

**SISTEMA BASEADO EM CONHECIMENTO  
PARA DIAGNÓSTICO DE ERROS NA  
APRESENTAÇÃO DA *FIM*  
(FÁBRICA INTEGRADA MODELO)<sup>1</sup>**

**SOLANGE OLIVEIRA REZENDE  
CHANDLER WELLINGTON CAULKINS  
RODRIGO MOTTA E SILVA  
VALTEMIR DE ALENCAR E SILVA**

**Nº 55**

**RELATÓRIOS TÉCNICOS DO ICMSC**

São Carlos  
Mai./1997

|             |        |
|-------------|--------|
| SYSNO       | 946817 |
| DATA        | / /    |
| ICMC - SBAB |        |

**Sistema Baseado em Conhecimento  
para Diagnóstico de Erros na  
Apresentação da FIM  
(Fábrica Integrada Modelo)<sup>1</sup>**

**Solange Oliveira Rezende  
Chandler Wellington Caulkins  
Rodrigo Motta e Silva  
Valtemir de Alencar e Silva**

Universidade de São Paulo / ILTC  
Instituto de Ciências Matemáticas de São Carlos  
Departamento de Computação e Estatística  
Caixa Postal 668, 13560-970 - São Carlos - SP

e-mail: [solange@icmsc.sc.usp.br](mailto:solange@icmsc.sc.usp.br)

Abril de 1997

---

<sup>1</sup>Trabalho realizado com o auxílio do CNPq e FAPESP

## Conteúdo

|                                                                                            |           |
|--------------------------------------------------------------------------------------------|-----------|
| <b>1. INTRODUÇÃO .....</b>                                                                 | <b>1</b>  |
| 1.1 LOCALIZAÇÃO .....                                                                      | 1         |
| 1.2 OBJETIVOS.....                                                                         | 2         |
| 1.3 ORGANIZAÇÃO DO TRABALHO.....                                                           | 2         |
| <b>2. ALGUNS CONCEITOS DE INTELIGÊNCIA ARTIFICIAL.....</b>                                 | <b>2</b>  |
| 2.1 SISTEMAS BASEADOS EM CONHECIMENTO .....                                                | 4         |
| 2.1.1 <i>Estrutura Básica de um SBC</i> .....                                              | 4         |
| 2.1.2 <i>Domínios de Conhecimento</i> .....                                                | 5         |
| 2.1.3 <i>Etapas de Desenvolvimento de um SBC</i> .....                                     | 6         |
| <b>3. DESCRIÇÃO DO PROJETO .....</b>                                                       | <b>8</b>  |
| 3.1 SISTEMA BASEADO EM CONHECIMENTO PARA DIAGNÓSTICO DE ERROS NA APRESENTAÇÃO DA FIM ..... | 8         |
| 3.1.1 <i>Aquisição do Conhecimento</i> .....                                               | 9         |
| 3.1.2 <i>Estruturação do Conhecimento Usando Conceitos de Orientação a Objetos</i> .....   | 9         |
| 3.1.3 <i>Estruturação do Conhecimento Usando Regras de Produção</i> .....                  | 14        |
| 3.2 IMPLEMENTAÇÃO DO SISTEMA.....                                                          | 15        |
| <b>4. CONCLUSÕES.....</b>                                                                  | <b>17</b> |
| <b>5. REFERÊNCIAS BIBLIOGRÁFICAS .....</b>                                                 | <b>18</b> |
| <b>6. APÊNDICE A - A BASE DE CONHECIMENTO .....</b>                                        | <b>19</b> |
| <b>7. APÊNDICE B - A JANELA PRINCIPAL DA INTERFACE GRÁFICA .....</b>                       | <b>30</b> |
| <b>8. APÊNDICE C - A JANELA DE DIAGNÓSTICOS.....</b>                                       | <b>35</b> |

## Índice de Figuras

|                                                                                                                        |           |
|------------------------------------------------------------------------------------------------------------------------|-----------|
| <i>Figura 1 - Áreas Afins e Subáreas de Inteligência Artificial.....</i>                                               | <i>3</i>  |
| <i>Figura 2- A Inteligência Artificial, os Sistemas Baseados em Conhecimento e os<br/>Sistemas Especialistas .....</i> | <i>4</i>  |
| <i>Figura 3- Estrutura básica de um Sistema Baseado em Conhecimento .....</i>                                          | <i>5</i>  |
| <i>Figura 4- Etapas de Desenvolvimento de um SBC .....</i>                                                             | <i>7</i>  |
| <i>Figura 5 - Visualização do Conteúdo de um Objeto.....</i>                                                           | <i>10</i> |
| <i>Figura 6- Hierarquia de Classes na Herança Múltipla.....</i>                                                        | <i>11</i> |
| <i>Figura 7- Representação Gráfica de Alguns Conceitos de Orientação a Objetos.....</i>                                | <i>13</i> |
| <i>Figura 8 - Modelagem orientada a objetos do conhecimento .....</i>                                                  | <i>14</i> |
| <i>Figura 9- Modelagem orientada a objetos do conhecimento (Tela do Smart Elements) .</i>                              | <i>15</i> |
| <i>Figura 10- Janela Principal - DiagMod.PrincipalWin (Tela do Smart Elements).....</i>                                | <i>31</i> |
| <i>Figura 11 - Janela de Diagnósticos - DiagMod.DiagnosticoWin (Tela do Smart<br/>Elements) .....</i>                  | <i>35</i> |



## **1. Introdução**

### **1.1 Localização**

O projeto de desenvolvimento de um Sistema Baseado em Conhecimento para Diagnóstico de Erros na Apresentação da FIM encontra-se inserido no Projeto de Manufatura Integrada por Computador (CIM), em desenvolvimento no Laboratório de Máquinas Ferramentas (LAMAFE) e fomentado pelo RHAE, visando auxiliar as empresas instaladas no Brasil na aplicação das novas filosofias e tecnologias de produção, contribuindo, assim, para a sua sobrevivência perante a concorrência mundial. Um dos objetivos do Projeto CIM é testar e integrar componentes comerciais CIM em soluções piloto, oferecendo às empresas referências para suas decisões. Desenvolvem-se ainda metodologias de aplicação da tecnologia de integração.

A fim de se divulgar e difundir a tecnologia CIM, assim como outros conceitos, implantou-se uma Fábrica Integrada Modelo (FIM) no LAMAFE. Com a FIM, pretende-se simular empresas que começam a automatizar somente parte do seu parque de máquinas, mas que têm a necessidade de integrar suas antigas instalações. Ela é composta de um setor de planejamento, que integra as funções de engenharia e planejamento de produção, de um sistema flexível de fabricação com duas células flexíveis e uma oficina mecânica convencional.

Atualmente, a FIM encontra-se inserida no seu cenário 3 de integração, onde todas as tarefas são realizadas por microcomputadores. Os visitantes participam de apresentações pelas quais se verificam duas fases distintas de trabalho: uma fase de orçamento integrado e uma fase onde esse orçamento torna-se o pedido do visitante, que então vê desde a programação CN (de Comando Numérico) até a produção de seu produto no centro de usinagem.

Em algumas dessas apresentações foram observados erros de diferentes naturezas. Estes podem ser classificados em dois grupos principais:

- erros de sistema: são provocados por falhas na rede ou nos softwares que compõem as diversas fases da FIM;
- erros de operação: são provocados por falhas dos usuários durante as apresentações da FIM.

Dada a frequência dos erros ocorridos nas apresentações, verificou-se a possibilidade de catalogá-los e propor diagnósticos que facilitassem suas correções, permitindo assim oferecer uma maior segurança às pessoas responsáveis pelas apresentações. Em consequência disto, viabilizou-se a implementação de um sistema que automatize este processo de diagnósticos de erros a partir de uma interação amigável com os responsáveis.

## **1.2 Objetivos**

O Sistema Baseado em Conhecimento para Diagnósticos de Erros na Apresentação da FIM procura auxiliar uma pessoa responsável pela apresentação a encontrar e solucionar os possíveis erros surgidos no processo, a partir de dados fornecidos por ele. Com a utilização desse sistema pretende-se minimizar o tempo gasto na procura das possíveis causas da ocorrência de um erro e nas prováveis soluções necessárias para saná-lo de forma eficiente.

## **1.3 Organização do Trabalho**

O trabalho está organizado da seguinte forma:

- Na seção 2, são apresentados conceitos básicos de Inteligência Artificial relacionados ao desenvolvimento dos Sistemas Baseados em Conhecimento, incluindo os utilizados para Diagnóstico.
- A seção 3 descreve o projeto de desenvolvimento de um Sistema Baseado em Conhecimento para Diagnóstico de Erros na Apresentação da FIM, abordando a aquisição, estruturação e representação do conhecimento (seção 3.1) bem como aspectos da implementação do sistema (seção 3.2);
- Algumas conclusões a respeito da utilização de SBCs na FIM e em fábricas em geral são feitas na Seção 4; e
- na seção 5 são colocadas as referências bibliográficas.

## **2. Alguns conceitos de Inteligência Artificial**

Pode-se definir a Inteligência Artificial como:



*“O estudo de como fazer os computadores realizarem tarefas nas quais, até agora, o ser humano tem um melhor desempenho que a máquina.”*

[Rich 93]

Atualmente, um dos grandes desafios de IA reside no domínio da tecnologia de processamento do conhecimento através da criação de grandes sistemas, capazes de manipular conceitos, fatos, analogias, relações e modelos, com capacidade para interagir naturalmente com outros sistemas inteligentes (homens e máquinas) numa relação amigável, cada um deles executando as tarefas que melhor dominar. A idéia geral é que estes sistemas sejam elementos ampliadores da capacidade cognitiva do homem em todos os domínios de sua atividade. Como os objetivos de IA são bastante abrangentes, os pesquisadores contam com o apoio do conhecimento de outras áreas de pesquisa com atividades afins, tais como: Lógica Matemática, Engenharia, Biologia, Psicologia, Linguística, entre outras, surgindo desta forma várias subáreas de pesquisa, tais como: Sistemas Baseados em Conhecimento, Planejamento, Redes Neurais, Processamento de Linguagem Natural, etc., como ilustra a figura 1.

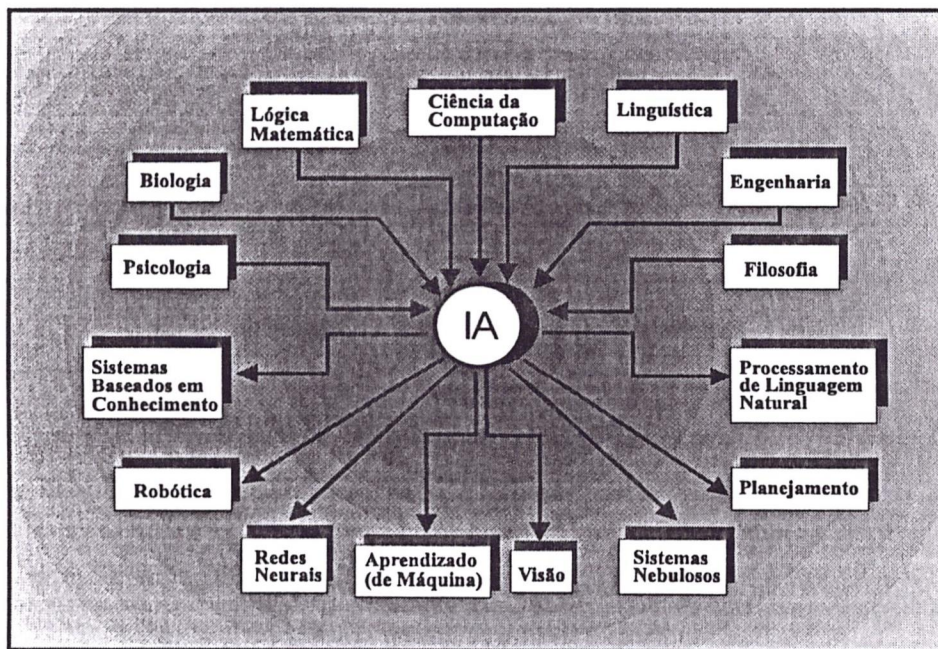


Figura 1 - Áreas Afins e Subáreas de Inteligência Artificial

Os Sistemas Baseados em Conhecimento (SBC) consistem na subárea de interesse na elaboração deste projeto, e sendo assim são tratados com maiores detalhes na próxima seção.

