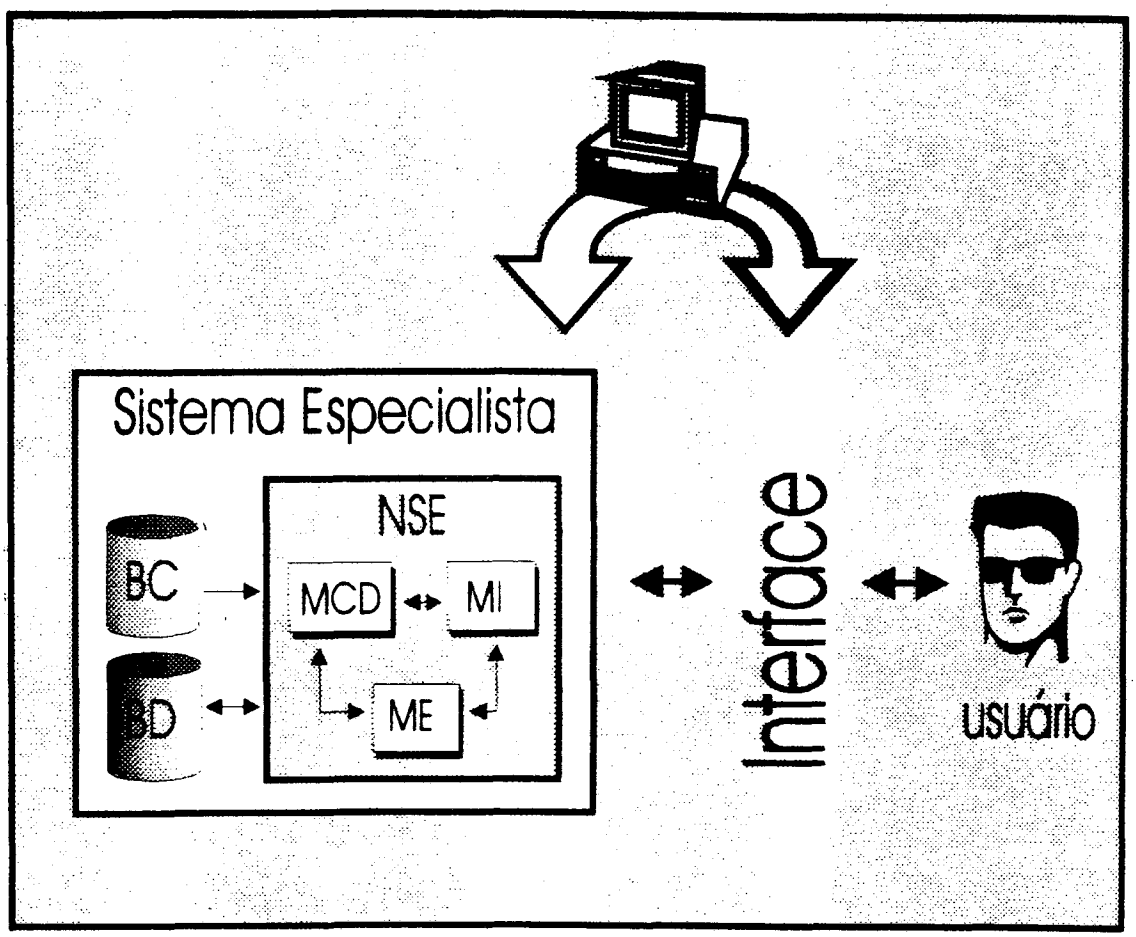

Figura 7: Estrutura Modular do NSE

3.2.1 O Módulo Coletor de Dados (MCD)

Este é o módulo do NSE responsável pela interação com o usuário, obtendo informações através de perguntas [4]. Quando ativado pelo MI, o MCD faz as perguntas necessárias e valida as respostas baseando-se em funções de crítica pré-estabelecidas na BC. Tais funções verificam a corretude das respostas. Se estas forem válidas, o MCD as transfere ao MI, para que o mesmo possa continuar a desenvolver seu raciocínio (figura 8).

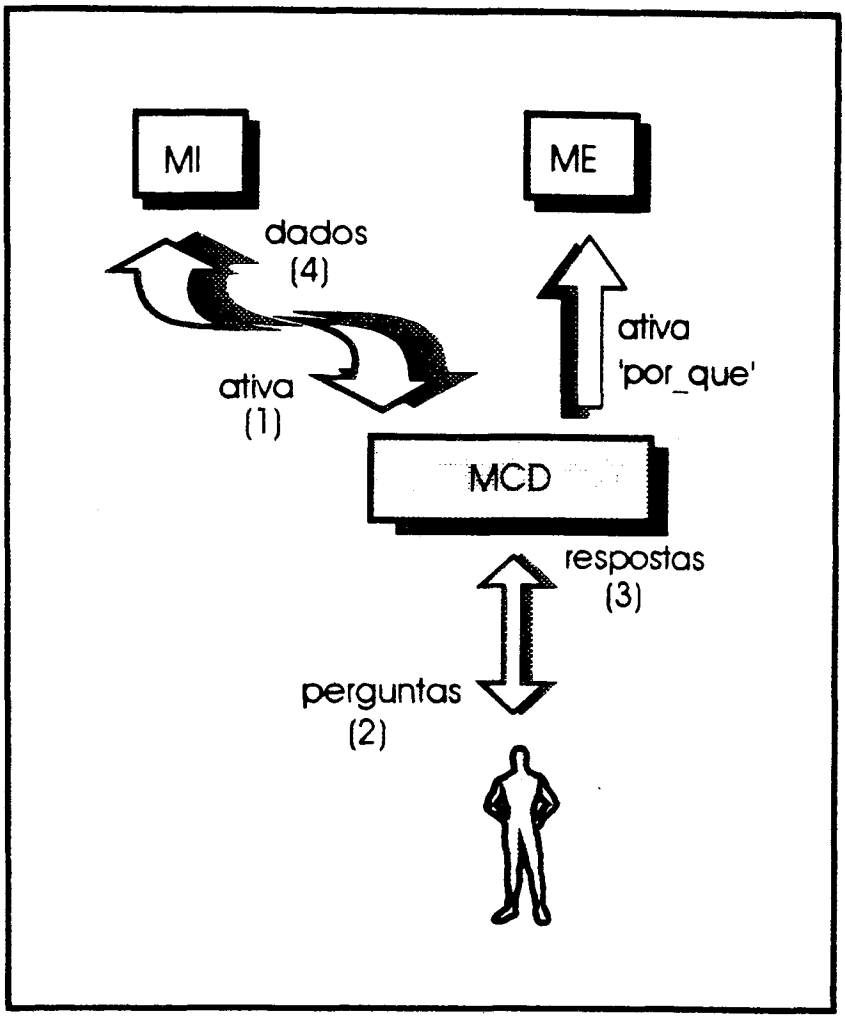

Figura 8: O Fluxo de Informações no MCD

No fluxo de informações do MCD:

- os números da figura representam a sequência de atividades do sistema;
- se o usuário responder *'porque'* a uma pergunta, o ME é ativado, de modo a fornecer as justificativas cabíveis. Depois disso, o MCD torna a fazê-la, recomeçando todo o processo. Caso o usuário responda normalmente às perguntas, o MCD simplesmente repassa as informações obtidas ao MI.

Como mencionado anteriormente, a BC é constituída de relações Prolog, as quais dividem-se em cinco grandes grupos, de acordo com sua natureza. Um destes grupos, as **relações perguntáveis**, é manipulado somente pelo MCD.

Para cada relação perguntável declarada na BC, haverão informações específicas que indicarão ao MCD o tipo de pergunta a ser formulada, o texto da pergunta, a responsabilidade de leitura e a crítica das respostas [4].

Tanto as relações perguntáveis quanto os programas algorítmicos de escrita e crítica de dados localizam-se em uma área especial da BC: a Base Algorítmica (**Ba**) (figura 9).

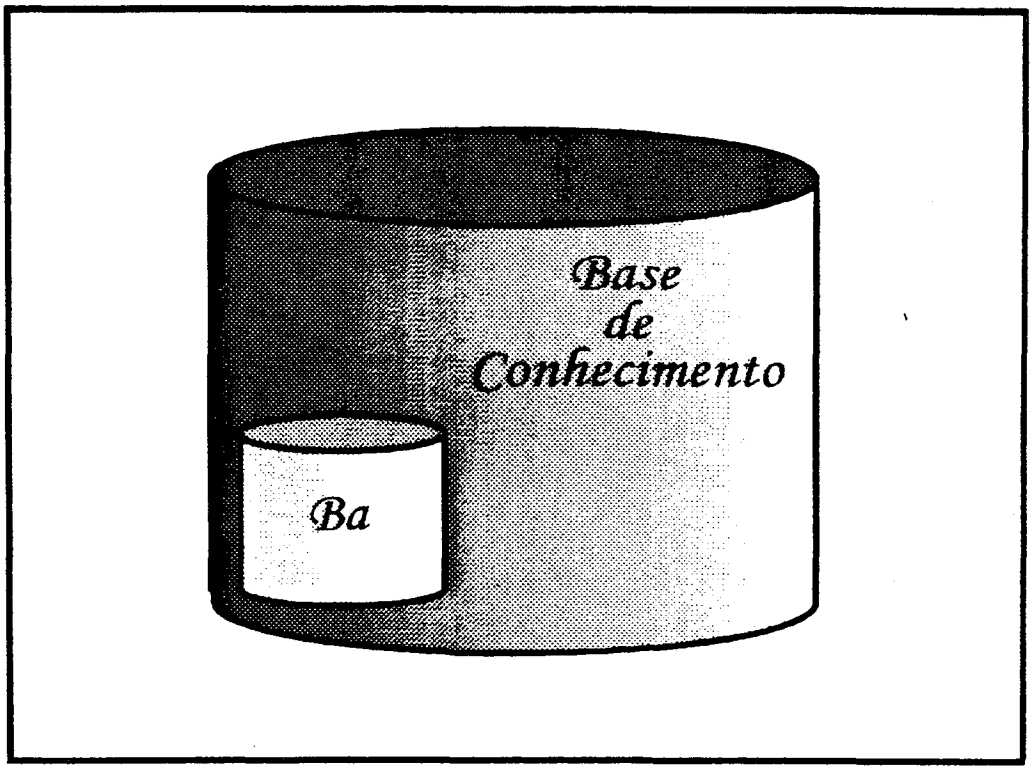

Figura 9: A Base de Conhecimento e a Base Algorítmica

A leitura e validação das respostas do usuário podem ser de responsabilidade tanto do MCD quanto da Ba. A fim de identificar claramente as funções destes dois módulos em relação a uma meta perguntável, basta inserí-la em um dos dois grandes grupos de metas localizados na Ba. São eles [4]:

