UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Um Sistema de Apoio à Gerência de Redes Locais

por

Fernando Luís Dotti

Dissertação submetida como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Profa. Liane Margarida Rockenbach Tarouco
Orientadora

Porto Alegre, março de 1992.

CIP - CATALOGAÇÃO NA PUBLICAÇÃO

Dotti, Fernando Luís

Um Sistema de Apoio à Gerência de Redes Locais / Fernando Luís Dotti.—Porto Alegre: CPGCC da UFRGS, 1992.

190 p.: il.

Dissertação (mestrado)—Universidade Federal do Rio Grande do Sul, Curso de Pós-Graduação em Ciência da Computação, Porto Alegre, 1992. Orientador: Tarouco, Liane Margarida Rockenbach

Dissertação: Comunicação de Dados
Gerência de Redes, Redes Locais, Sistemas Especialistas

AGRADECIMENTOS

Gostaria de agradecer ao CNPq, à CAPES e à IBM-Brasil pelo apoio financeiro prestado, custeando minha bolsa de mestrado.

À Dra. Liane M. R. Tarouco, pelo apoio pessoal e profissional dedicados, agradeço sinceramente.

Aos demais, que me cercaram e me apoiaram, obrigado.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

- **ARP** - Address Resolution Protocol
- **AT&T** - American Telephone and Telegraph
- **BNA** - Burroughs Network Architecture
- **CCITT** - International Telegraph and Telephone Consultative Commitee
- **CMIP** - Common Management Information Protocol
- **CMIS** - Common Management Information Service
- **CMOT** - CMIP Over TCP
- **CSMA/CD** - Carrier Sense Multiple Access / Collision Detection
- **DEC** - Digital Equipment Corporation
- **DNA** - DEC Network Architecture
- **DARPA** - Defense Advanced Research Projects Agency
- **FTP** - File Transfer Protocol
- **ICMP** - Internet Control Message Protocol
- **IEEE** - Intitute of Electrical and Eletronics Engineers
- **IBM** - International Business Machines Corporation
- **IP** - Internet Protocol
- **ISO** - International Organization for Standardization
- **LAN** - Local Area Network
- **MIB** - Management Information Base
- **OSI** - Open Systems Interconnection
- **OSINM** - Open Systems Interconnection Network Management
- **PDU** - Protocol Data Unit
- **SMAE** - Systems Management Application Entity
- **SMAP** - Systems Management Application Proccess
- **SMTP** - Simple Mail Transfer Protocol
- **SNMP** - Simple Network Management Protocol

- **TCP** - Transmission Control Protocol
- **UDP** - User Datagram Protocol

RESUMO

O fluxo de informações na sociedade moderna é crescente e mais intenso a cada dia. Cada vez mais as ferramentas de tratamento da informação ou seja, os computadores, necessitam interconexão de forma a proporcionar acesso a dados importantes e serviços especializados. Neste contexto, as redes de computadores agem como catalizadores do processo de disseminação e tratamento da informação. A demanda por serviços distribuídos cresce a cada dia, fazendo com que as redes cresçam também. Muitas vezes este processo é rápido e desordenado, a entropia do sistema aumenta e seu controle torna-se uma tarefa árdua.

Para que possa exercer maior controle sobre as redes de computadores o gerente da rede deve dispor de ferramentas (e métodos) de auxílio. Assim pode-se manter a prestação de serviços a que a rede se propõe. Em Gerência de Redes, duas arquiteturas têm sido apontadas como "modelos" a serem adotados: a arquitetura OSI para gerência de redes (OSINM) [DOT91] e a arquitetura INTERNET [SCH91]. Ambas arquiteturas consideram que os diferentes componentes inseridos no domínio de gerência são capazes de emitir e decodificar mensagens de gerência. Por outro lado, tem sido destacada a necessidade de gerenciar componentes que não suportam funções de gerência ou que não possuem as mesmas funções de gerência que o restante da rede, desta forma pretende-se que o sistema de gerência da rede seja completo e confiável.

O sistema especificado e prototipado se encaixa neste contexto pois é orientado à gerência de Redes Locais do tipo CSMA/CD que ainda não possuem funções de gerência. Defende-se a idéia de que, a partir do tráfego gerado, pode-se inferir o perfil ou estado da rede, derivando ou antecipando os problemas da instalação que não poderiam ser detectados pelo administrador da rede de outra maneira [DOT90]. Assim sendo, uma estação da rede é escolhida para este trabalho. Esta estação captura o tráfego da rede e o submete continuamente a um processo de análise que detecta as diversas situações anormais que podem ser verificadas nos

padrões de tráfego. As ocorrências de situações anormais, denominadas alarmes, são repassadas a um processo de diagnose. O processo de diagnose, contendo uma base de conhecimento em forma de regras e a descrição estrutural da rede, leva em conta estes alarmes para reconhecer o problema. Uma vez reconhecido o problema, formula-se um procedimento de correção que é levado ao Gerente da Rede.

O sistema foi especificado com o uso de dois métodos formais:

- Para especificar a interação do sistema com o ambiente em que se insere foi usado o método CSP *(Communicating Sequential Processes)*, proposto por Hoare em 1978 [HOA87] e reformulado também por Hoare em 1985 [HOA85];
- Para especificar o sistema em si, o funcionamento do processo de agregação de conhecimento, foi utilizado o método denotacional VDM *(Vienna Development Method)*, [BJØ78] e [JON86].

Para a construção do protótipo foram utilizadas as linguagens "C" e "PROLOG".

Futuramente, quando inserido em um Domínio de Gerência (segundo a terminologia OSI/ISO), este sistema poderá reportar os problemas que fogem ao escopo da base de conhecimento local para o Processo Gerente do Domínio que responderá com os procedimentos de solução cabíveis à porção da rede gerenciada pelo sistema.

Palavras-chave:

Gerência de Redes, Redes Locais, Sistemas Especialistas.

ABSTRACT

The information flow in the modern society becomes bigger and faster day by day. Aidding the task of information treating, the computers need to be interconnected in order to proportionate access to important data and specialized services. In this context, the computer networks are like catalyzers of the process of dissemination and treating of information. The demand for distributed services is increasing, making the computer networks increase too. Many times this growth is fast and disordered and the network control becomes a hard task.

For achieving more control over the computer network, powerful tools and methods for problem solving must be provided to the Network Manager. With this, the services offered by the network can be kept running. In Network Management, two architectures have been pointed as "models" to be used: the OSI network management architecture (OSINM) [DOT91] and the INTERNET architecture [SCH91]. Both consider that every network component must be capable of sending and receiving management messages. However, there are many components that do not support management functions, or that have management functions that are distinct of the rest of the network. This different components must be managed because without this the management system would not be representative and trustworthy.

The specified and prototyped system deals with the problem above boarded. It is oriented to the management of a CSMA/CD Local Area Networks that do not support management functions. The central idea is that, through traffic analysis, the network state can be obtained (or infered) and the problems arising in the network can be detected or even predicted.

A network station is chosen to perform this task. This station gathers all traffic and submit it continuously to an analysis process that detects the different anormal situations over the network. The occurrences of anormal situations, called

alarms, feeds a diagnosis process. This process has a knowledge base with rules and the structural description of the network, it uses this knowledge and the alarms to recognize the network problems. Once the problems are recognized, the system is ready for advise the possible corrections to the Network Manager.

The system was specified using two formal methods:

- To specify the interaction of the system with the environment in which it is embeeded was used CSP (Communicating Sequential Processes), proposed by Hoare in 1978 [HOA87] and reformulated by Hoare too in 1985 [HOA85];
- To specify the software, the way the knowledge aggregation task acts, was used the denotational method VDM (Vienna Development Method), [BJØ78] and [JON86].

For the prototyping were used the languages "C" and PROLOG.

In the future, when embeded in a Management Domain, this system will be capable of reporting the problems that are not treated by the local knowledge base (or that are not in the scope of the system) to the Management Process of the domain. The Management Process will give some advising and the local system will obey to it.

Keywords:

Network Management, Local Area Networks, Expert Systems.

1 INTRODUÇÃO

1.1 Contexto

Observando-se a evolução da nossa sociedade, pode-se notar que os eventos marcantes geralmente estão acompanhados de situações de crise e inversões na escala comum de valores. Uma vez que, atualmente, as pessoas e corporações devem ser cada vez mais eficientes em suas funções, a "informação" torna-se um elemento indispensável ao processo decisório. O elemento "informação" está subindo na escala comum de valores.

Este fenômeno social naturalmente estimula o processo de desenvolvimento tecnológico, fazendo com que novos rumos sejam tomados. O surgimento e evolução acelerados de novas formas de tratamento e disseminação da informação são fortes indicativos deste processo, revelando a necessidade crescente, por parte de organizações e pessoas, de se alimentar de informações e deduzir sobre estas.

O fluxo crescente de informação gera um conjunto maior de opções capazes de sanar problemas ou necessidades. A nível organizacional, o resultado deste processo é o surgimento de novos produtos e serviços, e a competitividade e eficiência na sua prestação. A nível pessoal, a população se torna cada vez mais heterogênea, pois os indivíduos dispõem das opções que mais lhe aprazem. Esta heterogeneidade crescente de opiniões e serviços age como multiplicadora da quantidade de informação no meio.

A informática se encaixa neste contexto como uma ferramenta aceleradora do processo de informação e tomada de decisão, pois as máquinas viabilizam o tratamento da grande quantidade de dados disponíveis. A disseminação da informação, por sua vez, acontece entre estas máquinas de apoio, de modo a alimentá-las com

dados, formando as redes de computadores. Desta forma, o processo de informação da sociedade moderna está sendo reformulado, tornando-se bem mais ágil.

As redes de computadores estão surgindo e crescendo rapidamente, formando um cenário em que a construção de software na área de comunicação de dados, bem como a gerência de redes, são tarefas árduas. O diagnóstico de problemas fica mais difícil à medida que a rede cresce, principalmente devido à dificuldade na obtenção de informacões relevantes para este fim.

Por outro lado, tem surgido na comunidade pesquisadora e usuária, um conjunto de ferramentas orientadas à monitoração do tráfego na rede. As informacões assim obtidas propiciam o acompanhamento periódico da rede, obtendo-se dados estatistícos sobre erros, *timeouts*, utilização de recursos, além de outras informacões relevantes, podendo levar o gerente da rede ou construtor do software a apontar possíveis problemas com mais exatidão e rapidez. Contudo, o excesso de informações derivadas da simples monitoração e tabulação dos dados concernentes ao tráfego da rede pode inibir uma análise mais acurada, fenômeno conhecido como "indigestão de informações". Nota-se, ainda, a necessidade de uma ferramenta que efetue um tratamento preliminar dos dados "brutos", de forma a apresentá-los em volume e forma mais adequados, realizando um processo de agregação de informações.

1.2 Objetivos

Dentro do contexto anteriormente descrito, verifica-se a utilidade de uma ferramenta para auxiliar o Gerente da Rede, fornecendo apoio na tomada de decisões.

A gerência de uma rede local de porte considerável é uma tarefa complexa que não se esgota somente no trabalho de garantir a conectividade entre os componentes de hardware da rede. Deve-se realizar um trabalho mais extenso, envolvendo:

detecção e isolamento de falhas, avaliação de desempenho, cuidados com a segurança dos recursos, contabilização de utilização e manutenção da configuração.

Neste sentido, este trabalho apresenta *Um Sistema de Apoio à Gerência de Redes Locais.* Defende-se a idéia de que, a partir do tráfego gerado, pode-se inferir o perfil ou estado da rede, derivando ou antecipando os problemas da instalação que não poderiam ser "sentidos" pelo administrador da rede de outra maneira [DOT90]. A partir desta idéia foi projetada a estrutura de uma ferramenta de apoio à gerência de redes, desenvolvida na UFRGS.

1.3 Estruturação deste trabalho

Este trabalho está disposto basicamente em seis partes principais:

- considerações importantes a este trabalho sobre Gerência de Redes, seguindo, principalmente, as normas da ISO para este fim;
- o estado da arte em Gerência de Redes, ressaltando diversos aspectos na área, como: arquiteturas de gerência, sistemas de gerência, compatibilização de equipamentos heterogêneos, e outras práticas;
- o projeto proposto para o sistema, abordando o processo de modelagem do conhecimento específico da área, as estruturas de representação para este conhecimento e os mecanismos de inferência sobre o mesmo;
- o protótipo implementado, ressaltando o processo de agregação do conhecimento (envolvendo o conhecimento do especialista codificado em regras e heurísticas);
- as considerações finais sobre o trabalho, ressaltando trabalhos futuros e analisando o trabalho desenvolvido;

- dois apêndices, o primeiro contendo a base de regras e a base de conhecimento estrutural, e o segundo contendo os procedimentos de correção que o sistema pode aconselhar.

2 GERÊNCIA DE REDES DE COMPUTADORES NA ARQUITETURA OSI

2.1 Introdução

Para o desenvolvimento de padrões na área de gerência de redes, a ISO estabeleceu o SC21/WG4 em Março de 1985.

Nos capítulos posteriores, serão necessárias algumas noções sobre gerência de redes. Assim sendo, neste capítulo são apresentados alguns conceitos já sedimentados da arquitetura OSI da ISO para gerência de redes. O Modelo Básico de referência OSI (as sete camadas do modelo para comunicação de computadores) é bastante disseminado na literatura corrente e não será apresentado, sugerindo-se consultar [TAN88] e/ou [TAR86] para obter maiores detalhes sobre o modelo.

2.2 Arquitetura de Gerência OSI

A Gerência OSI é definida, em [ISO86], como: "as facilidades proporcionadas pela operação de gerência de sistemas e gerência de camadas para supervisionar e controlar os recursos OSI". Desta forma, o ambiente de Gerência OSI consiste de dados e serviços necessários para controlar e supervisionar as atividades de interconexão e qualquer objeto gerenciado associado. A arquitetura de Gerência OSI é, assim, distribuída pelas estações da rede, realizando tarefas que vão desde a captação de pequenas porções de informação sobre elementos remotos até a agregação destas informações em um nodo gerente, inferindo o estado da comunicação.

A seguir, serão apresentados alguns conceitos necessários à compreensão desta arquitetura.

2.2.1 Domínio de Gerência

Domínio de Gerência é a extensão gerenciada da rede. Um domínio de gerência pode ser composto de uma coleção de Sistemas Gerentes e uma coleção de Sistemas Gerenciados.

2.2.2 Sistema Gerente

Um Sistema Gerente é responsável por uma fração do Domínio Gerenciado agregando as informações dos Sistemas Gerenciados de sua responsabilidade. Para obtenção destas informações, o Sistema Gerente, através de um Processo Gerente, invoca o Sistema Gerenciado, mais especificamente o Processo Agente no Sistema Gerenciado, que tem acesso aos Objetos Gerenciados naquele nodo.

2.2.3 Sistema Gerenciado

O Sistema Gerenciado abriga uma coleção de Objetos Gerenciados e responde, através do Processo Agente, aos pedidos do Sistema Gerente de informação sobre seus objetos. Ao Agente cabe, também, reportar ao Sistema Gerente os eventos ocorridos com os seus Objetos Gerenciados.

2.2.4 Objeto Gerenciado

Objetos Gerenciados *(Managed Objects - MO)* são o alvo de todas as operações de gerência OSI. Um Objeto Gerenciado é a representação interna do sistema para um recurso de comunicação gerenciado. Um Objeto Gerenciado consiste de:

- Atributos - representam valores do recurso relacionado (sendo gerenciado) que podem ser lidos e modificados pelo Sistema Gerente ao qual este objeto pertence;
- Eventos - são mensagens pré-definidas que serão reportadas do Sistema Gerenciado para o Gerente no caso de uma transição de estado relevante;
- Ações - podem ser iniciadas no Sistema Gerenciado, isto permite ao Sistema Gerente pedir a um sistema aberto que inicialize seus recursos, reinicialize-se, ou mesmo realize funções de teste, mantendo o nível de abstração;
- outros Objetos Gerenciados contidos neste objeto - esta característica leva a uma estrutura hierárquica (em árvore) em que, por exemplo, o objeto de nível mais alto é da classe "Sistema" e representa toda estação através de uma composição de objetos.

Uma instância de Objeto Gerenciado é definida pelo seu nome (Identificador do Objeto) e seu tipo (Classe do Objeto), que inclui as operações possíveis sobre a instância e suas características (atributos, eventos, ações, objetos contidos).

2.2.5 Banco de Informações de Gerência

O Banco de Informações de Gerência *(Management Information Base - MIB)* é um banco de dados formado pela coleção de todos Objetos Gerenciados

contidos no Domínio Gerenciado. A MIB é distribuída sobre todos os Sistemas Gerentes e Gerenciados da rede.

2.2.6 Entidade de Camada

Uma Entidade de Camada - N *(Layer Entity - LE)* é responsável pela monitoração e controle das comunicações relativas à camada N, utilizando-se para isso dos protocolos da camada N.

2.2.7 Entidade de Gerência de Camada

Uma Entidade de Gerência de Camada N *(Layer Management Entity - LME)* é responsável pelas operações de gerência restritas à camada N, utilizando-se para isso dos protocolos de gerência da camada N.

2.2.8 Entidade de Aplicação

Uma Entidade de Aplicação *(Application Entity - AE)* é um Processo de Aplicação que presta serviços ao usuário (humano ou não) usando a Entidade de Camada de nível mais alto para isso.

2.2.9 Entidade de Aplicação para Gerência de Sistemas

Uma Entidade de Aplicação para Gerência de Sistemas *(Systems Management Application Entity - SMAE)* utiliza-se de um protocolo para gerência de sistemas que possibilita o acesso aos dados de gerência de todas camadas em um nodo. Com tal poder, a SMAE fornece um eficiente serviço de apoio aos Processos

de Aplicação para Gerência de Sistemas (SMAP - exposto no item seguinte), que devem trocar diretivas de gerência. O interfaceamento entre o SMAE e o SMAP acontece através de um Elemento de serviço de Aplicação para Gerência de Sistemas *(SMASE - Systems Management Application Service Element)*, que define um conjunto de primitivas para cada área funcional de gerência. Estas primitivas podem ser mapeadas diretamente em primitivas do Elemento de Serviço Comum para Informações de Gerência *(CMISE - Common Management Information Service Element)*, que é relacionado com o Protocolo Comum de Informações de Gerência *(CMIP - Common Management Information Protocol)*.

O CMISE é baseado nos Elementos de serviço de Aplicação ACSE *Association Control Service Element)* para controle das associações do CMIS *(Common Management Information Service)*, e no ROSE *(Remote Operations Service Elements)* para a execução de operações remotas. A figura 2.1 mostra o relacionamento entre os componentes do SMAE.

2.2.10 Processo de Aplicação para Gerência de Sistemas

O Processo de Aplicação para Gerência de Sistemas *(SMAP - Systems Management Application Process)* tem acesso, através da MIB, aos dados de gerência de todas camadas do nodo nela contidos. Um SMAP assume o papel de um Processo Gerente ou Processo Agente, ou ambos. Qualquer SMAP pode comunicar-se com um SMAP remoto para trocar informações de gerência, utilizando-se para isso do SMAE.

A estrutura até então descrita pode ser visualizada na figura 2.2.

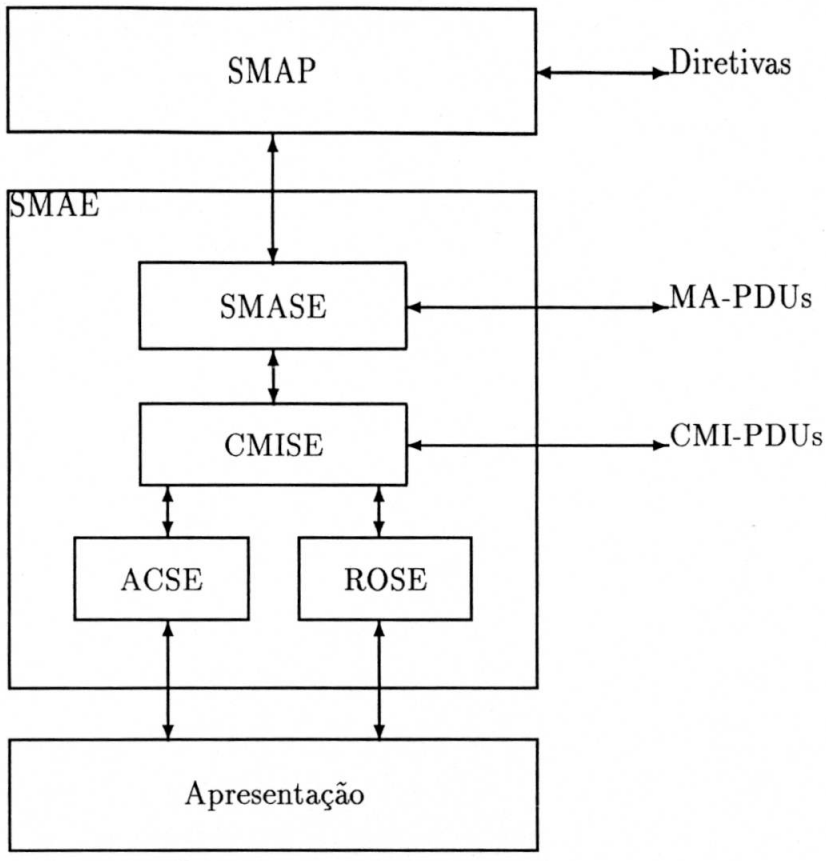

Figura 2.1: A camada de Aplicação SMAE

2.2.11 Áreas Funcionais de Gerência

Na padronização OSI, as diferentes tarefas de gerência em uma rede são classificadas em cinco grupos nomeados Áreas Funcionais Específicas de Gerência *(SMFAs - Specific Management Functional Areas)*, são elas:

1. Gerência de Falhas *(Fault Management)*: envolve um conjunto de facilidades que permitem a detecção, isolamento e correção de uma operação anormal da rede. Para isso, existem vários testes definidos, tais como testes de conectividade, testes de saturação de conexões, saturação de dados, entre outros.

2. Gerência de Configuração *(Configuration Management)*: envolve tanto a gerência lógica quanto física. A configuração lógica inclui nomeação, parâmetros de

interfaces, parâmetros de protocolos, etc. A configuração física trata da localização física dos nodos, configuração de hardware, etc. Para que isso ocorra, o acesso aos Objetos Gerenciados é provido, de modo que os atributos destes podem ser lidos e setados com propositos de configuração.

3. Gerência de Desempenho *(Performance Management)*: realiza a avaliação, controle e predição do desempenho da rede; esta avaliação envolve o acompanhamento das diversas camadas do Modelo OSI. Para isso, existem definidas no âmbito desta função de gerência uma série de facilidades para obtenção de dados estatísticos dos Objetos Gerenciados de cada camada, possibilitando a derivação do desempenho da rede, pontos críticos, futuros gargalos, e etc.

4. Gerência de Segurança *(Security Management)*: envolve o controle de acesso aos recursos e sub-redes do Domínio de Gerência, protegendo-os contra uso indevido. Isto pode ser realizado através de autenticações, codificação de dados (*encryption*), controle de acesso, e etc.

5. Gerência de Contabilizações *(Accounting Management)*: envolve dar o custo de cada comunicação estabelecida. As facilidades para este fim providas permitem ao gerente determinar a distribuição da utilização dos recursos por usuário, alocando custos. Isto é necessário para uma avaliação financeira da rentabilidade da rede.

2.2.12 Áreas Funcionais de Gerência

Na padronização OSI, as diferentes tarefas de gerência em uma rede são classificadas em cinco grupos nomeados Áreas Funcionais Específicas de Gerência *(SMFAs - Specific Management Functional Areas)*, são elas:

1. Gerência de Falhas *(Fault Management)*: envolve um conjunto de facilidades que permitem a detecção, isolamento e correção de uma operação anormal da

rede. Para isso, existem vários testes definidos, tais como testes de conectividade, testes de saturação de conexões, saturação de dados, entre outros.

2. Gerência de Configuração *(Configuration Management)*: envolve tanto a gerência lógica quanto física. A configuração lógica inclui nomeação, parâmetros de interfaces, parâmetros de protocolos, etc. A configuração física trata da localização física dos nodos, configuração de hardware, etc. Para que isso ocorra, o acesso aos Objetos Gerenciados é provido, de modo que os atributos destes podem ser lidos e setados com propositos de configuração.

3. Gerência de Desempenho *(Performance Management)*: realiza a avaliação, controle e predição do desempenho da rede; esta avaliação envolve o acompanhamento das diversas camadas do Modelo OSI. Para isso, existem definidas no âmbito desta função de gerência uma série de facilidades para obtenção de dados estatísticos dos Objetos Gerenciados de cada camada, possibilitando a derivação do desempenho da rede, pontos críticos, futuros gargalos, e etc.

4. Gerência de Segurança *(Security Management)*: envolve o controle de acesso aos recursos e sub-redes do Domínio de Gerência, protegendo-os contra uso indevido. Isto pode ser realizado através de autenticações, codificação de dados (*encryption*), controle de acesso, e etc.

5. Gerência de Contabilizações *(Accounting Management)*: envolve dar o custo de cada comunicação estabelecida. As facilidades para este fim providas permitem ao gerente determinar a distribuição da utilização dos recursos por usuário, alocando custos. Isto é necessário para uma avaliação financeira da rentabilidade da rede.

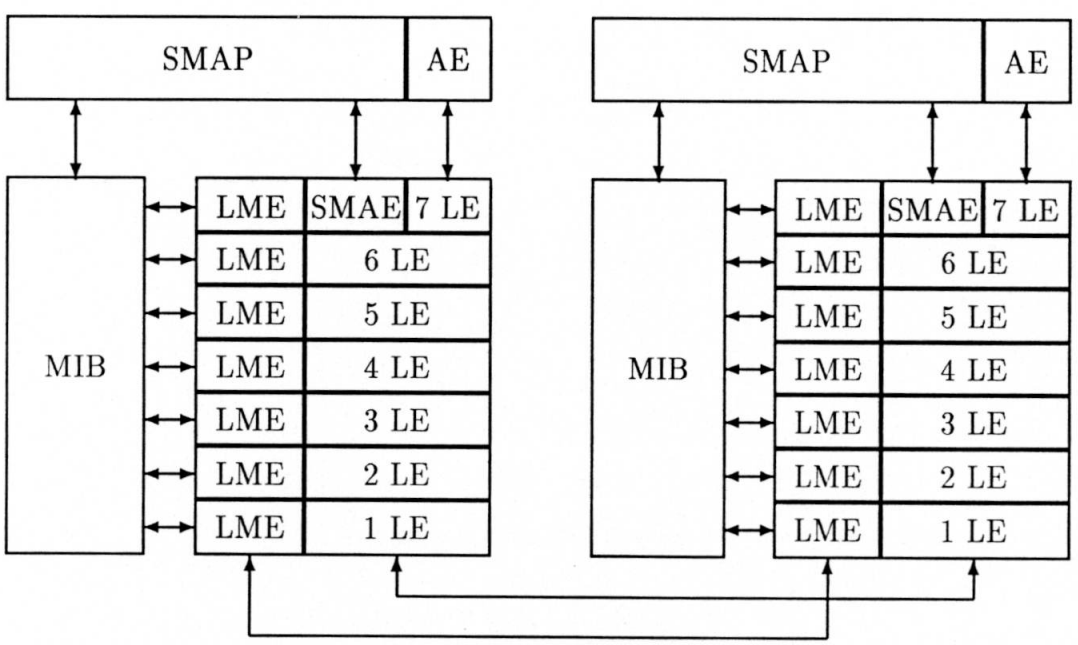

Figura 2.2: Ambiente de Gerência OSI

3 ESTADO DA ARTE EM GERÊNCIA DE REDES

3.1 Introdução

Há algum tempo, a problemática central no tema "redes de computadores" era a interoperabilidade entre as diversas máquinas. Esforços foram investidos para a solução deste problema e diversas arquiteturas proprietárias para comunicação de dados surgiram apontando seus caminhos para tornar possível a cooperação entre máquinas. As soluções apresentadas diferiam de arquitetura para arquitetura e o consumidor que possuia um ambiente heterogêneo tinha um problema complexo de interoperabilidade ainda não resolvido .

Os protocolos da arquitetura INTERNET (TCP/IP e demais relacionados à arquitetura usada na rede) se disseminaram de forma rápida devido a seus conceitos já bem sedimentados implatados no Departamento de Defesa Americano, colocando-se como "padrões de fato". A ISO construiu o modelo OSI, desencadeando a busca pela compatibilização por parte dos vendedores. Foram construídos Gateways entre o modelo de referência e os protocolos proprietários, o modelo OSI foi implementado e formas híbridas de transição foram estruturadas [TYE90] [GRO86] [WAL89] [RC87].

Com estes esforços as deficiências de interoperabilidade foram sendo resolvidas e as redes tomaram volume e importância maior. Com isso, o problema de gerência destas redes começou a emergir, tornando-se um importante foco de atenção.

As arquiteturas proprietárias mais uma vez apresentaram suas soluções para o problema mesmo antes dos padrões estarem bem definidos para este fim. A IBM foi o primeiro fabricante a anunciar sua solução integrada para gerência de re-

des através do NetView [KAN88], em 1986. A DEC, com o Enterprise Management Achitecture (EMA) [FEH89] [SYL91], e a AT&T, com o Unified Network Management Architecture (UNMA) [CKP88] anunciaram suas arquiteturas de gerência de uma forma bem mais "orientada a OSI". Para os protocolos da arquitetura INTERNET também foram elaboradas formas de gerência de redes.

Outras organizações além das citadas, responsáveis por arquiteturas proprietárias, também dispendem esforços para alcancar o objetivo de melhor manutenção na comunicação de seus equipamentos [DSW88].

Assim sendo, o cenário atual mostra várias ênfases no que tange à gerência de redes:

- A padronização OSI está em fase adiantada, continua avançando e se mostra como forma de solução a ser seguida no futuro, seus conceitos (vide capítulo 2) já estão sendo utilizados na prática;
- A gerência de redes na arquitetura INTERNET também tem forte impulso, visto que existe uma boa base instalada de protocolos tradicionais (sem gerência) facilitando a introdução da gerência de redes. Há uma tendência de migração das camadas superiores ao TCP/IP para os padrões OSI também no contexto de gerência de redes [WAB89];
- As formas de gerência proprietárias são ou estão sendo compatibilizadas de modo a operar em um ambiente heterogêneo, para isso são buscados os conceitos padronizados.