## 2.1 Sistemas Baseados em Conhecimento

Os Sistemas Baseados em Conhecimento - SBC - são de grande interesse nas pesquisas em Inteligência Artificial. Tais sistemas são programas de computador que resolvem ou ajudam a resolver problemas específicos, os quais geralmente requerem inteligência humana na sua solução [Aran 93]. Quando estes sistemas tratam domínios bastante restritos, atingindo um alto grau de especialidade, são chamados de Sistemas Especialistas. Na figura 2, podemos localizar os Sistemas Especialistas e os Sistemas Baseados em Conhecimento na área de Inteligência Artificial.

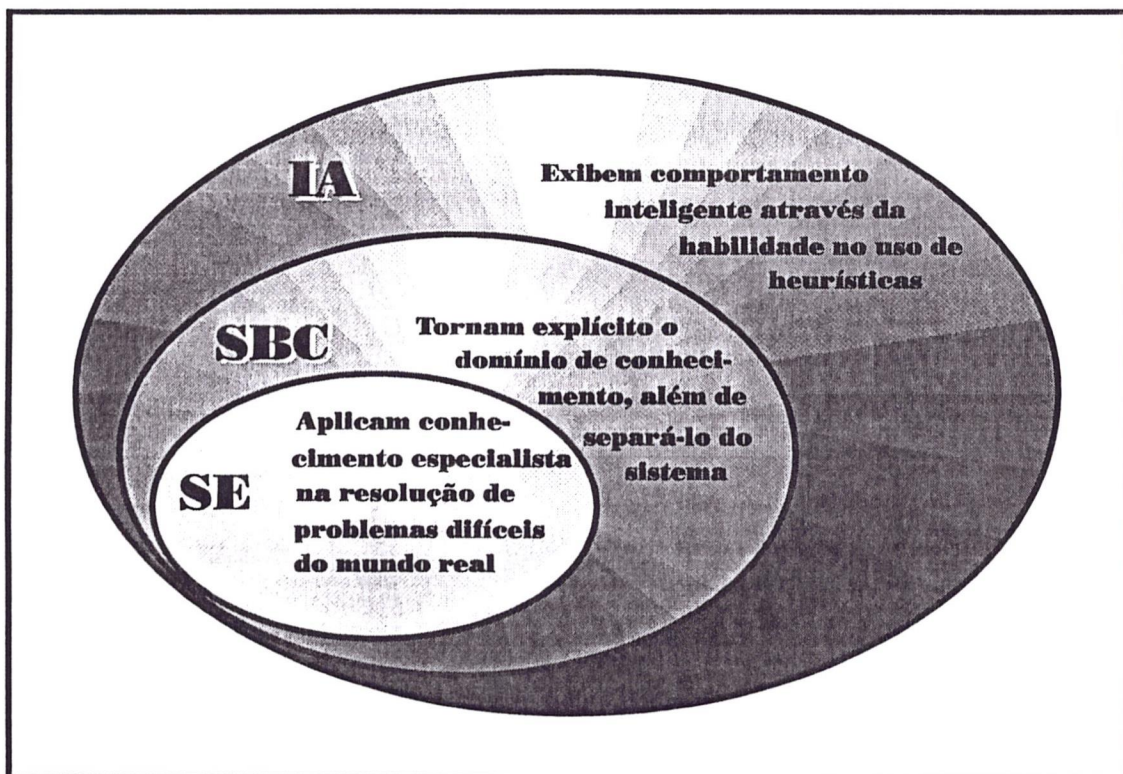


Figura 2- A Inteligência Artificial, os Sistemas Baseados em Conhecimento e os Sistemas Especialistas

### 2.1.1 Estrutura Básica de um SBC

O SBC pode ser dividido em três módulos principais [Rodr 93]:

- *Base de Conhecimento (BC)*: contém a modelagem do conhecimento específico do domínio de aplicação. Isto inclui asserções sobre o domínio de conhecimento, regras que descrevem relações neste domínio e, em alguns casos, heurísticas e métodos de resolução de problemas;



- *Motor de Inferência (MI)*: é responsável pelo processamento do conhecimento da BC, usando alguma linha de raciocínio. Contém as estratégias de inferência e controle;
- *Base de Dados (BD)*: consiste na área de trabalho do sistema, armazenando dados e conclusões intermediárias por ele obtidos.

A estrutura básica de um Sistema Baseado em Conhecimento pode ser visualizada na figura 3.

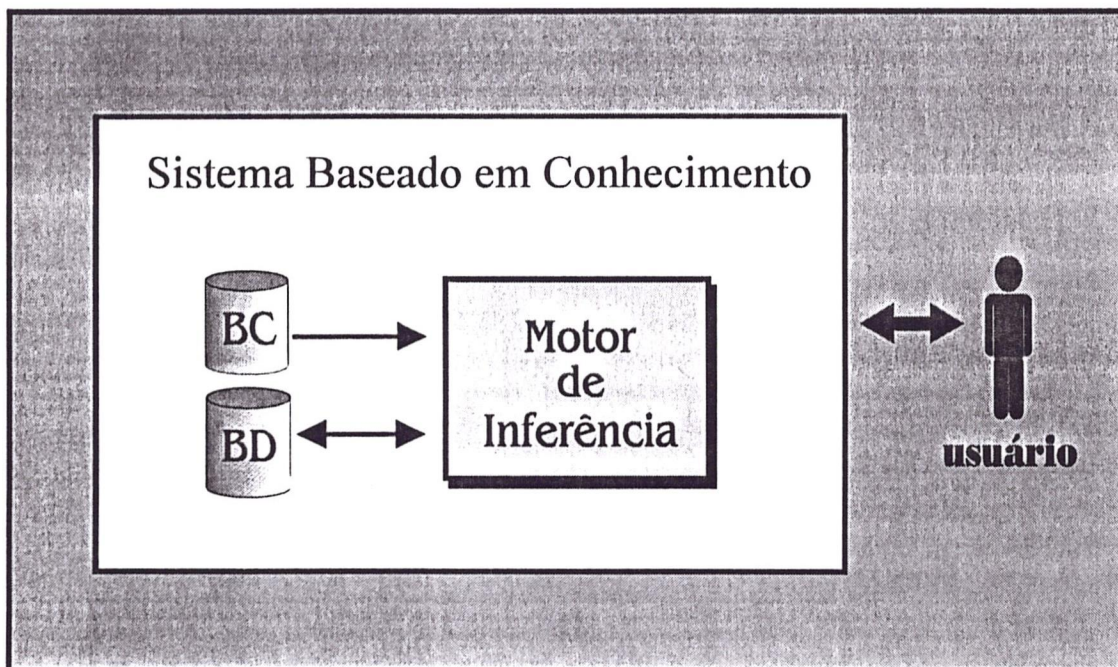


Figura 3- Estrutura básica de um Sistema Baseado em Conhecimento

### 2.1.2 Domínios de Conhecimento

O conhecimento e o processo de resolução de problemas são os pontos centrais no desenvolvimento de um SBC. Quanto ao conhecimento, verifica-se que existem domínios cujos problemas não são adequadamente solucionados através de SBCs. Por isso, uma análise do domínio e das características dos problemas a serem abordados é essencial quando se pretende desenvolver um sistema deste tipo.

Existe uma grande variedade de domínios de conhecimento comumente tratados em SBCs. Entre eles, pode-se citar:

- Interpretação: consiste na análise de dados para determinação de seu significado. O problema central desta classe é que, frequentemente, esses dados possuem ruído. Ex.: espectroscopia de massa, processamento de imagens, reconhecimento de fala, análise de circuitos elétricos, etc.;
- Diagnose: consiste no processo de determinação de falhas em um sistema, dado um conjunto de sintomas. Ex.: a maioria das aplicações médicas relativas à determinação de doenças em seres vivos, determinação de falhas de máquinas, etc.;
- Monitoramento: consiste no processo de observação contínua do comportamento de um sistema a fim de realizar ações quando alguma situação específica acontece. Ex.: monitoramento de centrais de energia nuclear, controle de tráfego aéreo, pacientes após ou durante cirurgia, etc.;
- Planejamento: consiste no processo de determinação da sequência de ações que devem ser realizadas para atingir uma dada meta. Ex.: planejamento de operações de robôs, experimentos em genética molecular, ações militares, planejamento de processos, etc.;
- Instrução: consiste no desenvolvimento de tutoriais para estudantes, com o objetivo de suprir pontos fracos nos seus conhecimentos. Ex.: atividades de ensino em diversas áreas;
- Configuração: consiste na configuração de sistemas, fornecendo estratégias a serem seguidas. Ex: configuração de sistemas de processamento de dados, reprojeto de circuitos visando atender a novas especificações, planejamento de experimentos de laboratório, etc...

Este trabalho consiste no desenvolvimento de um SBC para diagnósticos, e envolve as etapas de desenvolvimento descritas de forma generalizada na seção a seguir.

### **2.1.3 Etapas de Desenvolvimento de um SBC**

O desenvolvimento de um Sistema Baseado em Conhecimento envolve diversas etapas, como mostra a figura 4:



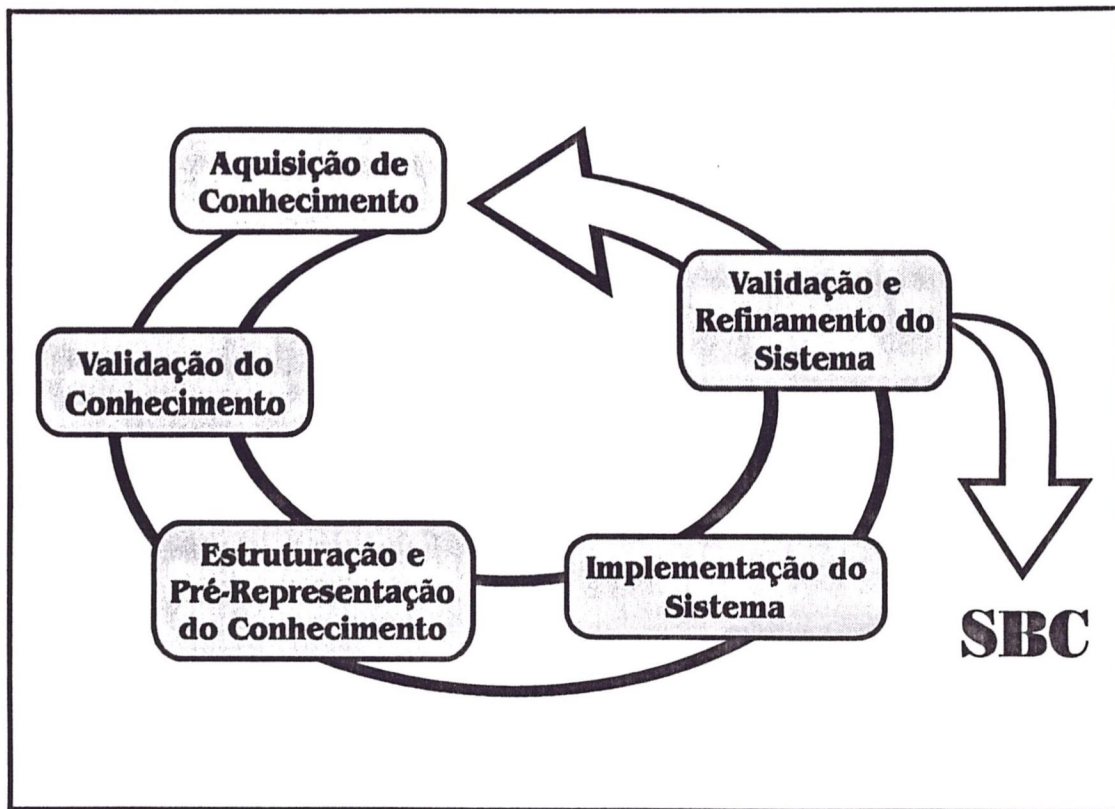


Figura 4- Etapas de Desenvolvimento de um SBC

- *Aquisição de conhecimento*: consiste na extração do conhecimento do especialista no domínio, bem como pesquisa em documentos, livros, handbooks, etc.;
- *Validação*: trata-se da validação do conhecimento com especialistas;
- *Estruturação e representação do conhecimento*: consiste em estruturar o conhecimento obtido na forma de regras, redes semânticas, frames, e outras formas de representação;
- *Implementação do sistema*: consiste na representação do conhecimento no computador, em linguagens de programação ou através de *shells* ou ferramentas que auxiliam a construção de SBCs;
- *Validação e Refinamento do sistema*: consiste em submeter o sistema a testes reais de forma a avaliar seu desempenho.