- *pergunta/3 ou pergunta/4*, que indicam ser o MCD o responsável pela leitura e crítica das respostas;
- *pergunta_livre/3 ou pergunta_livre/4*, que delegam tais responsabilidades ao implementador da Ba.

O número de argumentos em cada grupo difere pelo fato de que *pergunta/4* e *pergunta-livre/4* admitem variáveis que são instanciadas com informações fornecidas pelo usuário,

como seu nome, por exemplo. Tais variáveis podem ser utilizadas na elaboração das perguntas, e por isso possibilitam um tratamento diferenciado com relação ao usuário. A definição dos grupos pode ser visualizada na figura 10.

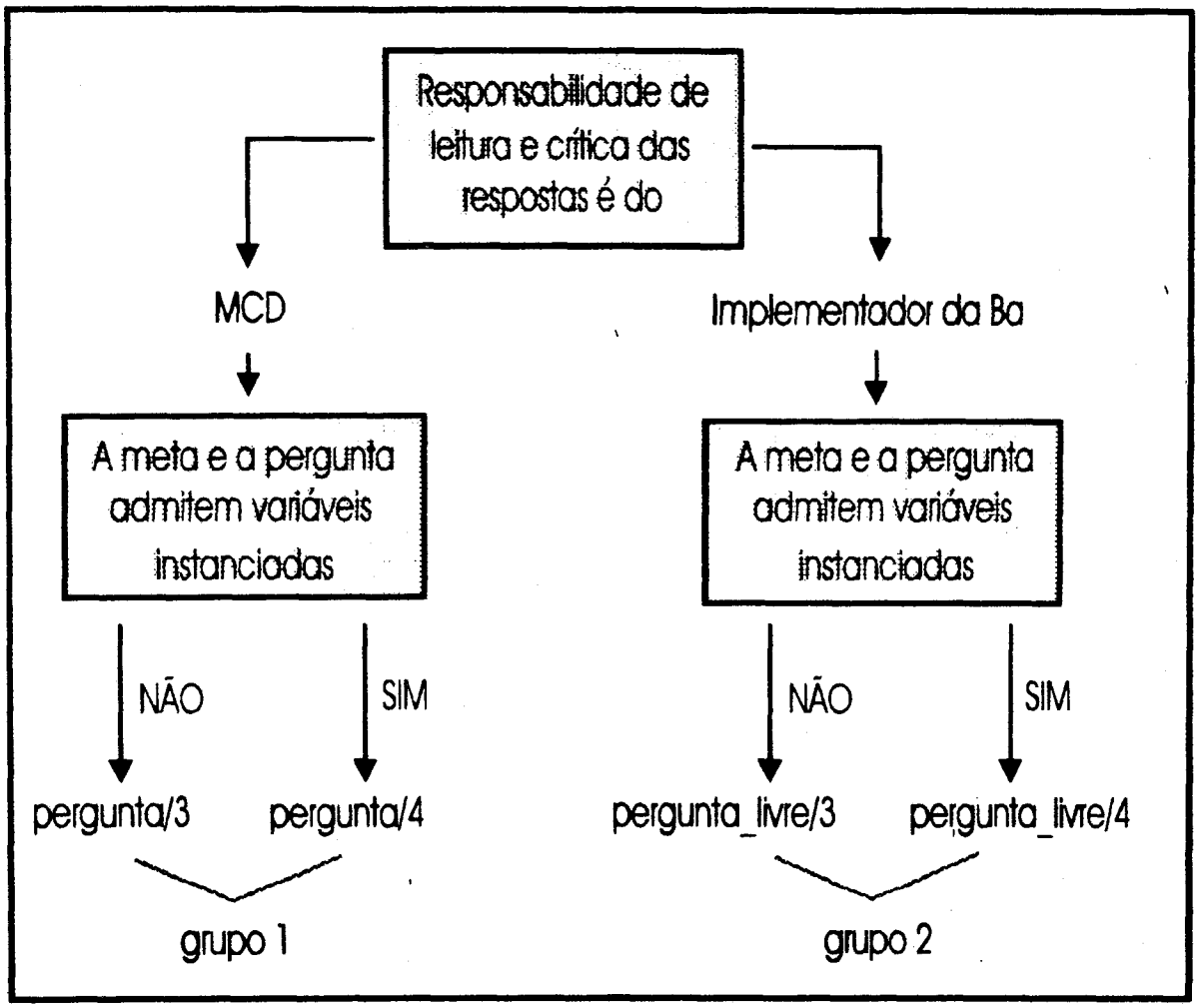

Figura 10: Grupos da Base Algorítmica

Os predicados *pergunta/3* e *pergunta/4* devem ser declarados com os seguintes argumentos:

pergunta(Meta,Perguntas,Critica), ou ainda

pergunta(Meta,Inst,Perguntas,Critica), onde

- **Meta** é a meta perguntável que está sendo provada pelo MI. Seus argumentos serão preenchidos com as respostas fornecidas pelo usuário;
- **Inst** é uma lista que contém os argumentos de Meta que permanecerão instanciados e que serão utilizados na elaboração das perguntas;
- **Perguntas** corresponde a uma meta cujos argumentos são programas de escrita das perguntas e que serão ativados pelo MCD. Estes programas devem estar declarados na Ba;
- **Critica** corresponde a uma lista cujo primeiro elemento indica o tipo de pergunta a ser realizada, e os elementos restantes correspondem a programas para validação das respostas, os quais serão ativados imediatamente após a leitura das mesmas.

A seguir temos a implementação da BC em várias versões, cada qual utilizando um dos tipos de perguntas citados. Esta BC representa o conhecimento do gerente de um banco, cuja função é selecionar o tipo de aplicação adequada a um determinado cliente. Para chegar a uma solução, o sistema deverá interrogar o usuário, de modo a extrair as informações necessárias ao desenvolvimento do seu raciocínio.

```
% Exemplo de regras do tipo pergunta/3:

% Relacoes perguntaveis

perguntavel(outro_banco).
perguntavel(renda(Renda)).
perguntavel(prazo(Prazo)).

% Regras do tipo pergunta/3, sem variaveis instanciadas, e que indicam ser
% o MCD responsavel pela leitura e validacao das respostas

pergunta(outro_banco,pg(perg1),[s-n]).
pergunta(renda(Renda),pg(perg2),[u,critica_renda]).
pergunta(prazo(Prazo),pg(perg3),[u,critica_prazo]).

% Texto das perguntas do tipo pergunta/3

perg1:-
  write('Voce e cliente de algum outro banco?'),
  nl.
```

```
perg2:-
  write('Qual a sua renda mensal em salarios minimos?'),
  nl.

perg3:-
  write('Voce deseja uma aplicacao a curto, medio'),
  write(' ou a longo prazo?'),
  nl.


% Exemplo de regras do tipo pergunta/4

% Relacoes perguntaveis

perguntavel(outro_banco(Cliente)).
perguntavel(renda(Cliente,Renda)).
perguntavel(prazo(Cliente,Prazo)).

% Regras do tipo pergunta/4, com uma variavel instanciada -
% o nome do cliente - e que indica ser o MCD responsavel pela
% leitura e critica das respostas

pergunta(outro_banco(Cliente),[Cliente],pg(perg1(Cliente)),[s-n]).

pergunta(renda(Cliente,Renda),[Cliente],pg(perg2(Cliente)),
         [u,critica_renda]).

pergunta(prazo(Cliente,Prazo),[Cliente],pg(perg3(Cliente)),
         [u,critica_prazo]).

% Texto das perguntas do tipo pergunta/4

perg1(Cliente):-
  write(Cliente),
  write(', voce ja e cliente de algum outro banco?'),
  nl.

perg2(Cliente):-
  write(Cliente),
  write(', qual e a sua renda mensal em salarios minimos?'),
  nl.
```

```
perg3(Cliente):-
  write(Cliente),
  write(', voce deseja uma aplicacao a curto, medio '),
  write(' ou longo prazo?'),
  nl.


% Programas que correspondem a funcoes de critica das respostas

critica_renda(Renda):-
  integer(Renda),
  renda>=0,
  !.

critica_renda(_):-
  write('O valor aceitavel para a renda e inteiro e '),
  write('maior que zero.'), nl,
  write('Responda novamente, por favor.'), nl,
  fail.

critica_prazo(Prazo):-
  pertence(Prazo,[curto,medio,longo]),
  !.

critica_prazo(_):-
  write('Os valores aceitaveis para o prazo sao curto, medio'),
  write(' ou longo.'), nl,
  write('Responda novamente, por favor.'), nl,
  fail.

% Programa auxiliar

pertence(X,[X|_]).

pertence(X,[Y|Cauda]):-
  pertence(X,Cauda).
```

As perguntas implementadas na Ba podem ser de três tipos [4]:

- **afirmativas ou negativas**, para perguntas que exigem resposta sim ou não;
- **únicas**, para perguntas que são feitas uma única vez;
- **várias**, para perguntas que admitem várias respostas. Neste caso, uma mesma pergunta é feita várias vezes, até que o usuário responda *basta* ou *pare*.