O assunto como um todo abriga vários focos de pesquisa, discutindo e empregando idéias com o objetivo de efetivar a gerência de redes. Bons avanços estão acontecendo nestes diferentes focos.

O objetivo deste capítulo é expor o que está sendo pesquisado em gerência de redes. A exposição está centrada em três tópicos bastante importantes do assunto:

arquiteturas de gerência para redes heterogêneas (formas de agregar as informações), as tendências a nível de processos de aplicação para gerência de redes, e uma breve exposição sobre a representação interna de informações de gerência.

3.2 Gerência de Redes Heterogêneas

Os serviços prestados através de redes de computadores crescem em número e tipo, cada vez mais se faz necessário poder contar com os mais diferentes equipamentos em uma instalação. Isto leva à inevitável existência de redes heterogêneas apoiando as diferentes tarefas que podem ser realizadas.

As estruturas para garantir a operação de redes heterogêneas já são conhecidas. O gerência destas instalações é um ponto em aberto, sendo que várias propostas para a solução deste problema estão surgindo e sendo discutidas.

Para alcançar a gerência de redes heterogêneas, é proposta em [SWE90] uma estrutura (voltada a redes locais) permitindo agregação das informações de gerência em um centro. Tal estrutura é composta de três entidades principais:

- O Gerenciador de Estação é implementado em cada estação da rede e está ligado com os gerenciadores de camadas que o alimentam com informações. Este módulo deve: monitorar a comunicação da estação, gerar informações para o gerenciador de rede, reconfigurar seus recursos durante falhas ou modificações das instalações, e executar comandos do gerenciador de rede.

- O Gerenciador de Rede recebe as informações das entidades das estações, filtra estas informações e reporta ao gerenciador central as informações concernentes à comunicação inter-redes. O que pertencer ao escopo local é ali decidido gerando comandos aos gerenciadores de estação. O gerenciador de rede deve possuir conhecimento sobre as áreas funcionais que se deseja gerenciar.

- O Gerenciador Central deve controlar os vários aspectos complementares das redes interconectadas. Este módulo possui uma visão global permanente dos recursos e atividades. São recebidas informações dos gerenciadores de rede, faz-se um trabalho de controle com estas informações e são gerados comandos aos gerenciadores de rede.

No artigo referido, considera-se comunicação baseada em TCP/IP [DAR81b, DAR81a], estes protocolos têm se destacado como os mais utilizados em redes locais (e também redes metropolitanas). Como protocolos para apoiar a gerência são utilizados subconjuntos do SNMP [CFS90] e CMOT [WAB89].

Em [CGP90] é proposto um método, chamado IDEA *(Intelligence, Diagnostic, Expertise, Administration)*, que provê uma arquitetura e ambiente de software para gerência de redes heterogêneas (multi-fabricantes, multi-estruturas, e multi-dados - ou seja: voz, dados, imagem).

O método tem o objetivo de integrar os diferentes ambientes de gerência de redes dos fabricantes, sem que sua modificação seja necessária. Para isso, as aplicações de gerência de cada fabricante possuem um MAP *(Manufacturer Access Point)* e o IDEA traduz o conjunto de MAP's em um único UAP *(User Access Point)*. O usuário neste caso é o operador ou gerente, que poderá, então, se valer de um processo de gerência agindo sobre a rede heterogênea.

O IDEA também oferece a possibilidade de pontos de gerência distribuídos (caracterizando os domínios de gerência na terminologia OSI) através do protocolo EAP *(Exchange Administration Protocol)*. Desta forma existirão vários UAPs agregando informação de direrentes MAPs e trocando informações através o EAP.

Em [WAS89] é apresentada uma arquitetura semelhante ao IDEA. Propõe-se que a gerência global da rede se dê em um único nodo físico. Este nodo roda todas as "pilhas" de protocolos que a rede utiliza, comunicando-se com todas as subredes lógicas. As mensagens de gerência de cada "pilha" são traduzidas para uma forma

comum e submetidas ao processo de gerência global. Esta arquitetura evita que vários equipamentos da rede estejam envolvidos em tarefas de gerência de subredes, e que as mensagens de gerência tenham que ser traduzidas por equipamentos intermediários.

Esta abordagem permite realizar a gerência integrada de redes heterogêneas, proporcionando, ainda, uma plataforma de estudos de problemas de interação entre as "pilhas" de protocolos.

Em [PAU90] propõe-se que a gerência de redes locais interconectadas (especialmente gerência de configuração e desempenho) se dê através de agentes nas subredes e agentes nos dispositivos de interconexão das subredes. Os agentes das subredes trabalham como monitores somente vendo a carga local, eles não administram nenhum dispositivo mas coletam dados de gerência importantes. Os agentes dos dispositivos de interconexão devem, por sua vez, realizar funções como: aprendizado de configuração, reconhecimento de reconfiguração, construção de tabelas de acesso, filtragem de protocolos, autorizações, e estatísticas.

Os agentes acima mencionados reportam suas informações a um processo gerente global do domínio de gerência em questão (tal como na arquitetura de gerência OSI). Nota-se que especial atenção é dada aos dispositivos de interconexão das subredes, isto se deve aos propósitos de gerência fixados (configuração e desempenho).

Esta forma de gerência está em implantação na rede LINK (Local Informatics Net Karlsruhe), composta por 180 *workstations*, 120 PC's, 6 *mainframes*, 80 terminais em concentradores ligados nos segmentos de rede, além de pontes, repetidores e *gateways*, distribuídos nos quase 20 segmentos de rede envolvendo campus e cidade.

Uma forma de tradução entre mensagens de gerência globais e mensagens dependentes de equipamento se dá através de *proxy agents* (ou agentes procurado-

res) [DUL91]. Um agente procurador é um elemento da rede que realiza a tarefa de "representar" um dispositivo remoto cuja forma de gerência é heterogênea, para o processo gerente a heterogeneidade não existe. A estrutura destes elementos geralmente é a mesma para diferentes dispositivos a serem representados. Basicamente, eles devem realizar tradução de protocolos, manter uma MIB relativa ao dispositivo representado, suportar um *driver* para o dispositivo e realizar *polling* sobre o dispositivo, buscando seu estado.

Enfim, a gerência de redes heterogêneas está sendo alcançada por diversos caminhos, conforme ilustrado. Seja através do uso de uma sub-estrutura de comunicação comum nos diferentes equipamentos (frequentemente TCP/IP), seja pela compatibilização de sistemas de gerência diversos oferecidos pelos fabricantes ou pela monitoração das redes e seus equipamentos de interconexão, as diferentes arquiteturas de gerência frequentemente possuem características em comum, tais como:

- Divisão da rede a ser gerenciada em domínios, cada qual com seu processo gerente e vários agentes;
- Agregação das informações de forma hierárquica;
- Utilização de áreas funcionais de gerência;
- Representação interna de objetos gerenciados.

As características comuns identificam claramente conceitos definidos na estrutura OSI de gerência de redes, evidenciando a validação prática do modelo em definição e a tendência a sua utilização. As arquiteturas de sistemas de gerência já contam com normas e boas experiências realizadas, sendo que novas formas surgem e são submetidas a testes práticos. Estas arquiteturas tem o objetivo comum de apoiar o processo de aplicação para gerência de rede, a ser abordado a seguir.

3.3 Processos de Aplicação para Gerência de Redes

Os processos de gerência são, em suma, os detentores do "conhecimento" de gerencia da rede em questão. Estes processos não possuem forma e nem limite de atuação especificados. Suas funções podem ser desde simples filtros de eventos auxiliando o operador da rede até complexos sistemas capazes de aprender por observação e agir sobre a instalação sem intervenção humana.

À medida que se queira tornar estes processos mais capazes de gerenciar a rede, existe a transferência da atividade humana do gerente para o processo de aplicação. A atividade de gerência é uma atividade especializada, envolvendo conhecimento e experiência no ramo gerenciado. Para que o processo de aplicação suporte tal atividade se faz necessário o uso de ferramentas de inteligência artificial, mais especificamente sistemas especialistas, para a representação do conhecimento do gerente. Atualmente, muitos dos processos de aplicação para gerência de redes fazem uso de sistemas especialistas, portando-se de forma satisfatoria.

Grande parte dos processos gerentes já em fase de teste se destinam ao tratamento de falhas. Em [KBW89] é exposta uma análise dos sistemas correntes encontrados na literatura voltados a este tipo de gerência. A maioria dos processos de gerência são sistemas especialistas baseados em regras heurísticas. Alguns deles são:

- O REACT *(Real-time Expert Analysis and Control)* é um protótipo para manipular falhas em telecomunicações, ele monitora o comportamento dos componentes e gera alarmes que são submetidos à inferência com encadeamento regressivo em um corpo de regras.

- O TOPAS-ES é um sistema em tempo-real, distribuído, para manutenção de redes comutadas, usa encadeamento progressivo em um conjunto de regras.

- O StarKeeper foi desenvolvido para simplificar a tarefa de isolamento de falhas. O sistema especialista, baseado em heurísticas com encadeamento progressivo e ajuste condicional de probabilidades, identifica os componentes falhos e formula um plano de correção.

- ACE *(Automated Cable Expertise)* e *PROPHET (Pro-active Rehabilitation of Outside Plant Using Heuristic Expert Techniques)* são sistemas automáticos (não interativos) para manutenção e reparo de redes. O usuário pode fazer perguntas sobre o estado do sistema que roda em *background.* Usam regras e heurísticas; o ACE usa encadeamento progressivo.

- O FIESTA *(Fault Isolation System for TDRSS Applications)* combina os dados da situação de erro reportada com seu conhecimento pré-armazenado (regras, proposições, e *frames*) derivando conclusões sobre o sistema. Usa encadeamento progressivo e regressivo durante a inferência.

É natural que o tratamento de falhas receba uma dose de atenção inicial maior pois é objetivo comum manter a rede operante. As demais áreas funcionais também estão sendo abordadas e os resultados já são visíveis. Em [PAU90] são tratados os problemas de gerência de desempenho e de configuração; em [CST89] é detalhada uma estrutura para gerência de desempenho e seus aspectos comuns com outras áreas funcionais de gerência; [MAL89] trata de gerência de segurança seguindo as especificações do Departamento de Defesa Americano.

Existem, também, experiências de processos de gerência dedicados a ambientes mais complexos. Em [JOH90] é descrito o OMS *(Operations Management System)* que controla o complexo ambiente de uma estação espacial, envolvendo: a rede de equipamentos, sistema de força, controle ambiental, controle térmico e navegação, entre outros. O OMS pode ser visto como uma extensão da gerência de redes para a gerência do ambiente, vários conceitos OSI-NM são utilizados: áreas funcionais de gerência, uso de CMIP/CMIS e estrutura de objetos gerenciados. Em alguns pontos foram identificadas deficiências do modelo de gerência OSI para o

sistema em questão , tais deficiências foram sanadas com o uso de ferramentas adicionais (Ex.: a necessidade de tratamento de mensagens de gerência em tempo-real levou à adoção adicional do MMS-Manufacturing Message Specification).

O YES/MVS *(Yorktown Expert System/MVS)* da IBM [MIL86] também é um exemplo de sistema especialista para controle de um ambiente complexo, dedicando-se ao domínio da operação de um centro computacional.

Além dos acima citados, vários outros sistemas de gerência para as diferentes áreas funcionais estão sendo desenvolvidos. Algumas vezes são encontrados sistemas que tratam híbridos de mais de uma área funcional, isto se deve à necessidade de compartilhamento de dados e dependências funcionais [KOB89] (por exemplo: tratamento de desempenho necessita conhecimento estrutural sobre a rede, estes dados estão relacionados com a gerência de configuração).

Existe uma forte tendência, ratificada por experiências positivas, em se utilizar sistemas especialistas para a modelagem dos processos de gerência de redes. Estes processos geralmente são voltados às áreas funcionais de gerência como descrito na arquitetura OSI da ISO (Falhas, Configuração, Contabilização, Desempenho e Segurança), fazendo com que o sistema exerça gerência sobre uma ou mais áreas funcionais.

O tratamento destes problemas é um tanto complexo, mesmo com o uso de Inteligência Artificial (I.A.) quase que invariavelmente os processos de gerência têm que ser acompanhados por uma pessoa especializada. As pesquisas em I.A. podem ajudar a capacitar os processos gerentes para que estes desenvolvam atividades complexas.

Uma técnica de I.A. que tem sido apontada como impulsionadora de novas formas de gerência é a I.A. Distribuída [LIE88]. Segundo [GOY91], arquiteturas de sistemas especialistas que suprem as necessidades de várias áreas do conhecimento não são capazes de resolver o problema de gerência de redes devido à característica

de distribuição no domínio do problema. Em [YGY91] é estruturada uma forma de distribuir funções de gerência entre os gerentes e agentes, enfatizando que a gerência centralizado não é capaz de dominar o problema.

Juntamente com I.A. Distribuída, algumas outras técnicas em pesquisa atualmente como formas de aprendizagem simbólica automática, e modelos para representação de conhecimento, entre outras, poderão ser ferramentas para uma próxima geração de processos de gerência.

Os mais diferentes dispositivos físicos e lógicos devem ser gerenciados, envolvendo um processo de troca de informações entre processo gerente e dispositivos. Para isso existe o mapeamento destas entidades físicas e lógicas para uma representação interna ao sistema de gerência. À representação interna de uma entidade chamamos Objeto Gerenciado seguindo a terminologia OSI (capítulo 2), no item a seguir serão abordadas algumas tendências para a representação destas informações de gerência.

3.4 Representação Interna de Informações de Gerência

Na arquitetura OSI, o conjunto de todos os Objetos Gerenciados no mesmo domínio fazem parte de uma MIB (Management Information Base). A MIB suporta, portanto, uma representação interna dos recursos gerenciados ao longo do domínio referido. Quando uma aplicação remota (ex.: processo gerente) necessita mudar o comportamento do sistema gerenciado, atua sobre a MIB usando um protocolo de operações remotas. As modificações comandadas pela aplicação remota são efetuadas sobre os Objetos Gerenciados que, acoplados a entidades físicas e lógicas da rede, resultam em mudanças na operação do sistema. Este processo está bem exemplificado em [PAT89].

Para suportar a funcionalidade desejada, aponta-se como paradigma de representação para as informações de gerência (no ambiente OSI) a orientação a objetos [KOB89]. Tal paradigma envolve conceitos como herança, encapsulação e polimorfismo, e descreve os princípios para definição de classes, atributos e esquemas de nomeação. Estes conceitos não serão aqui abordados, alguns detalhes podem ser buscados em [BAN88, TLX90, TAK89].

A natureza distribuída da MIB resulta em problemas para a implementação da orientação a objetos. Considere-se a migração de um objeto de um nodo para outro ou sua distribuição pelos nodos da rede. Existem estudos para o tratamento destes problemas, porém não estão completos pois assumem que a rede é homogênea ou que existe capacidade de tradução local. Ainda para o mesmo exemplo, seria necessário que cada nodo possuísse conhecimento sobre todas as classes do domínio de gerência, isso levaria ao cadastramento de cada classe introduzida no sistema em cada nodo do domínio gerenciado.

Em [WEL90] é descrita uma arquitetura orientada a objetos para desenvolver um sistema de gerência capaz de se adaptar a mudanças no ambiente gerenciado, abordando os problemas acima mencionados.

Em [CHL90] considera-se a orientação a objetos certamente necessária mas não suficiente para uma boa modelagem da rede a ser gerenciada. São propostas estruturas de mais alto nível sobre a orientação a objetos, melhor dispondo os objetos gerenciados, estas estruturas são as "views", ou "perspectivas". Os objetos são agrupados em "views" conforme seu significado, estas "views" devem servir a propósitos genéricos e não a apenas uma ou duas áreas funcionais de gerência. Uma forma de estrututuração possível seria agrupar os objetos em "views" físicas e lógicas, onde a "view" física provê a organização ou estrutura da rede e a "view" lógica proporciona as funções e serviços prestados. O relacionamento entre objetos de diferentes "views" é importante, realizando o mapeamento entre as mesmas. Isto enfatiza a relação entre os objetos e entre as "views" envolvidas, dando valor semântico maior à estrutura armazenada.

A adoção de orientação a objetos tem sido um ponto discutido na comunidade pesquisadora. Alguns aceitam tal paradigma como necessário e suficiente para seu propósito em gerência de redes [BML91]. Outros tem adotado o modelo, auxiliando-se de ferramentas de mais alto nível [CHL90] ou distribuindo o modelo [RAE91]. Há ainda pesquisadores que criticam o uso deste modelo, afirmando ser "pobre" para a gerência de uma rede real [ROS91].

De qualquer forma, a orientação a objetos tem sido adotada frequentemente para a representação das informações de gerência. Estas experiências tem resultado em críticas que deverão levar a reformulações do modelo atual.

3.5 Considerações

Além dos acima referidos, outros experimentos relacionados com gerência de redes estão sendo realizados, por exemplo:

- estudos para o desenvolvimento de interfaces "user-friendly" voltadas aos requisitos de gerência de redes [LIK90] [CRA89];
- gerência de redes digitais de serviços integrados [OWE89] [STU89];
- integração ao sistema de gerência de dispositivos sem primitivas de gerência [PAU90] [DUL91];
- construção de ambientes para desenvolvimento de sistemas de gerência OSI [NKI91].

Alguns tópicos, porém, necessitam de maiores estudos, por exemplo:

- relacionamento entre as áreas funcionais específicas de gerência;
- tratamento da MIB em redes heterogêneas;

- estruturação do processo gerente.

As pesquisas em gerência de redes estão se avolumando, os problemas complexos aos poucos estão sendo dominados com o uso de técnicas modernas de tratamento com a informação. Em um futuro próximo as conclusões destas pesquisas indicarão as estruturas corretas a serem adotadas, de forma a proporcionar serviços mais ágeis e econômicos ao usuário final.

4 ESPECIFICAÇÃO FORMAL DO SISTEMA

4.1 Os Métodos Utilizados

As metodologias modernas de desenvolvimento de *software* propiciam meios corretos e objetivos para descrição e inter-relacionamento das várias entidades envolvidas no processo de criação de *software*. O uso destas metodologias neste trabalho visa proporcionar correção na implementação e eliminar a ambigüidade no entendimento, de parte do leitor, acarretada pela linguagem natural.

Observando-se algumas características do sistema a ser implementado, nota-se que ele é composto de alguns blocos operando em paralelo. Desta forma, optou-se pelo uso de um método de especificação suportando paralelismo para representar a interação entre estes blocos concorrentes, e um método de especificação sem paralelismo para denotar o comportamento seqüencial de cada bloco em separado.

Para especificar o funcionamento do sistema a nível mais alto foi utilizado o método operacional CSP (Communicating Sequential Processes), proposto por Hoare em 1978 [HOA87] e reformulado também por Hoare em 1985 [HOA85]. Em CSP os sistemas são descritos em termos de processos, explicitando-se a maneira como estes devem se comportar. A observação do comportamento dos processos pode ser feita por *traces* que contém seqüências de eventos que acontecem no sistema. O método CSP pode ser encontrado de forma sucinta em [RIB90] ou de forma mais detalhada em [HOA85]. Este método foi escolhido para a representação da interação que acontece entre as entidades paralelas envolvidas no funcionamento do sistema: *Rede, Estação e Usuário*, sendo que o processo *Estação* é composto por dois subprocessos operando em paralelo, os processos *Driver e MAG (Módulo de Apoio à Gerência).*

Para especificar o funcionamento das partes seqüenciais do sistema, foi utilizado o método denotacional VDM (Vienna Development Method), desenvolvido no centro de pesquisa da IBM em Viena durante os anos 70. Em VDM, constrói-se um modelo explícito do sistema utilizando-se elementos matemáticos (conjuntos e relações) e lógica de primeira ordem. Sobre este modelo são construídas, utilizando-se lógica, as operações a serem realizadas no sistema. O método VDM pode ser encontrado bem sucintamente em [RIB90], de forma mais completa em [MAR90], ou por seus autores em [BJØ78] e [JON86].

Descrições de software utilizando diferentes paradigmas de especificação são frequentes, geralmente trazendo vantagens pois são utilizadas as características principais de cada método. Os motivos para a adoção desta forma de especificação foram:

- A característica algorítmica de VDM, tornando a especificação acessível a um número maior de leitores,
- Utilizar o poder de tratamento de conjuntos proporcionado por VDM na especificação do sistema baseado em conhecimento,
- Evitar a complexidade de métodos operacionais e algébricos,
- Existência de poucos blocos paralelos trocando dados em pontos bem definidos, facilitando a forma de especificação tal como adotada.

A especificação em CSP define, assim, os blocos concorrentes e a especificação em VDM define o comportamento em separado de cada um destes blocos de forma seqüencial. Os processos de CSP são modelados em VDM como funções e os eventos de CSP são retornos de funções, *buffers* ou parâmetros de mesmo nome em VDM.

Algumas funções em VDM apresentam somente o cabeçalho pois seu comportamento é dependente do ambiente de funcionamento do sistema. Desta forma

os detalhes de cada ambiente são evitados e parte-se de uma interface comum que é considerada na fase de especificação.

Outras funções em VDM, se analizadas separadamente, não terão significado pois serão somente produtoras ou consumidoras de dados, deve-se atentar para o contexto maior onde existe paralelismo e cada função tem seu papel bem definido. Quando necessário, as equivalências entre CSP e VDM estão comentadas junto das especificações.

4.2 Interação dos Módulos

Como explicitado anteriormente, a interação entre os módulos está especificada em CSP. Na figura 4.1 temos esta interação.

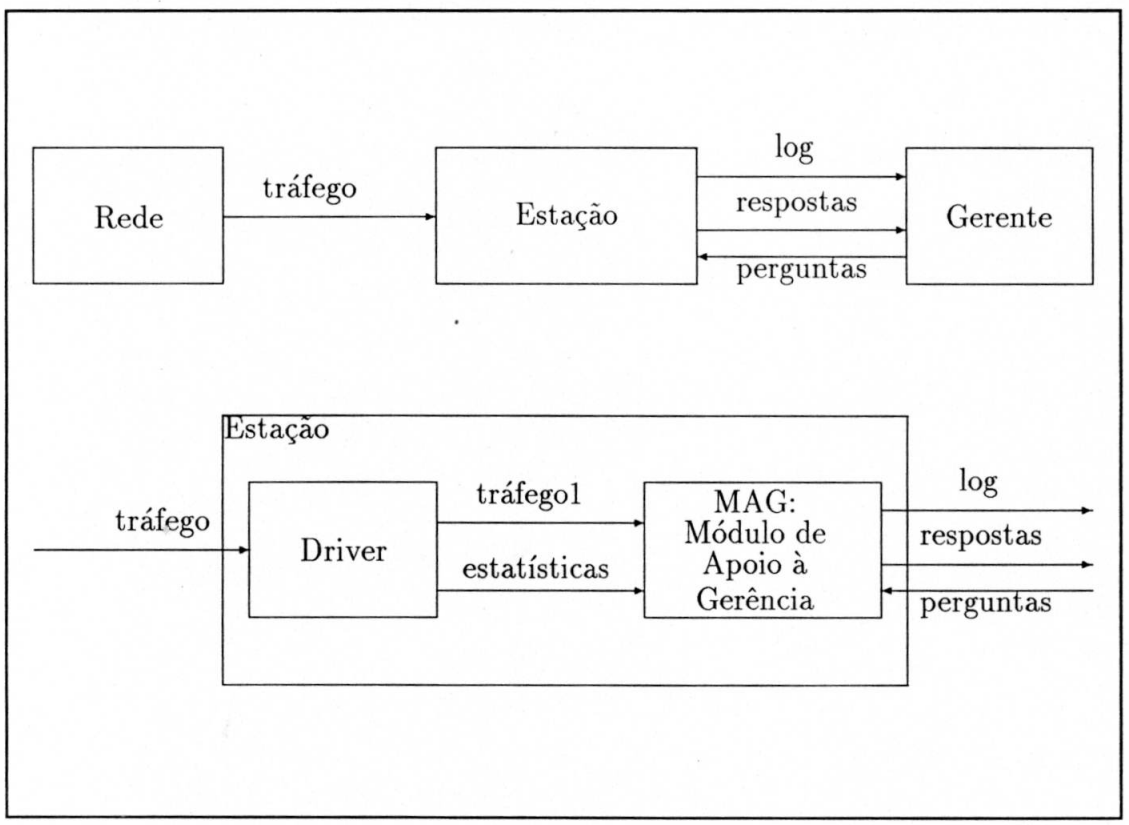

Figura 4.1: Entidades Concorrentes do Sistema

$$
\begin{array}{lrcl}
(1) & Sistema & = & (Rede \parallel Estacão \parallel Gerente) \\
(2) & Estacão & = & (Driver \parallel MAG) \\
(3) & Rede & = & tráfego \rightarrow Rede \\
(4) & Driver & = & tráfego \rightarrow tráfego1 \rightarrow (Driver \mid \\
 & & & estatísticas \rightarrow Driver) \\
(5) & MAG & = & tráfego1 \rightarrow (MAG \mid \\
 & & & estatísticas \rightarrow MAG \mid \\
 & & & log \rightarrow (MAG \mid \\
 & & & estatísticas \rightarrow MAG) \mid \\
 & & & perguntas \rightarrow respostas \rightarrow MAG) \\
(6) & Gerente & = & (log \rightarrow Gerente \mid \\
 & & & perguntas \rightarrow respostas \rightarrow Gerente) \\
(7) & Restricões & = & R1 + R2 + R3 + R4 \\
(8) & R1 & = & (tr \downarrow tráfego1) \leq (tr \downarrow tráfego) \\
(9) & R2 & = & (tr \downarrow respostas) = (tr \downarrow perguntas) \\
(10) & R3 & = & (tr \downarrow log) \leq (tr \downarrow tráfego1) \\
(11) & R4 & = & (tr \downarrow estatísticas) < (tr \downarrow tráfego1)
\end{array}
$$

A especificação em CSP acima denota o seguinte:

- Em (1) define-se o sistema como a composição de três processos operando em paralelo: *Rede, Estação e Usuário.* Apesar do *software* implementado estar contido no processo Estação, os processos *Rede e Usuário* foram também especificados para que fiquem definidas as possíveis interações entre as entidades envolvidas. Em CSP, quando o comportamento do ambiente em que o sistema (a ser implementado) se insere é bem definido, é normal modelar parte

do ambiente como processo(s) de forma que a interação das entidades fique formalizada.

- Em (2) define-se o processo *Estação* como o paralelismo dos processos *Driver* e *MAG*.

- Por (3) fica definido que *Rede* pode gerar *tráfego*, ou simplesmente estar inoperante.

- Em (4) fica definido que o *Driver* recebe informações da *Rede (tráfego)* e passa este *tráfego* para o processo *MAG*, chama-se *tráfego1* pois adiciona-se um *time-stamp* aos *frames* que chegam de *Rede*. O *Driver* também tem a função de suprir o processo *MAG* com *estatísticas* de utilização do nível físico.

- Em (5) define-se que o processo *MAG* é alimentado pelo *Driver* com *tráfego1 (time-stamp) e estatísticas*, processa estas informações e adiciona os problemas detectados (através da análise do *tráfego* de rede) e possíveis soluções em um *log* de problemas/soluções disponível ao *Gerente*. Este processo responde, ainda, a perguntas do *Gerente*.

- *Estatistícas* são informações sobre a utilização do dispositivo físico. Os tipos de dados disponíveis variam de dispositivo para dispositivo, dependendo do ambiente em que o sistema se encontra. Assim, as informações que fazem parte de *estatísticas* serão abordadas no capítulo relativo à implementação do sistema.

- Por (6) fica definido que o *Gerente* recebe periodicamente um *log* com os problemas detectados pelo sistema e as possíveis soluções, e elabora consultas ao sistema, que retorna as respostas.

- Em (7) foram definidas quatro restrições de consistência para o sistema, que podem ser observadas sobre os *traces*.

- A restrição R1 especifica que cada *frame* que chega da rede deverá ser repassado para o *MAG* pelo *Driver*. Situações de perda de pacotes pelo *Driver* podem acontecer.

- A restrição R2 define que a cada *pergunta* feita pelo *Gerente* deverá existir uma *resposta.*

- Pela restrição R3 define-se que não podem ser detectados mais problemas pelo sistema do que o número de *frames* recuperados.

- Na restrição R4 fica especificado que o processo *Driver* não pode gerar mais *estatísticas* do que *frames* para o processo *MAG.*

Os processos *Rede e Gerente* ficam, assim, bem definidos. O processo *Estação* (onde está o software construído) é composto pelo paralelismo de dois sub-processos: *Driver e MAG.* Estes dois processos estão especificados de forma separada em VDM, a seguir.

4.3 O Módulo Driver de Rede

O *Driver* de Rede é composto de um conjunto de funções que denotam o comportamento do processo *Driver* em CSP. A função *Driver* em VDM é a principal função deste Módulo. Os *buffers* aqui mencionados (*BUF_TRAFEGO1 e BUF_ESTATISTICAS*) equivalem aos eventos de saída do processo *Driver* em CSP (*trafego1 e estatísticas*) e o evento de entrada do processo (*tráfego*) equivale, em VDM, ao retorno da função *Pega_Trafego_da_Rede().*

4.3.1 Domínios Semânticos

Abaixo temos os domínios semânticos utilizados pelas funções do Módulo Driver de Rede:

$$
\begin{aligned}
&(12) \quad BUFFERS &&:: BUF\_TRAFEGO1 \times BUF\_ESTATISTICAS \\
&(13) \quad BUF\_TRAFEGO1 &&= BYTE^* \\
&(14) \quad BUF\_ESTATISTICAS &&= BYTE^* \\
&(15) \quad I &&= \mathbf{N}
\end{aligned}
$$

- Por (12) temos a estrutura entre o processo *Driver* e o processo *MAG* composta por dois *buffers*.
- Cada *buffer* é uma seqüência de *bytes*.