### 3. Descrição do Projeto

#### 3.1 Sistema Baseado em Conhecimento para Diagnóstico de Erros na Apresentação da FIM

Dentro de uma visão geral, as apresentações da FIM assemelham-se a um organismo, de forma que o sistema de diagnósticos especificado é tipicamente baseado em sintomas, como são os sistemas da área médica. Neste caso, o sistema exerce a função do especialista. A partir da manipulação de regras e de fatos que refletem a experiência deste e de dados estruturais do centro de trabalho, ele realiza inferências e, como resultado, identifica desvios, especifica causas e sugere ações corretivas.

Por se tratar de um sistema de apoio à decisão, que manipula informações por vezes incompletas ou pouco estruturadas, optou-se pela especificação de um sistema baseado em conhecimento. Através dele, a perícia do especialista pode ser assimilada de forma mais completa, fazendo com que o diagnóstico se torne uma atividade mais sistemática, uma vez que o empirismo é operacionalizado.

As atividades envolvidas neste trabalho fazem parte das seguintes fases de desenvolvimento:

- **Aquisição do Conhecimento** – obtenção da informação e da lógica utilizada pelo Especialista (no caso todos os responsáveis pela elaboração dos sistemas desenvolvidos na FIM) para resolver certos problemas em determinados domínios;
- **Modelagem do Conhecimento Usando Conceitos de Orientação a Objetos** – representação dos erros encontrados utilizando-se conceitos de orientação a objetos, promovendo a implementação de uma base de conhecimento concisa e expansível;
- **Estruturação do Conhecimento Usando Regras de Produção** – manipulação de dados e processamento do conhecimento realizados através de regras do tipo *Se (premissas) -> Então (ações) Senão (ações)*.

Estas fases serão detalhadas nas seções a seguir.



### 3.1.1 Aquisição do Conhecimento

Para a aquisição do conhecimento necessário para implementação do sistema proposto foram realizadas entrevistas com os responsáveis pelo desenvolvimento de cada fase da FIM. Além disso, utilizaram-se alguns documentos que apresentavam todas essas fases numa visão geral assim como as relações existentes entre elas (SADT).

### 3.1.2 Estruturação do Conhecimento Usando Conceitos de Orientação a Objetos

#### 3.1.2.1 Alguns Conceitos de O.O.

Da evolução natural dos paradigmas de desenvolvimento de software surgiu o **Paradigma Orientado a Objetos**, que se preocupa, entre outras coisas, com a integração das fases de desenvolvimento e com o aspecto de reusabilidade.

O ciclo de vida de um software orientado a objetos também apresenta as três fases usuais de desenvolvimento de software: análise, projeto e implementação. Os limites entre estas fases são praticamente indistinguíveis, já que os itens de interesse de cada uma delas são os mesmos: os objetos e seus relacionamentos. Estes são identificados tanto na análise quanto no projeto. Aqueles identificados na análise são encarados não somente como uma entrada da fase de projeto, mas também como um nível inicial da mesma. Desta forma, analistas, projetistas e programadores trabalham com um conjunto comum de elementos.

A seguir serão descritos alguns conceitos básicos de OO necessários para o desenvolvimento deste trabalho (e cada uma das fases que compõem o ciclo de vida de um software orientado a objetos):

- **OBJETO**: Um objeto modela alguma parte da realidade e é, portanto, alguma coisa que existe no tempo e no espaço. Outros importantes tipos de objetos são invenções do processo de projeto que produzem alguns procedimentos de alto nível. Isto nos leva a mais definições, como a de Smith e Tockey, que sugere: “um objeto representa um item individual, identificável, uma unidade ou entidade tanto no real quanto no abstrato, que possui um papel bem definido no domínio do problema”.
- **MENSAGEM**: Uma mensagem provoca uma ação do objeto que a recebe. O objeto então seleciona um de seus métodos para ser executado, baseado



na natureza da mensagem. Tal mensagem contém parâmetros como uma chamada de função. Após a execução do método, o objeto retornará o controle ao objeto que lhe enviou a mensagem. O conjunto de mensagens possíveis que podem ser enviadas a um objeto é especificado na descrição da classe para o objeto.

- **MÉTODO:** Métodos são operações que podem ser executadas nos dados do objeto. São eles que determinam como o objeto agirá quando receber uma mensagem. Somente os métodos do objeto podem manipular os valores das variáveis de instância, que armazenam dados do próprio objeto. Os métodos podem também enviar mensagens aos objetos para solicitar informações, ou requisitar alguma ação, como pode ser observado na figura 5.



Figura 5 - Visualização do Conteúdo de um Objeto

**HERANÇA:** É um mecanismo que permite que uma classe herde propriedades de classes mais genéricas, reutilizando integralmente ou parcialmente estas classes existentes. Este mecanismo constitui uma característica extremamente importante da programação orientada a objetos, pois representa economia de tempo e facilidade na construção de classes. Sem a herança, cada classe seria uma unidade independente, desenvolvida a partir do zero. Esta característica permite organizar, construir e reutilizar classes existentes, criando assim novas classes, acrescentando, modificando ou subtraindo métodos da classe-pai. A classe que herda características pode ser chamada de subclasse, classe descendente ou derivada, e a classe que fornece características pode ser chamada de base, superclasse, antepassado ou classe-pai. Outro tipo de hereditariedade refere-se à herança múltipla, onde uma subclasse pode herdar dados e métodos de uma ou mais classes, tornando-se útil na criação de comportamento composto por mais de um ramo de hierarquia de classe, como visto na figura 6.

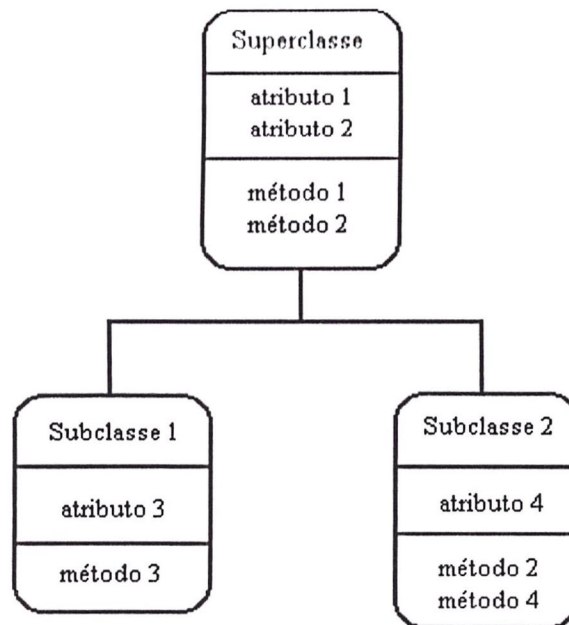


Figura 6- Hierarquia de Classes na Herança Múltipla

- **ENCAPSULAMENTO:** É a junção dos dados e métodos de um objeto, sendo que somente através destes métodos podem ser manipulados os dados do objeto. De acordo com Britton e Parnas, encapsulamento é o processo de esconder todos os detalhes de um objeto que não contribuam para suas características essenciais. Com o encapsulamento, o usuário tem visibilidade das rotinas do objeto, porém não tem visibilidade de seus dados, que ficam protegidos dentro da cápsula objeto. O encapsulamento tem três finalidades:
  - proteção aos dados de um objeto de exposição excessiva, já que estes somente são acessados por funções definidas dentro do objeto;
  - facilita o uso dos dados do objeto uma vez que todas as ações ocorrem pela interface bem definida (operações associadas ao objeto);
  - ocultamento de detalhes dos dados armazenados e implementados.
- **POLIMORFISMO:** O polimorfismo permite que um programa siga um curso de ação enviando uma mensagem para um objeto sem se preocupar sobre como o sistema implementa a ação. Isto significa que a mesma mensagem pode provocar ações diferentes quando recebida por objetos diferentes.
- **ABSTRAÇÃO:** Pode ser definida como o princípio de ignorar aspectos de um assunto não relevantes ao propósito em questão, tornando possível



uma maior concentração nos seus aspectos principais. Esta técnica permite que ignoremos a complexidade do conceito considerado, levando em conta somente suas partes relevantes ao invés de observá-lo como um todo.

- **PERSISTÊNCIA:** Um objeto ocupa um espaço e existe por um determinado tempo. Atkinson acredita que há uma continuidade da sua existência, passando de objetos transitórios a objetos em um banco de dados, que permanece após a execução como um todo. Na maioria das linguagens orientadas a objeto, as instâncias de classes são criadas em tempo de execução. Algumas instâncias são necessárias apenas por um curto espaço de tempo. Quando um objeto não é mais necessário, ele é destruído e o espaço alocado é recuperado. Após um programa ser executado, os objetos construídos não ficam armazenados permanentemente e são tratados como persistentes.
- **CLASSE:** A construção de classes é usada para o encapsulamento e ocultamento de dados. Uma classe é formada por objetos e métodos, sendo os objetos acessados através de mensagens. Os dados apresentam-se sob variáveis de classe, que têm valores armazenados em uma classe e variáveis de instância, que contêm valores armazenados unicamente a cada instância ou objeto criado a partir de uma classe. Uma classe pode ter três seções:
  - *private*: dados e métodos não podem ser acessados diretamente;
  - *protected*: dados e métodos somente podem ser acessados por classes derivadas;
  - *public*: dados e métodos podem ser acessados diretamente.

Algumas classes são específicas para um problema, outras podem ser reutilizadas em muitas aplicações e ficam disponíveis em bibliotecas de software, que são classes bases.

Uma classe pode também resumir elementos comuns para um conjunto de subclasses ou classes derivadas. Uma classe pode herdar características de outra classe, para reutilizar a implementação da classe base, aumentar, diminuir ou modificar métodos da classe base.

A fim de se tornar mais claros alguns dos conceitos definidos nesta seção, a figura 7 é apresentada a seguir como um exemplo de representação gráfica destes.