A seguir temos o exemplo de um pequena BC que contém os três tipos de perguntas.

Tomemos uma BC que representa o conhecimento de um orientador vocacional, cuja função é selecionar um curso universitário adequado a um estudante, no caso o usuário do sistema. Para chegar às suas conclusões, o sistema deverá extrair diversas informações do usuário, fazendo-lhe sucessivas perguntas.

```
% Relacoes perguntaveis

% Exemplos de perguntas do tipo afirmativa-negativa

perguntavel(dedicacao).
perguntavel(criatividade).
perguntavel(memorizacao).

% Exemplos de perguntas do tipo unica

perguntavel(idade_tempo(Idade,Tempo)).
perguntavel(aprendizado(Aprend)).
perguntavel(desempenho(Desemp)).

% Exemplos de perguntas do tipo varias

perguntavel(disciplinas(D)).
perguntavel(cursos(C)).

% Perguntas referentes as metas perguntaveis

pergunta(dedicacao,perg(escreve('Voce se considera um aluno
        dedicado?')),[s-n]).

pergunta(criatividade,perg(escreve('Voce e uma pessoa criativa?')),
        [s-n]).
```

```
pergunta(memorizacao,perg(escreve('Voce tem facilidade de
        memorizacao?')),[s-n]).

pergunta(idade_tempo(Idade,Tempo),perg(escreve('Qual e a sua idade
        (anos)?','Ha quanto tempo voce concluiu o segundo grau
        (anos)?')),[u,critica_idade,critica_tempo]).

pergunta(aprendizado(Aprend),perg(escreve('Voce prefere o aprendizado
        pratico ou teorico?')),[u,critica_aprend]).

pergunta(desempenho(Desemp),perg(escreve('Em que area voce apresenta
        melhor desempenho?')),[u,critica_desemp]).

pergunta(disciplinas(D),perg(escreve('Quais sao as disciplinas de sua
       preferencia?')),[v,critica_disc]).

pergunta(cursos(C),perg(escreve('Que cursos voce ja pensou em fazer?')),
       [v,critica_cursos]).

% Programas que representam funcoes de critica das respostas

critica_idade(Idade):-
       Idade>0,
       !.

critica_idade(_):-
       write('A sua idade deve ser um numero positivo.'),nl,
       write('Responda novamente, por favor.'),nl,
       fail.

critica_tempo(Tempo):-
       Tempo>=0,
       !.

critica_tempo(_):-
       write('A resposta deve ser um numero positivo.'),nl,
       write('Caso voce ainda esteja cursando o 2o grau,'),nl,
       write('responda "0"'.),nl,
       fail.
```

```
critica_aprend(Aprend):-
        pertence(Aprend,[pratico,teorico]),
        !.

critica_aprend(_):-
        write('As respostas validas sao: pratico ou teorico.'),nl,
        write('Responda novamente, por favor.'),nl,
        fail.

critica_desemp(Desemp):-
        pertence(Desemp,[exatas,humanas,biologicas]),
        !.

critica_desemp(_):-
        write('As respostas validas sao: exatas,'),nl,
        write('humanas ou bilogicas.'),nl,
        write('Responda novamente, por favor.'), nl,
        fail.

critica_disc(Disc):-
        pertence(Disc,[matematica,fisica,quimica,ingles,
            historia,geografia,biologia,portugues,educacao_fisica]),
        !.

critica_disc(_):-
        write('Resposta invalida.'),nl,
        write('Responda novamente, por favor.'),nl,
        fail.

critica_curso(Curso):-
        pertence(Curso,[medicina,odontologia,administracao,direito,
              engenharia,fisica,quimica,matematica,geografia,historia,
              computacao,artes,arquitetura,biologia,veterinaria,
              fisioterapia,farmacia,pedagogia,letras,estatistica]),
        !.

critica_curso(_):-
        write('Resposta invalida.'),nl,
        write('Responda novamente, por favor.'),nl,
        fail.
```

3.2.2 O Motor de Inferência (MI)

Este é o módulo do NSE responsável pelo desenvolvimento de uma linha de raciocínio, que o permitirá tirar conclusões dado um determinado problema.

Para tanto, o MI dispõe [5]:

- do conhecimento de um especialista, o qual estará armazenado sob a forma de cláusulas Prolog na BC;
- das informações do problema em estudo, obtidas pelo MCD através de perguntas feitas ao usuário;
- dos dados temporários gravados na BD durante a execução do sistema.
- de método(s) de inferência que o permitirão manipular todos os ítens acima de forma a chegar a uma solução satisfatória dado um problema. Os principais métodos, o *backward chaining* e o *forward chaining*, serão descritos posteriormente.

Após a exposição das conclusões obtidas pelo MI, o usuário poderá perguntar qual a linha de raciocínio desenvolvida até chegar-se às mesmas. Se assim acontecer, o ME será ativado, e explicará todo o processo (figura 11).

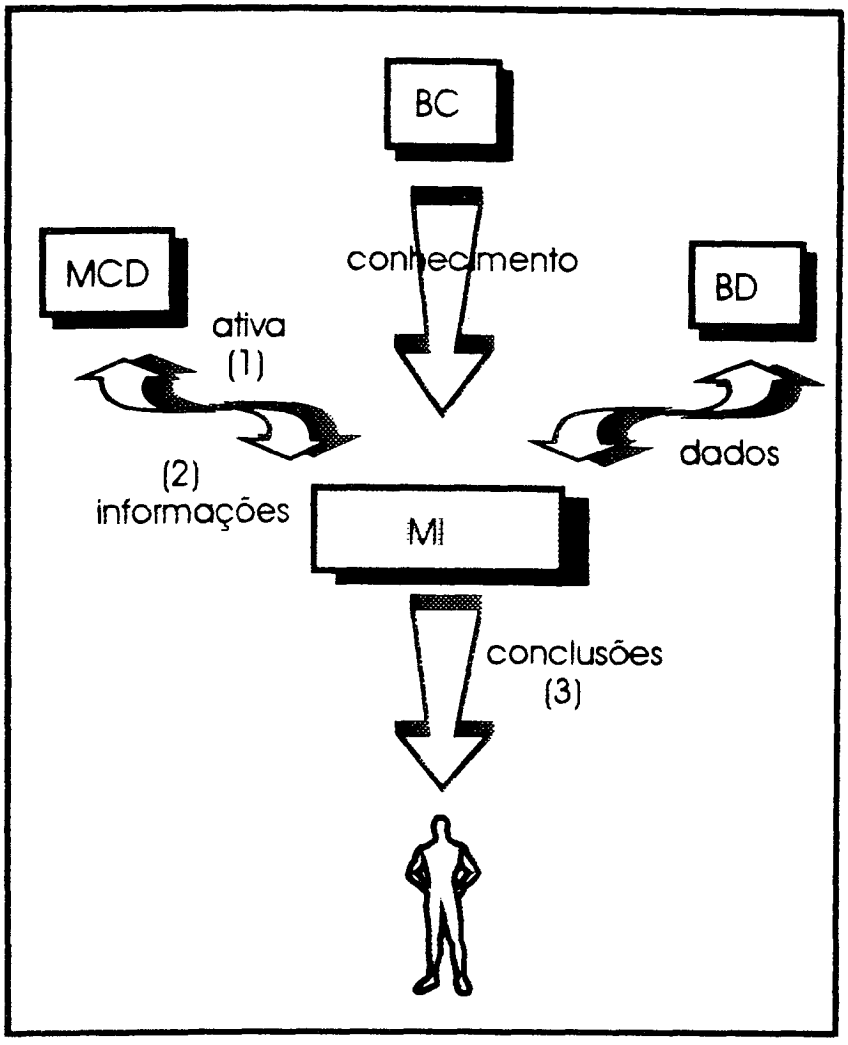

Figura 11: O Fluxo de Informações no MI

Para um MI que manipula regras, a linha de raciocínio pode ser desenvolvida por dois processos [5]:

- **encadeamente regressivo ou "backward chaining".** Aqui parte-se da suposição de que cada provável solução é verdadeira. Feito isso, tenta-se reunir evidências que comprovem ser correta a solução previamente considerada. Tais evidências são buscadas nas informações fornecidas pelo usuário (figura 12).