4.3.2 Funções

As funções do Módulo Driver de Rede são:

$$
\begin{aligned}
&(16) \quad Driver &&: BUFFERS \times \mathbf{N} \times \mathbf{N} \rightarrow BUFFERS \\
&(17) \quad Pega\_Trafego\_da\_Rede &&: \rightarrow BYTE^* \\
&(18) \quad Pega\_Estatisticas\_da\_Rede &&: \rightarrow BYTE^* \\
&(19) \quad Pega\_Instante\_Atual &&: \rightarrow I
\end{aligned}
$$

A seguir, serão detalhadas as funções.

```
16. Driver : BUFFERS × N × N → BUFFERS

16. Driver(mk-BUFFERS(trafego1, estatisticas),
    cont_traf, const_cont_traf) ≙
    let t = Pega_Trafego_da_Rede() in
    if t =< > / * vazio, naohatrafego * /
    then Driver(mk-BUFFERS(trafego1, estatisticas),
         cont_traf, const_cont_traf)
    else let i = Pega_Instante_Atual() in
         let trafego1' = conc trafego1, i, t in
         if (cont_traf mod const_cont_traf = 0)
         then let e = Pega_Estatisticas_da_Rede() in
              Driver(mk-BUFFERS(trafego1', conc estatisticas, i, e),
              1, const_cont_traf)
         else Driver(mk-BUFFERS(trafego1', estatisticas),
              cont_traf + 1, const_cont_traf)
```

Tráfego1 e *estatísticas* são *buffers* alimentados por esta função, que trabalha como produtora de dados (por isso fica em *loop*). Estes *buffers* servem de entrada para o *Módulo de Apoio à Gerência*, que consome os dados.

A função *Pega_Trafego_da_Rede()* retorna o *tráfego* da rede, se o *tráfego* for vazio nada é feito, caso contrário, este *tráfego* é colocado no *buffer trafego1* juntamente com o *instante* em que os dados chegaram, usando a função *Pega_Instante_Atual()*. É mantido um contador de pacotes de tráfego que são passados ao *buffer* de tráfego e quando este contador atinge múltiplos de uma constante *const_cont_trafego* é gerado um pacote de *estatísticas* no *buffer estatisticas*, usando

a função *Pega_Estatisticas_da_Rede()* juntamente com o *instante* em que o pacote é gerado.

Desta forma a função *Driver* (16) alimenta os *buffers* com tráfego e estatísticas.

17. $Pega\_Trafego\_da\_Rede : \rightarrow BYTE^*$

17. $Pega\_Trafego\_da\_Rede() \triangleq$

    /* *retorna todos pacotes em trafego na rede* */

    /* *dependente do equipamento e tipo de rede* */

18. $Pega\_Estatisticas\_da\_Rede : \rightarrow BYTE^*$

18. $Pega\_Estatisticas\_da\_Rede() \triangleq$

    /* *retorna estatisticas de utilizacao de nivel fisico da rede* */

    /* *dependente do equipamento e tipo de rede* */

19. $Pega\_Instante\_Atual : \rightarrow I$

19. $Pega\_Instante\_Atual() \triangleq$

    $/* \ retorna\ instante\ atual\ acessando\ o\ equipamento\ */$

    $/* \ dependente\ do\ equipamento\ */$

As funções *Pega_Trafego_da_Rede()*, *Pega_Estatisticas_da_rede()* e *Pega_Instante_Atual()* retornam informações acessando o equipamento (*hardware*) local, por isso elas não possuem parâmetros de entrada. Esta é uma situação pouco normal em especificações VDM, mas no caso presente sua adoção é cabível e modela bem a realidade.

O *Driver* de Rede é colocado em funcionamento disparando-se a função *Driver*, fornecendo como parâmetros:

- os *buffers* onde serão colocados o *tráfego* capturado e as *estatísticas*,
- 1 no segundo parâmetro (que é um contador de pacotes recebidos, iniciado em 1),
- e o número desejado de pacotes de *tráfego* para cada pacote de *estatísticas*.

Desta forma, o Módulo Driver de Rede alimenta o Módulo de Apoio à Gerência, que será exposto a seguir.

4.4 O Módulo de Apoio à Gerência

Este módulo é alimentado pelo Módulo *Driver* de Rede e realiza o processamento dos *frames* capturados da rede, detectando problemas e derivando possíveis soluções.

Os *buffers* de saída do processo *Driver* servem de entrada para este processo, em CSP estes *buffers* são caracterizados por eventos entre os dois processos. O módulo *MAG* fornece resultados e responde a perguntas do *Gerente*, em VDM isso acontece novamente através de *buffers* e em CSP a comunicação se dá por eventos que acontecem entre *MAG e Gerente.*

Para realizar isto, este módulo está dividido em cinco submódulos:

- Submódulo de Interpretação
- Submódulo de Detecção
- Submódulo de Diagnose
- Submódulo de Correção
- Submódulo de Tratamento do Usuário

Cada um destes submódulos citados será a seguir detalhado.

4.4.1 Submódulo de Interpretação

Este submódulo recebe *tráfego* e *estatísticas* do Módulo *Driver* de Rede através dos *buffers* anteriormente mencionados para este fim. O *tráfego* da rede chega separado por *frames* de forma "crua", isto é, seqüências de *bytes* da forma como são capturados da rede em funcionamento adicionadas de um *time-stamp.*

As *estatísticas*, buscadas periodicamente do dispositivo físico, são também passadas como seqüências de *bytes* através do *buffer* para este fim.

O Submódulo de Interpretação deve agir sobre os *frames* capturados separando campos e valores para cada unidade de dado de cada nível de protocolo utilizado na montagem dos *frames*. Os pacotes de estatísticas também devem ser separados em seqüências de campos e valores.

Como os *frames* que trafegam na rede podem ser formados por diversos tipos de protocolos, a interpretação das unidades de dados acontece como o reconhecimento de gramáticas. Existe, então, um conjunto de procedimentos capaz de reconhecer as unidades de dados de cada protocolo de cada nível empregado, gerando uma representação interna que alimenta o Submódulo de Detecção. Os pacotes de estatísticas são também interpretados, gerando uma representação interna.

4.4.1.1 Domínios Semânticos

Estruturas utilizadas no Submódulo de Interpretação:

$$
\begin{array}{lrcl}
(20) & GRAMATICAS & :: & GRAM\_TRAFEGO \times \\
 & & & \times GRAM\_ESTATISTICAS \\
(21) & GRAM\_TRAFEGO & = & NIVEL \xrightarrow{m} GRAM\_NIVEL \\
(22) & GRAM\_NIVEL & = & GRAMATICA \\
(23) & GRAM\_ESTATISTICAS & = & GRAMATICA
\end{array}
$$

$$
\begin{aligned}
&(24)\quad TRAFEGO &&= I \times NIVEL \times CV\text{-set}\\
&(25)\quad NIVEL &&= \{1,2,3,4,5,6,7\}\\
&(26)\quad CV &&:: CAMPO \times VALOR\\
&(27)\quad CAMPO &&= STRING\\
&(28)\quad VALOR &&= STRING
\end{aligned}
$$

A estruturação interna de *GRAMATICA* não foi especificada, deixando-se para a fase de implementação, onde pode-se usar as facilidades do ambiente de desenvolvimento. Sabe-se apenas a forma dos dados de entrada e a representação interna a ser gerada na fase de Interpretação.

4.4.1.2 Funções

Funções do Submódulo de Interpretação:

$$
\begin{aligned}
&(29)\quad Busca\_UD &&: GRAMATICAS \times BUFFERS \rightarrow TRAFEGO\\
&(30)\quad Interpreta\_Traf &&: GRAM\_TRAFEGO \times BYTE^* \rightarrow TRAFEGO\\
&(31)\quad Interpreta\_Est &&: GRAM\_ESTATISTICAS \times BYTE^* \rightarrow TRAFEGO\\
&(32)\quad Antes &&: BYTE^* \times BYTE^* \rightarrow BOOL
\end{aligned}
$$

A seguir serão detalhadas as funções deste submódulo:

29. $Busca\_UD : GRAMATICAS \times BUFFERS \rightarrow TRAFEGO$

29. $Busca\_UD(mk\text{-}GRAMATICAS(gram\_trafego, gram\_estatisticas),$
$mk\text{-}BUFFERS(buff\_trafego, buff\_estatisticas)) \triangleq$
if $Antes(buff\_trafego, buff\_estatisticas)$
then $Interpreta\_Traf(gram\_trafego, buff\_trafego)$
else $Interpreta\_Est(gram\_estatisticas, buff\_estatisticas)$

Esta função busca os dados brutos dos dois *buffers (tráfego1 e estatísticas)*, conforme os *time-stamps* dos pacotes contidos nos *buffers*. Dispara a interpretação (desencapsulamento) das unidades de dados separando-as em listas de campos e valores, e retorna o resultado, (*tráfego*), que será utilizado no Módulo de Detecção. Os *buffers* de *tráfego e estatísticas* são alimentados pelo *Driver* de Rede, que funciona em paralelo com este módulo.

32. $Antes : BYTE^* \times BYTE^* \rightarrow BOOL$

32. $Antes(buff1, buff2) \triangleq$
$/* \ retorna\ verdadeiro\ se\ o\ time\text{-}stamp\ de\ buff1\ */$
$/* \ for\ anterior\ ao\ de\ buff2\ */$

A função *Antes* analiza os *time-stamps* nos *buffers* de *tráfego e estatísticas* e retorna o valor verdadeiro se os dados do primeiro argumento antecederem os dados do segundo argumento, e falso caso contrário.

30. $Interpreta\_Traf : GRAM\_TRAFEGO \times BYTE^* \rightarrow TRAFEGO$

30. $Interpreta\_Traf(gram\_trafego, fita) \triangleq$

    $/* \ consome\ a\ fita\ de\ entrada\ utilizando\ gramatica \ */$

    $/* \ para\ trafego \ */$

31. $Interpreta\_Est : GRAM\_ESTATISTICAS \times BYTE^* \rightarrow TRAFEGO$

31. $Interpreta\_Est(gram\_estatisticas, fita) \triangleq$

    $/* \ consome\ a\ fita\ de\ entrada\ utilizando\ gramatica \ */$

    $/* \ para\ estatisticas \ */$

Estas funções interpretam uma fita de entrada consumindo-a conforme o reconhecimento da gramática parâmetro, gerando uma saída da forma de *TRAFEGO* (24) anteriormente especificado.

4.4.2 Submódulo de Detecção

O Submódulo de Detecção recebe as unidades de dados já separadas em duplas de campos e valores por nível, trabalho efetuado pelo Submódulo de Interpretação.

As unidades de dados de cada nível são, então, submetidas ao teste de uma série de predicados sobre seu conteúdo. A aplicação destes predicados gera eventos que são contabilizados em estruturas separadas. Durante esta contabilização são verificados limites de ocorrências para estes eventos em função do tempo, gerando ou não alarmes. Os alarmes são mantidos em históricos denotando o comportamento da rede, e colocados em listas de alarmes (separados por nível) juntamente com o instante em que foi disparado, de forma a alimentar o Submódulo de Diagnose.

4.4.2.1 Domínios Semânticos

Estruturas utilizadas no Submódulo de Detecção:

$$
\begin{array}{lrcl}
(33) & DETECTA & = & AREA\_GER \xrightarrow{m} DET\_AREA \\
(34) & DET\_AREA & = & NIVEL \xrightarrow{m} DET\_N \\
(35) & DET\_N & :: & PRED\_OC\_TRAF\text{-set} \times \\
 & & & \times PRED\_AUS\_TRAF\text{-set} \times \\
 & & & \times CONT \times HIST \times ESTR\_ALARM\text{-set}
\end{array}
$$

$$
\begin{array}{lrcl}
(36) & PRED\_OC\_TRAF & :: & EXPR \times EVENTO \\
(37) & PRED\_AUS\_TRAF & :: & EXPR \times EVENTO \times INTERVALO \times \\
 & & & \times MOM\_LIM \\
(38) & CONT & = & EVENTO \xrightarrow{m} (MOMENTO^* \times N\_MAX \times \\
 & & & \times T\_MAX \times INT\_AN \times \\
 & & & \times LIM\_OC \times ESTR\_ALARME) \\
(39) & HIST & = & ALARME \xrightarrow{m} (MOMENTO^* \times N\_MAX) \\
(40) & ESTR\_ALARM & :: & ALARME \times MOMENTO \\
(41) & ALARME & :: & NOME\_ALARME \times VALOR^* \mid EVENTO \\
(42) & NOME\_ALARME & = & STRING \\
(43) & EXPR & = & CV \mid NOT(CV) \mid EXPRBOOL \\
(44) & EXPRBOOL & :: & EXPR \times OPBOOL \times EXPR \\
(45) & OPBOOL & = & \{\wedge, \vee\} \\
(46) & EVENTO & :: & OC\_AUS \times NOME\_EVENTO \times VALOR^* \\
(47) & OC\_AUS & = & \{\text{Ocorrencia}, \text{Ausencia}\} \\
(48) & NOME\_EVENTO & = & STRING \times CV^* \\
(49) & AREA\_GER & = & \{\ \text{Falhas}, \text{Configuracao}, \\
 & & & \text{Contabilizacao}, \text{Performance}, \\
 & & & \text{Seguranca}\ \} \\
(50) & INTERVALO & = & \mathbf{N} \\
(51) & MOM\_LIM & = & \mathbf{N} \\
(52) & MOMENTO & = & \mathbf{N} \\
(53) & N\_MAX & = & \mathbf{N} \\
(54) & T\_MAX & = & \mathbf{N} \\
(55) & INT\_AN & = & \mathbf{N} \\
(56) & LIM\_OC & = & \mathbf{N}
\end{array}
$$

As principais construções denotam:

- Por (33) sabe-se que a detecção de problemas se dará de forma separada por cada área de gerenciamento OSI.
- Em (34) detalha-se que para cada área de gerenciamento, a detecção de problemas será separada conforme os níveis de protocolo em utilização.
- Em (35) detalham-se as estruturas necessárias para a detecção de problemas a cada nível: os predicados que as unidades de dados devem satisfazer, as estruturas de *CONT*abilização, os *HIST*óricos, e os alarmes gerados naquele nível.
- Em (36) tem-se os predicados sobre ocorrências. Estes predicados, se satisfeitos, geram contabilizações, isto é, se a *EXPR*essão for verdadeira então a estrutura da contabilização para *EVENTO* é modificada.
- Em (37) tem-se os predicados sobre ausências. Estes, se não satisfeitos dentro de um *INTERVALO* de tempo, geram contabilizações, da mesma forma como (36).
- Na linha (38) está especificada a forma de uma estrutura de contabilização: para cada *EVENTO* temos uma lista de *MOMENTO*s que este evento foi detectado, um *N*úmero *MAX*imo de momentos que devem permanecer na lista, um *T*amanho *MAX*imo para a lista, um *INT*ervalo de tempo para *AN*álise sobre a lista em que se existir um número maior do que *LIM*ite de *OC*orrências *MOMENTO*s na lista, será disparado o *ALARME*, que possui em sua estrutura um prazo de validade.
- Em (39) fica definido que existe um *HIST*órico para cada *ALARME*, sendo sua estrutura composta por uma lista de *MOMENTO*s que aconteceram o *ALARME* e o *N*úmero *MAX*imo de *ALARME*s que podem existir na lista.
- Em (40) define-se a lista de alarmes para cada nível como sendo um conjunto de duplas *ALARME* e *MOMENTO* em que foi disparado.

- Por (41) temos que um *ALARME* é identificado por um nome igual ao do *EVENTO* que o gera ou pode ser composto de um nome específico e uma seqüência nula ou não de valores encontrados nos dados de tráfego, servindo como subsídio para o processo de diagnose.

- Um *EVENTO*, por (46), denota ocorrência de determinada circunstância (disparada na computação sobre (36)) ou denota ausência de determinada circunstância (disparada na computação sobre (37)). O *EVENTO* é composto por um nome e por uma seqüência nula ou não de valores encontrados nos dados de tráfego que se queira passar para o histórico e posteriormente para o processo de diagnose.

4.4.2.2 Funções

Funções do Submódulo de Detecção:

(57)	*Deteccao*	:	$AREA\_GER\text{-set} \times NIVEL\text{-set} \times$ $\times TRAFEGO \rightarrow DETECTA \rightarrow DETECTA$
(58)	*Detecta_Niveis*	:	$NIVEL\text{-set} \times TRAFEGO \times$ $\rightarrow DET\_AREA \rightarrow DET\_AREA$
(59)	*Atualiza_L_LAlarm*	:	$DET\_N \times TRAFEGO \rightarrow DET\_N$

(60)	*Contab_Ocorrencias*	:	$DET\_N \times TRAFEGO \rightarrow DET\_N$
(61)	*Contab_Ausencias*	:	$DET\_N \times TRAFEGO \rightarrow DET\_N$
(62)	*Contab_Ausencias2*	:	$PRED\_AUS\_TRAF\text{-set} \times I \times$
			$\times CV\text{-set} \rightarrow DET\_N \rightarrow DET\_N$
(63)	*Aval_Pred_Oc*	:	$PRED\_OC\_TRAF\text{-set} \times CV\text{-set} \times$
			$\rightarrow EVENTO\text{-set} \rightarrow EVENTO\text{-set}$
(64)	*Avalia_Expr*	:	$EXPR \times CV\text{-set} \rightarrow BOOL$
(65)	*Atualiza_Contab*	:	$I \times EVENTO\text{-set} \rightarrow DET\_N \rightarrow DET\_N$
(66)	*Verif_Existencia_Cont*	:	$CONT \times EVENTO \rightarrow CONT$
(67)	*Verif_Existencia_Hist*	:	$HIST\text{-set} \times ALARME \rightarrow HIST\text{-set}$
(68)	*Atu_Lista_Mom*	:	$N\_MAX \times T\_MAX \times I \rightarrow$
			$\rightarrow MOMENTO^* \rightarrow MOMENTO^*$
(69)	*Atualiza_Hist*	:	$ALARME \times I \times HIST \rightarrow HIST$
(70)	*Det_Get_L_Alarm*	:	$AREA\_GER \times NIVEL \rightarrow$
			$\rightarrow ALARM\text{-set}$
(71)	*Det_Create_Alarm*	:	$AREA\_GER \times$
			$\times NIVEL \times CONT \rightarrow BOOLEAN$
(72)	*Det_Delete_Alarm*	:	$AREA\_GER \times$
			$\times NIVEL \times ALARM \rightarrow BOOLEAN$
(73)	*Det_Create_Pred*	:	$AREA\_GER \times NIVEL \times$
			$\times CONT \rightarrow BOOLEAN$
(74)	*Det_Delete_Pred*	:	$AREA\_GER \times NIVEL \times$
			$\times EVENTO \rightarrow BOOLEAN$

A seguir serão detalhadas as funções deste submódulo:

57. $Deteccao : AREA\_GER\text{-set} \times NIVEL\text{-set} \times TRAFEGO$
$\rightarrow DETECTA \rightarrow DETECTA$

57. $Deteccao(areas, niveis, trafego)(detecta) \triangleq$
if $areas = \{\}$
then $detecta$
else let $a \in areas$ in
let $det\_area = detecta(a)$ in
let $det\_area2 = Detecta\_Niveis(niveis, trafego)(det\_area)$ in
$Deteccao(areas - a, niveis, trafego)(detecta \dagger [a \mapsto det\_area2])$

A detecção de eventos acontece nas diversas áreas de gerenciamento definidas. Para cada área é disparada a detecção nos diferentes níveis de protocolo através da função *Detecta_Niveis*.

Este processo gera subsídios para análise posterior, no Submódulo de Diagnose.

58. $Detecta\_Niveis : NIVEL\text{-}\mathsf{set} \times TRAFEGO \rightarrow DET\_AREA$
$\rightarrow DET\_AREA$

58. $Detecta\_Niveis(niveis, trafego)(det\_area) \triangleq$
if $niveis = \{\}$
then det_area
else let $n \in niveis$ **in**
let $det\_n = det\_area(n)$ **in**
let $det\_n1 = Atualiza\_L\_Alarm(det\_n, trafego)$ **in**
let $det\_n2 = Contab\_Ocorrencias(det\_n1, trafego)$ **in**
let $det\_n3 = Contab\_Ausencias(det\_n2, trafego)$ **in**
$Detecta\_Niveis(niveis - n, trafego)(det\_area \dagger [n \mapsto det\_n3])$

A função 58 dispara a atualização da lista de alarmes (conforme seu tempo de validade) e a contabilização de "Ocorrências" e "Ausências" para todos os níveis de protocolo dentro de uma mesma área de gerenciamento.

59. $Atualiza\_L\_Alarm : DET\_N \times TRAFEGO \rightarrow DET\_N$

59. $Atualiza\_L\_Alarm(det\_n, mk\text{-}TRAFEGO(i, nivel, cvs)) \triangleq$
    let $mk\text{-}DET\_N(pred\_oc, pred\_aus, cont, hist, l\_alarm) = det\_n$ **in**
    if $\exists mk\text{-}ESTR\_ALARM(al, valid) \in l\_alarm \cdot i > valid$
    then let $a = mk\text{-}ESTR\_ALARM(al, valid)$ **in**
        $Atualiza\_L\_Alarm(mk\text{-}DET\_N(pred\_oc, pred\_aus, cont,$
        $hist, l\_alarm - a), mk\text{-}TRAFEGO(i, nivel, cvs))$
    else det_n

Os *alarmes* têm um tempo de validade, isto é, um tempo que devem ficar ativos por ocasião de seu disparo. A função acima torna os *alarmes* não válidos inativos, seguindo seu tempo de validade.

60. $Contab\_Ocorrencias : DET\_N \times TRAFEGO \rightarrow DET\_N$

60. $Contab\_Ocorrencias(det\_n, mk\text{-}TRAFEGO(i, nivel, cvs)) \triangleq$
    let $mk\text{-}DET\_N(pred\_oc\_traf, pred\_aus\_traf, cont, hist, l\_alarm) =$
    det_n **in**
    let $eventos\_a\_contabilizar =$
    $Avalia\_Pred\_Oc(pred\_oc\_traf, nivel, cvs)(\{\})$ **in**
    $Atualiza\_Contab(i, eventos\_a\_contabilizar)(det\_n)$

Esta é a função de mais alto nível na contabilização de ocorrências de eventos, ela obtém os eventos a serem contabilizados através da avaliação dos predicados de ocorrência sobre o tráfego, e dispara a atualização das estruturas necessárias através de *Atualiza_Contab*.

61. $Contab\_Ausencias : DET\_N \times TRAFEGO \rightarrow DET\_N$

61. $Contab\_Ausencias(det\_n, mk\text{-}TRAFEGO(i, nivel, cvs)) \triangleq$
    let $mk\text{-}DET\_N(pred\_oc\_traf, pred\_aus\_traf, cont, hist, l\_alarm) =$
        det_n in
    $Contab\_Ausencias2(pred\_aus\_traf, i, cvs)$
    $(mk\text{-}DET\_N(pred\_oc\_traf, \{\}, cont, hist, l\_alarm))$

Esta é a função de mais alto nível relativa à contabilização de ausências de eventos. Ela dispara a função recursiva *Contab_Ausencias2*.

62. $Contab\_Ausencias2 : PRED\_AUS\_TRAF\text{-set} \times I \times CV\text{-set}$
$\rightarrow DET\_N \rightarrow DET\_N$

62. $Contab\_Ausencias2(p\_aus, i, cvs)(det\_n) \triangleq$
if $p\_aus = \{\}$
then det_n
else let $mk\text{-}DET\_N(p\_oc, p\_aus2, cont, hist, l\_alarm) = det\_n$ in
let $pred \in p\_aus$ in
let $mk\text{-}PRED\_AUS\_TRAF(expr, evento, intervalo, mom\_lim) =$
$pred$ in
if $mom\_lim >= i$
then if $Avalia\_Expr(expr, cvs)$
then $Contab\_Ausencias2(p\_aus\text{-}pred, i, cvs)$
$(mk\text{-}DET\_N(p\_oc, p\_aus2 \cup mk\text{-}PRED\_AUS\_TRAF$
$(expr, evento, intervalo, i+intervalo), cont, hist, l\_alarm))$
else $Contab\_Ausencias2(p\_aus\text{-}pred, i, cvs)$
$(mk\text{-}DET\_N(p\_oc, p\_aus2 \cup mk\text{-}PRED\_AUS\_TRAF$
$(expr, evento, intervalo, mom\_lim), cont, hist, l\_alarm))$
else let $mk\text{-}DET\_N(p\_oc', p\_aus', cont', hist', l\_alarm') =$
$Atualiza\_Contab(i, evento)(det\_n)$ in
$Contab\_Ausencias2(p\_aus\text{-}pred, i, cvs)$
$(mk\text{-}DET\_N(p\_oc', p\_aus' \cup mk\text{-}PRED\_AUS\_TRAF$
$(expr, evento, intervalo, i + intervalo), cont, hist, l\_alarm))$

Esta função verifica os predicados de ausência sobre o tráfego e dispara a contabilização de eventos quando estes predicados não são satisfeitos dentro de um tempo limite. Também é feito o controle destes tempos durante a verificação.

```
63. Aval_Pred_Oc : PRED_OC_TRAF-set × CV-set → EVENTO-set
                   → EVENTO-set

63. Aval_Pred_Oc(preds, cvs)(evs) ≜
    if preds = { }
    then evs
    else let p ∈ preds in
        let mk-PRED_OC_TRAF(expr, evento) = p in
        if Avalia_Expr(expr, cvs)
        then Aval_Pred_OC(preds-p, cvs)(evs ∪ evento)
        else Aval_Pred_OC(preds-p, cvs)(evs)
```

Esta função avalia os predicados de ocorrência sobre o tráfego, gerando um conjunto de eventos que devem ser contabilizados.

```
64. Avalia_Expr : EXPR × CV-set → BOOL

64. Avalia_Expr(expr, cvs) ≜
    /* avalia a expressao recursivamente obedecendo */
    /* as precedencias previamente definidas para os */
    /* operadores logicos */
```

A expressão *expr* é avaliada sobre o conjunto de campos e valores *cvs*, gerando um valor boleano.

65. $Atualiza\_Contab : I \times EVENTO\text{-set} \rightarrow DET\_N \rightarrow DET\_N$

65. $Atualiza\_Contab(i, eventos)(det\_n) \triangleq$

if $eventos = \{\}$

then det_n

else let $e \in eventos$ in

let $mk\text{-}DET\_N(p\_oc, p\_aus, cont, hist, l\_alarm) = det\_n$ in

let $cont' = Verif\_Existencia\_Cont(cont, e)$ in

let $(l\_mom, n\_max, t\_max, int\_an, lim\_oc, alarme) = cont'(e)$ in

let $l\_mom' = \text{conc}\, l\_mom, i$ in

let $nro\_oc = \text{card}\,(x \in \text{elems}\, l\_mom' \mid x > i - int\_an)$ in

if $(nro\_oc < lim\_oc)$

then let $l\_mom'' = Atu\_Lista\_Mom(n\_max, t\_max, i)(l\_mom')$ in

let $cont'' = cont' \dagger$

$[e \mapsto (l\_mom'', n\_max, t\_max, int\_an, lim\_oc, alarme)]$ in

$Atualiza\_Contab(i, eventos - e,$

$mk\text{-}DET\_N(p\_oc, p\_aus, cont'', hist, l\_alarm)$

else let $mk\text{-}ESTR\_ALARM(al, valid) = alarme$ in

let $hist' = Atualiza\_Hist(al, i, hist)$ in

let $cont'' =$

$cont' \dagger [e \mapsto (<>, n\_max, t\_max, int\_an, lim\_oc, alarme)]$ in

let $l\_alarm' = \text{conc}\, l\_alarm, < al, i + valid >$ in

$Atualiza\_Contab(i, eventos - e,$

$mk\text{-}DET\_N(p\_oc, p\_aus, cont'', hist', l\_alarm')$

Esta função atualiza as estruturas de contabilização de eventos e, quando os limites de ocorrência "estouram", são disparados alarmes gerando atualizações em históricos e aumentando também as lista de alarmes por nível.

66. $Verif\_Existencia\_Cont : CONT \times EVENTO \rightarrow CONT$

66. $Verif\_Existencia\_Cont(cont, ev) \triangleq$

    if $ev \in$ dom $cont$

    then $cont$

    else $cont \cup [ev \mapsto (<>, 0, 0, 0, 0,$

        $mk\text{-}ESTR\_ALARM(mk\text{-}ALARME(nao\_definido, ev), 10))]$

A função acima verifica a existência da contabilização para o alarme disparado e, se não existir, é criada uma estrutura de contabilização *default*.

67. $Verif\_Existencia\_Hist : HIST$-set $\times ALARME \rightarrow HIST$-set

67. $Verif\_Existencia\_Hist(hists, alarme) \triangleq$

    if $alarme \in$ dom $hist$

    then $hist$

    else $hist \cup [alarme \mapsto (<>, n\_max\_hist)]$

A função acima verifica a existência do histórico de um alarme sendo alterado e, se este não existir, é criado com valores *default*.

68. $Atu\_Lista\_Mom : N\_MAX \times T\_MAX \times I \rightarrow MOMENTO^* \rightarrow MOMENTO^*$

68. $Atu\_Lista\_Mom(n\_max, t\_max, i)(l\_mom) \triangleq$
    if $(\mathsf{card}\,(x \in \mathsf{elems}\, l\_mom) > n\_max) \vee$
    $(\mathsf{hd}\, l\_mom < i - t\_max)$
    then $Atu\_Lista\_Mom(n\_max, t\_max, i)(\mathsf{tl}\, l\_mom)$
    else l_mom

Esta função mantém as listas de momentos de ocorrências de eventos conforme os parâmetros tamanho e tempo de permanência na lista, particulares a cada estrutura de contabilização de eventos.

69. $Atualisa\_Hist : ALARME \times I \times HIST \rightarrow HIST$

69. $Atualisa\_Hist(alarme, i, hist) \triangleq$

$\quad$ let $hist' = Verif\_Existencia\_Hist(hist, alarme)$ in

$\quad$ let $(l\_mom, n\_max) = hist'(alarme)$ in

$\quad$ let $l\_mom1 = \mathsf{conc}\, l\_mom, i$ in

$\quad$ if $\mathsf{card}\,(x \in elemsl\_mom1) > n\_max$

$\quad$ then $hist' \dagger [alarme \mapsto (\mathsf{tl}\, l\_mom1, n\_max)]$

$\quad$ else $hist' \dagger [alarme \mapsto (l\_mom1, n\_max)]$

Esta função mantém as listas de momentos de ocorrências de alarmes conforme o parâmetro tamanho da lista, particular a cada estrutura de histórico.