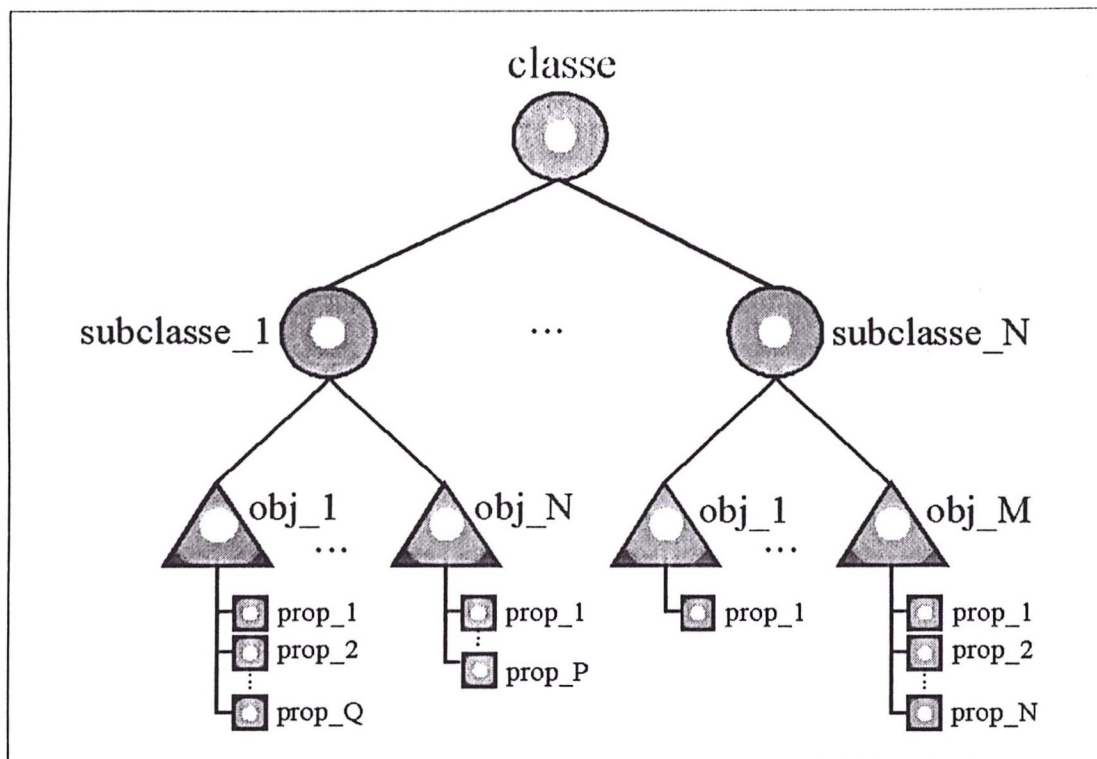


Figura 7- Representação Gráfica de Alguns Conceitos de Orientação a Objetos

### 3.1.2.2 A Modelagem do Conhecimento

Na fase de estruturação do conhecimento baseada em conceitos de orientação a objetos, foram propostos e analisados dois modelos iniciais. O primeiro visou apresentar em foco principal a **FIM** como um todo (representada como uma Classe-pai), todas as suas fases ou **etapas** (representadas como Sub-Classes) e seus respectivos **erros** (representados como Atributos ou Propriedades). Esse modelo, porém, mostrou-se falho devido ao fato de que ao se considerar a FIM como núcleo dessa modelagem deixou-se de lado o objetivo principal do projeto: os **erros** ocorridos nas apresentações.

Com o objetivo de deixar a modelagem mais centralizada nas falhas e erros ocorridos, partiu-se para um segundo modelo. A partir deste, existiria uma Classe-pai denominada Erros, donde se partiria para classes inferiores e mais específicas denominadas Erros\_Fim (onde os erros ocorridos na parte de execução do sistema propriamente dito seriam relatados), Erros\_Rede (onde os erros ocorridos na parte de conexão de rede - parte anterior à execução do sistema - seriam relatados) e Erros\_Rh (onde se tentaria encontrar os membros responsáveis pelas apresentações que possivelmente não estivessem presentes). Esta modelagem está representada na figura 8:

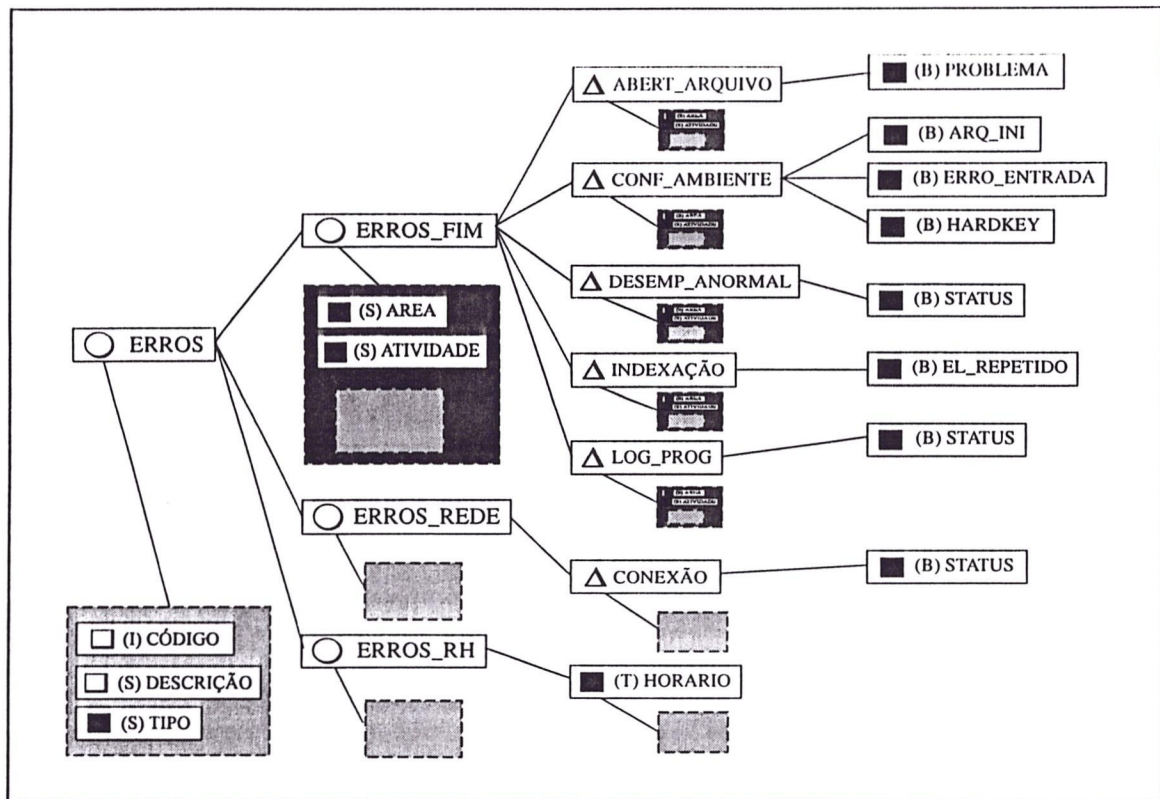


Figura 8 - Modelagem orientada a objetos do conhecimento

### 3.1.3 Estruturação do Conhecimento Usando Regras de Produção

As regras especificadas para o sistema são regras de produção do tipo:

**SE (premissas)**  
**ENTÃO (ações)**  
**SENÃO (ações)**

Cada premissa é composta por um conjunto de cláusulas que formam uma combinação booleana. Cada cláusula apresenta os seguintes elementos:

- Atributos (ou propriedades);
- Operadores;
- Valores (uma variável, um inteiro, um real ou um estado - como: orçamento, funções de CAD, entre outros).

Cada premissa satisfeita leva à execução de uma hipótese. Esta, por sua vez, pode refletir uma causa do erro ou indicar uma ação corretiva.

A seguir é dado um exemplo de uma regra hipotética para a verificação dos arquivos de inicialização.

**SE**

atividade é Solicitar Orçamento e  
houve problemas de abertura de arquivos

**ENTÃO**

verificar se arquivos de inicialização estão corretos

### 3.2 Implementação do Sistema

Visando o aproveitamento dos conceitos de Orientação a Objetos e da manipulação de dados via regras de produção, foi utilizado o software Smart Elements, da Neuron Data.

A figura a seguir mostra a tela do Smart Elements que visualiza a representação do conhecimento utilizando-se de conceitos de Orientação a Objetos.

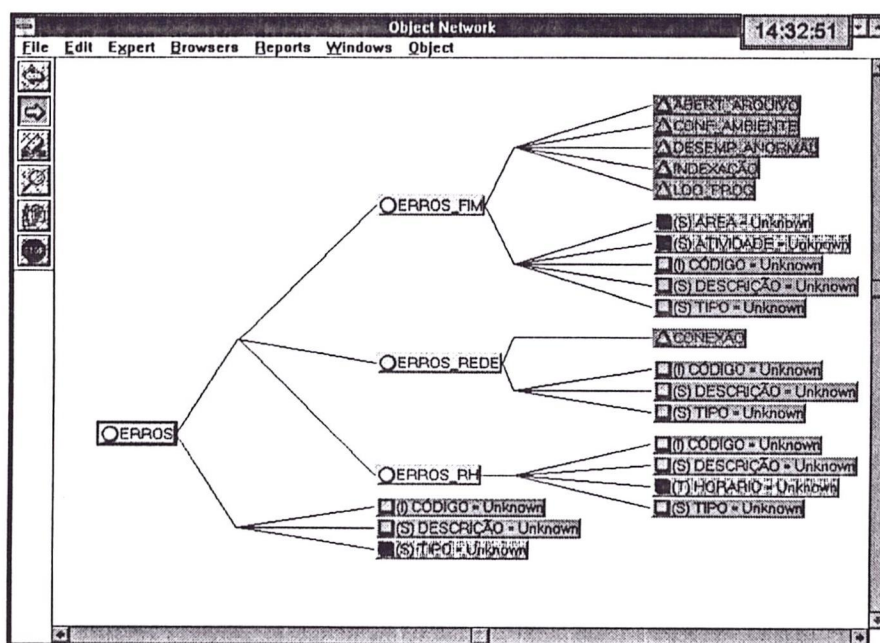


Figura 9 - Modelagem orientada a objetos do conhecimento (Tela do Smart Elements)

A fase de implementação consistiu de duas etapas principais:



- **Construção da Interface:** Preocupou-se em promover uma interação o mais amigável possível com o usuário na obtenção dos dados de entrada necessários e no retorno dos erros encontrados pelo sistema bem como suas ações corretivas. O código relacionado com a interface é apresentado nos apêndices B e C;
- **Construção da Base de Conhecimento:** Preocupou-se em estruturar o conhecimento da forma mais eficiente possível, a fim de haver uma adequada manipulação dos dados de entrada e uma possibilidade de expandi-la facilmente, algo muito frequente no tipo de aplicação em questão. O código relacionado com a Base de Conhecimento é apresentado no apêndice A.

Para se concluir o sistema, houve a sua validação e testes junto aos especialistas (no caso, os responsáveis pela implementação de cada fase da FIM).

O processo de diagnóstico se inicia a partir dos dados obtidos periodicamente referentes a cada centro de trabalho. O sistema deve, então, buscar para cada dado fornecido os fatos e regras que se apliquem a ele. É feita uma análise das premissas e se elas forem satisfeitas, então aplica-se uma hipótese que pode refletir uma causa do erro ou uma sugestão de ação corretiva. No caso de falha das premissas, outra regra é tentada até que sejam esgotadas todas as possibilidades. Nesse caso, pode-se emitir mensagens ao usuário denotando insuficiência de dados fornecidos ou de conhecimento.

### **As Janelas de Interface**

A interface do sistema proposto é composta de duas janelas:

- *Janela Principal* (DiagMod.PrincipalWin): responsável pela obtenção de dados fornecidos pelo usuário e necessários para o processamento do conhecimento. Uma particularidade desta janela é com relação ao tipo de meta sendo raciocinada - se for um problema de RH, “aparece” na parte inferior dela uma caixinha de entrada de dados para passar o horário durante o qual a pessoa procurada se encontra na FIM. Enquanto não se sabe se o erro é desse tipo, esta caixinha fica “escondida” e no meio da tela, “aparece” uma caixa de escolhas preenchida com respostas possíveis para a meta sendo processada naquele momento.
- *Janela de Diagnósticos* (DiagMod.DiagnosticoWin): retorna ao usuário os erros encontrados bem como ações corretivas que possam ser realizadas.

Os respectivos códigos de implementação das janelas, usando a linguagem de “script” nativa do Smart Elements, são mostrados nos apêndices B e C, junto com figuras demonstrativas de suas aparências.

#### **4. Conclusões**

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

As aplicações destes sistemas abrangem desde interpretação e monitoramento até planejamento e diagnóstico. De particular interesse desse projeto são os sistemas que permitem diagnosticar problemas de modo a oferecer soluções atualizadas e coerentes, como por exemplo alguns sistemas médicos.