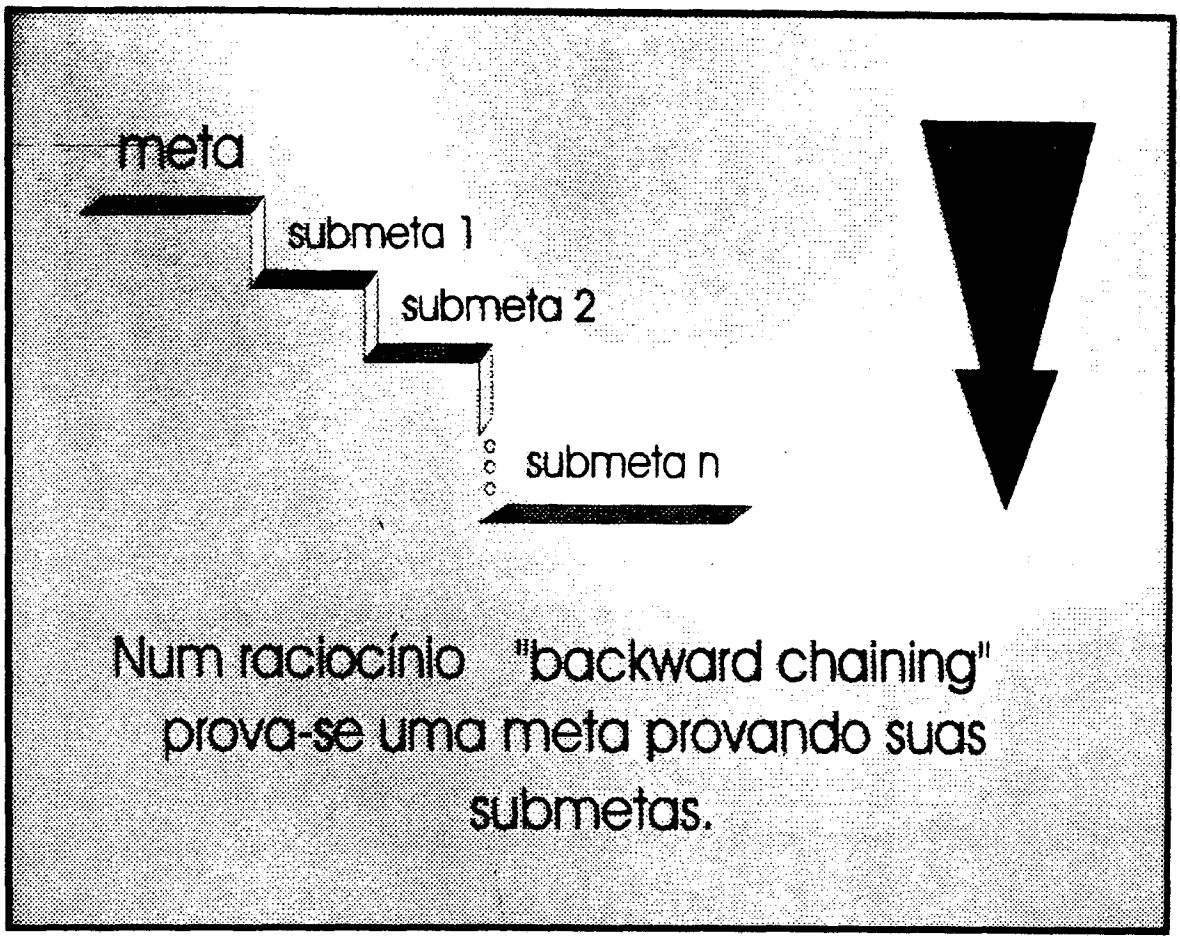

Figura 12: O Raciocínio Backward Chaining

Resumindo, esta estratégia corresponde a perguntar:

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

Este é o raciocínio embutido no NSE do ANSEsp.

- **encadeamento progressivo ou "forward chaining"**. Aqui chega-se a uma solução para o problema a partir das informações fornecidas pelo usuário. Este processo consiste em analisar estes dados, baseando-se nas regras de conhecimento, até chegar-se a uma conclusão (figura 13).

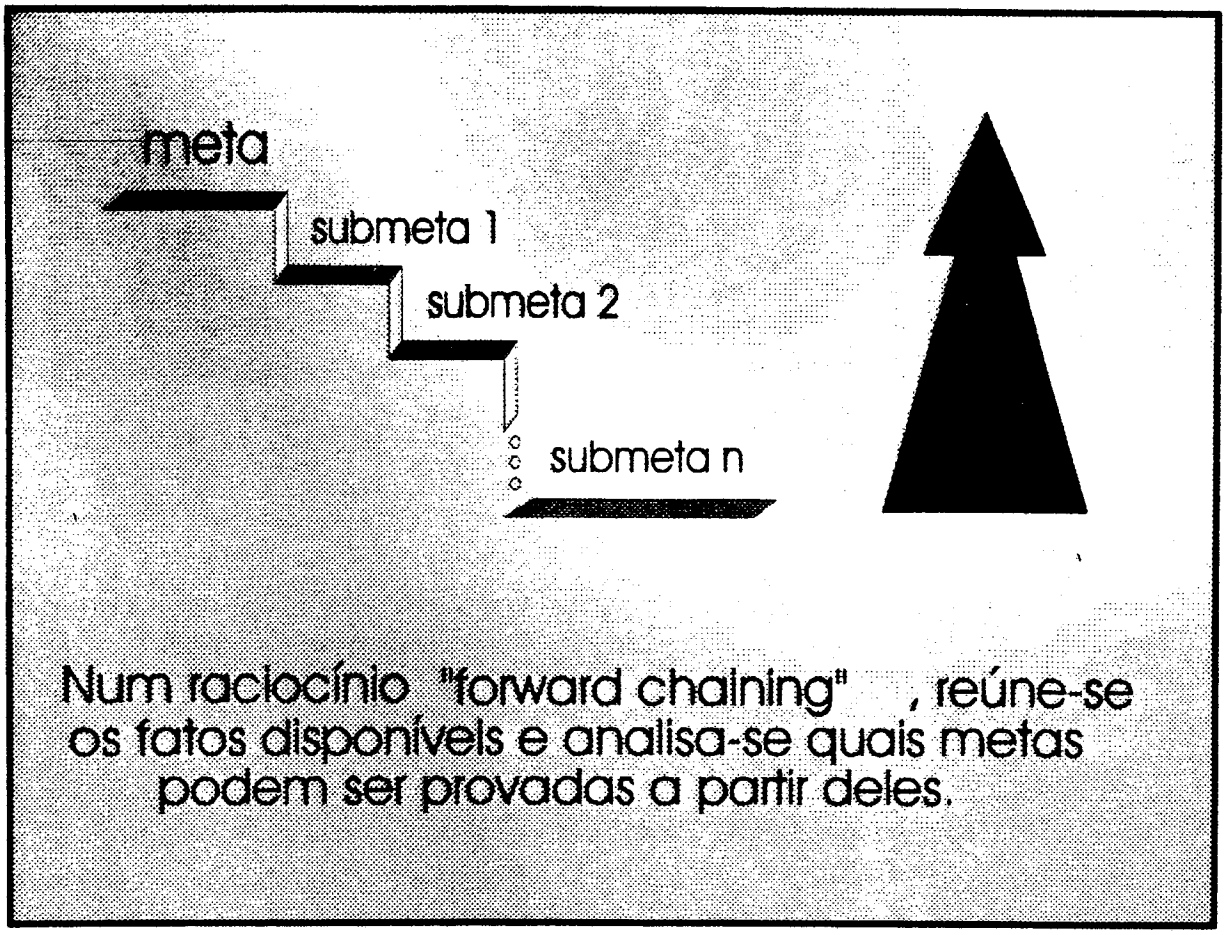

Figura 13: O Raciocínio Forward Chaining

Resumindo, esta estratégia corresponde a perguntar:

" O que é possível concluir a partir dos dados que temos?"

O MI tem como principal ferramenta um meta-interpretador, o *prove/3*, o qual utiliza um raciocínio do tipo "backward chaining". Isto significa que o programa *prove* parte de uma hipótese e tenta encontrar evidências que a suportem. Como a BC está internamente representada por cláusulas Prolog, o programa *prove/3* tenta provar uma meta provando suas submetas [5].

O programa *prove/3* é constituído de várias cláusulas, sendo que uma ou um grupo delas trata de um tipo específico de conhecimento.

Ele é representado da seguinte forma:

prove(Meta,Regras,Prova), onde:

- **Meta:** argumento de entrada, consiste na meta a ser provada. Pode pertencer a qualquer um dos tipos de relação da BC (perguntável, primitiva, decisão, sistema ou definida), ou pode ser uma negação, conjunção ou disjunção de metas;
- **Regras:** argumento de entrada ou de saída, contém as regras provadas até o momento, na ordem em que foram utilizadas. As informações armazenadas neste argumento são repassadas ao ME pelo MCD sempre que o usuário perguntar porque foi interrogado;
- **Prova:** este é um argumento de saída. Se o programa *prove* é bem sucedido e este argumento encontra-se instanciado, o mesmo conterá a árvore de prova com todas as inferências realizadas durante o processo. As informações armazenadas neste argumento são utilizadas pelo ME sempre que o usuário perguntar como foi realizada a prova corrente.

3.2.3 O Módulo de Explicação (ME)

É o módulo do NSE responsável por esclarecer as dúvidas do usuário, durante ou após o processamento do raciocínio.