O Submódulo de Detecção oferece, ainda, ao Submódulo de Diagnose, uma interface composta de funções que encapsulam suas estruturas. Desta forma, toda vez que informações de detecção forem necessárias na fase de diagnóstico, ou o módulo de diagnóstico modificar alguns dados do módulo de detecção, estas funções serão disparadas.

70. $Det\_Get\_L\_Alarmes : AREA\_GER \times NIVEL \rightarrow ALARM$-set

70. $Det\_Get\_L\_Alarmes(teoria, subteoria) \triangleq$

rd $detecta$: $DETECTA$

let $det\_area = detecta(teoria)$ in

let $det\_n = det\_area(subteoria)$ in

let $mk\text{-}DET\_N(\_,\_,\_,\_,l) = det\_n$ in

l

A função acima retorna a lista de alarmes relativa à subteoria de uma teoria.

71. $Det\_Create\_Alarm : AREA\_GER \times NIVEL \times CONT \rightarrow BOOLEAN$

71. $Det\_Create\_Alarm(teoria, subteoria, pred, cont) \triangleq$

wr $detecta$: $DETECTA$

let $det\_area = detecta(teoria)$ in

let $det\_n = det\_area(subteoria)$ in

let $mk\text{-}DET\_N(p\_oc, p\_aus, contabs, hist, l\_alarm) = det\_n$ in

let $[ev \mapsto (l\_mom, n\_max, t\_max,$

$int\_an, lim\_oc, mk\text{-}ESTR\_ALARM(alarme, valid))] = cont$ in

if $\exists c \in cont \cdot dom(c) = dom(cont) \vee$

$\exists h \in hist \cdot dom(h) = alarme \vee$

$\exists po \in p\_oc \cdot po = pred \vee$

$\exists pa \in p\_aus \cdot pa = pred$

then $False$

else let $contabs' = contabs \cup cont$ in

let $hist' = hist \bigcup$

$[alarme \mapsto (<>, n\_max\_hist)]$ in

if $is\text{-}PRED\_OC\_TRAF(pred)$

then let $det\_n' = mk\text{-}DET\_N(p\_oc \cup pred, p\_aus,$

$contabs', hist', l\_alarm)$ in

let $det\_area' = det\_area \dagger [subteoria \mapsto det\_n']$ in

let $detecta = detecta \dagger [teoria \mapsto det\_area']$ in

$True$

else let $det\_n' = mk\text{-}DET\_N(p\_oc, p\_aus \cup pred,$

$contabs', hist', l\_alarm)$ in

let $det\_area' = det\_area \dagger [subteoria \mapsto det\_n']$ in

let $detecta = detecta \dagger [teoria \mapsto det\_area']$ in

$True$

Esta função cria uma contabilização sobre um evento de tráfego. O usuário desta função (o submódulo de Diagnose) deve informar a subteoria e a teoria em que será adicionada a nova contabilização, bem como passar o predicado que a dispara e os parâmetros da estrutura de contabilização que definem seu comportamento, anteriormente explanado. O procedimento verifica a existência de predicados ou estruturas que inviabilizem a criação desta nova e, se não existir impedimento algum, adiciona-se um novo predicado, uma nova estrutura de contabilização e uma nova estrutura de histórico.

72. $Det\_Delete\_Alarm : AREA\_GER \times NIVEL \times ALARM \rightarrow BOOLEAN$

72. $Det\_Delete\_Alarm(teoria, subteoria, alarm) \triangleq$

wr $detecta$: $DETECTA$

let $det\_area = detecta(teoria)$ in

let $det\_n = det\_area(subteoria)$ in

let $mk\text{-}DET\_N(p\_oc, p\_aus, contabs, hist, l\_alarm) = det\_n$ in

let $hist' = hist/alarm$ in

let $contabs(ev) = (l\_mom, n\_max, t\_max, int\_an, lim\_oc,$

$mk\text{-}ESTR\_ALARM(alarm, valid))$ in

let $cont' = contabs/ev$ in

let $b = Det\_Delete\_Pred(teoria, subteoria, ev)$ in

let $det\_n' = mk\text{-}DET\_N(p\_oc, pred, p\_aus, cont', hist', l\_alarm)$ in

let $det\_area' = det\_area \dagger [subteoria \mapsto det\_n']$ in

let $detecta = detecta \dagger [teoria \mapsto det\_area']$ in

$True$

Esta função deleta todas as estruturas de contabilização para um alarme dado. O usuário desta função (o submódulo de Diagnose) deve informar a subteoria e a teoria de que será retirada o alarme.

73. $Det\_Create\_Pred : AREA\_GER \times NIVEL \times CONT \rightarrow BOOLEAN$

73. $Det\_Create\_Pred(teoria, subteoria, pred) \triangleq$

wr $detecta$: $DETECTA$

let $det\_area = detecta(teoria)$ in

let $det\_n = det\_area(subteoria)$ in

let $mk\text{-}DET\_N(p\_oc, p\_aus, contabs, hist, l\_alarm) = det\_n$ in

if $is\text{-}PRED\_OC\_TRAF(pred)$

then let $mk\text{-}PRED\_OC\_TRAF(expr, evento) = pred$ in

if $\exists mk\text{-}PRED\_OC\_TRAF(expr1, evento1) \in pred\_oc \cdot$

$evento1 = evento$

then let $det\_n' = mk\text{-}DET\_N(p\_oc \cup pred, p\_aus,$

$contabs, hist, l\_alarm)$ in

let $det\_area' = det\_area \dagger [subteoria \mapsto det\_n']$ in

let $detecta = detecta \dagger [teoria \mapsto det\_area']$ in

$True$

else $False$

else let $mk\text{-}PRED\_AUS\_TRAF(expr, evento, interv, mom\_lim) = pred$ in

if $\exists mk\text{-}PRED\_AUS\_TRAF(ex1, ev1, int1, mom1) \in pred\_oc \cdot$

$ev1 = evento$

then let $det\_n' = mk\text{-}DET\_N(p\_oc, p\_aus \cup pred,$

$contabs, hist, l\_alarm)$ in

let $det\_area' = det\_area \dagger [subteoria \mapsto det\_n']$ in

let $detecta = detecta \dagger [teoria \mapsto det\_area']$ in

$True$

else $False$

A função acima cria um novo predicado que dispara contabilizações em uma estrutura já existente, para o caso de vários eventos contribuirem para a formação de um alarme.

74. $Det\_Delete\_Pred : AREA\_GER \times NIVEL \times EVENTO \rightarrow BOOLEAN$

74. $Det\_Delete\_Pred(teoria, subteoria, evento) \triangleq$

wr $detecta$: $DETECTA$

let $det\_area = detecta(teoria)$ in

let $det\_n = det\_area(subteoria)$ in

let $mk\text{-}DET\_N(p\_oc, p\_aus, contabs, hist, l\_alarm) = det\_n$ in

let $po = \{p \in p\_oc \mid p = mk\text{-}PRED\_OC\_TRAF(e, evento)\}$ in

let $pa = \{p \in p\_aus \mid$

$p = mk\text{-}PRED\_AUS\_TRAF(e, evento, intm, mom)\}$ in

let $p\_oc' = p\_oc\text{-}po$ in

let $p\_aus' = p\_aus\text{-}pa$ in

let $det\_n' = mk\text{-}DET\_N(p\_oc', p\_aus', contabs, hist, l\_alarm)$ in

let $det\_area' = det\_area \dagger [subteoria \mapsto det\_n']$ in

let $detecta = detecta \dagger [teoria \mapsto det\_area']$ in

$True$

A função acima retira todos os predicados relativos ao mesmo evento da base de predicados.

4.4.3 Submódulo de Diagnóstico

Os alarmes captados no Submódulo de Detecção são submetidos a este submódulo com o objetivo de diagnosticar possíveis problemas na instalação em análise. Após a diagnose, o resultado é passado ao Submódulo de Correção que fornece uma recomendação para solucionar o problema em questão.

O Submódulo de Diagnóstico usa, para isso, do conhecimento de um especialista em forma de regras, alarmes, bem como outros subsídios. As estruturas e as funções envolvidas estão especificadas e comentadas a seguir.

4.4.3.1 Domínios Semânticos

As estruturas utilizadas no Submódulo de Diagnóstico são as apresentadas a seguir:

(75) $BANCO\_DE\_CONHEC :: CJ\_TEORIAS \times CONHEC\_ESTRUT$

(76) $CONHEC\_ESTRUT = FATO\_SOBRE\_REDE\text{-set}$

(77) $FATO\_SOBRE\_REDE :: OBJETO\times$
$\times ATRIBUTO \times VALOR$

(78) $CJ\_TEORIAS = AREA\_GER \xrightarrow{m} TEORIA$

(79) $TEORIA :: NIVEL \xrightarrow{m} SUBTEORIA\times$
$\times inter\text{-}reLACOES$

(80) $SUBTEORIA = REGRA\text{-set}$

(81) $INTERRELACOES = REGRA\text{-set}$

(82) $REGRA :: NRO\_REGRA \times CONDICAO\times$
$\times ASSERCAO \times FATOR\_DE\_CERTEZA$

(83) $CONDICAO = COND\_COMPOSTA \mid$
$COND\_ELEMENTAR$

(84) $COND\_COMPOSTA :: CONDICAO \times OPBOOL \times CONDICAO$

(85) $COND\_ELEMENTAR = ALARME \mid$
$FATO\_SOBRE\_REDE \mid NO(CONDICAO)$

(86) $ASSERCAO = FATO\_SOBRE\_REDE \mid DIAGNOSE$

(87) $DIAGNOSE = PROBL \times TRACE\times$
$\times FATOR\_DE\_CERTEZA$

(88) $FATOR\_DE\_CERTEZA = \mathbf{R}$

(89) $NRO\_REGRA = \mathbf{N}$

(90) $OBJETO = STRING$

(91) $ATRIBUTO = STRING$

(92) $PROBL = STRING$

(93) $TRACE = STRING$

- Em (75) define-se o *Banco de Conhecimento* dividido em *conhecimento estrutural e em teorias.* O *Conhecimento Estrutural* corresponde à estrutura lógica e física da rede. As *Teorias* contém conhecimento especialista para tratamento de problemas em diferentes áreas.

- O *Conhecimento Estrutural* (76) é um conjunto de fatos verdadeiros sobre a rede, seguindo a estrutura OAV (Objeto/Atributo/Valor). Através desta estruturação pretende-se criar uma forma bastante permissível de representação de situações lógicas e físicas sobre a instalação (criando uma rede semântica) [GMS89]. Utiliza-se o mesmo *banco de conhecimento estrutural* para a análise em qualquer *teoria.*

- Um *Fato Sobre a Rede* (77) é uma tripla do tipo OAV (Objeto/Atributo/Valor) através da qual pode-se criar assertivas sobre a rede.

- O *Conjunto de Teorias* (78) divide o conhecimento especialista em várias teorias conforme as áreas de gerenciamento OSI (descritas anteriormente). Assim cada teoria é alimentada por alarmes distintos, gerando diagnoses distintas.

- Cada *Teoria* (79) para cada área de gerenciamento OSI é, por sua vez, dividida em subteorias seguindo os níveis de protocolos em utilização, e uma subteoria à parte para realizar o inter-relacionamento das diagnoses das subteorias dos diversos níveis. Esta estruturação faz com que o conhecimento de cada área de gerenciameto fique dividido para cada nível de protocolo, auxiliando no domínio da complexidade.

- Cada *Subteoria* (80) para cada nível é composta por um conjunto de regras (codificando heurísticas), resultado da codificação do conhecimento de um especialista por engenharia de conhecimento. Estas regras associam alarmes e conhecimento estrutural, podendo se valer de históricos também.

- As *subteorias de inter-relacionamento* (??) são também conjuntos de regras. As regras de inter-relacionamento associam *diagnoses* das subteorias dos níveis

(ou diagnoses parciais) validando, combinando ou cancelando diagnoses. O *banco de conhecimento estrutural* também é importante neste processo.

- Uma *Regra* (82) é composta por um número que a identifica univocamente dentro de sua *Subteoria*, uma condição que deve ser satisfeita para que a regra seja aplicada, a ação resultante da aplicação da regra, e um fator de certeza associado à regra que auxilia na computação do fator de certeza global para as diagnoses.

- *Condições* (83), segundo sua estrutura, podem ser compostas ou elementares.

- *Condições Compostas* (84) contém operadores booleanos de aridade 2, no caso: AND's e OR's.

- *Condições Elementares* (85) são testes quanto à existência de alarmes, fatos sobre a rede, ou são negações de uma condição.

- Uma *Asserção* (86) significa a afirmação de algo sobre a rede, no caso pode-se chegar a um diagnóstico, ou pode-se afirmar sobre a rede (utilizando 77) criando subsídios para posterior análise.

- Uma *Diagnose* (87) é composta por um problema ou situação causadora, a árvore de dedução ou *trace* deste problema utilizando-se o corpo de regras da subteoria devida, e o *fator de certeza* para a dedução do problema.

- Cada regra tem um *fator de certeza* associado, este fator é limitado entre zero e um. A forma de cálculo de fatores de certeza empregada é bastante comum e foi retirada de [ARI89]. Os *fatores de certeza* (88) são calculados da seguinte forma:

  em conjunções de expressões booleanas (ANDs), utiliza-se o menor fator de certeza entre os dois termos;

  em disjunções de expressões booleanas (ORs), utiliza-se o maior entre os fatores de certeza;

em negações de expressões booleanas, utiliza-se o complemento de 1 da expressão negada;

na aplicação encadeada de regras utiliza-se o produto dos fatores de certeza.

4.4.3.2 Funções

Funções do Submódulo de Diagnóstico:

(94) $Diagnostico : I \times BANCO\_DE\_CONHEC \times$
$\times AREA\_GER\text{-set} \times NIVEL\text{-set} \rightarrow$
$\rightarrow DIAGNOSE\text{-set}$

(95) $Diag\_Teorias : BANCO\_DE\_CONHEC \times$
$\times AREA\_GER\text{-set} \times NIVEL\text{-set} \rightarrow$
$\rightarrow DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$

(96) $Diag\_Subteorias : BCO\_DE\_CONHEC \times AREA\_GER \times$
$\times NIVEL\text{-set} \rightarrow DIAGNOSE\text{-set} \rightarrow$
$\rightarrow DIAGNOSE\text{-set}$

(97) $Infere\_Prof\_Tras : BCO\_DE\_CONHEC \times AREA\_GER \times$
$\times NIVEL \times NRO\_REGRA\text{-set} \times$
$\times NRO\_REGRA\text{-set} \rightarrow DIAGNOSE\text{-set}$

(98) $Prova\_Goal\_PT : SUBTEORIA \times CONHEC\_ESTRUT \times$
$\times ASSERCAO \times AREA\_GER \times NIVEL \times$
$\times SUBTEORIA \rightarrow setof\, DIAGNOSE$

(99) $Prova\_Cond\_PT : SUBTEORIA \times CONHEC\_ESTRUT \times$
$\times ASSERCAO \times AREA\_GER \times$
$\times NIVEL \rightarrow setof\, DIAGNOSE$

(100) $Prova\_Cond\_Elementar\_PT : SUBTEORIA \times CONHEC\_ESTRUT \times$
$\times ASSERCAO \times AREA\_GER \times$
$\times NIVEL \rightarrow DIAGNOSE\text{-set}$

(101)	*Nega*	:	$DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$
(102)	*Prova_Cond_Composta_PT*	:	$SUBTEORIA \times CONHEC\_ESTRUT \times$ $\times ASSERCAO \times AREA\_GER \times$ $\times NIVEL \rightarrow DIAGNOSE\text{-set}$
(103)	*Combina*	:	$DIAGNOSE\text{-set} \times DIAGNOSE\text{-set} \rightarrow$ $\rightarrow DIAGNOSE\text{-set}$
(104)	*Combina2*	:	$DIAGNOSE \times DIAGNOSE\text{-set} \rightarrow$ $\rightarrow DIAGNOSE\text{-set}$
(105)	*Atu_Diags*	:	$REGRA \times DIAGNOSE\text{-set} \rightarrow$ $\rightarrow DIAGNOSE\text{-set}$
(106)	*Diag_inter-relacionamento*	:	$BANCO\_DE\_CONHEC \times AREA\_GER \times$ $\times DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$
(107)	*Infere_Amplit_Frente*	:	$REGRAS\text{-set} \times REGRAS\text{-set} \times$ $\times CONHEC\_ESTRUT \times DIAGNOSE\text{-set} \times$ $\times DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$
(108)	*Prova_Cond_AF*	:	$CONDICAO \times DIAGNOSE\text{-set} \times$ $\times CONHEC\_ESTRUT \rightarrow DIAGNOSE$
(109)	*Prova_Cond_Elementar_AF*	:	$COND\_ELEMENTAR \times DIAGNOSE\text{-set} \times$ $\times CONHEC\_ESTRUT \rightarrow DIAGNOSE$
(110)	*Prova_Cond_Composta_AF*	:	$COND\_COMPOSTA \times DIAGNOSE\text{-set} \times$ $\times CONHEC\_ESTRUT \rightarrow DIAGNOSE$

A seguir, cada função deste Submódulo está especificada.

94. $Diagnostico : I \times BANCO\_DE\_CONHEC \times AREA\_GER\text{-set} \times$
$\times NIVEL\text{-set} \rightarrow DIAGNOSE\text{-set}$

94. $Diagnostico(i, bco\_de\_conhec, teorias, subteorias) \triangleq$
wr $i\_ant: I$
if $i = i\_ant$
then $\{\}$
else let $i\_ant = i$ in
$Diag\_Teorias(bco\_de\_conhec, teorias, subteorias)(\{\})$

Para o *diagnóstico*, submete-se o último instante i em que foram recebidos dados da rede, *bco_de_conhec* o banco de conhecimento acima descrito, além das *teorias e subteorias* sobre as quais deve haver inferência.

A diagnose é somente disparada quando existe a troca de instante i pois isto significa a passagem de todos os níveis de uma PDU pelos predicados de detecção, uma vez que o *loop* de busca de dados e detecção de alarmes acontece por nível de protocolo em cada PDU. Disparando as diagnoses desta maneira, todas as subteorias de cada nível de protocolo poderão atuar sobre informações renovadas, ao invés de repetir computações já realizadas.

95. $Diag\_Teorias : BANCO\_DE\_CONHEC \times AREA\_GER\text{-set} \times$
$\times NIVEL\text{-set} \rightarrow DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$

95. $Diag\_Teorias(bco\_de\_conhec, teorias, subteorias)(diagnoses) \triangleq$
if $teorias = \{\}$
then $diagnoses$
else let $teoria \in teorias$ in
let $diags = Diag\_Subteorias(bco\_de\_conhec, teoria, subteorias)(\{\})$
in
let $diags2 = Diag\_inter\text{-}relacionamento(bco\_de\_conhec, teoria, diags)$
in
$Diag\_Teorias(bco\_de\_conhec, teorias\text{-}teoria,$
$subteorias)(diagnoses \cup diags2)$

A função *Diag_Teorias* realiza o diagnóstico para cada *teoria* contida em *teorias*. Em cada *teoria* ela dispara os diagnósticos sobre as subteorias através da função *Diag_Subteorias* e inter-relaciona os resultados destas subteorias atvés da função *Diag_inter-relacionamento*. Note que os diagnósticos finais são somente os que passaram pelo inter-relacionamento dos diagnósticos dos diversos níveis.

96. $Diag\_Subteorias : BCO\_DE\_CONHEC \times AREA\_GER \times NIVEL$-set
$\rightarrow DIAGNOSE$-set $\rightarrow DIAGNOSE$-set

96. $Diag\_Subteorias(bco\_de\_conhec, teoria, subteorias)(diagnoses) \triangleq$
if $subteorias = \{\}$
then $diagnoses$
else let $subteoria \in subteorias$ in
let $diags = Infere\_Prof\_Tras(bco\_conhec, teoria, subteoria, \{\})$ in
$Diag\_Subteorias(bco\_de\_conhec, teoria,$
$subteorias\text{-}subteoria)(diagnoses \cup diags)$

A função acima realiza o diagnóstico para cada *subteoria* de *subteorias* contida em *teoria*. Para cada *subteoria* é disparada a função *Infere_Prof_Tras*.

97. $Infere\_Prof\_Tras : BCO\_DE\_CONHEC \times AREA\_GER \times$
$\times NIVEL \times NRO\_REGRA\text{-set}$
$\rightarrow DIAGNOSE\text{-set}$

97. $Infere\_Prof\_Tras(bco\_de\_conhec, teoria,$
$subteoria, regras\_goal) \triangleq$
let $mk\text{-}BCO\_DE\_CONHEC(teorias, conhec\_estr) = bco\_de\_conhec$ in
let $mk\text{-}TEORIA(teoria\_area, inter\text{-}relacionamento) = teorias(teoria)$ in
let $teoria\_nivel = teoria\_area(subteoria)$ in
if $\exists r = mk\text{-}REGRA(n, cond, acao, fc) \in teoria\_nivel \cdot$
$n \notin regras\_goal \wedge is\text{-}ASSERCAO(acao)$
then let $diags = Prova\_Goal\_PT(teoria\_nivel, conhec\_estr,$
$acao, teoria, subteoria, teoria\_nivel)$ in
$diags \cup Infere\_Prof\_Tras(bco\_de\_conhec, teoria,$
$subteorias, regras\_goal \cup r)$
else $\{\}$

A função *Infere_Prof_Tras* realiza uma inferência em profundidade para trás (ou encadeamento regressivo) sobre as regras da *subteoria* indicada. A forma de inferência escolhida foi esta já que o conjunto de estados iniciais (alarmes) é mais numeroso que os estados finais (diagnoses) [WEK84] [HAK88], pois diferentes configurações de alarmes podem resultar em um mesmo diagnóstico. Desta forma, procura-se, sobre toda a subteoria em inferência, pela existência de regras que levem a asserções (diagnósticos ou afirmações sobre a rede). Para cada regra deste tipo, tenta-se prová-la através da função *Prova_Goal_PT*.

98. $Prova\_Goal\_PT : SUBTEORIA \times CONHEC\_ESTRUT \times ASSERCAO \times$
$\times AREA\_GER \times NIVEL \times SUBTEORIA$
$\rightarrow setof\, DIAGNOSE$

98. $Prova\_Goal\_PT(regras, conhec\_estr, goal,$
$teoria, subteoria, regras\_goal) \triangleq$
if $\exists r = mk\text{-}REGRA(n, cond, goal, fc) \cdot \in regras\_goal$
then let $diags = Prova\_Cond\_PT(regras, conhec\_estr, cond, teoria,$
$subteoria)$ in
let $diags2 = Atu\_Diags(r, diags)$ in
$diags2 \cup Prova\_Goal\_PT(regras, conhec\_estr, goal,$
$teoria, subteoria, regras\_goal\text{-}r)$
else $\{\}$

A função acima tenta provar um objetivo em profundidade e regressivamente (para Trás). Ela busca, então, cada regra que chega ao objetivo desejado e tenta provar sua condição através da função *Prova_Cond_PT*. O resultado da prova da condição é submetido à função *Atu_Diags* que calcula o *trace* e o *fator de certeza* resultantes da aplicação da regra.

99. $Prova\_Cond\_PT : SUBTEORIA \times CONHEC\_ESTRUT \times ASSERCAO \times$
$\times AREA\_GER \times NIVEL \rightarrow setof\, DIAGNOSE$

99. $Prova\_Cond\_PT(regras, conhec\_estr, goal,$
$teoria, subteoria) \triangleq$
if $is\text{-}COND\_ELEMENTAR(goal)$
then $Prova\_Cond\_Elementar\_PT(regras,$
$conhec\_estr, goal, teoria, subteoria)$
else if $is\text{-}COND\_COMPOSTA(goal)$
then $Prova\_Cond\_Composta\_PT(regras,$
$conhec\_estr, goal, teoria, subteoria)$
else if $\exists r \in regras \cdot r = mk\text{-}REGRA(\_, \_, goal, \_)$
then $Prova\_Goal\_PT(regras, conhec\_estr, goal,$
$teoria, subteoria, regras)$
else $mk\text{-}DIAGNOSE(goal, \text{conc}\,(goal, \text{nao provado}), 0)$

Esta função prova as condições submetidas utilizando encadeamento em profundidade e regressivo. As condições podem ser elementares ou compostas (como especificadas anteriormente), ou podem ainda referir ações (*goals*) de outras regras evidenciando a necessidade de provar estas regras (o que é feito pela chamada de *Prova_Goal_PT*), realizando o encadeamento das regras.

100. $Prova\_Cond\_Elementar\_PT : SUBTEORIA \times CONHEC\_ESTRUT \times$
$\times ASSERCAO \times AREA\_GER \times$
$\times NIVEL \rightarrow DIAGNOSE$-set

100. $Prova\_Cond\_Elementar\_PT(regras,$
$conhec\_estr, goal, teoria, subteoria) \triangleq$
let $l\_alarm = Det\_Get\_L\_Alarmes(teoria, subteoria)$ in
if $goal \in l\_alarm$
then $diagnoses' = mk\text{-}DIAGNOSE(goal,$ conc $(goal,$ é um alarme, $1)$
else if $goal \in conhec\_estr$
then $mk\text{-}DIAGNOSE(goal,$ conc $(goal,$ é conhecido, $1)$
else if $goal = NO(goal1)$
then let $diags = Prova\_cond(regras, conhec\_estr,$
$goal1, teoria, subteoria)$ in
$Nega(diags)$
else $mk\text{-}DIAGNOSE(goal,$ conc $(goal,$ não resolvido, $0)$

A função *Prova_Cond_Elementar* verifica se o *goal* a ser provado é um alarme ou faz parte do conhecimento estrutural sobre a rede. Por condição elementar entende-se também a negação de uma condição qualquer.

Note-se que as *regras* podem se referir a dados pertencentes ao Submódulo de Detecção. Para separar bem estes submódulos todos os acessos a dados de Detecção feitos na fase de Diagnose são realizados via chamadas de funções especiais que implementam uma interface entre os módulos. Esta estrutura é análoga à adotada no modelo de gerência OSI, onde o processo Gerente (Submódulos de Diagnose e Correção) requisita dados da MIB (estruturas do Submódulo de Detecção) através de primitivas de interação (funções de interface).

101. $Nega : DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$

101. $Nega(diags) \triangleq$

if $diags = \{\}$

then $\{\}$

else let $d = mk\text{-}DIAGNOSE(g, t, fc) \in diags$ in

$mk\text{-}DIAGNOSE(not(g), \text{conc}\,(not(g), \text{não}, t), 1\text{-}fc)$

$\cup\, Nega(diags\text{-}d)$

A função acima nega um conjunto de diagnósticos submetidos. A negação de um diagnóstico é feita substituindo-se o *fator de certeza* pelo seu complemento de um, e negando-se o *trace*.

102. $Prova\_Cond\_Composta\_PT : SUBTEORIA \times CONHEC\_ESTRUT \times$
$\times ASSERCAO \times AREA\_GER \times$
$\times NIVEL \rightarrow DIAGNOSE$-set

102. $Prova\_Cond\_Composta\_PT(regras,$
$conhec\_estr, goal, teoria, subteoria) \triangleq$
let $mk\text{-}COND\_COMPOSTA(c1, op, c2)$ in
let $diags1 = Prova\_Cond\_PT(regras, conhec\_estr, c1,$
$teoria, subteoria)$ in
let $diags2 = Prova\_Cond\_PT(regras, conhec\_estr, c2,$
$teoria, subteoria)$ in
if $op = \wedge$
then $Combina(and, diags1, diags2)$
else $Combina(or, diags1, diags2)$

A função *Prova_Cond_Composta* tem como objetivo provar condições envolvendo operadores boleanos de aridade 2. Isto é feito seguindo encadeamento em profundidade e regressivo.

```
103. Combina : DIAGNOSE-set × DIAGNOSE-set → DIAGNOSE-set
103. Combina(op, d1, d2) ≜
     if d1 = { }
     then { }
     else let d ∈ d1 in
          let diags = Combina2(op, d, d2) in
          diags ∪ Combina(op, d1-d, d2)
```

A função acima tem o mesmo objetivo da função *Nega*, diferindo no tipo de operador. *Nega* modifica um conjunto de diagnósticos devido à ação de uma negação, *Combina* combina dois conjuntos de diagnósticos devido à ação de um *AND* ou *OR*.

104. $Combina2 : DIAGNOSE \times DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$

104. $Combina2(op, d, ds) \triangleq$

if $ds = \{\}$

then $diagnose' = \{\}$

else let $d2 \in ds$ in

let $d = mk\text{-}DIAGNOSE(g1, t1, fc1)$ in

let $d2 = mk\text{-}DIAGNOSE(g2, t2, fc2)$ in

if $op = AND$

then $mk\text{-}DIAGNOSE(g1\ and\ g2, \text{conc}\,(t1, \text{e}, t2), fc1 * fc2)$

$\cup\ Combina2(op, d, ds\text{-}d2)$

else if $fc1 > fc2$

then $mk\text{-}DIAGNOSE(g1\ or\ g2, \text{conc}\,(t1, \text{ou}, t2), fc1)$

$\cup\ Combina2(op, d, ds\text{-}d2)$

else $mk\text{-}DIAGNOSE(g1\ or\ g2, \text{conc}\,(t1, \text{ou}, t2), fc2)$

$\cup\ Combina2(op, d, ds\text{-}d2)$

A função *Combina*2 auxilia *Combina* realizando a modificação dos *traces* e computando os fatores de certeza.

105. $Atu\_Diags : REGRA \times DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$

105. $Atu\_Diags(r, diags) \triangleq$

if $diags = \{\}$

then $\{\}$

else let $r = mk\text{-}REGRA(n, cond, acao, fcr)$ in

let $diag = mk\text{-}DIAGNOSE(g, t, fcd) \in diags$ in

$mk\text{-}DIAGNOSE(acao,$ conc $(acao,$ provado pela regra, $n,$

onde, $t), fcr * fcd) \cup Atu\_Diags(r, diags\text{-}diag)$

Esta função calcula o *trace e o fator de certeza* resultante da aplicação de uma *regra* na formação de um diagnóstico.