Os problemas que ocorrem nas apresentações da FIM podem ser assemelhados a doenças por possuírem sinais que indicam que estão ocorrendo. Desta forma, percebeu-se que seria interessante construir um Sistema Baseado em Conhecimento para Diagnóstico, onde se pudesse armazenar os “sintomas” mais frequentes e soluções conhecidas para eles.

Conhecimento dos problemas foi adquirido junto a responsáveis pela elaboração dos sistemas usados na FIM. Este conhecimento foi então estruturado utilizando conceitos de orientação a objetos e regras de produção, construindo uma base de conhecimento. Tendo esta, especificou-se uma interface gráfica para tornar a interação entre o usuário e a base mais amigável.

O Sistema de Diagnóstico de Erros na Apresentação da Fábrica Integrada Modelo tem como objetivo fornecer suporte aos usuários responsáveis por tal atividade. Por se tratar de um SBC, o sistema apresentado neste documento deve ser apto a absorver mudanças ou novos conhecimentos surgidos em seu domínio de conhecimento. Este fato ainda é reforçado quando se considera que a Fábrica Integrada Modelo consiste de uma proposta em fase de pesquisa, e portanto seus cenários são frequentemente aperfeiçoados. Novos erros surgem a qualquer momento e devem ser incluídos na Base de Conhecimento do sistema. Este encontra-se ainda com um conjunto limitado de erros, mas deverá ser constantemente atualizado de



modo a acompanhar de forma consistente e completa a evolução da Fábrica Integrada Modelo.

## 5. Referências Bibliográficas

- [Aran 95] Arantes, A. C. N.; Caulkins, C. W.; Oliveira, M. A. de; *Desenvolvimento de um Sistema Baseado em Conhecimento para Configuração de Tornos Mecânicos*, projeto de iniciação científica na área de Inteligência Artificial, Universidade de São Paulo - Campus São Carlos, 1995.
- [Coad 92] Coad, P.; Yourdon, E.; *Análise Baseada em Objetos*, série Yourdon Press - Editora Campus, 1992.
- [Durk 96] Durkin, J.; *Expert Systems: A View of the Field*, IEEE Expert, abril, 1996.
- [Luge 89] Luger, G. F.; Stubblefield, W. A. *Artificial Intelligence and the Design of Expert Systems*. The Benjamin/Cummings Publishing Company Inc., 1989.
- [Peat 88] Peat, F.D. *Artificial Intelligence: How Machine Think*. New York, Bean Publishing Enterprises, 1988.
- [Rich 93] Rich, E.; Knight, K. *Inteligência Artificial*. (2da Ed.) McGraw-Hill, São Paulo, 1993.
- [Rodr 93] Rodrigues, S. R. *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.
- [Senz 94] Senzako, E. Y.; Figueiredo, M. B.; *Análise e Projeto Orientado a Objetos*, trabalho de Engenharia de Software, Universidade de São Paulo - Campus São Carlos, 1994.
- Documentação e folhetos explicativos existentes no LAMAFE.



## 6. Apêndice A - A Base de Conhecimento

Nesta seção é mostrada em sua íntegra a Base de Conhecimento elaborada com a ajuda dos responsáveis pelas apresentações da FIM e com o software Smart Elements.

```
(@VERSION=      030)
(@PROPERTY=     AREA @TYPE=String;)
(@PROPERTY=     ARQ_INI @TYPE=Boolean;)
(@PROPERTY=     ATIVIDADE @TYPE=String;)
(@PROPERTY=     CURSOR @TYPE=Integer;)
(@PROPERTY=     CÓDIGO @TYPE=Integer;)
(@PROPERTY=     DESCRIÇÃO @TYPE=String;)
(@PROPERTY=     EL_REPETIDO @TYPE=Boolean;)
(@PROPERTY=     ERRO_ENTRADA @TYPE=Boolean;)
(@PROPERTY=     HARDKEY @TYPE=Boolean;)
(@PROPERTY=     HOR @TYPE=String;)
(@PROPERTY=     HORARIO @TYPE=Time;)
(@PROPERTY=     HOR_ENTRADA @TYPE=String;)
(@PROPERTY=     HOR_SAIDA @TYPE=String;)
(@PROPERTY=     H_ENTRADA @TYPE=Time;)
(@PROPERTY=     H_SAIDA @TYPE=Time;)
(@PROPERTY=     LOCAL @TYPE=String;)
(@PROPERTY=     NOME @TYPE=String;)
(@PROPERTY=     PROBLEMA @TYPE=Boolean;)
(@PROPERTY=     STATUS @TYPE=Boolean;)
(@PROPERTY=     TIPO @TYPE=String;)

(@CLASS= ERROS
  (@SUBCLASSES=
    ERROS_FIM
    ERROS_REDE
    ERROS_RH
  )
  (@PROPERTIES=
    CÓDIGO
    DESCRIÇÃO
    TIPO
  )
)

(@CLASS= ERROS_FIM
  (@PROPERTIES=
    AREA
    ATIVIDADE
    CÓDIGO
    DESCRIÇÃO
    TIPO
  )
)
```

```
(@CLASS= ERROS_REDE
  (@PROPERTIES=
    CÓDIGO
    DESCRIÇÃO
    TIPO
  )
)

(@CLASS= ERROS_RH
  (@PROPERTIES=
    CÓDIGO
    DESCRIÇÃO
    HORARIO
    TIPO
  )
)

(@CLASS= saida
  (@PROPERTIES=
    H_ENTRADA
    H_SAIDA
    LOCAL
    NOME
  )
)

(@OBJECT= ABERT_ARQUIVO
  (@CLASSES=
    ERROS_FIM
  )
  (@PROPERTIES=
    AREA
    ATIVIDADE
    CÓDIGO
    DESCRIÇÃO
    PROBLEMA
    TIPO
  )
)

(@OBJECT= AUX
  (@PROPERTIES=
    CURSOR
    HOR
    HOR_ENTRADA
    HOR_SAIDA
    H_ENTRADA
    H_SAIDA
    LOCAL
    NOME
  )
)

(@OBJECT= CONEXÃO
  (@CLASSES=
```

```

        ERROS_REDE
    )
    (@PROPERTIES=
        CÓDIGO
        DESCRIÇÃO
        STATUS
        TIPO
    )
)

(@OBJECT= CONF_AMBIENTE
    (@CLASSES=
        ERROS_FIM
    )
    (@PROPERTIES=
        AREA
        ARQ_INI
        ATIVIDADE
        CÓDIGO
        DESCRIÇÃO
        ERRO_ENTRADA
        HARDKEY
        TIPO
    )
)

(@OBJECT= DESEMP_ANORMAL
    (@CLASSES=
        ERROS_FIM
    )
    (@PROPERTIES=
        AREA
        ATIVIDADE
        CÓDIGO
        DESCRIÇÃO
        STATUS
        TIPO
    )
)

(@OBJECT= HIP_ABERT_ARQUIVO
    (@PROPERTIES=
        Value @TYPE=Boolean;
    )
)

(@OBJECT= HIP_CAD
    (@PROPERTIES=
        Value @TYPE=Boolean;
    )
)

(@OBJECT= HIP_CAPP
    (@PROPERTIES=
        Value @TYPE=Boolean;
    )
)

```



```

    )
)

(@OBJECT= HIP_FABRICAÇÃO
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_INDEXAÇÃO
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_INICIO_DIAGNOSTICO
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_ORÇAMENTO
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PCP
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PLAN_ATIV
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PROBLEMA_CAD
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PROBLEMA_CAPP
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PROBLEMA_ORÇAMENTO
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

```

```

)

(@OBJECT= HIP_PROBLEMA_PCP
  (@PROPERTIES=
    Value  @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PROBLEMA_PLAN_ATIV
  (@PROPERTIES=
    Value  @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PROBLEMA_REDE
  (@PROPERTIES=
    Value  @TYPE=Boolean;
  )
)

(@OBJECT= HIP_PROBLEMA_ÁRVORE
  (@PROPERTIES=
    Value  @TYPE=Boolean;
  )
)

(@OBJECT= HIP_RH
  (@PROPERTIES=
    Value  @TYPE=Boolean;
  )
)

(@OBJECT= HIP_SEM_DIAGNÓSTICO
  (@PROPERTIES=
    Value  @TYPE=Boolean;
  )
)

(@OBJECT= HORARIO_VALIDO
  (@PROPERTIES=
    Value  @TYPE=Boolean;
  )
)

(@OBJECT= INDEXAÇÃO
  (@CLASSES=
    ERROS_FIM
  )
  (@PROPERTIES=
    AREA
    ATIVIDADE
    CÓDIGO
    DESCRIÇÃO
    EL_REPETIDO
    TIPO
  )
)

```

```

    )
)

(@OBJECT=LOG_PROG
  (@CLASSES=
    ERROS_FIM
  )
  (@PROPERTIES=
    AREA
    ATIVIDADE
    CÓDIGO
    DESCRIÇÃO
    STATUS
    TIPO
  )
)

(@OBJECT=RECUPERA_HORARIO
  (@PROPERTIES=
    Value @TYPE=Boolean;
  )
)

(@META= IERROS!.TIPO
  @PROMPT="Forneça o tipo de erro:";
  @QUESTWIN="ModTeste.Win1";
)

(@META= IERROS_FIM!.AREA
  @PROMPT="Em qual fase da apresentação ocorreu o erro?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= IERROS_FIM!.ATIVIDADE
  @PROMPT="Qual a função em execução no CIM-Manager no momento do erro?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= IERROS_RH!.HORARIO
  @PROMPT="Qual é o horário em que se quer encontrar os responsáveis pela apresentação?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= ABERT_ARQUIVO.DESCRICÃO
  (@INITVAL= "Problemas na abertura de arquivo")
)

(@META= ABERT_ARQUIVO.PROBLEMA
  @PROMPT="Houve problema de abertura de arquivo?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= AUX.HOR
  @PROMPT="Qual é o horário em que se quer encontrar os responsáveis pela apresentação?";
  @QUESTWIN="ModTeste.Win1";
)

```



```

)

(@META= CONEXÃO.STATUS
  @PROMPT="Houve problemas com a utilização de diretórios após a execução do comando
login?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= CONF_AMBIENTE.ARQ_INI
  @PROMPT="Você configurou os arquivos de inicialização do sistema?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= CONF_AMBIENTE.DESCRICÃO
  (@INITVAL= "Ambiente com configuração incorreta")
)

(@META= CONF_AMBIENTE.ERRO_ENTRADA
  @PROMPT="O software em questão mostra uma inicialização anormal?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= CONF_AMBIENTE.HARDKEY
  @PROMPT="A hardkey do software CAD está na porta paralela?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= DESEMP_ANORMAL.STATUS
  @PROMPT="O sistema em questão está mais lento ou com um desempenho insatisfatório?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= INDEXAÇÃO.EL_REPETIDO
  @PROMPT="A árvore do produto possui elementos repetidos, como duas bases iguais ou
dois pinos iguais?";
  @QUESTWIN="ModTeste.Win1";
)

(@META= LOG_PROG.STATUS
  @PROMPT="Você notou problemas de lógica no software em questão?";
  @QUESTWIN="ModTeste.Win1";
)

(@RULE= R_ABERT_ARQUIVO
  (@LHS=
    (Yes (ABERT_ARQUIVO.PROBLEMA))
  )
  (@HYPO= HIP_ABERT_ARQUIVO)
)

(@RULE= R_ARVORE
  (@LHS=
    (Yes (HIP_ORÇAMENTO))
    (= (IERROS_FIM.ATIVIDADE) ("ÁRVORE DO PRODUTO"))
    (Yes (HIP_INDEXAÇÃO))
  )
)