O ME pode fornecer três tipos de explicações [8,2]:

- **porque:** enquanto busca as soluções para o problema em questão, o MI necessita de algumas informações. O MCD é então ativado de modo a questionar o usuário. Se o mesmo perguntar o *porquê* da pergunta, o ME será ativado pelo MCD, e justificará a necessidade da mesma (figura 14).

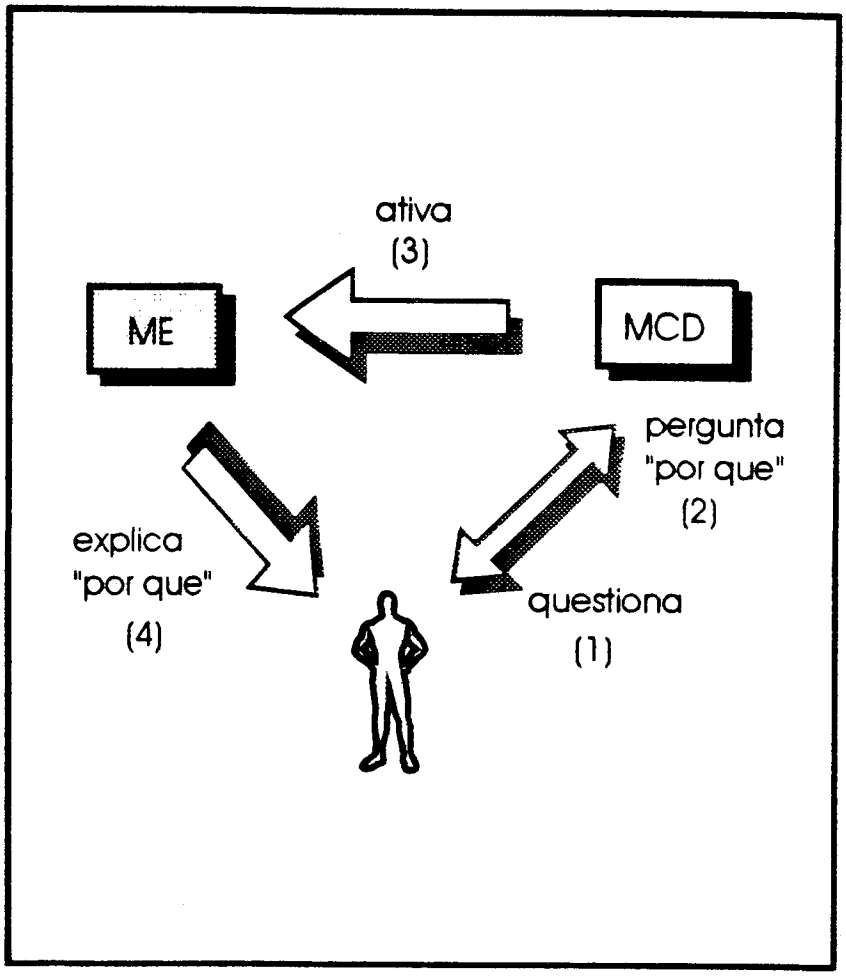

Figura 14: A Explicação '*Por_Que*'

- **como:** ao final do processo de inferências, o MI chega às suas conclusões e as expõe ao usuário, se for requisitado. Se, além disso, o usuário quiser saber *como* as conclusões foram obtidas, o ME será ativado, e fornecerá toda a árvore de prova do processo. Esta contém todas as provas intermediárias e relevantes realizadas durante o desenvolvimento do raciocínio. O ME fornece as conclusões tiradas a partir destas provas até chegar à solução final (figura 15).

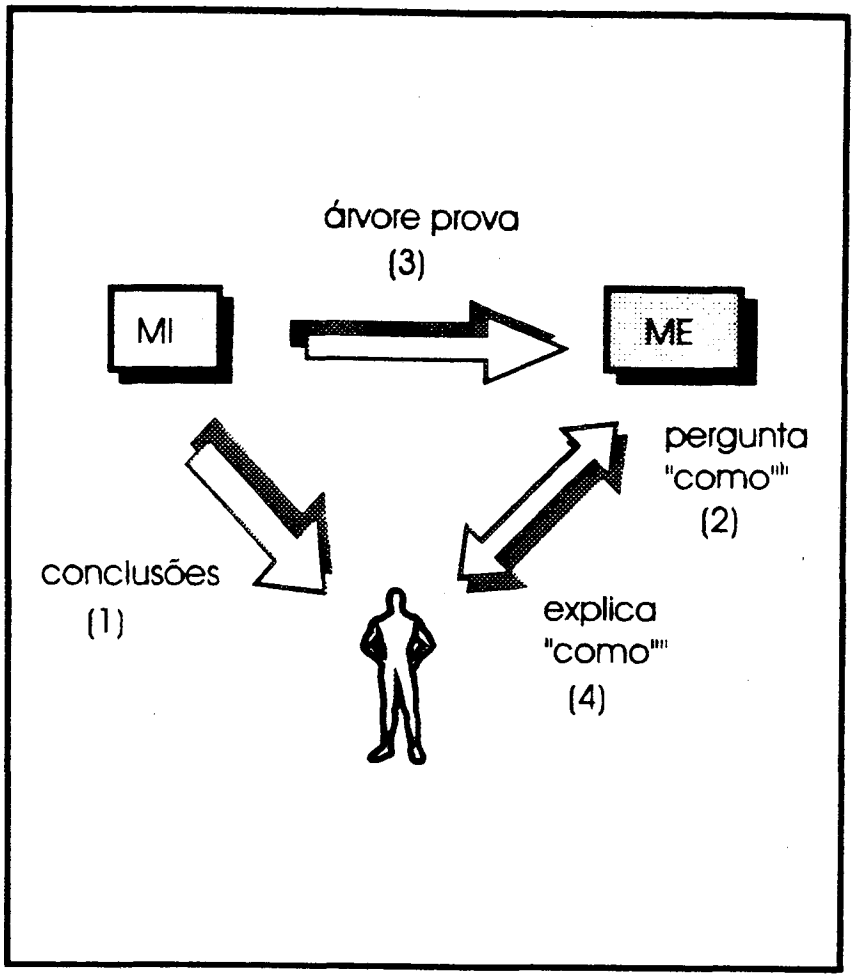

Figura 15: A Explicação '*Como*'

- **what_if:** caso o usuário deseje saber qual seria o resultado obtido se uma ou mais informações fossem modificadas, perguntará *what_if* ao final da execução do sistema. Se assim acontecer, o ME perguntará quais as condições a serem substituídas, apagará da BD antigas informações referentes a tais condições e ativará o MI, que reiniciará o processo de inferências. Assim, novas conclusões serão obtidas (figura 16).

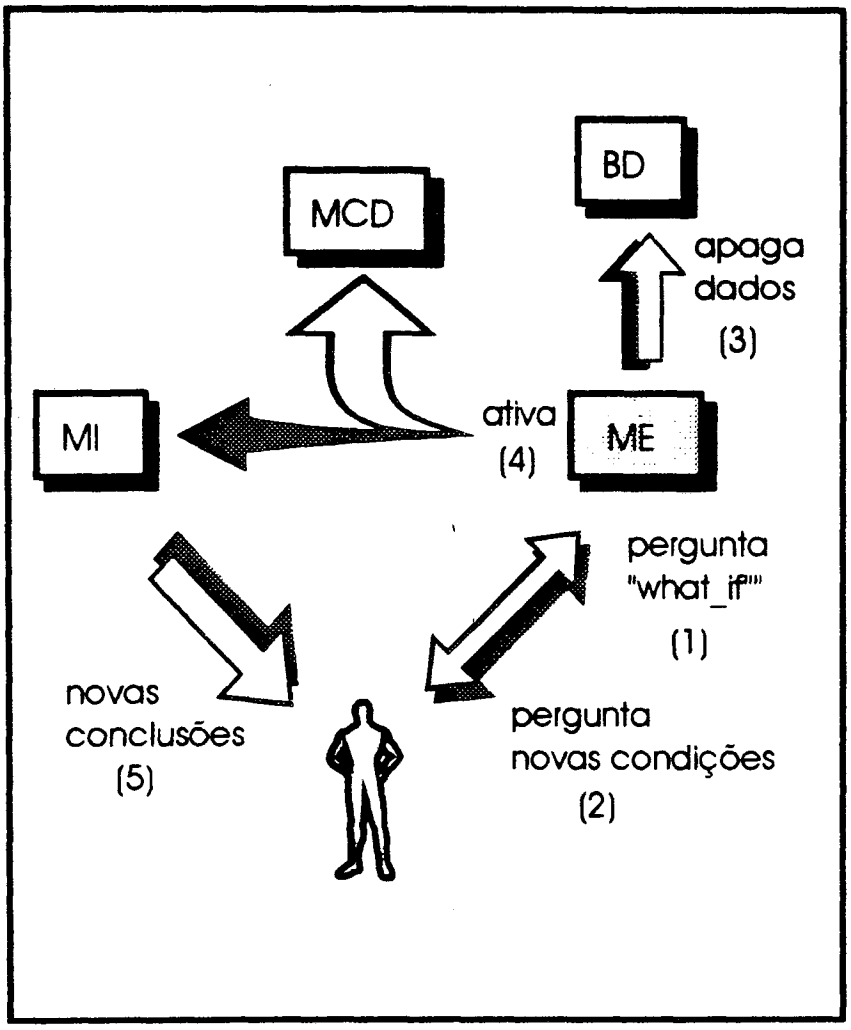

Figura 16: A Explicação '*What_If*'

A seguir temos a descrição da Base de Dados utilizada pelo Sistema Especialista construído a partir do ANSEsp.