106. $Diag\_inter\text{-}relacionamento : BANCO\_DE\_CONHEC \times AREA\_GER \times$

$\times DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$

106. $Diag\_inter\text{-}relacionamento(bco\_de\_conhec, area, diags) \triangleq$

let $mk\text{-}BANCO\_DE\_CONHEC(teorias, conhec\_estr) = bco\_de\_conhec$ in

let $mk\text{-}TEORIA(niv\_sub, inter\text{-}relacoes) = teorias(area)$ in

$Infere\_Amplit\_Frente(regras, regras, conhec\_estr, diags, <>)$

Após realizadas as inferências sobre as *subteorias* dos diferentes níveis pertencentes a uma mesma *teoria* (área de gerenciamento) deve-se fazer o inter-

relacionamento dos diagnósticos parciais obtidos. Este trabalho é realizado através da função *Diag_interrelacionamento* que deve validar, cancelar ou combinar as diagnoses obtidas nas subteorias.

107. $Infere\_Amplit\_Frente : REGRAS\text{-set} \times REGRAS\text{-set} \times$
$\times CONHEC\_ESTRUT \times DIAGNOSE\text{-set} \times$
$\times DIAGNOSE\text{-set} \rightarrow DIAGNOSE\text{-set}$

107. $Infere\_Amplit\_Frente(regras, regras\_disp,$
$conhec\_estr, diagnoses, diags\_finais) \triangleq$
if $regras\_disp = \{\}$
then $diagnoses' = diags\_finais$
else let $r = mk\text{-}REGRA(n, cond, acao, fc) \in regras\_disp$ in
let $mk\text{-}DIAGNOSE(g, t, fc1) =$
$Prova\_Cond\_AF(cond, diagnoses, conhec\_estr)$ in
if $g = t = fc1 =$ "-"
then $Infere\_Amplit\_Frente(regras, regras\_disp\text{-}r,$
$diagnoses, diags\_finais)$
else let $diag = mk\text{-}DIAGNOSE(acao,$ conc $(acao,$ derivado pela regra,
$r,$ onde, $t), fc * fc1)$ in
if $is\text{-}ASSERCAO(acao)$
then $Infere\_Amplit\_Frente(regras, regras\_disp\text{-}r,$
$diagnoses, diags\_finais \cup diag)$
else $Infere\_Amplit\_Frente(regras, regras\_disp\text{-}r,$
$diagnoses \cup diag, diags\_finais)$

Para realizar o inter-relacionamento citado utiliza-se de inferência em amplitude e para frente (ou progressiva). Este tipo de inferência foi escolhido pois se adequa melhor à tarefa em questão [WEK84] [HAK88]. Uma vez que os diagnósticos serão validados, cancelados ou combinados, o número de resultados finais possíveis é maior que as diagnoses parciais de entrada.

Na inferência em amplitude para frente são disparadas todas as regras possíveis, os resultados destas regras são adicionados aos subsídios já existentes para alimentar a inferência, as regras já disparadas são separadas para que não o sejam novamente, repete-se o processo com as demais regras.

O processo pára quando não existem mais regras disparáveis com os subsídios disponíveis. Os diagnósticos finais gerados são encontrados entre os subsídios (também chamados memória de trabalho).

108. $Prova\_Cond\_AF : CONDICAO \times DIAGNOSE\text{-set} \times CONHEC\_ESTRUT \rightarrow DIAGNOSE$

108. $Prova\_Cond\_AF(cond, diags, conhec\_estr) \triangleq$
if $is\text{-}COND\_ELEMENTAR(cond)$
then $Prova\_Cond\_Elementar\_AF(cond, diags, conhec\_estr)$
else $Prova\_Cond\_Composta\_AF(cond, diags, conhec\_estr)$

A função *Prova_Cond_AF* verifica a estrutura da *cond*ição e dispara a função de prova adequada.

109. $Prova\_Cond\_Elementar\_AF : COND\_ELEMENTAR \times DIAGNOSE\text{-set} \times$
$\times CONHEC\_ESTRUT$
$\rightarrow DIAGNOSE$

109. $Prova\_Cond\_Elementar\_AF(cond, diags, conhec\_estr) \triangleq$

if $cond \in conhec\_estr$

then $mk\text{-}DIAGNOSE(cond, \text{conc}(cond, \text{é conhecido}), 1)$

else if $\exists d \cdot d = mk\text{-}DIAGNOSE(cond, t, fc1) \in diags$

then d

else if $cond = NO(cond1)$

then let $mk\text{-}DIAGNOSE(g\_n, t\_n, fc\_n) =$
$Prova\_Cond\_AF(cond1, diags, conhec\_estr)$ in

if $t\_n = g\_n = fc\_n =$ "-"

then $mk\text{-}DIAGNOSE(\text{-}, \text{-}, \text{-})$

else $mk\text{-}DIAGNOSE(no(g\_n), \text{conc}(\text{não}, t), 1\text{-}fc\_n)$

else $mk\text{-}DIAGNOSE(\text{-}, \text{-}, \text{-})$

Prova_Cond_Elementar realiza a prova de condições elementares, ou seja, verifica afirmações sobre o *conhecimento estrutural* bem como verifica se determinadas condições não estão provadas no conjunto de diagnósticos parciais submetido. O tratamento de negações também se dá por esta função.

110. $Prova\_Cond\_Composta\_AF : COND\_COMPOSTA \times DIAGNOSE\text{-set} \times$
$\times CONHEC\_ESTRUT \rightarrow DIAGNOSE$

110. $Prova\_Cond\_Composta\_AF(cond, diags, conhec\_estr) \triangleq$
let $mk\text{-}COND\_COMPOSTA(c1, op, c2)$ in
let $diags1 = Prova\_Cond\_AF(c1, diags, conhec\_estr)$ in
let $diags2 = Prova\_Cond\_AF(c2, diags, conhec\_estr)$ in
if $op = \wedge$
then $Combina(and, diags1, diags2)$
else $Combina(or, diags1, diags2)$

A função acima trata condições com operadores booleanos de aridade 2 (ou seja: $AND's$ e $OR's$), em amplitude e progressivamente, combinando os resultados obtidos nas provas das condições parciais.

4.4.4 Submódulo de Correção

O Submódulo de Correção recebe diagnósticos de problemas do Submódulo de Diagnose e retorna procedimentos de correção para estes problemas. Estas correções são, então, adicionadas a um *log* de problemas e correções da instalação que ficará a disposição do Gerente da Rede.

Este Submódulo não é complexo pois o Sistema é passivo, isto é, não gera mensagens de gerenciamento e sim "aconselha" ou "adverte" o gerente da rede no caso de situações anormais, conforme as áreas de gerenciamento definidas no contexto OSI de gerenciamento de redes.

A seguir está a especificação comentada para este Submódulo.

4.4.4.1 Domínios Semânticos

Estruturas utilizadas no Submódulo de Correção:

(111)	*CORRECOES*	=	$PROBL \xrightarrow{m} CORRECAO\text{-set}$
(112)	*CORRECAO*	::	$PROCEDIMENTO \times FATOR\_DE\_CERTEZA$
(113)	*PROCEDIMENTO*	=	*STRING*
(114)	*ELEM_LOG*	::	$DIAGNOSE \times CORRECAO\text{-set}$

As estruturas acima significam:

- Por (111) fica especificado que para cada *problema* encontrado durante a diagnose podem existir várias formas de correção;
- Cada *Correção* (112) é composta por um procedimento que a implementa e por um *Fator de Certeza* para o procedimento;
- Um *Procedimento* (113) é um conjunto de ações para orientar ou adverter o gerente em situações anormais;
- Um *Elemento do Log* (114), que será apresentado ao gerente, é composto por um Diagnóstico e o conjunto de possíveis Correções para o problema diagnosticado.

Os Fatores de Certeza, atualmente, não possuem forma alguma de correção dinâmica. Futuramente, pode-se implantar um esquema de ajuste dinâmico destes fatores através de um mecanismo de realimentação ("Feedback Processing" em

[MAR88] por exemplo), possivelmente interno ao Submódulo de Correção. Com isto o sistema pode se adapatar a algumas características particulares da instalação sendo gerenciada.

4.4.4.2 Funções

Funções do Submódulo de Correção:

(115) $Corrige$: $CORRECOES \times DIAGNOSE$-set $\rightarrow$
$\rightarrow ELEM\_LOG$-set $\rightarrow ELEM\_LOG$-set

115. $Corrige$: $CORRECOES \times DIAGNOSE$-set $\rightarrow$
$ELEM\_LOG$-set $\rightarrow ELEM\_LOG$-set

115. $Corrige(correcoes, diags)(el) \triangleq$
if $diags = \{\}$
then el
else let $mk\text{-}DIAGNOSE(probl, trace, fc) = diag \in diags$ in
let $corr = correcoes(probl)$ in
let $el2 = el \cup mk\text{-}ELEM\_LOG(diag, corr)$ in
$Corrige(correcoes, diags\text{-}diag)(el2)$

A função Corrige não apresenta complexidade, parte-se do pressuposto que as várias formas de correção estão já armazenadas e é necessário apenas um trabalho de recuperação destas informações. Submete-se um conjunto de Diagnoses e retorna-se os diversos conjuntos de Correções para estas Diagnoses.

Note-se que o trabalho de relacionamento de problemas e suas correções, codificando os procedimentos a serem executados, bem como a atribuição de fatores de certeza a estes procedimentos fazem parte do trabalho de modelagem de conhecimento feito pelo engenheiro de conhecimento com os subsídios prestados pelo especialista na área.

4.4.5 Submódulo de Tratamento do Usuário

Este Submódulo trata as perguntas do gerente (o usuário do sistema), gerando respostas ao mesmo. As perguntas submetidas podem ser consultas, deleções, alterações ou adições envolvendo as diversas estruturas do domínio semântico do sistema:

- No Submódulo de Interpretação, o gerente tem o poder de agir sobre o conjunto de gramáticas dos protocolos aceitos pelos sistema;
- No Submódulo de Detecção o gerente pode agir sobre as estruturas de contabilização;
- No Submódulo de Diagnose, o gerente o pode agir sobre a base de regras e a base de conhecimento estrutural da instalação;
- O conjunto de Correções também pode ser modificado conforme ordem do gerente.

Enfim, o gerente tem acesso irrestrito sobre os domínios semânticos do sistema. A modificação destas estruturas é responsabilidade do gerente.

4.4.5.1 Domínios Semânticos

Abaixo, as estruturas para permitir tal acesso por parte do gerente.

(116)	*BUFF_USUARIO*	::	*BUFF_PERGUNTA*×
			×*BUFF_RESPOSTA*
(117)	*BUFF_PERGUNTA*	::	*OPERACAO* × *ESTRUTURA*
(118)	*BUFF_RESPOSTA*	::	*STRING*
(119)	*OPERACAO*	=	CONSULTA | CRIACAO |
			| DELECAO
(120)	*ESTRUTURA*	=	*ESTRUT_INTERP* |
			| *ESTRUT_DETECCAO* |
			| *ESTRUT_DIAGNOSE* |
			| *ESTRUT_CORRECAO*
(121)	*ESTRUT_INTERP*	::	*GRAM_TRAFEGO* |
			| *GRAM_ESTATISTICAS*
(122)	*ESTRUT_DETECCAO*	::	*AREA_GER* × *NIVEL*×
			×(*PRED_OC_TRAF* |
			| *PRED_AUS_TRAF* |
			| *CONT* | *HIST*)
(123)	*ESTRUT_DIAGNOSE*	::	*FATO_SOBRE_REDE* |
			| (*AREA_GER* × *NIVEL* × *REGRA*)
(124)	*ESTRUT_CORRECAO*	=	*PROBL* × *CORRECAO*

As estruturas acima denotam:

- *BUFF_USUARIO* (116) é composto por dois *buffers*, um para *perguntas* e outro para *respostas*;
- O *BUFF_PERGUNTA* (117) é o *buffer* através do qual o gerente pode submeter Operações ao sistema;
- O *BUFF_RESPOSTA* (118) é um *stream* de caracteres que contém uma resposta ao usuário, podendo ser os dados de sua consulta ou respostas afirmativas ou negativas, conforme o teor da operação disparada;
- O gerente pode disparar uma *OPERACAO* (119) que será uma *CONSULTA*, uma *CRIACAO* ou uma *DELECAO* sobre os domínios semânticos do sistema;
- Conforme (120), a estrutura sendo consultada/criada/deletada pode ser uma estrutura do submódulo de interpretação, do submódulo de detecção, do submódulo de diagnose, ou do submódulo de correção;
- Para o submódulo de Interpretação (121), podem ser realizadas operações sobre Gramáticas para Tráfego ou para os dados de Estatística, tal como definido naquele submódulo;
- No submódulo de Detecção (122), podem ser consultados/criados/deletados em cada subteoria: os predicados de ocorrência ou ausência, as estruturas de contabilização e os históricos;
- No submódulo de Diagnose (123) pode-se realizar operações sobre os Fatos da Rede ou sobre as Regras de cada subteoria de cada teoria;
- Para o submódulo de Correção (124) pode-se operar sobre as correções para cada problema.

Na especificação do sistema, o Usuário foi modelado como uma entidade paralela ao Módulo de Apoio. A troca de *perguntas e respostas*, tal como definido na especificação em CSP, se dá através de BUFF_USUARIO.

A modelagem da forma de tratamento do usuário através de *buffers* facilita uma futura modificação para possibilitar que o gerente acompanhe o sistema por uma estação remota.

4.4.5.2 Funções

Abaixo as funções principais do Submódulo de Tratamento do Usuário. A estrutura MAG está especificada na fórmula (130).

$$
\begin{array}{lrcl}
(125) & \mathit{Trata\_Usuario} & : & MAG \rightarrow MAG \\
(126) & \mathit{Trata\_Interp} & : & OPERACAO \times \\
 & & & \times ESTRUTURA \times MAG \rightarrow MAG \\
(127) & \mathit{Trata\_Detecta} & : & OPERACAO \times \\
 & & & \times ESTRUTURA \times MAG \rightarrow MAG \\
(128) & \mathit{Trata\_Diagnose} & : & OPERACAO \times \\
 & & & \times ESTRUTURA \times MAG \rightarrow MAG \\
(129) & \mathit{Trata\_Correcao} & : & OPERACAO \times \\
 & & & \times ESTRUTURA \times MAG \rightarrow MAG
\end{array}
$$

A função abaixo é disparada a cada ciclo do sistema. Ela verifica se existe alguma operação submetida ao sistema pelo usuário, quando isto acontece são disparadas funções de tratamento conforme os domínios semânticos sobre os quais as operações agem.

125. $Trata\_Usuario : MAG \rightarrow MAG$

125. $Trata\_Usuario(mag) \triangleq$

let $mk\text{-}MAG(buffers, gram, det, bco\_conhec,$

$corrs, buff\_usuario) = mag$ in

let $mk\text{-}BUFF\_USUARIO(buff\_perg, buff\_resp) = buff\_usuario$ in

if $buff\_perg = <>$

then mag

else let $mk\text{-}BUFF\_PERGUNTA(op, estr) = buff\_perg$ in

if $is\text{-}ESTRUT\_INTERP(estr)$

then $Trata\_Interp(op, estr, mag)$

else if $is\text{-}ESTRUT\_DETECTA(estr)$

then $Trata\_Detecta(op, estr, mag)$

else if $is\text{-}ESTRUT\_DIAGNOSE(estr)$

then $Trata\_Diagnose(op, estr, mag)$

else $Trata\_Correcao(op, estr, mag)$

As operações sobre os domínios semânticos não estão especificadas totalmente pois as facilidades do ambiente de desenvolvimento do sistema podem ser bastante utilizadas neste submódulo. Por exemplo: existência de uma linguagem de banco de dados embutida, ou mesmo utilização de linguagens de maior nível de abstração (PROLOG, LISP) facilitando as consultas/criações/deleções sobre os domínios semânticos.

Abaixo estão os cabeçalhos das funções específicas sobre os domínios semânticos.

126. $Trata\_Interp : OPERACAO \times ESTRUTRA \times MAG \rightarrow MAG$

126. $Trata\_Interp(op, estr, mag) \triangleq$

$/* \quad consulta/delecao/criacao\ sobre\ gramaticas \quad */$

127. $Trata\_Detecta : OPERACAO \times ESTRUTRA \times MAG \rightarrow MAG$

127. $Trata\_Detecta(op, estr, mag) \triangleq$

$/* \quad consulta/delecao/criacao\ sobre\ estruturas\ de\ deteccao \quad */$

128. $Trata\_Diagnose : OPERACAO \times ESTRUTRA \times MAG \rightarrow MAG$

128. $Trata\_Diagnose(op, estr, mag) \triangleq$

$/* \quad consulta/delecao/criacao\ sobre\ regras\ e\ fatos\ da\ rede \quad */$

129. $Trata\_Correcao : OPERACAO \times ESTRUTRA \times MAG \rightarrow MAG$

129. $Trata\_Correcao(op, estr, mag) \triangleq$

$/* \ \ consulta/delecao/criacao \ sobre \ estruturas \ de \ correcao \ \ */$

4.4.6 Escalonamento dos Submódulos

Para que os módulos anteriormente descritos funcionem em conjunto, é necessário um escalonamento cíclico dos mesmos, bem como a passagem da informação atualizada de um para outro. Desta forma teremos uma agregação dos domínios semânticos já especificados em uma estrutura de maior grau de abstração, como segue.

4.4.6.1 Domínios Semânticos

Estruturas de maior grau de abstração, servindo ao escalonamento dos submódulos:

(130) $MAG :: BUFFERS \times GRAMATICAS \times DETECTA \times$
$\times BANCO\_DE\_CONHEC \times CORRECOES \times$
$\times LOG \times BUFF\_USUARIO$

(131) $LOG = ELEM\_LOG$-set

4.4.6.2 Funções

A seguir as funções de mais alto nível no Módulo MAG, servindo ao escalonamento dos submódulos:

(132) $Inicia : \rightarrow MAG$

(133) $Modulo\_Apoio : MAG \rightarrow MAG$

Detalhando cada função:

132. $Inicia : \rightarrow MAG$

132. $Inicia() \triangleq$

$mag = carga()$

Esta função representa a carga do Módulo de Apoio à Gerência, passando a existir a estrutura MAG, tal como definida.

133. $Modulo\_Apoio : MAG \rightarrow MAG$

133. $Modulo\_Apoio(mk\text{-}MAG(buffers, gramaticas, detecta,$
$bco\_de\_conhec, corrs, log)) \triangleq$
let $t = Busca\_UD(gramaticas, buffers)$ in
let $mk\text{-}TRAFEGO(i, nivel, cvs) = t$ in
let $detecta' = Deteccao(\{Falhas\}, nivel, t)(detecta)$ in
let $d = Diagnostico(i, bco\_de\_conhec, \{Falhas\}, \{1,2,3,4,5,6,7\})$ in
let $c = Correcao(d, corrs)$ in
let $log' = log \cup c$ in
let $mag1 = mk\text{-}MAG(buffers, gramatica, detecta',$
$bco\_de\_conhec, corrs, log')$ in
let $mag2 = Trata\_Usuario(mag1)$ in
$Modulo\_Apoio(mag2)$

Esta função realiza o principal *loop* do Módulo de Apoio à Gerência, escalonando os diversos Submódulos já definidos.

Com isso, completa-se a especificação do sistema. O próximo capítulo aborda a implementação do mesmo, relatando a equivalência entre as estruturas.

5 IMPLEMENTAÇÃO DO PROTÓTIPO DO SISTEMA

5.1 Introdução

A especificação até então apresentada demonstra **O Que** o *software* deve fazer. Para validar esta especificação foi construído um protótipo para o sistema. Este capítulo descreve a implementação do protótipo, ressaltando **Como** o *software* realiza as funções especificadas. Para isso, o capítulo está dividido da seguinte forma:

- uma descrição do ambiente em que ocorreu a prototipação, os outros equipamentos da rede, softwares utilizados, etc;
- estruturação do "Módulo de Apoio à Gerência" e seus submódulos, escritos em PROLOG;

A implementação do módulo *Driver* não será abordada nesta dissertação pois traria muitos detalhes que fogem do objetivo principal do trabalho. Como o Driver foi construído com a utilização de "Packet Drivers", e sendo esta construção uma das primeiras experiências no CPGCC-UFRGS no desenvolvimento de *software* sobre esta interface, foi gerado um relatório de pesquisa descrevendo a experiência [DOT92].

5.2 Ambiente de Prototipação

5.2.1 Equipamentos

A prototipação foi realizada sobre equipamento da linha PC, ligado à rede local do CPD-UFRGS. A rede local durante a prototipação estava configurada conforme a figura 5.1.

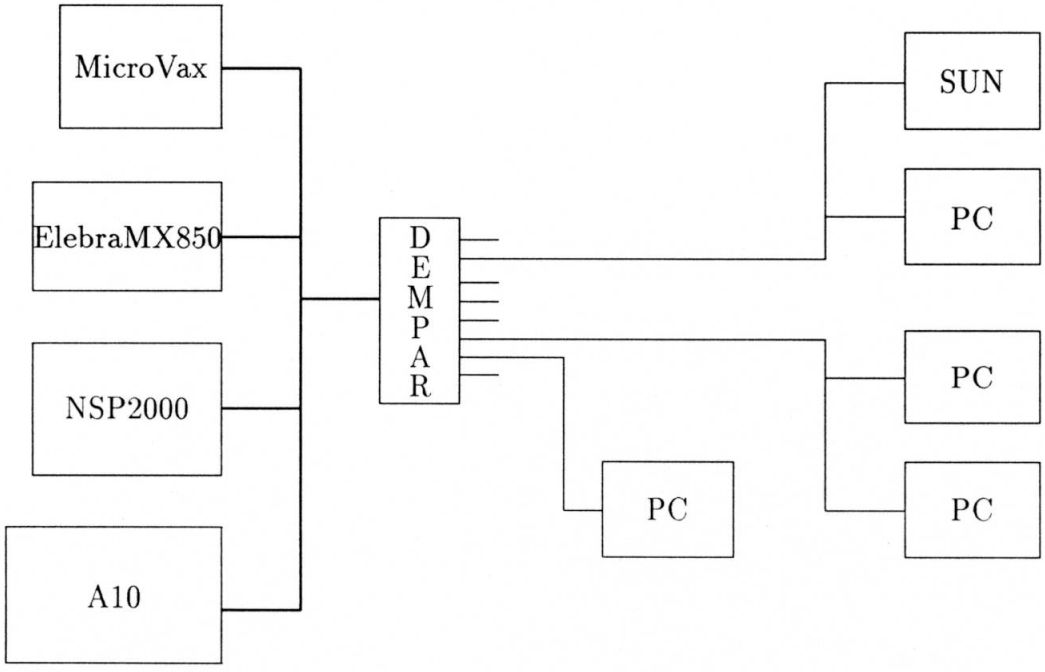

Figura 5.1: Rede local no CPD-UFRGS

5.2.2 Estrutura de Comunicação

5.2.2.1 Nível Físico

O segmento de rede unindo os equipamentos A10, NSP2000, Elebra MX-850, e o MicroVAX, segue as especificações IEEE 802.3 10BASE5 [MIL89]. Características principais:

- Tamanho máximo por segmento de cabo coaxial de 500 metros, com velocidade de propagação mínima de 0.77c (onde "c" é a velocidade da luz no vácuo - 300.000 metros por segundo)
- 10 Mbps;
- Máximo de 5 segmentos (4 repetidores) entre duas estações;
- Distância mínima de 2.5 metros entre os nodos;
- Máximo de 100 nodos por segmento;
- Conectores do tipo "N";
- *Transceiver* externo, com distância máxima até o nodo de 50 metros;
- Uso de codificação Manchester.

Os demais segmentos usam a especificação IEEE 802.3 10BASE2 [MIL89]. Características principais:

- Tamanho máximo por segmento de cabo coaxial de 185 metros, com velocidade de propagação mínima de 0.65c (onde "c" é a velocidade da luz no vácuo)
- 10 Mbps;
- Máximo de 5 segmentos (4 repetidores) entre duas estações;

- Distância mínima de 0.6 metros entre os nodos;
- Máximo de 30 nodos por segmento;
- Conectores do tipo BNC "T";
- *Transceiver* interno;
- Uso de codificação Manchester.

O equipamento "DEMPAR" é um repetidor multiportas (8 portas) que copia o sinal de cada segmento para os demais.

5.2.2.2 Nível de Enlace

O nível de enlace da rede segue o projeto IEEE 802, sendo subdividido em MAC *(Medium Access Control)* e LLC *(Logical Link Control).*

A subcamada MAC realiza o controle de acesso ao meio. Este controle está relacionado com o tipo de canal utilizado e topologia da rede, para o caso da rede em barra do CPD-UFRGS, utiliza-se a política CSMA/CD *Carrier Sense Multiple Access with Collision Detection.* Informações complementares (formato dos *frames*, forma de endereçamento, etc) para esta subcamada podem ser encontrados em [TAN88] e [SOA86].

A subcamada LLC proporciona multiplexação sobre a ligação de dados fornecendo DSAPs e SSAPs (Destination e Source Service Access Points, respectivamente). Esta camada proporciona uma interface comum com os níveis superiores eliminando as diferenças entre MACs. Provê controle de fluxo, controle de erros, confirmação, serviço de datagrama e orientado a conexão. Maiores detalhes (formato do *frame*, opções de controle, endereçamento de SAPs, etc) podem ser encontrados também em [TAN88] e [SOA86].

5.2.2.3 Nível de Rede

A partir do nível de rede começam a existir vários protocolos que podem trafegar sobre um *frame* LLC. Para a rede em questão podem ocorrer *frames* BNA *(Burroughs Network Architeture)*, DECnet, IP e ICMP.

Frames BNA podem acontecer entre o A10 e o *front-end* NSP2000, pois o A10 não possui o protocolo IP e para que este possa se comunicar com equipamentos que utilizam IP é necessária uma conversão por parte do NSP2000. Neste caso a comunicação A10-NSP2000 é feita utilizando-se *frames* BNA.

Os *frames* DECnet são utilizados pelo Elebra-MX850 e pelo MicroVax.

Todos os equipamentos, com exceção do A10, podem se comunicar usando o IP (*Internet Protocol* - [DAR81a]) diretamente.

5.2.2.4 Nível de Transporte

Neste nível podem ocorrer, novamente, os *frames* proprietários DECnet e BNA para o nível 4, *frames* TCP [DAR81b] e UDP. Todos os equipamentos suportam o protocolo TCP.

5.2.2.5 Nível de Aplicação

Os aplicativos mais utilizados na rede são os serviços ARPA: TELNET (terminal remoto - [PRE83]), FTP (*File Transfer Protocol* - [DAR81b]), e o SMTP *(Simple Mail Transfer Protocol)*, muito difundidos nas redes suportando TCP/IP. O serviço NFS (*Network File System* - [SMI89]) também pode ser oferecido.

A arquitetura INTERNET não possui níveis de Sessão e Apresentação tal como na estrutura OSI/ISO.

5.2.3 Ambientes de Desenvolvimento de Software

Para o desenvolvimento do software foram utilizadas duas linguagens, com propósitos diferentes.

Para a captura dos dados de tráfego foi construído o Módulo *Driver*, envolvendo características de baixo nível, para isso foi utilizada a linguagem "C" (compilador Turbo C da Borland Inc.).

O tratamento das informações capturadas, ou seja, o Módulo de Apoio à Gerência, agregando o conhecimento até a fase de Correção para o usuário, foi construído utilizando-se PROLOG, com o interpretador e compilador ARITY (Arity Corporation).

5.3 O Módulo de Apoio à Gerência

Esta seção detalha o *software* construído no MAG - Módulo de Apoio à Gerência.

O MAG, conforme sua especificação está dividido nos submódulos:

- Submódulo de Interpretação
- Submódulo de Detecção
- Submódulo de Diagnose
- Submódulo de Correção
- Submódulo de Tratamento do Usuário

Nesta seção são feitas considerações sobre a implementação de cada submódulo. Atenção maior é dada ao conhecimento modelado nas diversas fases do

protótipo. O *software* que usa estas estruturas e o formato dos dados de entrada de cada fase estão especificados no capítulo anterior e sua apresentação neste capítulo seria uma repetição.

5.3.1 Submódulo de Interpretação

As estruturas de reconhecimento para os diversos protocolos são implementadas através de diferentes predicados em PROLOG. Quando estes predicados são disparados, fornecendo os dados "brutos" de entrada vindos dos *buffers* *BUF_TRAFEGO1* e *BUF_ESTATISTICAS*, correspondem ao disparo das funções *Interpreta_Traf* (30) e *Interpreta_Est* (31) em VDM.

Cada predicado reconhece os dados relativos a um protocolo de um determinado nível, gerando uma representação interna que servirá ao submódulo de Detecção.

Para tratar o aninhamento de protocolos, o predicado que reconhece um protocolo de nível N-1 dispara, durante seu corpo de comandos, um predicado de reconhecimento para os protocolos de nível N. Da mesma forma, o predicado de reconhecimento de nível N dispara o reconhecimento do protocolo de nível N+1.

Se houver falha durante o reconhecimento dos protocolos, o procedimento de *backtraking* é garantido pelo PROLOG.

Como exemplo, tome-se o processo de separação dos dados dos protocolos dos níveis 2, 3 e 4.