```

```

    )
    (@HYPO=    HIP_PROBLEMA_ÁRVORE)
)

(@RULE=  R_CAD
  (@LHS=
    (Yes  (HIP_ORÇAMENTO))
    (=    (IERROS_FIM.ATIVIDADE)  ("FUNÇÕES CAD"))
  )
  (@HYPO=    HIP_CAD)
)

(@RULE=  R_CAD__1
  @INFCAT=2;
  (@LHS=
    (Yes  (HIP_CAD))
    (No   (CONF_AMBIENTE.HARDKEY))
  )
  (@HYPO=    HIP_PROBLEMA_CAD)
  (@RHS=
    (Assign (FALSE)      (DESEMP_ANORMAL.STATUS))
    (Assign (FALSE)      (CONF_AMBIENTE.ERRO_ENTRADA))
    (Assign (FALSE)      (LOG_PROG.STATUS))
  )
)

(@RULE=  R_CAD__2
  (@LHS=
    (Yes  (HIP_CAD))
    (Yes  (DESEMP_ANORMAL.STATUS))
    (No   (CONF_AMBIENTE.ARQ_INI))
  )
  (@HYPO=    HIP_PROBLEMA_CAD)
  (@RHS=
    (Assign (FALSE)      (CONF_AMBIENTE.ERRO_ENTRADA))
    (Assign (FALSE)      (LOG_PROG.STATUS))
  )
)

(@RULE=  R_CAD__3
  (@LHS=
    (Yes  (HIP_CAD))
    (Yes  (CONF_AMBIENTE.ERRO_ENTRADA))
  )
  (@HYPO=    HIP_PROBLEMA_CAD)
  (@RHS=
    (Assign (FALSE)      (DESEMP_ANORMAL.STATUS))
    (Assign (FALSE)      (LOG_PROG.STATUS))
  )
)

(@RULE=  R_CAD__4
  (@LHS=
    (Yes  (HIP_CAD))
    (Yes  (LOG_PROG.STATUS))
  )

```

```

)
(@HYPO=    HIP_PROBLEMA_CAD)
(@RHS=
    (Assign (FALSE)      (DESEMP_ANORMAL.STATUS))
    (Assign (FALSE)      (CONF_AMBIENTE.ERRO_ENTRADA))
)
)

(@RULE=    R_CAPP
  (@LHS=
    (Yes  (HIP_ORÇAMENTO))
    (=    (IERROS_FIM.ATIVIDADE)  ("FUNÇÕES CAPP"))
  )
  (@HYPO=    HIP_CAPP)
)

(@RULE=    R_CAPP__1
  (@LHS=
    (Yes  (HIP_CAPP))
  )
  (@HYPO=    HIP_PROBLEMA_CAPP)
)

(@RULE=    R_FIM_FABRICAÇÃO
  (@LHS=
    (=    (IERROS.TIPO) ("EXECUÇÃO DO SISTEMA"))
    (=    (IERROS_FIM.AREA) ("FABRICAÇÃO"))
  )
  (@HYPO=    HIP_FABRICAÇÃO)
)

(@RULE=    R_FIM_ORÇAMENTO
  (@LHS=
    (=    (IERROS.TIPO) ("EXECUÇÃO DO SISTEMA"))
    (=    (IERROS_FIM.AREA) ("ORÇAMENTO"))
  )
  (@HYPO=    HIP_ORÇAMENTO)
)

(@RULE=    R_HIP_RH
  (@LHS=
    (=    (IERROS.TIPO) ("LOCALIZAÇÃO DOS MEMBROS RESPONSÁVEIS
PELA APRESENTAÇÃO"))
  )
  (@HYPO=    HIP_RH)
  (@RHS=
    (Assign (AUX.HOR)      (AUX.HOR))
    (Assign (STR2TIME(AUX.HOR,"hh:mm"))      (IERROS_RH.HORARIO))
    (Assign (0)            (AUX.CURSOR))
    (Assign (RECUPERA_HORARIO)      (RECUPERA_HORARIO))
  )
)

(@RULE=    R_HORARIO_VALIDO
  (@LHS=

```



```

        (Assign (STR2TIME(AUX.HOR_ENTRADA,"hh:mm")) (AUX.H_ENTRADA))
        (<= (AUX.H_ENTRADA) (IERROS_RH1.HORARIO))
        (Assign (STR2TIME(AUX.HOR_SAIDA,"hh:mm")) (AUX.H_SAIDA))
        (<= (IERROS_RH1.HORARIO) (AUX.H_SAIDA))
    )
    (@HYPO= HORARIO_VALIDO)
    (@RHS=
        (CreateObject (\AUX.NOME\ (lsaidal))
        (Assign (AUX.H_ENTRADA) (\AUX.NOME\H_ENTRADA))
        (Assign (AUX.H_SAIDA) (\AUX.NOME\H_SAIDA))
        (Assign (AUX.LOCAL) (\AUX.NOME\LOCAL))
        (Assign (AUX.NOME) (\AUX.NOME\NOME))
    )
)

(@RULE= R_INDEXAÇÃO
    (@LHS=
        (Yes (INDEXAÇÃO.EL_REPETIDO))
    )
    (@HYPO= HIP_INDEXAÇÃO)
)

(@RULE= R_PCP
    (@LHS=
        (Yes (HIP_ORÇAMENTO))
        (= (IERROS_FIM1.ATIVIDADE) ("DEFINIR PRAZOS"))
    )
    (@HYPO= HIP_PCP)
)

(@RULE= R_PCP__1
    (@LHS=
        (Yes (HIP_PCP))
    )
    (@HYPO= HIP_PROBLEMA_PCP)
)

(@RULE= R_PLAN_ATIV
    (@LHS=
        (Yes (HIP_ORÇAMENTO))
        (= (IERROS_FIM1.ATIVIDADE) ("PLANEJAMENTO DE
ATIVIDADES"))
    )
    (@HYPO= HIP_PLAN_ATIV)
)

(@RULE= R_PLAN_ATIV__1
    (@LHS=
        (Yes (HIP_PLAN_ATIV))
        (Yes (DESEMP_ANORMAL.STATUS))
    )
    (@HYPO= HIP_PROBLEMA_PLAN_ATIV)
)

(@RULE= R_PROBL_ORÇAMENTO

```

```

    (@LHS=
      (Yes    (HIP_ORÇAMENTO))
      (=      (IERROS_FIM.ATIVIDADE) ("SOLICITAR ORÇAMENTO"))
      (Yes    (HIP_ABERT_ARQUIVO))
      (No     (CONF_AMBIENTE.ARQ_INI))
    )
    (@HYPO=    HIP_PROBLEMA_ORÇAMENTO)
  )

  (@RULE=    R_RECUPERA_HORARIO
    (@LHS=
      (>=     (AUX.CURSOR)(0))
    )
    (@HYPO=    RECUPERA_HORARIO)
    (@RHS=
      (Retrieve      ("horarios.nxp")
        (@TYPE=NXPD;@SLOTS=AUX.NOME,AUX.HOR_ENTRADA,\
        AUX.LOCAL,AUX.HOR_SAIDA;@FIELDS="NOME","HOR_ENTRADA",\
        "LOCAL","HOR_SAIDA";@CURSOR=AUX.CURSOR;))
        (Assign (HORARIO_VALIDO) (HORARIO_VALIDO))
        (Reset  (AUX.H_ENTRADA))
        (Reset  (AUX.H_SAIDA))
        (Reset  (HORARIO_VALIDO))
        (Reset  (RECUPERA_HORARIO))
      )
    )
  )

  (@RULE=    R_REDE_1
    (@LHS=
      (=      (IERROS.TIPO) ("CONEXÃO NA REDE"))
      (Yes    (CONEXÃO.STATUS))
    )
    (@HYPO=    HIP_PROBLEMA_REDE)
  )

  (@RULE=    R_SEM_DIAGNÓSTICO
    (@LHS=
      (No     (HIP_PROBLEMA_CAD))
      (No     (HIP_PROBLEMA_CAPP))
      (No     (HIP_PROBLEMA_ORÇAMENTO))
      (No     (HIP_PROBLEMA_PCP))
      (No     (HIP_PROBLEMA_PLAN_ATIV))
      (No     (HIP_PROBLEMA_REDE))
      (No     (HIP_RH))
      (No     (HIP_PROBLEMA_ÁRVORE))
    )
    (@HYPO=    HIP_SEM_DIAGNÓSTICO)
  )

```

## 7. Apêndice B - A Janela Principal da Interface Gráfica

Neste apêndice, é apresentado o código correspondente à Janela Principal da interface do Sistema Baseado em Conhecimento.

Na *Figura 10*, visualiza-se a aparência final da Janela Principal através do construtor de interfaces gráficas do Smart Elements. Os índices apontam para as diferentes partes da janela, sendo que a cada parte correspondem diferentes eventos descritos no código.

As partes da janela são (por ordem de índice):

- Botão ValidateBut (Índice 0) - é “apertado” quando se deseja passar alguma informação digitada para o motor de inferência, validando-a;
- Botão StartBut (Índice 1) - quando “pressionado”, inicia ou reinicia a sessão de diagnóstico, limpando da memória qualquer conclusão já encontrada;
- PromptLineField (Índice 2) - mostra a pergunta atual do motor de inferência, feita quando este necessita de mais informações;
- EscolhasCBox (Índice 3) - caixa de escolhas que mostra opções de resposta para aquilo que está sendo perguntado pelo motor de inferências do SBC em dado momento;
- Botão TerminateBut (Índice 4) - quando se deseja encerrar a sessão de diagnóstico, este botão deve ser clicado;
- TimeTEd (Índice 5) - neste editor de texto, digita-se o horário em que normalmente se encontra um responsável procurado, quando o problema é de RH.



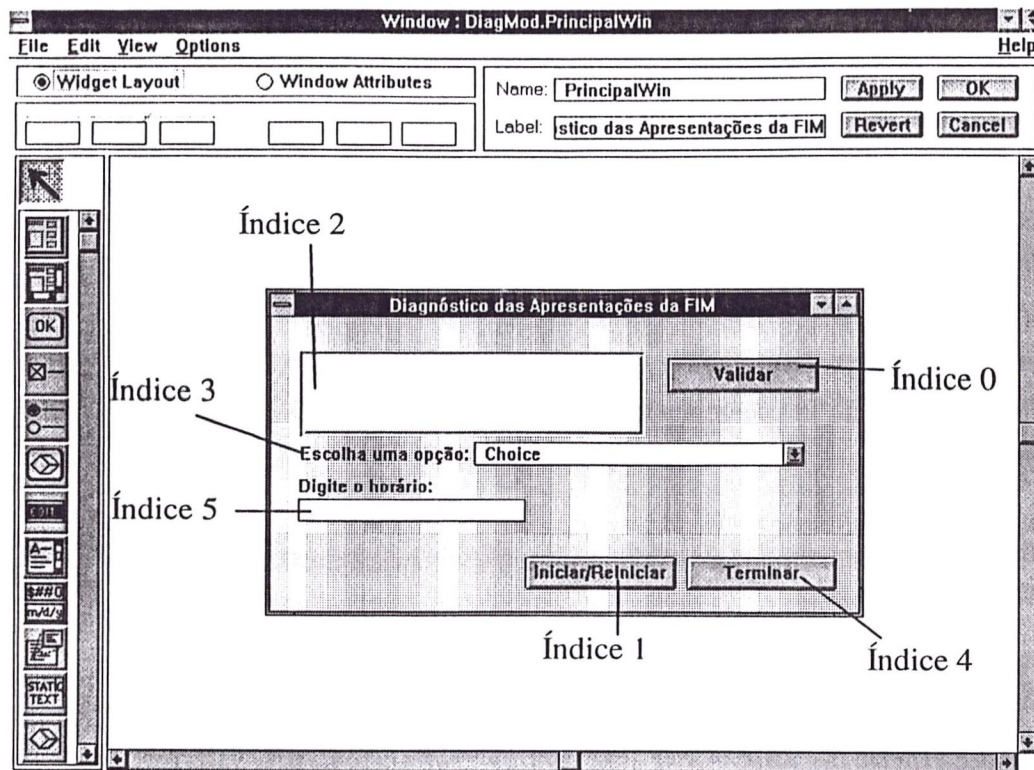


Figura 10- Janela Principal - DiagMod.PrincipalWin (Tela do Smart Elements)

#### // DiagMod.PrincipalWin

on event WIN\_OPENED

    // "Esconde" o TED utilizado para receber "horas procuradas"

    WGT\_SetInvisible(WIN\_GetNthWgt(WGT\_GetWin(SELF), 5));

end event

// DiagMod.PrincipalWin.ValidateBut (botão validar) - Índice 0

// Continua o "processamento de conhecimento" usando

// a resposta que o usuário deu à pergunta no

// PromptLineField.