3.3 A Base de Dados (BD)

Esta consiste na área de trabalho de um SE. Nesta são gravadas todas as conclusões intermediárias e provas realizadas durante a execução dos programas. Podemos dizer que a BD funciona como a "memória" do sistema, e portanto sua utilização traz duas vantagens [8]:

- Permite fornecer ao usuário toda a árvore de prova correspondente às conclusões obtidas;
- Não torna necessária e repetição de uma mesma pergunta ou de uma mesma prova.

4 A Idéia do Ambiente

O Ambiente para Construção de Núcleos Específicos de Sistemas Especialistas (ANSEsp) é constituído de um NSE completo, ou seja, capaz de manipular *qualquer* Base de Conhecimento [2].

O Engenheiro de Sistemas é o profissional responsável pelas alterações no sistema, de forma a torná-lo um NSE adequado a uma aplicação específica. Este processo pode ser visualizado na figura 17 [8].

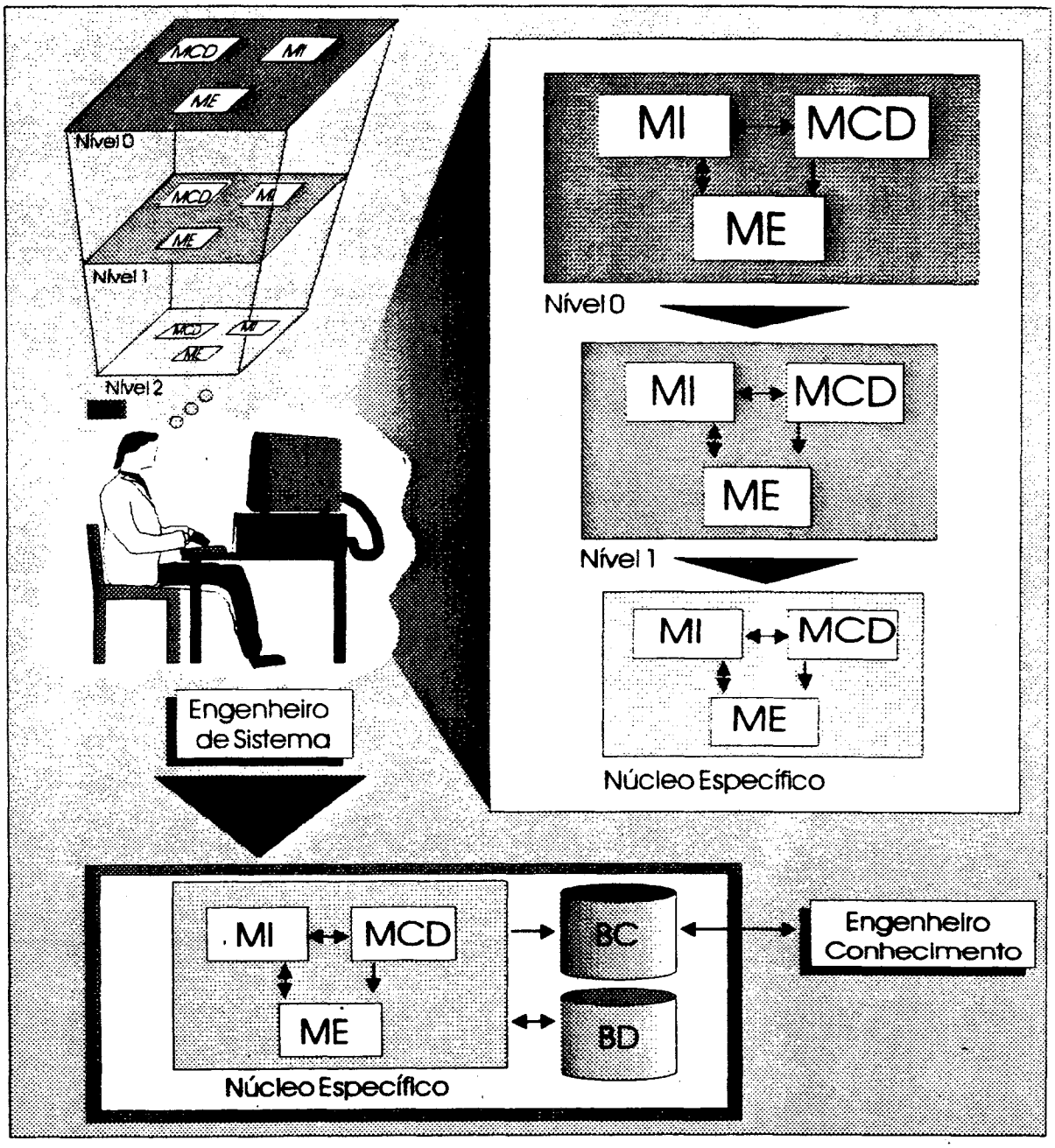

Figura 17: O Ambiente para Construção de Núcleos Específicos de Sistemas Especialistas-ANSEsp

O nível 0 do NSE corresponde ao núcleo completo do ANSEsp. A sua depuração pode ser realizada em duas fases:

- **Passagem do nível 0 para o nível 1:** consiste na eliminação de todo o código irrelevante para a aplicação desejada;
- **Passagem do nível 1 para o nível 2:** consiste na modelagem que o tornará realmente específico, através da adição de características particulares à aplicação.

4.1 A Passagem do Nível 0 para o Nível 1

Esta depuração neste nível está intimamente ligada á estruturação do código do ANSEsp. Para realizá-la, é importante saber como funciona a comunicação entre seus módulos e entre seus módulos e o restante do Sistema Especialista [8].

- **A comunicação entre o MCD e a BC:** para interagir com o usuário, o MCD necessita de informações como: texto da pergunta, tipo da pergunta, funções de crítica das respostas, etc. Estas informações estão armazenadas na Base Algorítmica, que corresponde a uma parte da BC.
- **A comunicação entre o MCD e o MI:** durante o processo de inferências, o MI necessita de informações referentes ao problema em questão. Para obtê-las, ele ativa o MCD, que faz as perguntas ao usuário e retorna as respostas obtidas ao MI. A comunicação entre estes dois módulos é feita através do programa *faz_pergunta/4*.
- **A comunicação entre o MCD e o ME:** enquanto questiona o usuário, o MCD pode obter um *por_que* como resposta, o que significa que se deseja esclarecimentos sobre a necessidade da pergunta. Neste caso, o MCD ativa o ME, responsável pelas justificativas. A comunicação entre estes dois módulos é feita através do programa *explica_porque/1*.
- **A comunicação entre o MI e a BC:** no processo de inferências, o MI utiliza o conhecimento de um ou mais especialistas, o qual é armazenado na BC sob a forma de cláusulas Prolog. Dessa forma, a BC é continuamente consultada enquanto o MI procura provar suas hipóteses e tirar suas conclusões.
- **A comunicação entre o MI e o ME:** após a exposição das conclusões obtidas pelo sistema, o usuário pode pedir todo o raciocínio desenvolvido até chegar-se a elas. Quando isso acontece, o MI ativa o ME através do programa *explica_como/1*, deixando sob sua responsabilidade as explicações.
- **A comunicação entre o MI e a BD:** enquanto desenvolve seu raciocínio, o MI grava e acessa na Base de Dados todas as conclusões intermediárias obtidas. Dessa forma, a BD funciona como a área de trabalho do sistema, onde dados são continuamente gravados e acessados.

- **A comunicação entre o NSE e o usuário:** o NSE, juntamente com a BC e a BD, formará um Sistema Especialista. Como tal, deverá interagir com o usuário de forma a imitar ao máximo o comportamento de um especialista humano. Esta comunicação é feita através dos seguintes programas:
  - *pergunta/3, pergunta/4, pergunta_livre/3* e *pergunta_livre/4*, através dos quais o NSE questiona o usuário. O módulo que manipula estes programas é o MCD.
  - *explica_porque/1*, programa manipulado pelo ME.
  - *explica_como/1*, cujo responsável é o ME.
  - *what_if/1*, também manipulado pelo ME.

O conhecimento de um ou mais especialistas é representado na BC sob a forma de cláusulas Prolog, onde podem ser utilizadas cinco tipos de relações, já detalhadas anteriormente: *perguntáveis, primitivas, definidas*, de *sistema* e de *decisão*.O NSE embutido no ANSEsp manipula qualquer BC porque seu Motor de Inferência é capaz de processar todas estas relações.

Se, por exemplo, a base não faz uso de relações do tipo decisão, o Engenheiro de Sistemas pode eliminar dos programas que compõem o MI todas as cláusulas que manipulam este tipo de relação.

Alterações também podem ser feitas no MCD, que é o módulo responsável pela interação com o usuário. Para elaborar as perguntas, ele necessita de informações contidas na BC.

Existem quatro tipos de perguntas que podem ser declaradas na BC, as quais diferem pelo número de argumentos e pelo módulo responsável pela validação de suas respostas. São elas: *pergunta/3, pergunta/4, pergunta_livre/3* e *pergunta_livre/4*. Caso a BC não utilize todos os grupos de perguntas permitidos, pode-se retirar do MCD todo o código correspondente ao tratamento dos grupos não declarados.