A interpretação do tráfego é disparada da seguinte forma:

```
interpreta_trafego([I,Tam|Dados]) :-
                    interpreta_ud(n2,I,Dados,Dout).
```

No nível 2, as informações disponíveis são os endereços, seguidos dos dados de nível 3:

```
interpreta_ud(n2,I,[Tam|Din],Dout) :-
            tira(6,Dstadd,Din,D1),
            tira(6,Srcadd,D1,D2),
            interpreta_ud(n3,I,[Srcadd,Dstadd|D2],Dout),
                        /* chamada nivel mais alto */
            asserta(buffer(
                    trafego(I,
                            n_2_1,
                            [ (tampdu, Tam),
                              (dstadd, Dstadd),
                              (srcadd, Srcadd) ] ))).
                        /* enddst - endereco destino da mensagem
                           endsrc - endereco fonte da mensagem
                        */
```

Abaixo o reconhecimento do protocolo de nível 3 IP, gerando a representação interna com suas informações:

```
interpreta_ud(n3,I,[Srcadd_n2,Dstadd_n2,8,0|Din],Dout) :-
              tira(1,[B],Din,D1),
                 Versio is (B>>4),
                 Hdleng is (B/\15),
              tira(1,Tpserv,D1,D2),
              tira(2,Totlen,D2,D3),
              tira(2,Identi,D3,D4),
              tira(2,[C,Fof2],D4,D5),
                 Df1 is (C>>6), Dntfrg is (Df1/\1),
```

```
   Mf1 is (C>>5), Morfrg is (Mf1/\1),
   Fof1 is (C/\62),
tira(1,Tmlive,D5,D6),
tira(1,[Protoc],D6,D7),
tira(2,Hdchck,D7,D8),
tira(4,Srcadd,D8,D9),
tira(4,Dstadd,D9,D10),
   Tamopt is ((Hdleng-5)*4),
tira(Tamopt,Option,D10,D11),
   ([H|T] = D11,
(interpreta_ud(n4,I,[Srcadd,Dstadd,Protoc,H|T],Dout)
   /* chamada de nivel mais alto */
    ;
    true),
asserta(buffer(trafego(I,
                n_3_1,
                [ (srcadd_n2, Srcadd_n2),
                  (dstadd_n2, Dstadd_n2),
                  (versio, [Versio]),
                  (hdleng, [Hdleng]),
                  (tpserv, Tpserv),
                  (totlen, Totlen),
                  (identi, Identi),
                  (dntfrg, [Dntfrg]),
                  (morfrg, [Morfrg]),
                  (fragof, [Fof1,Fof2]),
                  (tmlive, Tmlive),
                  (protoc, [Protoc]),
                  (hdchck, Hdchck),
                  (srcadd, Srcadd),
```

```
                    (dstadd, Dstadd),
                    (option, Option)   ] ))).
```

O reconhecimento do protocolo de nível 4 TCP é demonstrado no predicado abaixo.

```
interpreta_ud(n4,I,[Srcadd_n3,Dstadd_n3,6|Din],Dout) :-
            tira(2,Srcprt,Din,D1),
            tira(2,Dstprt,D1,D2),
            tira(4,Seqnum,D2,D3),
            tira(4,Pigack,D3,D4),
            tira(1,[B],D4,D5),
                 Hdleng is (B>>4),
            tira(1,[Flags],D5,D6),
                 Urgent is ((Flags>>5)/\1),
                 Acknow is ((Flags>>4)/\1),
                 Eofmsg is ((Flags>>3)/\1),
                 Reset  is ((Flags>>2)/\1),
                 Syn    is ((Flags>>1)/\1),
                 Fin    is (Flags/\1),
            tira(2,[W1,W2],D6,D7),
                 Window is ((W1 << 8) + W2),
            tira(2,Chksum,D7,D8),
            tira(2,Urgptr,D8,D9),
                 Tamopt is ((Hdleng-5)*4),
            tira(Tamopt,Option,D9,D10),
            interpreta_ud(n7,I,D10,Dout),
                 /* chamada de nivel mais alto */
            asserta(
            buffer(trafego(I,
                           n_4_1,
```

```
[ (i, I),
  (srcadd_n3, Srcadd_n3),
  (srcprt, Srcprt),
  (dstadd_n3, Dstadd_n3),
  (dstprt, Dstprt),
  (seqnum, Seqnum),
  (pigack, Pigack),
  (hdleng, [Hdleng]),
  (urgent, [Urgent]),
  (acknow, [Acknow]),
  (eofmsg, [Eofmsg]),
  (reset,  [Reset]),
  (syn,    [Syn]),
  (fin,    [Fin]),
  (window, Window),
  (chksum, Chksum),
  (urgptr, Urgptr),
  (option, Option)  ] ))).
```

Deve-se observar que podem existir mais de um protocolo para o mesmo nível, cada protocolo é tratado por um predicado de interpretação distinto. Tome como exemplo o tratamento para o protocolo ARP *(Address Resolution Protocol)*, utilizado antes do IP nas comunicações.

```
interpreta_ud(n3,I,[Srcadd_n2,Dstadd_n2,8,6|Din],Dout) :-
        tira(2,Hwadsp,Din,D1),
        tira(2,Ptadsp,D1,D2),
        tira(1,[Hwadln],D2,D3),
        tira(1,[Ptadln],D3,D4),
        tira(2,Opcode,D4,D5),
        tira(Hwadln,Hwsrad,D5,D6),
```

```
        tira(Ptadln,Ptsrad,D6,D7),
        tira(Hwadln,Hwdsad,D7,D8),
        tira(Ptadln,Ptdsad,D8,Dout),
        asserta(buffer(trafego(I,
                               n_3_3,
                               [ (hwadsp, Hwadsp),
                                 (ptadsp, Ptadsp),
                                 (hwadln, [Hwadln]),
                                 (ptadln, [Ptadln]),
                                 (opcode, Opcode),
                                 (hwsrad, Hwsrad),
                                 (ptsrad, Ptsrad),
                                 (hwdsad, Hwdsad),
                                 (ptdsad, Ptdsad),
                                 (resto,  Dout)    ] ))).
```

Conforme as *estatísticas* supridas pelo dispositivo físico, tem-se também um predicado para interpretar estes dados. Nesta implementação, o predicado é o seguinte:

```
interpreta_estatistica([I,Tam|D]) :-
        list_int4(D, Int1,D1),
        list_int4(D1,Int2,D2),
        list_int4(D2,Int3,D3),
        list_int4(D3,Int4,D4),
        list_int4(D4,Int5,D5),
        list_int4(D5,Int6,D6),
        assertz(buffer
                (estatistica(I,
                             n_1_1,
                             [ (i, I),
```

```
                    (pktsin, Int1),
                    (bytsin, Int2),
                    (errsin, Int3),
                    (pktsdr, Int4),
                    (pktsrb, Int5),
                    (pktspb, Int6) ] ))).

/* pktsin - pacotes recebidos no HW
   bytsin - bytes recebidos no HW
   errsin - pacotes errados recebidos
   pktsdr - pacotes nao capturados
   pktsrb - pacotes recebidos pelo buffer
   pktspb - pacotes perdidos pelo buffer
            circular (sem espaco)
*/
```

A função auxiliar *list_int4* pega os quatro primeiros octetos da lista (primeiro parâmetro) e transforma em um inteiro que é devolvido no segundo parâmetro, o terceiro parâmetro retorna o restante da lista.

A função auxiliar *tira* tira de uma lista (segundo parâmetro) a quantidade de elementos indicada no primeiro parâmetro. Estes elementos são devolvidos na mesma ordem em uma lista (terceiro parâmetro) e o restante da lista inicial é devolvida no quarto parâmetro.

Desta forma foi implementada a interpretação das unidades de dados, gerando a representação interna que alimenta o processo de detecção.

5.3.2 Submódulo de Detecção

Após desencapsuladas as unidades de dados, submete-se a representação interna gerada ao teste de uma série de predicados, estes testes são contabilizados e quando há a necessidade de gerar um alarme, este é documentado em um histórico. Estas são as três estruturas principais sobre as quais este módulo age (predicados, contabilizações e históricos).

A seguir, um exemplo comentado destas estruturas para gerar alarmes (em PROLOG):

```
pred_oc_traf( falhas,n_2_1,Cvs,
              ocorr(falhas,n_2_1,msg_menor_de(E))) :-
                  esta_em((srcadd,E),Cvs),
                  esta_em((tampdu,T),Cvs),
                  variavel_deteccao((tam_min_pdu,Tam)),
                  T < Tam.
default(contab( ocorr(falhas,n_2_1,msg_menor_de(E)),
                [],     /* lista de ocorrencias */
                10,     /* tamanho maximo da lista */
                210,    /* tempo maximo de uma
                           ocorrencia na lista */
                200,    /* intervalo de tempo para analise
                           de ocorrencias onde */
                5,      /* sendo encontradas 5 ocorrencias
                           do evento e' disparado o alarme
                           abaixo */
                (falhas,n_2_1,msg_menor_de(E),5)
                       /* alarme da teoria falhas,
                           subteoria de nivel 2,
                           indicando que a estacao "E"
```

```
                        emitiu mensagem menor que o
                        especificado,
                        tempo de validade do alarme e'
                        5 segundos */
        )       ).
default( hist( (falhas,n_2_1,msg_menor_de(E),TTL),
                [],     /* lista de instantes em que o
                           alarme foi disparado */
                50      /* tamanho maximo da lista */
        )       ).
```

O *pred_oc_traf* verifica se uma mensagem tem tamanho menor que o especificado, descobrindo o endereço fonte da mensagem. Se isto acontecer é disparada a contabilização do evento *ocorr(falhas,n_2_1,msg_menor_de(E))*, significando uma contabilização da teoria "falhas" e subteoria de nível 2, colocando-se o instante de ocorrência na lista. No momento em que se adiciona um elemento à lista de ocorrências, é disparada sua manutenção segundo o tamanho máximo e tempo máximo de permanência. Após o procedimento de manutenção verifica-se se o limite de ocorrências de eventos no intervalo de análise é excedido e, em caso positivo, é gerado o alarme que fica ativo durante um período de tempo determinado. O instante de disparo do alarme é documentado na lista do histórico, que possui um tamanho máximo.

Note que tanto as condições dos predicados quanto os parâmetros das estruturas de contabilização e histórico são definidas numa tarefa de engenharia do conhecimento de um especialista.

Existem várias formas de se tratar a questão da "validade" dos alarmes. Alguns sistemas tornam o alarme inativo depois de sua utilização em um diagnóstico, não servindo a outros diagnósticos. Outros sistemas contam com formas elaboradas de tratamento onde os alarmes apresentam dependências entre si (se um alarme está

ativo outros não estarão, e vice-versa). Neste sistema os alarmes são tratados de uma forma bastante simples. Cada alarme tem um tempo de validade determinado no módulo de Detecção, ao final deste tempo o alarme é tornado inativo. O módulo de Diagnose simplesmente é alimentado com a existência ou não do alarme, não exercendo tarefa alguma de manutenção sobre os mesmos.

Outros exemplos:

```
pred_oc_traf( falhas,n_2_1,Cvs, ocorr(falhas,n_2_1,broadcast)) :-
              esta_em((dstadd,[255,255,255,255,255,255]),Cvs).
contab( ocorr(falhas,n_2_1,broadcast),
                []),
                200, 130, 100, 150,
                (falhas,n_2_1,broadcasts_excessivos,5)
      ).
hist( (falhas,n_2_1,broadcasts_excessivos,TTL),
       [], 6
     ).


pred_oc_traf( falhas,n_3_2,Cvs,
              ocorr(falhas,n_3_2,msg_arp) ).
contab( ocorr(falhas,n_3_2,msg_arp),
                []),
                200, 100, 80, 150,
                (falhas,n_3_1,arps_excessivos,5)   ).
hist( (falhas,n_3_1,arps_excessivos,TTL),
       [], 20    ).


pred_oc_traf( falhas,n_3_3,Cvs,
```

```
                ocorr(falhas,n_3_3,end_unreach(Da,Sa,Rot)) ) :-
                esta_em((tipo,3),Cvs),
                esta_em((codigo,C),Cvs),
                (C=0; C=1; C=3),
                esta_em((srcadd_n3,Rot),Cvs),
                esta_em((srcadd_msg,Sa),Cvs),
                esta_em((dstadd_msg,Da),Cvs).
default( contab( ocorr(falhas,n_3_3,end_unreach(Da,Sa,R)),
                  [],
                  10, 100, 60, 2,
                  (falhas,n_3_1,end_unreach(Da,Sa,R),10)
        )        ).
default( hist( (falhas,n_3_1,end_unreach(Da,Sa,R),TTL),
                  [], 6
        )      ).


pred_oc_traf( falhas,n_3_3,Cvs,
                ocorr(falhas,n_3_3,sem_recursos(Da) )) :-
                esta_em((tipo,4),Cvs),
                esta_em((dstadd_msg,Da),Cvs).
default( contab( ocorr(falhas,n_3_3,sem_recursos(Da)),
                  [],
                  10, 100, 60, 5,
                  (falhas,n_3_1,sem_recursos(Da),10)
        )        ).
default( hist( (falhas,n_3_1,sem_recursos(Da),TTL),
                  [], 6
        )      ).
```

Desta forma os alarmes são disparados alimentando a fase de Diagnose, a seguir abordada.

5.3.3 Submódulo de Diagnose

Na especificação do sistema foi aconselhado o uso de duas formas de inferência (regressiva em profundidade e progressiva em amplitude) conforme o estágio do processo de diagnose. O protótipo implementa um motor de inferência que trabalha somente em profundidade e regressivamente. A construção de duas formas de inferência fugiria ao objetivo principal do trabalho, tendo em vista que a única perda considerada é em desempenho e que o resultado do processo é o mesmo.

É importante, também, observar que a negação de expressões boleanas, tal como implementada, permite somente a negação de fatos instanciados e não de regras. O problema da negação de regras é um tanto complexo, sendo discutido em [BRA86], que aconselha a solução adotada neste trabalho.

Na forma atual, a avaliação da negação de uma regra terá a forma:

$$\forall \chi \cdot \neg \Phi(\chi) \ ?$$

onde $\Phi(\chi)$ é uma regra envolvendo a variável χ. χ será universalmente quantificado ($\forall \chi \cdot$), o que não é desejado. O correto seria obtermos comportamento para a negação como o da expressão abaixo (usando o quantificador existencial $\exists \chi \cdot$):

$$\exists \chi \cdot \neg \Phi(\chi) \ ?$$

Além das regras e alarmes, a fase de Diagnose faz excessivo uso da base de conhecimento estrutural, em forma de rede semântica.

Abaixo, algums exemplos de regras. A base de conhecimento está exposta em sua totalidade no apêndice A-1.

```
teoria falhas
subteoria n_3_1
regra 25  : if alarme(prot_unreach(Da))
            then goal(diag_parcial(servico_nao_ativo_em(Da)))
            with 1.
```

A estação de endereço *Da* processa mensagem de nível 3 mas não existe processo ou *port* associado ativo para tratamento da mensagem. É gerado um diagnóstico parcial denotando esta situação.

```
teoria falhas
subteoria n_4_1
regra 32  : if oav(conexao(Sa,Sp,Da,Dp),estado,syn_wait(Iant)) and
               exec(instante_anterior(I)) and
               exec(I > (Iant + 10 ))
            then goal(diag_parcial(nao_responde_ped_con(Da,Dp)))
            with  1.
```

O acompanhamento de uma abertura de conexão do nível 4 (TCP) revela que uma estação não está respondendo o pedido de conexão. Também nesta situação é gerado um diagnóstico parcial.

```
teoria falhas
subteoria interrelacionamento
regra 22  : if diag_parcial(nao_responde_ped_con(A,P))  and
               diag_parcial(servico_nao_ativo_em(A))
            then goal(diag_final(nao_resp_nao_ativo(A,P)))
            with 0.8.
```

Combinando os dois diagnósticos parciais, ou seja, se a estação que não responde pedidos de conexão de transporte também notifica que seu serviço de transporte ou *port* de tratamento não está ativo, então sabe-se que a estação não está respondendo pois o serviço não foi instalado ou de alguma forma tornou-se inativo. O diagnóstico final gerado passa para a fase de Correção.

A seguir, exemplos de triplas O/A/V denotando o conhecimento estrutural. Outras triplas podem ser encontradas no apêndice A-1.

```
oav(rede,possui,segmento1).
oav(segmento1,tipo,dez_base_dois).
oav(estacao([170,0,4,0,195,185]),conectada_a,segmento1).
oav(estacao([170,0,4,0,195,185]),equipamento,vortex).
oav(estacao([170,0,4,0,195,185]),localizacao,'cpd - sala de maquinas').
oav(estacao([170,0,4,0,195,185]),endereco_ip,[143,54,1,2]).
oav(estacao([170,0,4,0,195,185]),une,(segmento1,renpac)).
```

Assim, a fase de Diagnose gera os diagnósticos finais que são passados à fase de Correção.

5.3.4 Submódulo de Correção

A partir do diagnóstico final gerado, o submódulo de Correção "aconselha" procedimentos corretivos ao Gerente. Para isso, este submódulo também utiliza do conhecimento estrutural da rede.

O conjunto total de procedimentos de correção está no apêndice A-2. O conhecimento estrutural está no apêndice A-1, juntamente com as regras. Abaixo, um exemplo comentado de procedimentos de correção que este submódulo oferece:

```
correcao(msg_menor_de(E),
```

```
        ['
         A estacao ',E,' esta emitindo
         mensagens menores que o permitido pelo padrao. Os
         parametros de instalacao  de seus servicos de co-
         municacao devem ser modificados.
         Localizacao da estacao: ',L,'
         Equipamento: ',Eq,'
        '],
        0.8) :- oav(estacao(E),localizacao,L),
                oav(estacao(E),equipamento,Eq).
```

O exemplo acima demonstra o procedimento corretivo aconselhado quando o processo de diagnose obtém *msg_menor_de(E)*. *E* é o endereço da estação, a partir deste se obtém sua localização e o tipo de equipamento na base de conhecimento estrutural. O número que segue o procedimento de correção denota o grau de certeza atribuído ao procedimento.

Outros exemplos:

```
correcao(nao_resp_nao_ativo(A,P),
        ['
         A estacao ',A,' nao esta respondendo
         pedidos de conexao do nivel de transporte  (TCP)
         pois seu servico  de transporte nao esta  ativo,
         ou o servico associado ao "port" ',P,'
         requisitado nao esta ativo.
         Verificar a instalacao do servico de transporte.
         Localizacao da estacao: ',L,'
         Equipamento: ',Eq,'
        '],
        0.7) :- oav(estacao(E),endereco_ip,A),
```

```
                oav(estacao(E),localizacao,L),
                oav(estacao(E),equipamento,Eq).


correcao(nao_resp_pela_rota_atual_caminho_alternativo(Da,Sa,Rot,C),
         ['
          O roteador ',Rot,' notifica que
          a estacao ',Da,' nao foi encontrada
          com o atual esquema de roteamento.  O caminho
          ',C,'
          pode ser sugerido como  rota para  a troca de
          mensagens entre ',Da,' e ',Sa,' .
         '],
         0.75).


correcao(nao_resp_estacao_externa_nao_alcancavel(Da,Sa,Rot),
         ['
          O roteador ',Rot,' notifica que
          a estacao ',Da,', externa ao
          dominio de gerenciamento, nao  foi encontrada
          com o atual esquema de roteamento.  O sistema',','
          nao encontrou um caminho alternativo para su-
          gerir como  rota para  a  troca de  mensagens
          entre ',Da,' e ',Sa,' .
         '],
         0.8).

correcao(nao_aceita_por_gargalo_roteador_roteador_alternativo(Sa,Da,E),
         ['
```

```
O roteador ',Sa,' nao esta aceitando
conexoes de transporte por  falta de recursos
para o tratamento  de novas conexoes.        ','
Verificar:
   1. Parametros  de instalacao do roteador.
   2. Eventual transferencia de servicos para
      ',E,'.
Localizacao do roteador: ',L,'
Equipamento: ',Eq,'
Localizacao do roteador alternativo:
          ',L1,'
Equipamento: ',Eq1,'.
'],
0.8) :- oav(estacao(Sa),localizacao,L),
        oav(estacao(Sa),equipamento,Eq).
        oav(estacao(E),localizacao,L1),
        oav(estacao(E),equipamento,Eq1).
```

Desta forma, a partir do diagnóstico final e de eventuais pesquisas à base de conhecimento estrutural, o procedimento de correção é recuperado.

O diagnóstico alcançado, o *trace* do diagnóstico (ou seja o encadeamento de regras/fatos da rede semântica/alarmes para derivar o diagnóstico), o fator de certeza atribuído ao diagnóstico, o procedimento de correção indicado, e o fator de certeza atribuído ao procedimento de correção são gravados em um *log* para o Gerente da Rede.

5.3.5 Exemplos de saídas do sistema

Exemplos do possível conteúdo do *log*:

```
Instante: 20:15:31
Area de Gerenciamento: falhas
   Problema: msg_menor_de([170,0,4,0,195,185])
   pelo processo de inferencia:
     goal(diag_final(msg_menor_de([170,0,4,0,195,185])))
     e' derivado com 1 certeza
     pela regra (16 , interrelacionamento , falhas) de
        goal(diag_parcial(msg_menor_de([170,0,4,0,195,185])))
        e' derivado com 1 certeza
        pela regra (12 , n_2_1 , falhas) de
           msg_menor_de([170,0,4,0,195,185]) e' um alarme

     Correcao apontada com 1 certeza:

          A estacao [143,54,1,2] esta emitindo
          mensagens menores que o permitido pelo padrao. Os
          parametros de instalacao  de seus servicos de co-
          municacao devem ser modificados.
          Localizacao da estacao: cpd - sala de maquinas
          Equipamento: vortex

Instante: 14:10:05
Area de Gerenciamento: falhas
   Problema: trafego_pesado_por_broadcasts_arps
   pelo processo de inferencia:
     goal(diag_final(trafego_pesado_por_broadcasts_arps))
     e' derivado com 0.6 certeza
     pela regra (12 , interrelacionamento , falhas) de
        goal(diag_parcial(trafego_pesado))
        e' derivado com 1 certeza
        pela regra (8 , n_1_1 , falhas) de
```

```
        trafego_pesado e' um alarme
       e
       goal(diag_parcial(broadcasts_excessivos))
       e' derivado com 1 certeza
       pela regra (18 , n_2_1 , falhas) de
        broadcasts_excessivos e' um alarme
      e
      goal(diag_parcial(arps_excessivos))
      e' derivado com 1 certeza
      pela regra (11 , n_3_1 , falhas) de
       arps_excessivos e' um alarme

    Correcao apontada com 0.8 certeza:

        Aviso: A rede apresenta trafego intenso de pacotes tipo
               broadcast devido a pacotes ARP.
               Verificar os parametros do  servico de resolucao
               de enderecos, mais especificamente se as tabelas
               de traducao de enderecos estao sendo mantidas de
               forma adequada.

Instante: 7:34:23
Area de Gerenciamento: falhas
   Problema: nao_resp_nao_ativo([143,54,1,1],[1,1])
   pelo processo de inferencia:
     goal(diag_final(nao_resp_nao_ativo([143,54,1,1],[1,1])))
     e' derivado com 0.8 certeza
     pela regra (22 , interrelacionamento , falhas) de
       goal(diag_parcial(nao_responde_ped_con([143,54,1,1],[1,1])))
       e' derivado com 1 certeza
       pela regra (32 , n_4_1 , falhas) de
```

```
        oav(conexao(sa,sp,[143,54,1,1],[1,1]),estado,syn_wait(27201.0508))
        esta' na BC estrutural
        e
        instante_anterior(27263.1425) e' verdadeiro
       e
       27263.1425 > 27201.0508 + 60 e' verdadeiro
     e
     goal(diag_parcial(servico_nao_ativo_em([143,54,1,1])))
     e' derivado com 1 certeza
     pela regra (25 , n_3_1 , falhas) de
      prot_unreach([143,54,1,1]) e' um alarme

   Correcao apontada com 0.9 certeza:

        A estacao [143,54,1,1] nao esta respondendo
        pedidos de conexao do nivel de transporte  (TCP)
        pois seu servico  de transporte nao esta  ativo,
        ou o servico associado ao "port" [1,1]
        requisitado nao esta ativo.
        Verificar a instalacao do servico de transporte.
        Localizacao da estacao: cpd - sala de maquinas
        Equipamento: a10

Instante: 18:07:44
Area de Gerenciamento: falhas
   Problema: nao_aceita_por_gargalo_roteador([143,54,1,2],[143,54,1,1])
   pelo processo de inferencia:
     goal(diag_final(nao_aceita_por_gargalo_roteador(
     [143,54,1,2],[143,54,1,1])))
     e' derivado com 0.64 certeza
     pela regra (34 , interrelacionamento , falhas) de
```

```
goal(diag_parcial(estacao_nao_aceita_conexao(
[143,54,1,2],[2,2],[143,54,1,1],[1,1])))
e' derivado com 0.8 certeza
pela regra (28 , n_4_1 , falhas) de
     nao_aceita_conexao([143,54,1,2],[2,2],[143,54,1,1],[1,1])
     e' derivado com 1 certeza
     pela regra (10 , n_4_1 , falhas) de
        msg_tcp(fin,[143,54,1,2],[2,2],[143,54,1,1],[1,1])
        e' um alarme
        e
        verif_conex([143,54,1,2],[2,2],[143,54,1,1],[1,1],
        [143,54,1,1],[1,1],[143,54,1,2],[2,2])
        e' derivado com 1 certeza
        pela regra (8 , n_4_1 , falhas) de
         oav(conexao([143,54,1,1],[1,1],[143,54,1,2],[2,2]),
         estado,syn_rcvd) esta' na BC estrutural
       e
       oav(conexao([143,54,1,1],[1,1],[143,54,1,2],[2,2]),
       estado,syn_rcvd) esta' na BC estrutural
     e
     oav(estacao([170,0,4,0,195,185]),endereco_ip,[143,54,1,2])
     esta' na BC estrutural
    e
    oav(estacao([170,0,4,0,195,185]),conectada_a,segmento1)
    esta' na BC estrutural
   e
   conta_nro_conexoes([143,54,1,2],0) e' derivado com 1 certeza
   pela regra (24 , n_4_1 , falhas) de
      conta_fatos_rede(oav(conexao([143,54,1,2],_18C8,_18CC,_18D0),
      estado,estab),0) e' verdadeiro
```

```
        e
        conta_fatos_rede(oav(conexao(_1914,_1918,[143,54,1,2],_1920),
        estado,estab),0) e' verdadeiro
       e
       0 is 0 + 0 e' verdadeiro
    e
    not
      oav(estacao([170,0,4,0,195,185]),n_max_conexoes,_19A0)
      nao esta' na BC estrtural
      e
      _19A0 > 0 nao e' verdadeiro
  e
  goal(diag_parcial(roteador_sem_recursos([143,54,1,2])))
  e' derivado com 0.9 certeza
  pela regra (27 , n_3_1 , falhas) de
     sem_recursos([143,54,1,2]) e' um alarme
     e
     oav(estacao([170,0,4,0,195,185]),endereco_ip,[143,54,1,2])
     esta' na BC estrutural
    e
    oav(estacao([170,0,4,0,195,185]),une,(segmento1 , renpac))
    esta' na BC estrutural

Correcao apontada com 0.8 certeza:

     O roteador [143,54,1,2] nao esta aceitando
     conexoes de transporte por  falta de recursos
     para o tratamento  de novas conexoes.  Nao ha
     equipamento alternativo  no mesmo segmento de
     rede para trabalhar como roteador.
```

```
Verificar os parametros  de instalacao do ro-
teador.
Localizacao do roteador: cpd - sala de maquinas
Equipamento: vortex
```

As regras usadas, conhecimento estrutral consultado e correções aconselhadas apontadas podem ser encontradas nos apêndices A-1 e A-2.

6 CONSIDERAÇÕES FINAIS

Neste capítulo serão feitas algumas apreciações sobre o sistema e aspectos importantes no seu desenvolvimento, concluindo o trabalho.

6.1 Quanto à Especificação

Especificar um sistema antes de construí-lo certamente é uma medida correta e que deve ser enfatizada no meio acadêmico. A arquitetura fica melhor concebida pois o *software* é projetado em sua totalidade antes que maiores esforços sejam investidos. Além disso, o uso de métodos formais diminui ou elimina a ambigüidade e uma documentação coerente é gerada.

O maior problema encontrado no processo de especificação e implementação foi a inexistência de ferramentas de apoio à geração de código ou mesmo apoio na manutenção da conformidade entre especificação e implementação. Uma mudança de planos na fase de implementação leva ao reestudo da especificação viabilizando a mudança e implantação das novas idéias tanto na especificação quanto na implementação. Este processo, quando não apoiado por ferramentas, é árduo e nada agradável pois deve-se lidar com duas estruturas lógicas e manter sua equivalência.

A especificação do sistema em VDM e a contrução do protótipo em PROLOG podem ser tarefas um pouco redundantes pois, devido ao alto nível de abstração proporcionado pelo PROLOG, o mesmo é tomado por muitos também como uma forma de especificação. Por isso deve-se enfatizar que o protótipo construído serve para validar a idéia e sua especificação. A eventual utilização prática deste trabalho exigiria cuidados com desempenho e disponibilidade de recursos (principalmente memória), podendo implicar na sua reconstrução com o uso de ferramentas apropria-

das para a geração de um produto mais eficiente. Para este processo a especificação gerada seria extremamente necessária.

6.2 Aplicação de Orientação a Objetos

As estruturas da fase de Detecção representam os recursos sendo gerenciados (pode-se dizer que as estruturas da fase de Detecção tem a mesma função dos objetos de uma MIB). Os predicados testados continuamente mapeiam a "imagem" da rede observada no tráfego para as contabilizações e históricos. Esta estrutura foi adotada tendo como base a arquitetura do sistema, ou seja, o fato do sistema ser um "espião" da rede. Desta forma, as situações mais heterogêneas podem ser acompanhadas.

O esquema adotado para esta fase respondeu às expectativas. Diversas situações foram modeladas de forma satisfatória com as estruturas propostas. Notou-se, porém, a necessidade de uma metodologia mais poderosa e um pouco mais estruturada nesta fase de forma a possibilitar maior agregação de suas informações e uma representação mais próxima da realidade (com maior valor semântico) para os recursos sendo gerenciados. A utilização de orientação a objetos, como apontado no contexto OSI de gerência de redes, poderia ser uma boa alternativa para sanar esta deficiência. Com isso, a fase de Detecção ganharia em poder e em flexibilidade, diminuindo em parte a complexidade da fase de Diagnose que receberia dados mais agregados. De qualquer forma, o mapeamento da "imagem" da rede para estas estruturas internas teria que acontecer de forma semelhante ao adotado, com testes freqüentes de predicados.

6.3 Conformidade com o Modelo

No estudo das principais arquiteturas para gerência de redes (OSINM e DARPA) têm sido destacada a necessidade de abrigar sistemas que não suportam funções de gerenciamento ou que não possuem as mesmas funções de gerenciamento que o restante da rede. Com isso pretende-se que o sistema de gerência da rede seja completo e confiável, representando fielmente a instalação.