// Variável global usada para distinguir entre problemas de

// localização de pessoas e os outros problemas (ORÇAMENTO, EXECUÇÃO, etc.)

global string Nome;

on event TBUT\_HIT

    // Se o problema não é de localização, a resposta será obtida

    // através do CBox (código de widget 3 - EscolhasCBox)

        if (Nome != "HOR")

            NOIR\_Volunteer(NOIR\_GetQuestionSlotId(), NXP\_DESC\_STR,  
                            CBOX\_ChosenGetLabel(WIN\_GetNthWgt(WGT\_GetWin(SELF),  
                            3)), NXP\_VSTRAT\_VOLFWRD);

    // Se o problema for de localização, a resposta será obtida

    // do TED (código de widget 5 - TimeTED)

        else

```

        NOIR_Volunteer(NOIR_GetQuestionSlotId(), NXP_DESC_STR,
            TED_GetStr(WIN_GetNthWgt(WGT_GetWin(SELF),
                5)), NXP_VSTRAT_VOLFWRD);
//Obtida a resposta, continua-se o processamento
    NOIR_Knowcess();
end event

on event NOIR_ENDQUESTION
//Depois que a resposta foi "passada adiante", o CBox é "limpado", ...
    CBOX_RemoveAllItems(WIN_GetNthWgt(WGT_GetWin(SELF), 3));
//... "desabilitado", ...
    WGT_Disable(WIN_GetNthWgt(WGT_GetWin(SELF), 3));
//... e o Ted também é "desabilitado" ...
    WGT_Disable(WIN_GetNthWgt(WGT_GetWin(SELF), 5));
//... e "limpado"
    TED_SetStr(WIN_GetNthWgt(WGT_GetWin(SELF), 5), "");
end event

// DiagMod.PrincipalWin.StartBut (botão iniciar/reiniciar) - Índice 1
// Inicializa/reinicializa a "sessão de processamento de
// conhecimento", sugerindo a hipótese "HIP_RH" como a inicial.

on event TBUT_HIT
//"Esconde" o Ted utilizado para receber "horas procuradas"
    WGT_SetInvisible(WIN_GetNthWgt(WGT_GetWin(SELF), 5));
//Inicializa/reinicializa a "sessão de processamento de conhecimento"
    NOIR_RestartSession();
//Sugere (ou coloca na agenda) a hipótese "HIP_RH"
    NOIR_Suggest( NOIR_GetAtomId("HIP_SEM_DIAGNÓSTICO", NXP_ATYPE_HYPO),
        NXP_SPRIO_SUG);
//Começa o "processamento"
    NOIR_Knowcess();
end event

// DiagMod.PrincipalWin.PromptLineField (widget que guarda as questões) - Índice 2
// Quando necessita-se fazer uma pergunta ao usuário
// este campo (janela) será "inicializado" (receberá)
// o "prompt line" atual. Ou seja, a pergunta definida
// para o valor desconhecido é passada ao PromptLineField.
// Quando não, será desabilitado.

on event NOIR_STARTQUESTION
//Coloca o "prompt line" atual no PromptLineField
    TED_SetStr(SELF, NOIR_GetQuestionPrompt());
end event

on event NOIR_ENDQUESTION
//"Limpa" o conteúdo do campo (janela)
    TED_SetStr(SELF, "");
//Desabilita o campo (janela)
    WGT_Disable(SELF);

```

end event

```
// DiagMod.PrincipalWin.EscolhasCBox(widget escolha uma opção) - Índice 3
// Quando necessita-se fazer uma pergunta, o "choice box" é
// inicializado com as opções de valor para o slot corrente (atual).
// Quando o usuário seleciona uma opção, o slot corrente é setado.
// The choice box is disabled when the question ends since
// the question panel is part of the application main window.
```

```
//Variável global usada para distinguir entre problemas de
//localização de pessoas e os outros problemas (ORÇAMENTO, EXECUÇÃO, etc.)
global string Nome;
```

on event NOIR\_STARTQUESTION

```
integer TipoValor;
pointer Prop;
```

```
Slot = NOIR_GetCurrentAtom(NXP_ATYPE_SLOT);
TipoValor = NOIR_GetIntInfo(Slot, NXP_AINFO_VALUETYPE);
Prop = NOIR_GetAtomInfo(Slot, NXP_AINFO_PROP);
Nome = NOIR_GetName(Prop);
if (Nome != "HOR")
{ WGT_SetVisible(SELF);
  WGT_SetInvisible(WIN_GetNthWgt(WGT_GetWin(SELF), 5));
  WGT_Enable(SELF);
  WGT_GainFocus(SELF);
  NOIR_CBoxSetSlotChoices(SELF, Slot);
  if (TipoValor == NXP_VTYPE_STR)
  { numItens = CBOX_GetNumItems(SELF);
    CBOX_AddItemById(SELF, numItens, "NãoConhecido", numItens+1,1);
  }
}
```

end event

on event NOIR\_ENDQUESTION

```
WGT_Disable(SELF);
end event
```

```
// DiagMod.PrincipalWin.TerminateBut (botão terminar) - Índice 4
```

on event TBUT\_HIT

```
WIN_Terminate(WGT_GetWin(SELF));
end event
```

```
// DiagMod.PrincipalWin.TimeTEd (widget digite horário) - Índice 5
```

global string Nome;

on event TED\_VALIDATE

```
NOIR_Volunteer(NOIR_GetCurrentAtom(NXP_ATYPE_SLOT), NXP_DESC_STR,
TED_GetStr(SELF), NXP_VSTRAT_VOLFWRD);
```



```
WGT_Disable(SELF);
TED_SctStr(SELF, "");
NOIR_Knowcess();
end event

on event NOIR_STARTQUESTION
  if (Nome == "HOR")
  { WGT_SetInvisible(WIN_GetNthWgt(WGT_GetWin(SELF), 3));
    WGT_SetVisible(SELF);
    WGT_Enable(SELF);
    WGT_GainFocus(SELF);
  }
end event
```

## 8. Apêndice C - A Janela de Diagnósticos

O código da Janela de Diagnóstico da interface do Sistema Baseado em Conhecimento encontra-se neste apêndice. A janela é composta por uma sub-janela de “Erros Encontrados” (Índice 0) e outra de “Ação (Ações) Corretiva(s)” (Índice 1). Esta relação é mostrada na *Figura 11*.

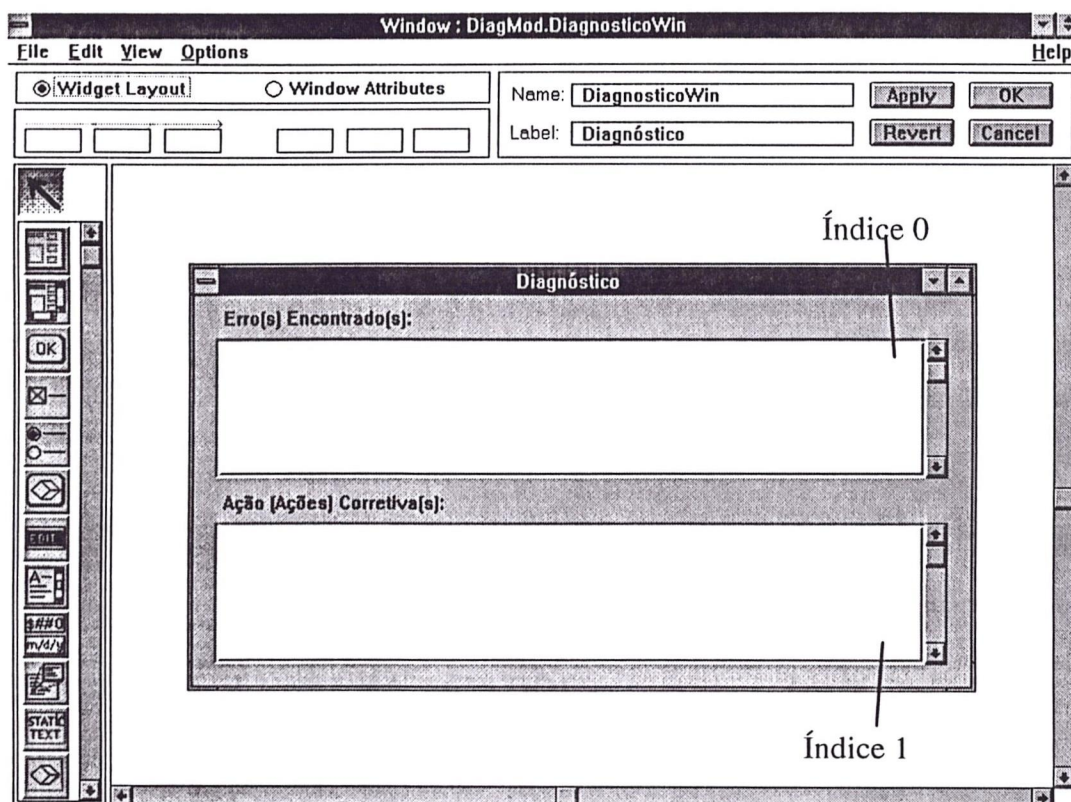


Figura 11 - Janela de Diagnósticos - DiagMod.DiagnosticoWin (Tela do Smart Elements)

### // DiagMod.DiagnosticoWin

```
// Ao ser aberta esta janela (DiagMod.DiagnosticoWin), todos os erros
// provados são recuperados e, através de seus campos de descrição, são
// retornados ao usuário na WIDGET de Erro(s) Encontrado(s).
// Após isso, uma hipótese final provada (um problema ou Sem_diagnósti-
// co) também é recuperada e então uma ação (ações) corretiva(s) é (são)
// apresentada(s) ao usuário na WIDGET de Ação(Ações) Corretiva(s).
```

```
on event WIN_OPENED
```

```
integer orc,arvore,plan_ativ,cad, // Variáveis que armazenam o valor
tipo_cad,capp,pcp,fabric, // das hipóteses (provadas ou não)
rede,rh,sem_diag, // para serem retornadas ao usuário
```

```

    val_erro; // Variável que guarda o valor de propriedades
               // (erros) relacionando-as com suas descrições e
               // retornando ao usuário.

    string text,aux,enter; // Variáveis que armazenam as descrições dos
                           // erros.

// ----- INICIALIZAÇÃO DAS VARIÁVEIS -----

orc = 0;
arvore = 0;
plan_ativ = 0;
cad = 0;
tipo_cad = 0;
capp = 0;
pcp = 0;
fabric = 0;
rede = 0;
rh = 0;
sem_diag = 0;
text = "";
enter = "
";

// ----- PARTE DOS ERROS ENCONTRADOS -----

if (NOIR_GetUnknownVal(NOIR_GetAtomId("ABERT_ARQUIVO.PROBLEMA",
NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{
    val_erro = NOIR_GetIntVal(NOIR_GetAtomId ("ABERT_ARQUIVO PROBLEMA",
NXP_ATYPE_SLOT));
    if (!val_erro) // Teste para verificar se houve problema na abertura
                  // de arquivo.
        aux = "";
    else
        aux = NOIR_GetStrVal(NOIR_GetAtomId("ABERT_ARQUIVO. DESCRIÇÃO",
NXP_ATYPE_SLOT));
    text = aux;
}

if (NOIR_GetUnknownVal(NOIR_GetAtomId("CONF_AMBIENTE. ARQ_INI",
NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{