Desta forma, o NSE resultante nesta passagem ficará restrito ao código necessário, sem que este fato altere seu funcionamento. Pelo contrário, a otimização de código resultará em um aumento de eficiência.

4.2 A Passagem do Nível 1 para o Nível 2

Nesta passagem não há uma simples redução do código. Aqui temos modificações em algumas características padrões do NSE, de forma que o sistema resultante no nível 2 tenha as particularidades de um NSE destinado a uma aplicação bastante específica.

As modificações podem ser feitas a nível de:

- **Interação com a BD;**
- **Interação com o usuário;**
- **Manipulação de informações imprecisas.**

A seguir temos o exemplo de uma Base de Conhecimento implementada de forma a ser manipulada pelo ANSEsp.

5 Base de Conhecimento sobre Seleção de Vinhos

Esta Base de Conhecimento contém regras definidas por especialistas na seleção de vinhos adequados a uma dada refeição. Todo o conhecimento está representado nas cláusulas, que consistem nas relações detalhadas na seção 3.1, e nos tipos de perguntas a serem feitas ao usuário no decorrer da execução do programa, detalhadas na seção 3.2.1.

```
% ************************   Base de hipoteses   ****************************

% Aqui temos todas as hipoteses a serem provadas pelo sistema. O usuario
% seleciona uma delas, o sistema supoe que a mesma seja verdadeira (raciocinio
% bacKward chaining), e inicia o processo de inferencias utilizando o
% conhecimento armazenado na BC e as respostas do usuario. Vale lembrar que
% a prova de uma hipotese pode ativar a prova de outras.

para_provar(vinho(fc,C,E,V)).
para_provar(cor_recomendada(fc,C,E,C)).
para_provar(corpo_recomendado(fc,C,E,C)).
para_provar(docura_recomendada(fc,C,E,C)).
para_provar(melhor_corpo(fc,C,E,C)).
para_provar(melhor_docura(fc,C,E,C)).


% **  rotina de leitura para ser usada pelo MCD (esta rotina e opcional) **

rotina_leitura:-
    write('-> ').
```

```
% ********************** Programas Auxiliares **************************

el([]):-!.
el([X|Z]):-
  write(X),
  nl,
  el(Z).

pertence(X,[X|_]).
pertence(X,[R|Z]):-
  pertence(X,Z).

min(X,[X]).

min(X,[X,Y]):-
   X =< Y,
   !.

min(Y,[X,Y]):-
   !.

min(Min,[X,Y|Z]):-
   X =< Y,
   !,
   min(Min,[X|Z]).

min(Min,[X,Y|Z]):-
   min(Min,[Y|Z]).

% ******************** Relacoes Perguntaveis ************************

% Estas sao as metas perguntaveis, as quais sao manipuladas pelo MCD.
% Enquanto desenvolve o raciocionio, o MI necessita de informacoes
% especificas do problema em estudo. Ele entao ativa o MCD,
% o qual obtem tais informacoes questionando o usuario.
% As perguntas sao definidas nas relacoes perguntaveis.

perguntavel(prato_principal(Prato,C)).
perguntavel(tem_molho).
perguntavel(molho(Tipo,C)).
perguntavel(sabor(Tipo,C)).
```

```
perguntavel(tem_peru).
perguntavel(tem_vitela).
perguntavel(corpo_preferido(Tipo)).
perguntavel(cor_preferida(Preferencia)).
perguntavel(docura_preferida(Preferencia)).


% Clausulas que contem o texto das perguntas e indicam as funcoes de critica
% a serem ativadas na validacao das respostas obtidas.

pergunta(prato_principal(Prato,C),
        pr(el(['O prato principal sera carne ave ou peixe ?',
               'Qual o coeficiente de certeza ?'])),
        [unica,critica_prato,critica_certeza]).

pergunta(tem_molho,pr(el(['O prato sera coberto por algum molho ?'])),
        [s-n]).

pergunta(molho(Tipo,C),
        pr(el(['O molho sera muito temperado, doce, cremoso ou tomate ?',
               'Qual o coeficiente de certeza ?'])),
        [unica,critica_molho,critica_certeza]).

pergunta(sabor(Tipo,C),
        pr(el(['O sabor da refeicao sera delicado medio ou forte?',
               'Qual o coeficiente de certeza ?'])),
        [unica,critica_sabor,critica_certeza]).

pergunta(tem_peru,pr(el(['O prato principal sera peru?'])),[s-n]).

pergunta(tem_vitela,pr(el(['O prato principal sera vitela?'])),[s-n]).

pergunta(corpo_preferido(Tipo),
        pr(el(['Voce geralmente prefere vinho leve medio ou pesado ?'])),
        [unica,critica_corpo]).

pergunta(cor_preferida(Preferencia),
        pr(el(['Voce geralmente prefere vinho tinto ou branco ?'])),
        [unica,critica_cor]).
```

```
pergunta(docura_preferida(Preferencia),
         pr(el(['Voce geralmente prefere vinho doce medio ou seco ?'])),
         [unica,critica_docura]).

% Funcoes de critica das respostas

critica_certeza(Cf):-
      Cf=<1,Cf>0,
      !.

critica_certeza(Cf):-
      write('O valor valido deve estar entre 0 e 1'),nl,
      write('Por favor, responda novamente.'),nl,
      fail.

critica_prato(Prato):-
      pertence(Prato,[carne,ave,peixe]).

critica_prato(Prato):-
      write('O valor valido e '),
      write('carne, ave ou peixe '),nl,
      write('Por favor, responda novamente.'),nl,
      fail.

critica_molho(Molho):-
      pertence(Molho,[muito_temperado,doce,cremoso,tomate]).

critica_molho(Molho):-
      write('O valor valido e '),
      write('muito_temperado, doce, cremoso ou tomate '),nl,
      write('Por favor, responda novamente.'),nl,
      fail.

critica_sabor(Sabor):-
      pertence(Sabor,[delicado,medio,forte]).

critica_sabor(Sabor):-
      write('O valor valido e '),
      write('delicado, medio ou forte '),nl,
      write('Por favor, responda novamente.'),nl,
      fail.
```

```
critica_corpo(Corpo):-
      pertence(Corpo,[leve,medio,pesado]).

critica_corpo(Corpo):-
      write('O valor valido e '),
      write('leve, medio ou pesado '),nl,
      write('Por favor, responda novamente.'),nl,
      fail.

critica_cor(Cor):-
      pertence(Cor,[tinto,branco]).

critica_cor(Cor):-
      write('O valor valido e '),
      write('tinto ou branco '),nl,
      write('Por favor, responda novamente.'),nl,
      fail.

critica_docura(Docura):-
      pertence(Docura,[doce,medio,seco]).

critica_docura(Docura):-
      write('O valor valido e '),
      write('doce, medio ou seco '),nl,
      write('Por favor, responda novamente.'),nl,
      fail.

% ********************  Relacoes Primitivas  ******************************

primitiva(min(Min,Lista)).

%*********************  Relacoes de Decisao  ******************************

% Estas relacoes envolvem coeficientes de certeza na prova das hipoteses

cor_recomendada(fc,C,1,tinto):-
    melhor_cor(fc,C,Q,tinto).

cor_recomendada(fc,C,1,branco):-
    melhor_cor(fc,C,Q,branco).
```

```
cor_recomendada(fc,C,0.2,tinto):-
    cor_preferida(tinto),
    melhor_cor(fc,C,Q,tinto).

cor_recomendada(fc,C,0.2,branco):-
    cor_preferida(branco),
    melhor_cor(fc,C,Q,branco).

corpo_recomendado(fc,C,1,leve):-
    melhor_corpo(fc,C,Q,leve).

corpo_recomendado(fc,C,1,medio):-
    melhor_corpo(fc,C,Q,medio).

corpo_recomendado(fc,C,1,pesado):-
    melhor_corpo(fc,C,Q,pesado).

corpo_recomendado(fc,C,0.2,leve):-
    corpo_preferido(leve),
    melhor_corpo(fc,C,Q,leve).

corpo_recomendado(fc,C,0.2,medio):-
    corpo_preferido(medio),
    melhor_corpo(fc,C,Q,medio).

corpo_recomendado(fc,C,0.2,pesado):-
    corpo_preferido(pesado),
    melhor_corpo(fc,C,Q,pesado).

corpo_recomendado(fc,C,1,medio):-
    corpo_preferido(leve),
    melhor_corpo(fc,C,Q,pesado).

corpo_recomendado(fc,C,1,medio):-
    corpo_preferido(pesado),
    melhor_corpo(fc,C,Q,leve).

docura_recomendada(fc,C,1,seco):-
    melhor_docura(fc,C,Q,seco).
```

```
docura_recomendada(fc,C,1,medio):-
    melhor_docura(fc,C,Q,medio).

docura_recomendada(fc,C,1,doce):-
    melhor_docura(fc,C,Q,doce).

docura_recomendada(fc,C,1,medio):-
    melhor_docura(fc,C,Q,seco).

docura_recomendada(fc,C,0.2,seco):-
    melhor_docura(fc,C,Q,seco),
    docura_preferida(seco).

docura_recomendada(fc,C,0.2,medio):-
    melhor_docura(fc,C,Q,medio),
    docura_preferida(medio).

docura_recomendada(fc,C,0.2,doce):-
    melhor_docura(fc,C,Q,doce),
    docura_preferida(doce).

docura_recomendada(fc,C,1,medio):-
    melhor_docura(fc,C,Q,doce),
    docura_preferida(seco).

docura_recomendada(fc,C,1,medio):-
    melhor_docura(fc,C,Q,seco),
    docura_preferida(doce).

docura_recomendada(fc,1,1,seco):-
    docura_preferida(seco).

docura_recomendada(fc,1,1,medio):-
    docura_preferida(medio).

docura_recomendada(fc,1,1,doce):-
    docura_preferida(doce).

caracteristica(temperado,C):-
    tem_molho,
    molho(muito_temperado,C).
```

```
vinho(fc,C,1,gamay):-
    cor_recomendada(fc,C1,E1,tinto),
    corpo_recomendado(fc,C2,E2,medio),
    (docura_recomendada(fc,C3,E3,medio);
       docura_recomendada(fc,C3,E3,doce)),
    min(C,[C1,C2,C3]).

vinho(fc,C,1,chablis):-
    cor_recomendada(fc,C1,E1,branco),
    corpo_recomendado(fc,C2,E2,leve),
    docura_recomendada(fc,C3,E3,seco),
    min(C,[C1,C2,C3]).

vinho(fc,C,1,savignon_blanc):-
    cor_recomendada(fc,C1,E1,branco),
    corpo_recomendado(fc,C2,E2,medio),
    docura_recomendada(fc,C3,E3,seco),
    min(C,[C1,C2,C3]).

vinho(fc,C,1,chardonnay):-
    cor_recomendada(fc,C1,E1,branco),
    (corpo_recomendado(fc,C2,E2,medio);
       corpo_recomendado(fc,C2,E2,pesado)),
    (docura_recomendada(fc,C3,E3,seco);
       docura_recomendada(fc,C3,E3,medio)),
    min(C,[C1,C2,C3]).

vinho(fc,C,1,soave):-
    cor_recomendada(fc,C1,E1,branco),
    corpo_recomendado(fc,C2,E2,leve),
    (docura_recomendada(fc,C3,E3,seco);
       docura_recomendada(fc,C3,E3,medio)),
    min(C,[C1,C2,C3]).

vinho(fc,C,1,riesling):-
    cor_recomendada(fc,C1,E1,branco),
    (corpo_recomendado(fc,C2,E2,leve);
       corpo_recomendado(fc,C2,E2,medio)),
    (docura_recomendada(fc,C3,E3,medio);
       docura_recomendada(fc,C3,E3,doce)),
```

```
    min(C,[C1,C2,C3]).

vinho(fc,C,1,geverztraminer):-
    cor_recomendada(fc,C1,E1,branco),
    corpo_recomendado(fc,C2,E2,pesado),
    caracteristica(Molho,C3),
    Molho=temperado,
    min(C,[C1,C2,C3]).

vinho(fc,C,1,chenin_blanc):-
    cor_recomendada(fc,C1,E1,branco),
    corpo_recomendado(fc,C2,E2,leve),
    (docura_recomendada(fc,C3,E3,medio);
       docura_recomendada(fc,C3,E3,doce)),
    min(C,[C1,C2,C3]).

vinho(fc,C,1,valpolicella):-
    cor_recomendada(fc,C1,E1,tinto),
    corpo_recomendado(fc,C2,E2,leve),
    min(C,[C1,C2]).

vinho(fc,C,1,cabernet_sauviggnon):-
    cor_recomendada(fc,C1,E1,tinto),
    (docura_recomendada(fc,C2,E2,seco);
       docura_recomendada(fc,C2,E2,medio)),
    min(C,[C1,C2]).

vinho(fc,C,1,zinfandel):-
    cor_recomendada(fc,C1,E1,tinto),
    (docura_recomendada(fc,C2,E2,seco);
       docura_recomendada(fc,C2,E2,medio)),
    min(C,[C1,C2]).

vinho(fc,C,1,pinot_noir):-
    cor_recomendada(fc,C1,E1,tinto),
    corpo_recomendado(fc,C2,E2,medio),
    docura_recomendada(fc,C3,E3,medio),
    min(C,[C1,C2,C3]).

vinho(fc,C,1,burgundy):-
    cor_recomendada(fc,C1,E1,tinto),
```

```
    corpo_recomendado(fc,C2,E2,pesado),
    min(C,[C1,C2]).

melhor_cor(fc,C,0.9,tinto):-
    prato_principal(ave,C),
    not(tem_vitela).

melhor_cor(fc,C,1,branco):-
    prato_principal(peixe,C).

melhor_cor(fc,C,0.8,tinto):-
    prato_principal(ave,C),
    tem_peru.

melhor_cor(fc,C,0.5,branco):-
    prato_principal(ave,C),
    tem_peru.

melhor_cor(fc,C,0.9,branco):-
    prato_principal(ave,C),
    not(tem_peru).

melhor_cor(fc,C,0.3,tinto):-
    prato_principal(ave,C),
    not(tem_peru).

melhor_cor(fc,C,1,tinto):-
    not(prato_principal(peixe,C1)),
    tem_molho,
    ( molho(tomate,C2);
      molho(cremoso,C2)),
    ifthen(var(C1),C1=0),
    C3 is 1 - C1,
    min(C,[C2,C3]).

melhor_corpo(fc,C,0.8,leve):-
    sabor(delicado,C).

melhor_corpo(fc,C,1,pesado):-
    tem_molho,
    molho(muito_temperado,C).
```

```
melhor_corpo(fc,C,0.3,leve):-
    sabor(medio,C).

melhor_corpo(fc,C,0.6,medio):-
    sabor(medio,C).

melhor_corpo(fc,C,0.3,pesado):-
    sabor(medio,C).

melhor_corpo(fc,C,0.4,medio):-
    sabor(forte,C).

melhor_corpo(fc,C,0.8,pesado):-
    sabor(forte,C).

melhor_corpo(fc,C,0.4,medio):-
    tem_molho,
    molho(cremoso,C).

melhor_corpo(fc,C,0.6,pesado):-
    tem_molho,
    molho(cremoso,C).

melhor_docura(fc,C,0.9,doce):-
    tem_molho,
    molho(doce,C).

melhor_docura(fc,C,0.4,medio):-
    tem_molho,
    molho(doce,C).
```