Para gerenciar estes sistemas é necessário incluir na rede componentes que os representem, os "Proxy Agents" [DUL91] [SCH91] (ou "Agentes Procuradores") realizam esta função:

- Para os equipamentos sem funções de gerência, os agentes monitoram seu comportamento (através do tráfego) e recebem/geram mensagens de gerência de/para o processo gerente. Um bom comportamento para este tipo de agente procurador é controlar constantemente o sistema representado, resolvendo localmente os problemas possíveis, e alertando o gerente central em situações que escapem do seu controle;

- Para os equipamentos com funções de gerência diferentes das funções "nativas" da rede, os agentes procuradores traduzem as mensagens de gerência de forma transparente para o processo gerente central.

O sistema desenvolvido, quando imerso em um domínio de gerenciamento abrangendo a rede por ele controlada, tem os requisitos principais para executar o papel de um agente procurador (do primeiro tipo). O sistema continua controlando da mesma forma a rede sob sua supervisão, quando alguma situação não for tratável pela sua base de conhecimento (foge ao escopo local), esta situação pode ser reportada ao gerente central. O gerente central responde com os procedimentos corretivos que são adaptados à realidade local pelo agente procurador.

Para se viabilizar este tipo de serviço, deve-se incorporar ao sistema um módulo agente e acrescentar conhecimento sobre os problemas que envolvem diagnóstico distribuído nas fases de Diagnóstico e Correção.

6.4 Trabalhos Futuros

Os esforços realizados viabilizam alguns trabalhos, por exemplo:

- modelagem de conhecimento relativo a outras áreas funcionais (Desempenho, Segurança, Contabilização, Configuração), criando novas teorias e subteorias para tratamento de problemas;
- adaptação ao sistema de um modelo orientado a objetos na fase de Detecção (como já comentado) e possível utilização deste modelo para representação do conhecimento estrutural da rede;
- estruturação de módulos para interação com um processo gerente, integrando o sistema em um domínio de gerência maior, bem como modelando a forma como esta interação se daria durante o processo de diagnose;
- remodelagem da fase de Correção, adicionando interação com o gerente, como também um mecanismo de realimentação informando sobre a taxa de sucessos nas correções apontadas.

6.5 Conclusão

Durante o projeto do sistema, especial atenção foi dada à necessidade de modularidade. Esta característica é muito importante para que o *software* seja aproveitado em estudos futuros. A fase de Interpretação pode servir de subsídio, por exemplo, em um monitorador de protocolos. A equivalência entre as funções

do submódulo de Detecção e de uma MIB, além da interface bem definida entre submódulo de Detecção e submódulo de Diagnose permitem que o sistema seja inserido em um contexto maior de gerenciamento com o papel de agente procurador, conforme abordado anteriormente (o processo de diagnose pode migrar para outra máquina). A fase de Correção é bastante simples mas bem definida, desta forma pode-se aprimorá-la (correção interativa acompanhada pelo gerente, realimentação conforme os resultados obtidos, etc). Deve-se ressaltar que a boa estrutura modular foi alcançada devido à especificação formal prévia do sistema.

A base de conhecimento conta com quase 70 regras em sua versão inicial. Muitas regras destinam-se à manutenção da base de conhecimento estrutural. A adoção de uma forma de representação mais poderosa (orientação a objetos) na fase de Detecção pode diminuir o número destas regras pois as informações repassadas à fase de diagnose seriam mais agregadas. Os problemas tratados são importantes e com grau de complexidade médio. O tratamento de problemas de maior complexidade pode ser alcançado com um trabalho mais aprofundado de engenharia de conhecimento, mapeando o conhecimento do especialista para a base de regras. As estruturas propostas para as diversas fases do sistema podem abrigar de forma tranquila o tratamento de problemas mais complexos.

Enfim, pode-se dizer que a arquitetura apresentada é uma boa opção para iniciar a gerência de uma instalação. A relação custo/benefício é compensadora pois não são necessárias modificações sobre a rede, apenas deve-se dedicar uma estação para que o serviço passe a ser prestado. Os demais equipamentos permanecem inalterados mas controlados. Os problemas que, anteriormente, existiam e cresciam até tornarem-se crônicos podem, com esta arquitetura, ser detectados e avisados ao Gerente. Com isso pode-se alcançar maior economia e eficiência na prestação de serviços à que a rede é destinada. O Gerente conta, assim, com uma ferramenta de auxílio em suas tarefas, um *Sistema de Apoio à Gerência.*

ANEXO A-1 REGRAS

Este apêndice apresenta as regras que fazem parte da base de conhecimento da versão inicial do *Sistema de Apoio à Gerência de Redes Locais.* Estas regras fazem parte da teoria para tratamento de "falhas", esta teoria está dividida em subteorias conforme os protocolos utilizados: Ethernet, IP, ICMP, ARP e TCP.

Algumas regras são destinadas à manutenção da base de conhecimento estrutural. O fator de certeza destas regras é sempre 1 pois a base estrutural não é parametrizada com relação à certeza de seus fatos.

Muitas regras estão comentadas, para outras este trabalho não é necessário pois seus mnemônicos dão o significado.

```
/* ###############    REGRAS N_1_1   ############## */

teoria falhas
subteoria n_1_1
regra  2  :  if   alarme(rajadas_de_erros) and
                  n (alarme(trafego_pesado))
             then goal(diag_parcial(rajadas_por_problema_instalacao))
             with 0.8.

teoria falhas
subteoria n_1_1
regra  4  :  if   alarme(rajadas_de_erros) and
                  alarme(trafego_pesado)
             then goal(diag_parcial(trafego_pesado_por_rajadas))
             with 0.7.

teoria falhas
```

```
subteoria n_1_1
regra  8 :  if   alarme(trafego_pesado)
            then goal(diag_parcial(trafego_pesado))
            with 1.

teoria falhas
subteoria n_1_1
regra  10  :  if   alarme(pacotes_nao_tratados)
              then goal(diag_parcial(acessos_nao_definidos))
              with 1.

teoria falhas
subteoria n_1_1
regra  12  :  if   alarme(pacotes_perdidos_buffer)
                   and ( trafego_pesado or trafego_medio )
              then goal(diag_parcial(buffer_pequeno_para_picos))
              with 1.

teoria falhas
subteoria n_1_1
regra  14  :  if   alarme(pacotes_perdidos_buffer)
                   and n ( trafego_pesado or trafego_medio )
              then goal(diag_parcial(buffer_pequeno))
              with 1.
```

Os dados disponíveis de nível físico são as estatísticas retornadas pelo *driver* do dispositivo de rede. Sobre as estatísticas são gerados os alarmes que alimentam as regras acima expostas. Assim, o serviço que esta camada pode prestar é altamente dependente da "completeza" dos dados que o dispositivo de rede retorna.

No caso em questão pouco pode-se agregar sobre o comportamento da rede com relação ao nível físico, os alarmes gerados são levados a diagnósticos parciais quase que diretamente e irão influir no inter-relacionamento dos diversos diagnósticos parciais de cada nível.

Note que as três últimas regras alertam para o funcionamento do próprio sistema denotando o não tratamento de alguns tipos de pacotes de tráfego ou a perda de pacotes pelo sistema.

```
/* ###############    REGRAS N_2_1   ############# */

teoria falhas
subteoria n_2_1
regra  2 : if alarme(endereco_local(End) ) and
              n ( oav(estacao_local,endereco,End) )
           then goal(exec(assert(oav(estacao_local,endereco,End))))
           with 1.

teoria falhas
subteoria n_2_1
regra  4 :  if alarme(estacao_ativa(E)) and
               n ( oav(estacao(E),estado,ativa) )
            then goal(
                 exec(
                 det_create(
                       (
                        pred_aus_traf(falhas,n_2_1,Cvs,120,ML,
                        aus(falhas,n_2_1,aus_msg_de(E))) :-
                        esta_em((srcadd,E), Cvs)
                                  ),
                        contab(aus(falhas,n_2_1,aus_msg_de(E)),
```

```
                    L_mom, 10, 600, 1, 1,
                    (falhas,n_2_1,estacao_inativa(E), 5) )
                    ,
                    hist((falhas,n_2_1,estacao_inativa(E),
                    TTL), LM, 10 )
                  )
                )
                )
        with 1.

teoria falhas
subteoria n_2_1
regra  6 :  if alarme(estacao_ativa(E))
        then goal(exec(
                  muda_valor(oav(estacao(E),estado,ativa))
                )    )
        with 1.

teoria falhas
subteoria n_2_1
regra  8 :  if alarme(estacao_inativa(E))
        then goal(exec(
                  muda_valor(oav(estacao(E),estado,inativa))
                )    )
        with 1.

teoria falhas
subteoria n_2_1
regra  10 :  if alarme(msg_maior_de(E))
         then goal(diag_parcial(msg_maior_de(E)))
```

```
            with 1.

teoria falhas
subteoria n_2_1
regra  12 :  if alarme(msg_menor_de(E))
             then goal(diag_parcial(msg_menor_de(E)))
             with 1.

teoria falhas
subteoria n_2_1
regra  14 :  if alarme(msgs_grandes)
             then goal(diag_parcial(msgs_grandes))
             with 1.

teoria falhas
subteoria n_2_1
regra  16 :  if alarme(estacao_ativa(E)) and
                n( oav(estacao(E), conectada_a, _ ) )
             then goal(diag_parcial(estacao_nao_conhecida(E)))
             with 1.

teoria falhas
subteoria n_2_1
regra  18 :  if alarme(broadcasts_excessivos)
             then goal(diag_parcial(broadcasts_excessivos))
             with 1.
```

As regras 2, 6 e 8 são destinadas à manutenção da base de conhecimento estrutural.

A regra 4 é um exemplo de manipulação de estruturas da fase de Detecção: quando é detectada a atividade de uma estação ela é dita "ativa" na base de conhecimento estrutural e são geradas estruturas para monitorar seu comportamento e derivar o instante em que não estiver mais ativa (ausência de mensagens durante um período de tempo considerável). A manutenção de um *status* de estação ativa ou inativa é importante para a fase de inter-relacionamento, pois diagnósticos parciais de níveis superiores podem ser anulados com esta informação.

As regras 10 a 18 tratam de problemas no nível 2. O protocolo de enlace é não orientado a conexão e bastante simples - seus campos são endereço fonte, endereço destino, tipo de pacote de nível 3 que trafega, e a unidade de dado do nível superior - isto faz com que a subteoria para tratamento do nível 2 seja relativamente simples.

```
/* ###############  REGRAS N_3_1  ############## */

teoria falhas
subteoria n_3_1
regra  1 :  if alarme(arp(Had,IPad)) and
               n(oav(estacao(Had),endereco_ip,IPad))
            then goal(exec(
                     assert(oav(estacao(Had),endereco_ip,IPad))
                    )    )
            with 1.

teoria falhas
subteoria n_3_1
regra  3 :  if alarme(msg_rede(Sa,Da,Sa_n2,Da_n2)) and
               n( oav(estacao(Sa_n2),endereco_ip,Sa) )
            then goal(exec(
                     assert(oav(estacao(Sa_n2),endereco_ip,Sa))
```

```
                    )    )
            with 1.

teoria falhas
subteoria n_3_1
regra  5 :  if alarme(msg_rede(Sa,Da,Sa_n2,Da_n2)) and
               n( oav(estacao(Da_n2),endereco_ip,Da) )
            then goal(exec(
                     assert(oav(estacao(Da_n2),endereco_ip,Da))
                    )    )
            with 1.

teoria falhas
subteoria n_3_1
regra  11 :  if alarme(arps_excessivos)
             then goal(diag_parcial(arps_excessivos))
             with 1.

teoria falhas
subteoria n_3_1
regra  13 :  if alarme(end_unreach(Da,Sa,E_rot))  and
                oav(E,endereco_ip,Da)             and
                oav(E,conectada_a,S)
             then estacao_local_nao_alcancavel(Da,Sa,E_rot)
             with 1.
/* estacao "Da" conectada a rede nao esta sendo atingida
   pela estacao "Sa" pelo roteador "E_rot" */

teoria falhas
subteoria n_3_1
```

```
regra  15 :  if exec(S1 = S2)
             then existe_caminho_alternativo_para(S1,S2,E_rot,C,C)
             with 1.

teoria falhas
subteoria n_3_1
regra  17 :  if ( oav(estacao(E_rot1),une,(S1,Saux)) or
                  oav(estacao(E_rot1),une,(Saux,S1))  ) and
                  exec(E_rot \= E_rot1)                  and
                  exec(not(esta_em(estacao(E_rot1),C)))   and
                  existe_caminho_alternativo_para(
                              Saux,S2,E_rot,[estacao(E_rot1)|C],Cr
                                                  )
             then existe_caminho_alternativo_para(S1,S2,E_rot,C,Cr)
             with 1.
/* regras 15 e 17 servem para encontrar uma rota alternativa
   unindo os segmentos "S1" e "S2", que nao use o roteador de
   endereco "E_rot"  */


teoria falhas
subteoria n_3_1
regra  19 :  if estacao_local_nao_alcancavel(Da,Sa,E_rot) and
                  oav(estacao(Sa),conectada_a,S1) and
                  oav(estacao(Da),conectada_a,S2) and
                  existe_caminho_alternativo(S1,S2,E_rot,[],C)
             then goal(diag_parcial(
                       caminho_alternativo_para(Da,Sa,E_rot,C)
                     )           )
             with 0.9.
```

```
/* esquema de roteamento aplicado nao alcanca estacao "Da",
   caminho alternativo apontado em "C" */

teoria falhas
subteoria n_3_1
regra  21 :  if estacao_local_nao_alcancavel(Da,Sa,E_rot) and
                oav(estacao(Sa),conectada_a,S1) and
                oav(estacao(Da),conectada_a,S2) and
                n( existe_caminho_alternativo(S1,S2,E_rot,[],C) )
             then goal(diag_parcial(
                       sem_caminho_alternativo_para(Da,Sa,E_rot)
                       )            )
            with 1.
/* esquema de roteamento aplicado nao alcanca estacao "Da",
   nao ha caminho alternativo possivel */

teoria falhas
subteoria n_3_1
regra  23 :  if alarme(end_unreach(Da,Sa,E_rot))  and
                n( oav(E,endereco_ip,Da)  and
                n( oav(E,conectada_a,S)  )
             then goal(diag_parcial(
                       estacao_externa_nao_alcancavel(Da,Sa,E_rot)
                       )            )
             with 0.9.
/* esquema de roteamento nao alcanca estacao externa "Da" */

teoria falhas
subteoria n_3_1
regra  25 :  if alarme(prot_unreach(Da))
```

```
            then goal(diag_parcial(servico_nao_ativo_em(Da)))
            with 1.
/* estacao "Da" processa mensagem de nivel 3 mas nao existe
   processo  ou port associado ativo para tratamento */

teoria falhas
subteoria n_3_1
regra  27 :  if alarme(sem_recursos(A))   and
               oav(E,endereco_ip,A)       and
               oav(E,une,_)
            then goal(diag_parcial(roteador_sem_recursos(A)))
            with 0.9.
/* roteador "A" nao e capaz de processar na velocidade
   que chegam os dados */

teoria falhas
subteoria n_3_1
regra  29 :  if alarme(sem_recursos(A))     and
               oav(E,endereco_ip,A)         and
               n(oav(E,une,_))
            then goal(diag_parcial(estacao_sem_recursos(A)))
            with 0.9.
/* estacao "A" nao e capaz de processar na velocidade
   que chegam os dados */
```

As regras 1, 3 e 5 mantém na base de conhecimento estrutural a relação entre endereços IP e endereços Ethernet. Para isso são usadas mensagens do tipo ARP e IP.

As demais regras desta subteoria servem ao diagnóstico de problemas. O ICMP - Internet Control Message Protocol - é uma importante fonte de informações

na subteoria de nível 3 por carregar dados de controle informando sobre diversos tipos de problemas acontecendo na rede.

```
/* ###############    REGRAS  N_4_1  ############## */

teoria falhas
subteoria n_4_1
regra  2 : if alarme(msg_tcp(syn,Sa,Sp,Da,Dp))  and
              exec(instante_anterior(I))
           then goal(exec(muda_valor(
                     oav(conexao(Sa,Sp,Da,Dp),estado,syn_wait(I))
                    )    )          )
           with 1.

teoria falhas
subteoria n_4_1
regra  4 : if alarme(msg_tcp(syn_ack,Sa,Sp,Da,Dp))   and
                     oav(conexao(Da,Dp,Sa,Sp1),estado,syn_wait(_))
           then goal(
                exec(
                troca(oav(conexao(Da,Dp,Sa,Sp1),estado,syn_wait(_)),
                      oav(conexao(Da,Dp,Sa,Sp),estado,syn_rcvd))
                    )
                    )
           with 1.

teoria falhas
subteoria n_4_1
regra  6 : if oav(conexao(Sa,Sp,Da,Dp),estado,_)
           then  verif_conex(Sa,Sp,Da,Dp,Sa,Sp,Da,Dp)
           with 1.
```

```
teoria falhas
subteoria n_4_1
regra  8 : if oav(conexao(Da,Dp,Sa,Sp),estado,_)
           then  verif_conex(Sa,Sp,Da,Dp,Da,Dp,Sa,Sp)
           with 1.

teoria falhas
subteoria n_4_1
regra  10 : if alarme(msg_tcp(fin,Sa,Sp,Da,Dp))          and
               verif_conex(Sa,Sp,Da,Dp,Sa1,Sp1,Da1,Dp1) and
               oav(conexao(Sa1,Sp1,Da1,Dp1),estado,syn_rcvd)
            then nao_aceita_conexao(Sa,Sp,Da,Dp)
            with  1.

teoria falhas
subteoria n_4_1
regra  12 : if alarme(msg_tcp(fin,Sa,Sp,Da,Dp))          and
               verif_conex(Sa,Sp,Da,Dp,Sa1,Sp1,Da1,Dp1) and
               oav(conexao(Sa1,Sp1,Da1,Dp1),estado,syn_rcvd)
            then goal(exec(muda_valor(
                     oav(conexao(Sa1,Sp1,Da1,Dp1),estado,fin_wait)
                   )    )          )
            with 1.

teoria falhas
subteoria n_4_1
regra  16 :  if alarme(msg_tcp(fin,Sa,Sp,Da,Dp))          and
                verif_conex(Sa,Sp,Da,Dp,Sa1,Sp1,Da1,Dp1) and
                oav(conexao(Sa1,Sp1,Da1,Dp1),estado,fin_wait)
```

```
            then goal(exec(muda_valor(
                      oav(conexao(Sa1,Sp1,Da1,Dp1),estado,closed)
                    )    )          )
            with 1.

teoria falhas
subteoria n_4_1
regra  18 : if alarme(msg_tcp(ack,Sa,Sp,Da,Dp))          and
               verif_conex(Sa,Sp,Da,Dp,Sa1,Sp1,Da1,Dp1) and
               oav(conexao(Sa1,Sp1,Da1,Dp1),estado,syn_rcvd)
            then goal(exec(muda_valor(
                      oav(conexao(Sa1,Sp1,Da1,Dp1),estado,estab)
                    )    )          )
            with 1.

teoria falhas
subteoria n_4_1
regra  20 : if alarme(msg_tcp(fin,Sa,Sp,Da,Dp))          and
               verif_conex(Sa,Sp,Da,Dp,Sa1,Sp1,Da1,Dp1) and
               oav(conexao(Sa1,Sp1,Da1,Dp1),estado,estab)
            then goal(exec(muda_valor(
                      oav(conexao(Sa1,Sp1,Da1,Dp1),estado,fin_wait)
                    )    )          )
            with 1.

teoria falhas
subteoria n_4_1
regra  22 : if alarme(msg_tcp(dados,Sa,Sp,Da,Dp))   and
               n ( oav(conexao(Sa,Sp,Da,Dp),estado,_)   or
               oav(conexao(Da,Dp,Sa,Sp),estado,_)    )
```

```
            then goal(exec(assert(
                      oav(conexao(Sa,Sp,Da,Dp),estado,estab)
                    )    )      )
            with 1.

/* as regras acima mantem o estado das conexoes TCP monitoradas
   pelo sistema - as regras estao encadeadas logicamente formando
   uma maquina de estados (a maquina de estados do rotocolo TCP).
   Tambem e' realizado um trabalho de verificacao de time-outs
   durante os procedimentos de abertura de conexao.
*/

teoria falhas
subteoria n_4_1
regra 24 : if exec(
              conta_fatos_rede(oav(conexao(A,X,Y,Z),estado,estab),N)
                  )  and
              exec(
              conta_fatos_rede(oav(conexao(P,Q,A,R),estado,estab),M)
                  )  and
              exec(N1 is N + M)
           then conta_nro_conexoes(A,N1)
           with  1.
/* regra para contar o numero de conexoes abertas em que a
   estacao de endereco "A" participa */

teoria falhas
subteoria n_4_1
regra 26 : if nao_aceita_conexao(Sa,Sp,Da,Dp) and
              oav(E,endereco_ip,Sa)           and
```

```
            oav(E,conectada_a,Seg)          and
            oav(E,n_max_conexoes,N_max)     and
            conta_nro_conexoes(Sa,N)        and
            exec(N = N_max)
        then goal(
             diag_parcial(nro_conexoes_esgotadas(Sa,Sp,Da,Dp))
                 )
        with  0.9.
/* uma estacao pode nao estar aceitando mais conexoes pois
   atingiu o numero maximo de conexoes a serem abertas */

teoria falhas
subteoria n_4_1
regra  28 : if nao_aceita_conexao(Sa,Sp,Da,Dp) and
            oav(E,endereco_ip,Sa)           and
            oav(E,conectada_a,Seg)          and
            conta_nro_conexoes(Sa,N)        and
            n( oav(E,n_max_conexoes,N_max) and
            exec(N_max > N)  )
         then goal(diag_parcial(
                  estacao_nao_aceita_conexao(Sa,Sp,Da,Dp)
                 )           )
         with  0.8.
/* estacao local nao aceita conexao sem motivo aparente
   no inter-relacionamento com outros niveis o problema
   pode ser identificado */

teoria falhas
subteoria n_4_1
regra 30 : if nao_aceita_conexao(Sa,Sp,Da,Dp) and
```

```
            n( oav(E,endereco_ip,Sa)   and
            oav(E,conectada_a,Seg)     )
         then goal(diag_parcial(
                  estacao_externa_nao_aceita_conexao(Sa,Sp,Da,Dp)
                )           )
         with  1.
/* estacao externa nao aceita conexao sem motivo aparente
   no inter-relacionamento com outros niveis o problema
   pode ser identificado */

teoria falhas
subteoria n_4_1
regra 32 : if oav(conexao(Sa,Sp,Da,Dp),estado,syn_wait(Iant)) and
            exec(instante_anterior(I)) and
            exec(I > (Iant + 60 ))
         then goal(diag_parcial(nao_responde_ped_con(Da,Dp)))
         with  1.
/* estacao nao esta respondendo pedidos de conexao - time-out */

teoria falhas
subteoria n_4_1
regra 34 : if oav(conexao(Sa,Sp,Da,Dp),estado,syn_wait(Iant)) and
            exec(instante_anterior(I)) and
            exec(I > (Iant + 60))
         then goal(exec(retract(
                  oav(conexao(Sa,Sp,Da,Dp),estado,syn_wait(Iant))
                )    )       )
         with  1.

teoria falhas
```

```
subteoria n_4_1
regra 36 : if alarme(janela_pequena(A,P))
           then goal(diag_parcial(janela_pequena(A,P)))
           with  1.

teoria falhas
subteoria n_4_1
regra 38 : if alarme(resets(A,P))
           then goal(diag_parcial(resets(A,P)))
           with  1.
/* estacao de endereco "A", port "P", esta resetando
   frequentemente a conexao */
```

As regras 4 a 22 têm como objetivo manter o *status* de cada conexão monitorada, acompanhando as fases de estabelecimento de conexão, troca de dados e encerramento da conexão. O fato do TCP ser orientado à conexão é positivo para que se possa inferir problemas a partir do tráfego, por exemplo: pode-se saber se existe atraso de resposta, o número de conexões abertas em determinado equipamento num certo instante, tempo de duração das conexões, e a massa de dados que trafega em cada conexão. Com tudo isso mais o auxílio da base de conhecimento estrutural pode-se detectar a intensidade dos acessos aos servidores (sejam servidores de arquivos, impressão, conexão, etc) e aconselhar a migração de serviços de um servidor para outro ou mesmo de um determinado instante para outro.

```
/* ##############   INTERRELACIONAMENTO   ############ */

teoria falhas
subteoria interrelacionamento
regra  2 : if diag_parcial(acessos_nao_definidos)
            then goal(diag_final(acessos_nao_definidos))
```

```
            with 1.

teoria falhas
subteoria interrelacionamento
regra  4 : if diag_parcial(buffer_pequeno_para_picos)
           then goal(diag_final(buffer_pequeno_para_picos))
           with 1.

teoria falhas
subteoria interrelacionamento
regra  6 : if diag_parcial(buffer_pequeno)
           then goal(diag_final(buffer_pequeno))
           with 1.

/* as regras 2 a 6 sao avisos do proprio sistema quanto ao seu
   funcionamento
*/

teoria falhas
subteoria interrelacionamento
regra  8 : if diag_parcial(trafego_pesado_por_rajadas)
           then goal(diag_final(trafego_pesado_por_rajadas))
           with 1.

teoria falhas
subteoria interrelacionamento
regra 10 : if diag_parcial(trafego_pesado) and
              diag_parcial(msgs_grandes)
           then goal(diag_final(trafego_pesado_por_msgs_grandes))
           with 0.7.
```

```
teoria falhas
subteoria interrelacionamento
regra 12 : if diag_parcial(trafego_pesado) and
              diag_parcial(broadcasts_excessivos) and
              diag_parcial(arps_excessivos)
           then goal(diag_final(trafego_pesado_por_broadcasts_arps))
           with 0.6.

teoria falhas
subteoria interrelacionamento
regra 14 : if n ( diag_parcial(trafego_pesado) )  and
              diag_parcial(excessivos_broadcasts) and
              diag_parcial(excessivos_arps)
           then goal(diag_final(excessivos_broadcasts_arps))
           with 0.8.

teoria falhas
subteoria interrelacionamento
regra 16 : if diag_parcial(msg_menor_de(E))
           then goal(diag_final(msg_menor_de(E)))
           with 1.

teoria falhas
subteoria interrelacionamento
regra 18 : if diag_parcial(msg_maior_de(E))
           then goal(diag_final(msg_maior_de(E)))
           with 1.

teoria falhas
```

```
subteoria interrelacionamento
regra 20 : if diag_parcial(estacao_nao_conhecida(E))
           then goal(diag_final(nova_estacao_inserida_na_rede(E)))
           with 1.

teoria falhas
subteoria interrelacionamento
regra 22 : if diag_parcial(nao_responde_ped_con(A,P))  and
              diag_parcial(servico_nao_ativo_em(A))
           then goal(diag_final(nao_resp_nao_ativo(A,P)))
           with 0.8.

teoria falhas
subteoria interrelacionamento
regra 24 : if diag_parcial(nao_responde_ped_con(Da,P))  and
              diag_parcial(caminho_alternativo_para(Da,Sa,Rot,C))
           then
           goal(diag_final(
           nao_resp_pela_rota_atual_caminho_alternativo(Da,Sa,Rot,C)
               )          )
           with 0.8.

teoria falhas
subteoria interrelacionamento
regra 26 : if diag_parcial(nao_responde_ped_con(Da,P))  and
              diag_parcial(sem_caminho_alternativo_para(Da,Sa,Rot,C))
           then
           goal(diag_final(
           nao_resp_pela_rota_atual_sem_caminho_alternativo(Da,Sa,Rot)
               )          )
```

```
            with 0.8.

teoria falhas
subteoria interrelacionamento
regra 28 : if diag_parcial(nao_responde_ped_con(Da,P))  and
               diag_parcial(estacao_externa_nao_alcancavel(Da,Sa,Rot))
            then
            goal(diag_final(
            nao_resp_estacao_externa_nao_alcancavel(Da,Sa,Rot)
                )          )
            with 0.9.

/* As regras 22, 24, 26 e 28 tratam a nao resposta de um pedido
   de conexao combinando com os possiveis problemas de nivel 3:
   - servico inativo na estacao que nao responde;
   - rota interrompida, e' aconselhada outra rota;
   - rota interrompida, nao ha outra rota aconselhada;
   - estacao exerna (nao pertence 'a rede local) nao
     e' alcancada com o atual esquema de roteamento.
*/

teoria falhas
subteoria interrelacionamento
regra 30 : if diag_parcial(nro_conexoes_esgotadas(Sa,Sp,Da,Dp))
            then goal(diag_final( nro_conexoes_esgotadas(Sa,Da,Dp) ))
            with 1.
/* estacao pode nao aceita mais conexoes pois
   atingiu o numero maximo de conexoes abertas */

teoria falhas
```

```
subteoria interrelacionamento
regra 32 : if diag_parcial(estacao_nao_aceita_conexao(Sa,Sp,Da,Dp))  and
              diag_parcial(roteador_sem_recursos(Sa)) and
              oav(E,endereco_ip,Sa) and oav(E,conectada_a,Seg) and
              oav(E2,conectada_a,Seg) and exec(E \= E2) and
              oav(E2,une,Segs)
           then
           goal(diag_final(
           nao_aceita_por_gargalo_roteador_roteador_alternativo(Sa,Da,E2)
               )          )
           with 0.8.

teoria falhas
subteoria interrelacionamento
regra 34   :  if diag_parcial(estacao_nao_aceita_conexao(Sa,Sp,Da,Dp))  and
                 diag_parcial(roteador_sem_recursos(Sa))
              then goal(
                   diag_final(nao_aceita_por_gargalo_roteador(Sa,Da))
                       )
              with 0.8.

teoria falhas
subteoria interrelacionamento
regra 36   :  if diag_parcial(estacao_nao_aceita_conexao(Sa,Sp,Da,Dp))  and
                 diag_parcial(estacao_sem_recursos(Sa)) and
                 oav(E,endereco_ip,Sa)  and
                 oav(E,funcao_na_rede,servidor(X)) and
                 oav(E,conectada_a,Seg) and
                 oav(E2,conectada_a,Seg) and exec(E \= E2) and
                 oav(E2,funcao_na_rede,servidor(X))
```

```
            then
            goal(diag_final(
            nao_aceita_por_gargalo_servidor_servidor_alternativo(Sa,Da,E2)
                )          )
            with 0.8.

teoria falhas
subteoria interrelacionamento
regra 38   :  if diag_parcial(estacao_nao_aceita_conexao(Sa,Sp,Da,Dp))  and
              diag_parcial(estacao_sem_recursos(Sa)) and
              oav(E,endereco_ip,Sa)  and
              oav(E,funcao_na_rede,servidor(X))
            then
            goal(diag_final(nao_aceita_por_gargalo_servidor(Sa,Da)))
            with 0.8.

teoria falhas
subteoria interrelacionamento
regra 40   :  if diag_parcial(estacao_nao_aceita_conexao(Sa,Sp,Da,Dp))  and
              diag_parcial(estacao_sem_recursos(Sa)) and
              oav(E,endereco_ip,Sa)  and
              n( oav(E,funcao_na_rede,servidor(X)) )
            then
            goal(diag_final(nao_aceita_por_gargalo_estacao(Sa,Da)))
            with 0.9.

teoria falhas
subteoria interrelacionamento
regra 42  : if diag_parcial(
             estacao_externa_nao_aceita_conexao(Sa,Sp,Da,Dp)
```

```
                    )
        then goal(diag_final(
                    estacao_externa_nao_aceita_conexao(Sa,Da)
                  )          )
        with 1.

/* As regras 30 a 40 tratam a nao aceitacao de conexoes:
   - por numero maximo de conexoes abertas;
   - por falta de recursos (estacao e' um roteador e
     e' fornecido um roteador alternativo);
   - por falta de recursos (estacao e' um roteador e
     nao ha' roteador alternativo a ser indicado);
   - por falta de recursos (estacao e' um servidor e
     e' indicado um servidor alternativo);
   - por falta de recursos (estacao e' um servidor e
     nao ha' servidor alternativo a ser indicado);
   - por falta de recursos da estacao.
/

teoria falhas
subteoria interrelacionamento
regra 44  : if diag_parcial(resets(A,P))
            then goal(diag_final(resets_frequentes_de(A,P)))
            with 1.
```

A subteoria de inter-relacionamento trata os problemas inter-níveis que possam ocorrer. Este tratamento é feito através dos diversos diagnósticos parciais das outras subteorias e do conhecimento estrutural da rede.