```



```
val_erro = NOIR_GetIntVal(NOIR_GetAtomId("CONF_AMBIENTE. ARQ_INI",
NXP_ATYPE_SLOT));
if (val_erro) // Teste para verificar se houve problema nos arquivos
    // iniciais.
    aux = "";
else
    aux = NOIR_GetStrVal(NOIR_GetAtomId("CONF_AMBIENTE. DESCRIÇÃO",
NXP_ATYPE_SLOT));
if (text == "")
    text = aux;
else
    text = text+enter+aux;
}

if (NOIR_GetUnknownVal(NOIR_GetAtomId("INDEXAÇÃO. EL_REPETIDO",
NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{
    val_erro = NOIR_GetIntVal(NOIR_GetAtomId("INDEXAÇÃO. EL_REPETIDO",
NXP_ATYPE_SLOT));
    if (!val_erro) // Teste para verificar se houve problema na indexação.
        aux = "";
    else
        aux = NOIR_GetStrVal(NOIR_GetAtomId("INDEXAÇÃO.DESCRICÃO",
NXP_ATYPE_SLOT));
    if (text == "")
        text = aux;
    else
        text = text+enter+aux;
}

if (NOIR_GetUnknownVal(NOIR_GetAtomId("DESEMP_ANORMAL. STATUS",
NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{
    val_erro = NOIR_GetIntVal(NOIR_GetAtomId("DESEMP_ANORMAL. STATUS",
NXP_ATYPE_SLOT));
    if (!val_erro) // Teste para verificar se houve problema no desempenho
        // do software.
        aux = "";
    else
        aux = NOIR_GetStrVal(NOIR_GetAtomId("DESEMP_ANORMAL. DESCRIÇÃO",
NXP_ATYPE_SLOT));
    if (text == "")
        text = aux;
    else
        text = text + enter + aux;
```

```
//enter+aux;
}

if (NOIR_GetUnknownVal(NOIR_GetAtomId("CONF_AMBIENTE. HARDKEY",
NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{
    val_erro = NOIR_GetIntVal(NOIR_GetAtomId("CONF_AMBIENTE. HARDKEY",
NXP_ATYPE_SLOT));
    if (val_erro) // Teste para verificar se houve problema com o
        // hardkey.
        aux = "";
    else
        aux = NOIR_GetStrVal(NOIR_GetAtomId("CONF_AMBIENTE. DESCRIÇÃO",
NXP_ATYPE_SLOT));
    if (text == "")
        text = aux;
    else
        text = text+enter+aux;
}

if (NOIR_GetUnknownVal(NOIR_GetAtomId("CONF_AMBIENTE. ERRO_ENTRADA",
NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{
    val_erro = NOIR_GetIntVal(NOIR_GetAtomId("CONF_AMBIENTE. ERRO_ENTRADA",
NXP_ATYPE_SLOT));
    if (!val_erro) // Teste para verificar se houve problema na inicializa-
        // ção do software.
        aux = "";
    else
        aux = NOIR_GetStrVal(NOIR_GetAtomId("CONF_AMBIENTE. DESCRIÇÃO",
NXP_ATYPE_SLOT));
    if (text == "")
        text = aux;
    else
        text = text+enter+aux;
}

if (NOIR_GetUnknownVal(NOIR_GetAtomId("LOG_PROG.STATUS",
NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{
    val_erro = NOIR_GetIntVal(NOIR_GetAtomId("LOG_PROG.STATUS",
```

```

        NXP_ATYPE_SLOT));
if (!val_erro) // Teste para verificar se houve problema na lógica
    // do programa.
    aux = "";
else
    aux = NOIR_GetStrVal(NOIR_GetAtomId("LOG_PROG.DESCRICÃO",
        NXP_ATYPE_SLOT));
if (text == "")
    text = aux;
else
    text = text+enter+aux;
}

if (NOIR_GetUnknownVal(NOIR_GetAtomId("CONEXÃO.STATUS",
    NXP_ATYPE_SLOT)) == 0)

// Teste para verificar se o valor deste slot (propriedade) não é
// Unknown.

{
    val_erro = NOIR_GetIntVal(NOIR_GetAtomId("CONEXÃO.STATUS",
        NXP_ATYPE_SLOT));
if (!val_erro) // Teste para verificar se houve problema na conexão
    // na rede.
    aux = "";
else
    aux = NOIR_GetStrVal(NOIR_GetAtomId("CONEXÃO.DESCRICÃO",
        NXP_ATYPE_SLOT));
if (text == "")
    text = aux;
else
    text = text+enter+aux;
}

// ----- PARTE DE INSERIR NA TELA OS ERROS ENCONTRADOS -----

TED_SetStr(WIN_GetNthWgt(SELF,0),text); // Esta função transfere a
// string text para a WIDGET Erro(s) Encontrado(s)

// ----- PARTE DAS AÇÕES CORRETIVAS -----

// ----- PROBLEMA ORÇAMENTO -----

orc = NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_    ORÇAMENTO",
    NXP_ATYPE_HYPO));

// A variável orc recebe o valor da hipótese HIP_PROBLEMA_ORÇAMENTO
// (True ou False)

```



```
// ----- PROBLEMA ÁRVORE -----
```

```
arvore      =      NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_      ÁRVORE",  
NXP_ATYPE_HYPO));
```

```
// A variável arvore recebe o valor da hipótese HIP_PROBLEMA_ÁRVORE  
// (True ou False)
```

```
// ----- PROBLEMA PLANEJAMENTO DE ATIVIDADES -----
```

```
plan_ativ   =      NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_      PLAN_ATIV",  
NXP_ATYPE_HYPO));
```

```
// A variável plan_ativ recebe o valor da hipótese HIP_PROBLEMA_  
// PLAN_ATIV (True ou False)
```

```
// ----- PROBLEMA NO CAD -----
```

```
cad = NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_CAD",  
NXP_ATYPE_HYPO));
```

```
// A variável cad recebe o valor da hipótese HIP_PROBLEMA_CAD  
// (True ou False)
```

```
tipo_cad = 0;
```

```
if (cad) // Teste para verificar se houve problema de CAD
```

```
if      (NOIR_GetUnknownVal(NOIR_GetAtomId("HIP_PROBLEMA_CAD.      TIPO_CAD",  
NXP_ATYPE_SLOT)) == 0)
```

```
// Teste para verificar se o valor deste slot (propriedade) não é  
// Unknown.
```

```
{
```

```
    tipo_cad  =      NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_CAD.      TIPO_CAD",  
NXP_ATYPE_SLOT));
```

```
    // A variável tipo_cad recebe o tipo do problema de CAD que foi provado
```

```
}
```

```
// ----- PROBLEMA NO CAPP -----
```

```
capp = NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_CAPP",  
NXP_ATYPE_HYPO));
```

```
// A variável capp recebe o valor da hipótese HIP_PROBLEMA_CAPP  
// (True ou False)
```

```
// ----- PROBLEMA NO PCP -----
```

```

pcp = NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_PCP",
    NXP_ATYPE_HYPO));

// A variável pcp recebe o valor da hipótese HIP_PROBLEMA_PCP
// (True ou False)

// ----- PROBLEMA DE FABRICAÇÃO -----

fabric = NOIR_GetIntVal(NOIR_GetAtomId("HIP_FABRICAÇÃO",
    NXP_ATYPE_HYPO));

// A variável fabric recebe o valor da hipótese HIP_FABRICAÇÃO
// (True ou False)

// ----- PROBLEMA REDE -----

rede = NOIR_GetIntVal(NOIR_GetAtomId("HIP_PROBLEMA_REDE",
    NXP_ATYPE_HYPO));

// A variável rede recebe o valor da hipótese HIP_PROBLEMA_REDE
// (True ou False)

// ----- PROBLEMA SEM DIAGNÓSTICO -----

sem_diag = NOIR_GetIntVal(NOIR_GetAtomId("HIP_SEM_    DIAGNÓSTICO",
    NXP_ATYPE_HYPO));

// A variável sem_diag recebe o valor da hipótese HIP_SEM_DIAGNÓS-
// TICO (True ou False)

// ---- PARTE DE INSERIR NA TELA AS AÇÕES CORRETIVAS ----

if (orc) // Teste para verificar se houve problema de orçamento
    TED_SetStr(WIN_GetNthWgt(SELF,1),
    "Há um problema provavelmente com o arquivo REQUISIC.DOC. Vá para o prompt do sistema
operacional e digite SETAR FIM, reinicializando o computador em seguida. ");

// Esta função transfere o texto acima para a WIDGET Ação(Ações)
// Corretiva(s)

if (arvore) // Teste para verificar se houve problema de árvore do produto
    TED_SetStr(WIN_GetNthWgt(SELF,1),
    "A base de dados de peças está inconsistente. Consulte os responsáveis pelas apresentações da FIM
para realizar manutenção na base.");

// Esta função transfere o texto acima para a WIDGET Ação(Ações)
// Corretiva(s)

if (plan_ativ) // Teste para verificar se houve problema de planejamento de atividades
    TED_SetStr(WIN_GetNthWgt(SELF,1),

```

"O arquivo SETDOC.DBF está muito grande! Vá para o prompt do sistema operacional e execute o comando SETUP.");

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

if (tipo\_cad == 1) // Teste para verificar se o problema de CAD foi do  
// tipo 1 (hardkey)

TED\_SetStr(WIN\_GetNthWgt(SELF,1),

"Coloque a hardkey do software de CAD na porta paralela do microcomputador.");

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

if (tipo\_cad == 2) // Teste para verificar se o problema de CAD foi do tipo 2  
// (desempenho insatisfatório ou está mais lento e arquivos  
// de inicialização não foram configurados)

TED\_SetStr(WIN\_GetNthWgt(SELF,1),

"As variáveis de ambiente para a execução do software de CAD não foram inicializadas. Certifique-se que você está mesmo no micro designado para apresentações do CAD. Caso a resposta seja positiva, vá para o prompt do sistema operacional e digite o comando SETAR FIM, reiniciando o computador em seguida.");

// Esta função transfere o texto acima para a WIDGET Ação (Ações)  
// Corretiva(s)

if (tipo\_cad == 3) // Teste para verificar se o problema de CAD foi do  
// tipo 3 (inicialização anormal)

TED\_SetStr(WIN\_GetNthWgt(SELF,1),

"Houve um erro durante alguma utilização anterior do software de CAD. Vá para o diretório onde o software de CAD está instalado e elimine o arquivo DEMO.DWG.");

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

if (tipo\_cad == 4) // Teste para verificar se o problema de CAD foi do  
// tipo 4 (lógica de programa)

TED\_SetStr(WIN\_GetNthWgt(SELF,1),

"Há problemas na lógica do programa. Este problema só terá solução nos próximos cenários da FIM.");

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

if (capp) // Teste para verificar se houve problema de capp

TED\_SetStr(WIN\_GetNthWgt(SELF,1),

"Há um problema com o capp");

// Esta função transfere o texto acima para a WIDGET Ação(Ações)



// Corretiva(s)

```
if (pcp) // Teste para verificar se houve problema de pcp
    TED_SetStr(WIN_GetNthWgt(SELF,1),
    "Há um problema com o pcp");
```

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

```
if (fabric) // Teste para verificar se houve problema na fabricação
    TED_SetStr(WIN_GetNthWgt(SELF,1),
    "Há um problema com a fabricação");
```

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

```
if (rede) // Teste para verificar se houve problema de rede
    TED_SetStr(WIN_GetNthWgt(SELF,1),
    "Durante o comando login não foi possível localizar os diretórios onde reside o software responsável pelas apresentações da FIM. Isto ocorreu porque o volume DEMO, na rede Novell, não está montado e assim é impossível o acesso à shell da FIM. Para resolver o problema, peça a algum DTI que monte o volume especificado.");
```

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

```
if (sem_diag) // Teste para verificar se não foi possível chegar a algum problema já editado na base
    TED_SetStr(WIN_GetNthWgt(SELF,1),
    "Não foi possível chegar a nenhum diagnóstico. Procure contactar algum responsável pelo sistema para a implementação deste novo erro na base de conhecimento.");
```

// Esta função transfere o texto acima para a WIDGET Ação(Ações)  
// Corretiva(s)

end event