6 Conclusões

Os Sistemas Baseados em Conhecimento consistem em um importante avanço da tecnologia no que diz respeito à automatização, ainda limitada, do conhecimento e do racioçínio humano. As ferramentas que auxiliam a construção de SBCs permitem a otimização de esforços no desenvolvimento de tais sistemas, enfatizando as etapas de aquisição, estruturação, representação e validação do conhecimento, o qual é assim adaptado à ferramenta utilizada.

O ANSEsp contém todo o código responsável pela manipulação do conhecimento e dos dados representativos de problemas pertencentes a um domínio selecionado. Este código pode ser adaptado às necessidades deste domínio, eliminando-se partes irrelevantes ou alterando-se algumas características. O Sistema Tutorial desenvolvido visa esclarecer e auxiliar os usuários em geral na manipulação do ANSEsp. Sendo assim, reúne conceitos gerais em que o mesmo se baseia, e características de implementação de seus módulos. Como um sistema de apoio, procura facilitar as atividades de Engenheiros de Conhecimento e de Sistemas.

Este documento apresenta uma breve descrição do Sistema Tutorial, o qual está disponível ao usuário, podendo ser acessado através da interface do ANSEsp. O sistema encontra-se sob a forma de um hipertexto, permitindo a livre navegação por suas telas. O software utilizado na sua construção é o HIPERGRAPH, desenvolvido pelo ILTC (Instituto de Lógica, Filosofia e Teoria da Ciência).

Referências

[1] Rodrigues, S.R.; Monard, M.C. *Sistemas Baseados em Conhecimento: Conceitos Fundamentais e Aplicações.* Anais da Primeira Jornada USP - SUCESU-SP de Informática e Telecomunicações, São Paulo, SP, junho 1993, pp 355-364.

[2] Monard, M.C.; Rodrigues, S.R. *An Environment for Developing Expert System's Shells.* Expert Systems World Congress Proceedings, Jay Liebowitz (Ed.), Pergamon Press, Vol. 3, december 1991, pp 1757-1765.

[3] Rodrigues, S.R.; Monard, M.C. *Um Subsistema de um Núcleo de Sistema Especialista para Interrogar o Usuário Admitindo diversas Categorias de Perguntas.* Anais VI SBIA, novembro 1989, pp 78-92.

[4] Rodrigues,S.R.; Monard, M.C. *Implementação Lógica de um Módulo Coletor de Dados para a Construção de Sistemas Especialistas.* Notas do ICMSC-USP, Número 63, 1990, 49 pg.

[5] Monard, M.C.; Rodrigues, S.R. *Implementação Lógica de um Motor de Inferência com Raciocínio "Backward Chaining" para a Construção de Sistemas Especialistas.* Notas do ICMSC-USP, Número 51, outubro 1989, 57 pg.

[6] Rodrigues, S.R.; Rozenfeld, H. *Criação e Aplicação de Módulos Automáticos para o Planejamento de Processo Assistido por Computador em Soluções Híbridas de Planejamento.* Tese de Doutorado, EESC-USP, 1993.

[7] Rich, E.; Knight, K.*Inteligência Artificial. (2da Ed.)* Mc Graw-Hill, São Paulo, 1993.

[8] Rodrigues, S.R. *Um Ambiente para Auxiliar a Construção de Núcleos de Sistemas Especialistas.* Dissertação de Mestrado, ICMSC-USP, 1990.