O conhecimento estrutural da rede onde o protótipo foi desenvolvido está exposto abaixo.

```
/* ##########  CONHECIMENTO ESTRUTURAL  ########## */

oav(rede,possui,segmento1).
oav(rede,possui,segmento2).
oav(rede,possui,segmento3).
oav(rede,possui,segmento4).
oav(rede,possui,segmento5).

oav(segmento1,tipo,dez_base_cinco).
oav(segmento2,tipo,dez_base_dois).
oav(segmento3,tipo,dez_base_dois).
oav(segmento4,tipo,dez_base_dois).

oav(estacao([170,0,4,0,195,185]),conectada_a,segmento1).
oav(estacao([170,0,4,0,195,185]),equipamento,vortex).
oav(estacao([170,0,4,0,195,185]),localizacao,'cpd - sala de maquinas').
oav(estacao([170,0,4,0,195,185]),endereco_ip,[143,54,1,2]).
oav(estacao([170,0,4,0,195,185]),une,(segmento1,renpac)).

oav(estacao([170,0,4,0,62,187]),conectada_a,segmento1).
oav(estacao([170,0,4,0,62,187]),equipamento,sbu).
oav(estacao([170,0,4,0,62,187]),localizacao,'cpd - sala de maquinas').

oav(estacao([8,0,11,0,0,19]),conectada_a,segmento1).
oav(estacao([8,0,11,0,0,19]),equipamento,a10).
oav(estacao([8,0,11,0,0,19]),localizacao,'cpd - sala de maquinas').
oav(estacao([8,0,11,0,0,19]),endereco_ip,[143,54,1,1]).
```

```
oav(estacao([8,0,11,254,95,88]),conectada_a,segmento1).
oav(estacao([8,0,11,254,95,88]),equipamento,nsp2000).
oav(estacao([8,0,11,254,95,88]),localizacao,'cpd - sala de maquinas').

oav(estacao([8,0,11,254,5,92]),conectada_a,segmento1).
oav(estacao([8,0,11,254,5,92]),equipamento,cp2000).
oav(estacao([8,0,11,254,5,92]),localizacao,'cpd - sala de maquinas').

oav(estacao([0,0,180,16,12,107]),conectada_a,segmento2).
oav(estacao([0,0,180,16,12,107]),equipamento,pcxt).
oav(estacao([0,0,180,16,12,107]),localizacao,'cpd - setor de redes').

oav(estacao([8,0,32,10,208,40]),conectada_a,segmento2).
oav(estacao([8,0,32,10,208,40]),equipamento,sun).
oav(estacao([8,0,32,10,208,40]),localizacao,'cpd - setor de redes').
oav(estacao([8,0,32,10,208,40]),endereco_ip,[143,54,1,20]).

oav(estacao([0,0,27,48,150,176]),conectada_a,segmento3).
oav(estacao([0,0,27,48,150,176]),equipamento,pc386).
oav(estacao([0,0,27,48,150,176]),localizacao,'nau').

oav(estacao([0,0,27,28,71,173]),conectada_a,segmento4).
oav(estacao([0,0,27,28,71,173]),equipamento,pcxt).
oav(estacao([0,0,27,28,71,173]),localizacao,'cpd - suporte').

oav(estacao([0,0,27,48,132,118]),conectada_a,segmento4).
oav(estacao([0,0,27,48,132,118]),equipamento,pc386).
oav(estacao([0,0,27,48,132,118]),localizacao,'cpd - suporte').
```

ANEXO A-2 CORREÇÕES

Este apêndice apresenta as correções para os diagnósticos finais alcançados na fase de diagnose.

```
/* ##############   CORRECAO   ############ */

correcao(acessos_nao_definidos,
         ['
          Existem tipos de pacotes  na rede para os quais o
          sistema nao define acesso. O acesso a estes paco-','
          tes deve ser  definido  para que  o sistema possa
          representar o estado da rede de forma verdadeira.
         '],
         1).

correcao(buffer_pequeno_para_picos,
         ['
          Durante momentos de trafego excessivo, o sistema esta
          perdendo pacotes.      O buffer de recepcao no modulo
          Driver deve ser aumentado.
         '],
         1).

correcao(buffer_pequeno,
         ['
          O sistema  esta  perdendo pacotes em situacoes de
          trafego normal.    O buffer de recepcao no modulo
          Driver deve ser aumentado de forma significativa.
         '],
```

```
        1).

correcao(trafego_pesado_por_rajadas,
        ['
         A rede apresenta trafego pesado devido 'a ocorrencia
         de rajadas de  erros e  consequentes retransmissoes.
         Procedimentos indicados:
             1. Verifique novas conexoes ou reorganizacoes','
                no sistema de cabeamento;
             2. Verifique se o cabo nao esta submerso em
                fortes campos magneticos, mesmo os even-
                tuais.
        '],
        0.7).

correcao(trafego_pesado_por_msgs_grandes,
        ['
         Aviso: Neste instante a rede apresenta  trafego
                intenso e mensagens grandes.  Esta forma
                de operacao pode levar a uma baixa  per-','
                formance. Considere a mudanca de parame-
                tros na instalacao dos  servicos de rede
                e enlace (fator de fragmentacao dos  pa-
                cotes).
        '],
        0.6).

correcao(trafego_pesado_por_broadcasts_arps,
        ['
         Aviso: A rede apresenta trafego intenso de pacotes tipo
```

```
                broadcast devido a pacotes ARP.
                Verificar os parametros do  servico de resolucao',',
                de enderecos, mais especificamente se as tabelas
                de traducao de enderecos estao sendo mantidas de
                forma adequada.
        '],
        0.8).

correcao(msg_menor_de(E),
        ['
         A estacao ',E,' esta emitindo
         mensagens menores que o permitido pelo padrao. Os
         parametros de instalacao  de seus servicos de co-
         municacao devem ser modificados.
         Localizacao da estacao: ',L,'
         Equipamento: ',Eq,'
        '],
        1) :- oav(estacao(E),localizacao,L),
              oav(estacao(E),equipamento,Eq).

correcao(msg_maior_de(E),
        ['
         A estacao ',E,' esta emitindo
         mensagens maiores que o permitido pelo padrao. Os
         parametros de instalacao  de seus servicos de co-
         municacao devem ser modificados.
         Localizacao da estacao: ',L,'
         Equipamento: ',Eq,'
        '],
        1) :- oav(estacao(E),localizacao,L),
```

```
            oav(estacao(E),equipamento,Eq).

correcao(nova_estacao_inserida_na_rede(E),
        ['
         A estacao ',E,' foi inserida na rede.
         Seus dados devem ser adicionados a base de conhe-
         cimento estrutural  deste  sistema  (localizacao
         fisica, tipo de equipamento, segmento  a que esta',',
         conectada,  funcao na  rede ou servico prestado,
         endereco IP (opcional) e numero maximo  de  cone-
         xoes (opcional).
        '],
        1).

correcao(nao_resp_nao_ativo(A,P),
        ['
         A estacao ',E,' nao esta respondendo
         pedidos de conexao do nivel de transporte  (TCP)
         pois seu servico  de transporte nao esta  ativo,
         ou o servico associado ao "port" ',P,'
         requisitado nao esta ativo.
         Verificar a instalacao do servico de transporte.
         Localizacao da estacao: ',L,'
         Equipamento: ',Eq,'
        '],
        0.7) :- oav(estacao(E),endereco_ip,A),
                oav(estacao(E),localizacao,L),
                oav(estacao(E),equipamento,Eq).

correcao(nao_resp_nao_ativo(A,P),
```

```
        ['
         A estacao ',E,' nao esta respondendo
         pedidos de conexao do nivel  de transporte (TCP)
         pois seu servico  de  transporte nao esta ativo,
         ou o servico associado ao "port" ',P,'
         requisitado nao esta ativo.
         Verificar a instalacao do servico de transporte.
         Localizacao da estacao: externa ao dominio de
                                 gerenciamento.
        '],
        0.7) :- oav(E,endereco_ip,A),
                not(oav(E,conectada_a,Seg)).


correcao(nao_resp_pela_rota_atual_caminho_alternativo(Da,Sa,Rot,C),
        ['
         O roteador ',Rot,' notifica que
         a estacao ',Da,' nao foi encontrada
         com o atual esquema de roteamento.  O caminho
         ',C,'
         pode ser sugerido como  rota para  a troca de
         mensagens entre ',Da,' e ',Sa,' .
        '],
        0.75).

correcao(nao_resp_pela_rota_atual_sem_caminho_alternativo(Da,Sa,Rot),
        ['
         O roteador ',Rot,' notifica que
         a estacao ',Da,' nao foi encontrada
         com o atual esquema de roteamento.  O sistema
```

```
        nao encontrou um caminho alternativo para su-
        gerir como  rota para  a  troca de  mensagens
        entre ',Da,' e ',Sa,' .
       '],
       0.8).

correcao(nao_resp_estacao_externa_nao_alcancavel(Da,Sa,Rot),
        ['
        O roteador ',Rot,' notifica que
        a estacao ',Da,', externa ao
        dominio de gerenciamento, nao  foi encontrada
        com o atual esquema de roteamento.  O sistema',','
        nao encontrou um caminho alternativo para su-
        gerir como  rota para  a  troca de  mensagens
        entre ',Da,' e ',Sa,' .
       '],
       0.8).

correcao(nro_conexoes_esgotadas(Sa,Da,Dp),
        ['
        A estacao servidora  ',Sa,' nao
        esta aceitando conexoes  pois esta com o nu-
        mero maximo permitido de conexoes
        abertas.
        Verificar:
           1. Parametros de instalacao do servico de
              transporte em ',Sa,'.
           2. Possibilidade de migrar parte do  ser-
              vico prestado para outra estacao.
        Localizacao da estacao: ',L,'
```

```
         Equipamento: ',Eq,'
        '],
        0.75) :- oav(estacao(Sa),localizacao,L),
                 oav(estacao(Sa),equipamento,Eq).

correcao(nao_aceita_por_gargalo_roteador(Sa,Da),
        ['
         O roteador ',Sa,' nao esta aceitando
         conexoes de transporte por  falta de recursos
         para o tratamento  de novas conexoes.  Nao ha
         equipamento alternativo  no mesmo segmento de',','
         rede para trabalhar como roteador.
         Verificar os parametros  de instalacao do ro-
         teador.
         Localizacao do roteador: ',L,'
         Equipamento: ',Eq,'
        '],
        0.8) :- oav(E,endereco_ip,Sa),
                oav(E,localizacao,L),
                oav(E,equipamento,Eq).

correcao(nao_aceita_por_gargalo_roteador_roteador_alternativo(Sa,Da,E),
        ['
         O roteador ',Sa,' nao esta aceitando
         conexoes de transporte por  falta de recursos
         para o tratamento  de novas conexoes.        ','
         Verificar:
            1. Parametros  de instalacao do roteador.
            2. Eventual transferencia de servicos para
               ',E,'.
```

```
          Localizacao do roteador: ',L,'
          Equipamento: ',Eq,'
          Localizacao do roteador alternativo:
                    ',L1,'
          Equipamento: ',Eq1,'.
         '],
         0.8) :- oav(estacao(Sa),localizacao,L),
                 oav(estacao(Sa),equipamento,Eq).
                 oav(estacao(E),localizacao,L1),
                 oav(estacao(E),equipamento,Eq1).

correcao(nao_aceita_por_gargalo_servidor(Sa,Da),
         ['
          O servidor ',Sa,' nao esta aceitando
          conexoes de transporte por  falta de recursos
          para o tratamento  de novas conexoes.  Nao ha',','
          equipamento alternativo  no mesmo segmento de
          rede para prestar o mesmo tipo de servico.   ',','
          Verificar os parametros  de instalacao do ro-
          teador.
          Localizacao do servidor: ',L,'
          Equipamento: ',Eq,'
         '],
         0.85) :- oav(estacao(Sa),localizacao,L),
                  oav(estacao(Sa),equipamento,Eq).

correcao(nao_aceita_por_gargalo_servidor_servidor_alternativo(Sa,Da,Serv),
         ['
          O servidor ',Sa,' nao esta aceitando
          conexoes de transporte por  falta de recursos
```

```
        para o tratamento  de novas conexoes.',',
        Verificar:
           1. Parametros  de instalacao do servidor.
           2. Eventual transferencia de servicos para
              ',Serv,'.
        Localizacao do servidor: ',L,'
        Equipamento: ',Eq,'
        Localizacao do servidor alternativo:
                   ',L1,'
        Equipamento: ',Eq1,'.
       '],
       0.9) :- oav(estacao(Sa),localizacao,L),
               oav(estacao(Sa),equipamento,Eq).
               oav(estacao(Serv),localizacao,L1),
               oav(estacao(Serv),equipamento,Eq1).

correcao(nao_aceita_por_gargalo_estacao(Sa,Da) ,
       ['
        A estacao ',Sa,' nao esta aceitando
        conexoes de transporte por  falta  de recursos
        para o tratamento  de novas conexoes.     ',',
        Verificar os parametros de  instalacao do ser-
        vico de transporte (alocacao de mais buffers).
        Localizacao da estacao: ',L,'
        Equipamento: ',Eq,'
       '],
       0.8) :- oav(estacao(Sa),localizacao,L),
               oav(estacao(Sa),equipamento,Eq).

correcao(estacao_externa_nao_aceita_conexao(Sa,Da),
```

```
        ['
         Aviso: estacao externa ao dominio de gerenciamento
                ',Sa,' nao esta aceitando
                pedidos de conexao de transporte.
         '],
         1).

correcao(resets_frequentes_de(A,P) ,
        ['
         Aviso: A estacao externa ',A,'
                esta resetando com frequencia sua conexao
                associada  ao "port"  ',P,'.
         Localizacao da estacao: ',L,'
         Equipamento: ',Eq,'
        '],
         1) :- not(oav(estacao(A),conectada_a,Seg)),
               oav(estacao(A),localizacao,L),
               oav(estacao(A),equipamento,Eq).
```

nocitewinston,tarouco,mefisto,rfcftp,rfcarp,rfcicmp

BIBLIOGRAFIA

[ARI89] ARITY Corporation. **Arity PROLOG Manual.** Arity Corporation. Concord, Massachussets: Arity Corporation, 1989. 202p.

[BAN88] BANCILHON, François. Object Oriented Database Systems. In: ACM SIGART-SIGMOD-SIGACT SYMPOSIUM ON PRINCIPLES OF DATABASE SYSTEMS, 7., March 1988, Austin, Texas. **Proceedings ...** S.l.: ACM, 1988. 10p.

[BJØ78] BJØRNER, D.; JONES, C. B. (eds.) **The Vienna Development Method: the Meta-language**, Berlim: Springer-Verlag, 1978. 382p. (Lecture Notes in Computer Science, 61).

[BML91] BOCHMANN, G. V.; MONDAIN-MONVAL, P.; LECOMTEC, L. Formal Description of Networks Management Issues. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2., Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North-Holland, 1991. p.77-92.

[BRA86] BRATKO, Ivan. **Prolog Programming for Artificial Intelligence**. Great-Britain: Addison-Wesley, 1986. 423p.

[DAR81a] Darpa Internet Program Protocol Specification. **Internet Protocol. (Request for Comments: 791)**. California: Information Sciences Institute University Of Southern California, Sept. 1981. 44p.

[DAR81b] Darpa Internet Program Protocol Specification. **Transmission Control Protocol. (Request for Comments: 793)**. California: Information Sciences Institute University Of Southern California, Sept. 1981. 85p.

[CGP90] CLAUDÉ, J.P.; CLAUDÉ, M.P.; GERVAIS, M.P.; .PENNA, M. Management of Open Networks in Heterogeneous Context. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGEMENT, 2., Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 81-100,

[CFS90] CASE, J.D.; FEDOR, M.; SCHOFFSTALL, M.L.; DAVIN, C. **Simple Network Management Protocol (SNMP). Request for Comments: 1157**, May. 1990. 36p.

[CHL90] CHAN, E.; LEE, C.H. Structuring Network Objects with Views. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGEMENT, 2., Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 159-171.

[CKP88] COHEN, Roberta S.; KAN, Hsin K.; PENNOTTI, J. Raymond. Unified Network Management from AT&T. **AT&T Technical Journal**, November/December 1988. p. 121-136.

[CRA89] J. CRAWFORD. Graphics for Network Management: An Interactive Approach. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 197-210,

[CST89] CELLARY, W.; STROINSKI, M. Performance Management Architecture for Protocol Entity. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 227-234.

[DUL91] DUATO, E,; LEMERCIER, B. SNPM for Non-TCP/IP Sub-networks: An Implementation. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2., Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North-Holland, 1991. p.201-212.

[DOT92] DOTTI, Fernando Luís. **Usando o Packet Driver.** Relatório Técnico. Porto Alegre: CPGCC da UFRGS, 1992.

[DSW88] DUNNING, Barton B.; SWILTIK, John. A Real-Time Expert System for Computer Network Monitor and Control. **DATA BASE**, Summer, 1988. p. 35-38.

[DOT90] DOTTI, F. L.; TAROUCO, L. M. R. Um Sistema de Apoio à Análise de Tráfego. In: TELEMÁTICA 90 - SIMPÓSIO NACIONAL DE REDES DE COMPUTADORES E SUAS APLICAÇÕES, Set. 1990, Porto Alegre. **Proceedings ...** Porto Alegre: SUCESU-RS, 1990.

[DOT91] DOTTI, F. L.; TAROUCO, L. M. R. Um Sistema de Apoio à Gerência de Redes Locais. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES, 9., Mai. 1991, Florianópolis-SC. **Proceedings ...** Florianópolis-SC: UFSC, 1991. p. 400-416.

[FEH89] FEHSKENS, L. An Architectural Strategy for Enterprise Management. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 41-60.

[GMS89] GOODMAN, R.; MILLER, J.; SMYTH, P. A Platform for Heterogeneous Interconnection Network Management. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 13-26.

[GOY91] GOYAL, S. K. Knowledge Technologies for Evolving Networks. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2., Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North-Holland, 1991. p.439-461.

[GRO86] GROENBAEK, Inge. Conversion Between the TCP And ISO Transport Protocols As A Method Of Achieving Interoperability Between Data Communications Systems. **IEEE Journal on Selected Areas in Communications**, v.4, n.2, p. 288-295. Mar. 1986.

[HAK88] HARMON, P.; KING, D. **Sistemas Especialistas: A Inteligência Artificial Chega ao Mercado**. Rio de Janeiro-RJ: Editora Campus, 1988.

[HOA85] HOARE, C. A. R. **Communicating Sequential Processes**. London: Prentice-Hall International, 1985.

[HOA87] HOARE, C. A. R. An Overview Of Some Formal Methods For Program Design. **IEEE Computer**, v.20, n.9, Sept. 1987.

[ISO86] ISO. **ISO DP 7498-4 : Information Processing Systems - OSI Reference Model Part 4: Management Framework**. S.l.: OSI, Out. 1986.

[JOH90] JOHNSON, M. J. Space Station Freedom Operations Management System. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGEMENT, 2. Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 135-146.

[JON86] JONES, C. B. **Systematic Software Development Using VDM**. London: Prentice-Hall International, 1986. 300p.

[KAN88] KANYUH, D. An Integrated Network Management Product. **IBM Systems Journal**, v.27, n.1, p.45-59. 1988.

[KBW89] KERSCHBERG, L.; BAUM, R.; WAISANEN, A.; HUANG, I.; YOON, J. **A Survey Of Automated Fault Management Systems For Telecommunications Networks.** Virginia - USA: Departament of Information Systems and Systems Engineering - Geroge Mason University, 1989. 14p.

[KOB89] KOBAYASHI, Yoshikazu. Standardization Issues In Integrated Network Management. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 79-92.

[LIE88] LIEBOWITZ, J. **Expert Systems Applications to Telecommunications**. John Wiley & Sons, U.S.A., 1988. 371p.

[LIK90] LIKAVEC, J. Management Realities Lan For Computer Graphics & Computer Graphics For LAN. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGEMENT, 2. Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 21-34.

[MAR88] MARQUES, T. E. A Symptom-driven Expert System For Isolating And Correcting Network Faults. **IEEE Communications Magazine**, v.26, n.3, p.6-13. Mar.1988.

[MAR90] MARTINS, Raul C. B. **VDM e Desenvolvimento de Software.** Rio de Janeiro-RJ: Centro Científico Rio - IBM Brasil, Relatório Técnico CCR 099, Jan. 1990. 59p.

[MIL86] MILLIKEN, K.R.; CRUISE, A.V.; ENNIS, R.L.; FINKEL, A.J.; HELLERSTEIN, J.L.; LOEB, D.J.; KLEIN, D.A.; MASULLO,

M.J.; VAN WOERKOM, H.M.; WAITE, N.B. YES/MVS And The Automation Of Operations For Large Computer Complexes. **IBM Systems Journal**, v.25, n.2, p.159-180. 1986.

[MIL89] MILLER, M. A. **LAN Troubleshooting Handboook**. California: M and T Books, 1989. 309p.

[MAL89] MAZUMDAR, S.; LAZAR, A. Knowledge-based Monitoring Of Integrated Networks. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 235–246.

[NKI91] NAKAI, S.; KIRIHA, Y.; IHARA, Y.; HASEGAWA, S. A Development Environment For OSI Systems Management. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2., Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North-Holland, 1991. p.157–168.

[OWE89] OWEN, S. A. An End-user Network Management Model For ISDN. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p.387–396.

[PAT89] PATRIDGE, C. Integrating Network Measurement Agents Into The OSI. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 219–226.

[PAU90] PAULISCH, Stephan. Configuration And Performance Management Of LANs. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGE-

MENT, 2. Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 259-276.

[PLU82] PLUMMER, David C. **An Ethernet Address Resolution Protocol Or Converting Network Protocol Addresses To 48.bit Ethernet Addresses For Transmission On Ethernet Hardware. Request for Comments: 826**. S.l: Darpa Internet Program Protocol Specification, Nov. 1982.

[POS81] POSTEL, J. **Internet Control Message Protocol. Request for Comments: 792**. S.l: Darpa Internet Program Protocol Specification, Sept. 1981. 21p.

[PRE83] POSTEL, J.; REYNOLDS, J. **TELNET Protocol Specification. Request for Comments: 854**. S.l: Darpa Internet Program Protocol Specification, May. 1983. 15p.

[PRE85] POSTEL, J.; REYNOLDS, J. **File Transfer Protocol (FTP). Request for Comments: 959**. S.l: Darpa Internet Program Protocol Specification, Oct. 1985. 69p.

[RAE91] RAEDER, G. OSFA: An Object-Oriented Distributed Approach To Network Management. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2., Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North-Holland, 1991. p.135-146.

[RC87] ROSE, Marshall; CASS, Dwight E. OSI Transport Services On Top Of The TCP. **Computer Networks and ISDN Systems**, n.12, 1987. p.159-173.

[RIB90] RIBEIRO, Leila. **Comparação De Abordagens Para Especificação Formal.** Rio de Janeiro-RJ: Centro Científico Rio - IBM Brasil, Relatório Técnico, Mar. 1990. 48p.

[ROS91] ROSE, M. Network Management Is Simple: You Just Need The Right Framework. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2., Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North- Holland, 1991. p.9-28.

[SCH91] SCHMITT, M. A. R.; BRONZATTI, R. A. Gerenciando Ambiente TCP/IP. In: TELEMÁTICA 91 - SIMPÓSIO NACIONAL DE REDES DE COMPUTADORES E SUAS APLICAÇÕES, Out. 1991, Porto Alegre. **Proceedings ...** Porto Alegre: SUCESU-RS, 1991.

[SMI89] SUN MICROSYSTEMS INC. NFS: Network File **System Protocol Specification. Request for Comments: 1094**. S.l.: S.n., March. 1989.

[SOA86] SOARES, L. F. G. **Redes Locais**. Rio de Janeiro: Campus, 1986. 250p.

[STU89] SY, K.B.; TURCU, P. N. Network Management Of ISDN Customer Premises Equipment. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p. 397-409.

[SWE90] SEKKAKI, A.; WESTPHALL, C.B. Heterogeneous LAN Management. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGEMENT, 2. Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 35-46.

[SYL91] SYLOR, M. DECnet/OSI Phase V Networks Management. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2.,

Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North-Holland, 1991. p.57-69.

[TLX90] TAKAHASHI, Tadao; LIESENBERG, Haus K. E.; XAVIER, Daniel Tavares. **Programação Orientada a Objetos: Uma Visão Integrada do Paradigma de Objetos.** São Paulo: IME/USP, 1990, 340 p. (VII Escola de Computação, São Paulo, 1990.)

[TAK89] TAKAHASHI, Tadao. **O Paradigma de Objetos: Introdução e Tendências.** Uberlândia: UFU, 1989, 96 p.

[TAN88] TANENBAUM, A.S. **Computer Networks.** New-Jersey: Prentice-Hall, 1988. 658p.

[TAR86] TAROUCO, L. M. R. **Redes de Computadores - Locais e de Longa Distância.** São Paulo: McGraw-Hill, 1986. 352p.

[TAR90] TAROUCO, L. M. R. Support of Network Management. In: INDC-90, CONFERENCE ON INFORMATION NETWORK AND DATA COMMUNICATION. Mar. 1990, Lillehammer. **Proceedings ...** Norway: [S.n.] 1990.

[TAD90] TAROUCO, L. M. R.; DOTTI, F. L. MEFISTO- Mechanism Efficient To Foster The Implementation Of Software Totally OSI. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGEMENT, 2. Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 221-228.

[TYE90] TILLMAN, M.A.; YEN, D. SNA and OSI: Three Strategies For Interconnection. **Communications of the ACM**, v.33, n.2, p.214-233. Feb. 1990.

[WAL89] WALLACE, David. How To Internetwork Between TCP/IP And OSI. **Telecommunications**, p.46-54. Apr. 1989.

[WAB89] WARRIER, U.; BESAW, L. **The Common Management Information Services And Protocol Over TCP/IP (CMOT). Request for Comments: 1095**. S.l: Darpa Internet Program Protocol Specification, 67p. April. 1989.

[WIN84] WINSTON, P. **Artificial Intelligence**. E.U.A.: Addison-Wesley, 1984.

[WEK84] S. M. WEISS and C. A. KULIKOWSKI. **A Practical Guide to Designing Expert Systems**. New Jersey: Rowland & Allanheld, 1984.

[WEL90] WENG, W..; LOADER, R. J. An Object Oriented Approach To Developing An Adaptable Network. In: IFIP TC6 WG6.4A INTERNATIONAL SIMPOSIUM ON LOCAL COMMUNICATIONS SYSTEMS MANAGEMENT, 2. Sept. 1990, University of Kent at Canterbury, U.K. **Proceedings ...** Holanda: North-Holland, 1990. p. 243-259.

[WAS89] WARRIER, U.; SUNSHINE, C. A Platform For Heterogeneous Interconnection Network Management. In: IFIP TC6 WG6.6 SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 1., May. 1989, Boston, MA, U.S.A. **Proceedings ...** Holanda: North-Holland, 1989. p.13–26.

[YGY91] YEMINI, Y.; GOLDSZMIDT, G.; YEMINI, S. Networks Management By Delegation. In: PROC. OF THE IFIP TC6 WG6.6 INTERNATIONAL SIMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2., Apr. 1991, Washington, DC. **Proceedings ...** Holanda: North-Holland, 1991. p.95-107.

Informática
UFRGS

Um sistema de apoio à gerência de redes locais.

Dissertação apresentada aos Srs.:

Prof. Dr. Carlos Becker Westphall

Prof. Dr. José Palazzo Moreira de Oliveira

Profa. Dra. Liane Margarida Rockenbach Tarouco

Prof. Dr. Joberto Sérgio Barbosa Martins (UFPb)

Vista e permitida a impressão.
Porto Alegre, 24/4/92.

Profa. Dra. Liane Margarida Rockenbach Tarouco
Orientador.

Prof. Dr. Ricardo A. da L. Reis,
Coordenador do Curso de Pós-Graduação
em Ciência da Computação.