UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Andrey Soares

ESPECIFICAÇÃO DE UMA MIB XML PARA O GERENCIAMENTO DE SISTEMAS

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Marcello Thiry Comicholi da Costa, Dr. Eng.

Florianópolis, Setembro de 2001.

UMA SOLUÇÃO PARA GERENCIAMENTO DE SISTEMAS USANDO MIB E XML

Andrey Soares

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação Área de Concentração (Sistemas de Computação) e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Fernando A. Ostuni Gauthier, Dr.
Coordenador do Curso

Banca Examinadora:

Marcello Thiry Comicholi da Costa, Dr.
Orientador

Alexandre Moraes Ramos, Dr.
Banca Examinadora

Elizabeth Sueli Specialski, Dra.
Banca Examinadora

Vitório Bruno Mazzola, Dr.
Banca Examinadora

DEDICATÓRIA

Dedico este trabalho aos meus pais,

Luiz e Neuza.

AGRADECIMENTOS

Agradeço primeiramente à Deus, por iluminar meu caminho, me dar forças para alcançar meus objetivos e pela oportunidade de aprender e crescer.

Aos meus pais Luiz e Neuza, pelo carinho, apoio, educação, lição de vida e muito mais. Vocês me ensinaram a acreditar e sonhar.

À Cristiane, pelo seu amor, carinho, paciência, compreensão e constante apoio nesta caminhada. A sua cumplicidade foi fundamental para a conquista deste sonho.

Ao professor Marcello Thiry, que antes de ser "orientador", é um grande amigo e conselheiro. Obrigado pela dedicação, apoio, otimismo, paciência e também por ter acreditado em mim.

Aos amigos professores do NIS - CEFET/SC, pelo incentivo e pelos momentos de descontração.

Aos professores Alexandre M. Ramos, Elizabeth S. Specialski e Vitório B. Mazzola, membros da banca examinadora, que contribuíram com suas críticas e sugestões para o enriquecimento deste projeto.

Ao locutor Everton Cunha (Mr. Pi), apresentador do programa Pijama Show da Rádio Atlântida FM, pela companhia incondicional nas madrugadas de estudo.

À todos os meus amigos e familiares que direta ou indiretamente contribuíram para a realização deste projeto.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

API	*Application Program Interface*
ASN.1	*Abstract Syntax Notation One*
B2B	*Business to Business*
CPU	*Central Process Unit*
CSS	*Cascading Style Sheet*
DHCP	*Dynamic Host Configuration Protocol*
DMA	*Direct Memory Access*
DOM	*Document Object Model*
DTD	*Document Type Definition*
GDMO	*Guidelines for Definition of Managed Object*
GPL	*General Public License*
HTML	*HiperText markup Language*
HTTP	*HiperText Transfer Protocol*
I/O	*Input / Output*
IA	Inteligência Artificial
IDC	*Internet Data Center*
IP	*Internet Protocol*
ISO	*International Standardization for Organization*
MIB	*Management Information Base*
OID	*Object Identifier*
OSI	*Open System Interconection*
PCDATA	*Parser Carachter Data*
PCI	*Peripheral Component Interconect*

RAM	*Random Access Memory*
RBC	Raciocínio Baseado em Casos
RMON	*Remote Monitoring*
SGML	*Standard Generalized Markup Language*
SMI	*Structured of Management Information*
SNMP	*Simple Network Management Protocol*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
UML	*Unified Modeling Language*
W3C	*World Wide Web Consortium*
WBEM	*Web-Base Enterprise Management*
XML	*Extended Markup Language*
XSL	*Extensible Stylesheet Language*

RESUMO

Este trabalho apresenta a utilização da XML no gerenciamento de sistemas. Mais especificamente, com o mapeamento de informações de uma MIB para documentos XML.

Inicialmente, são identificadas as arquiteturas de gerenciamento de sistemas baseadas em agentes SNMP e XML, os quais possuem o objetivo de manipular informações em uma MIB. Em seguida, são definidas as informações que farão parte da MIB proposta, bem como um resumo com os passos para o desenvolvimento de uma MIB.

O primeiro passo para o mapeamento MIB - XML é a criação dos arquivos DTD, que representam a estrutura das informações. A partir do arquivo DTD, gera-se o arquivo XML que contém as informações da MIB.

O trabalho ainda descreve alguns procedimentos para otimizar a geração de arquivos XML e para compatibilizar as MIBs.

Finalmente, é apresentada uma aplicação Web, como forma de validar e demonstrar a utilização de documentos XML, além dos resultados obtidos e as novas propostas de trabalho.

Palavras chaves: MIB, XML, Gerenciamento de Sistemas

ABSTRACT

This work present the using of XML at systems management. More specifically, with the mapping of information about a the MIB for XML documents.

Initially, are identifies the architecture of systems management, based in SNMP and XML agents, of whom own the objective of manipulate information in a MIB. Next, are defined the information that make part of the proposed MIB, wel how a summary with the steps to development the MIB.

The first step to MIB - XML mapping is the creation of DTD files, that represent the structure of information. From DTD file, conceive the XML file that contain the information of the MIB.

The work yet describe some proceeds to optimize the conceive the XML file and to compatibilit the MIBs.

Finaly, is presented an Web application, how validation and demonstration forms the using of XML documents, as well of results obtens and the news proposed of work.

Keywords: MIB, XML, Systems Management

1 INTRODUÇÃO

O crescimento contínuo, tanto em número quanto em diversidade dos componentes das redes de computadores tem exigido cada vez mais da atividade de gerenciamento. As principais causas que tornam árduo o trabalho de isolamento e teste de problemas são (BRISA, 1993):

- diversidade dos níveis humanos – técnicos, gerentes e engenheiros;
- diversidade nas formas de controle e monitoração.

Os pontos críticos da tarefa de gerenciar uma rede são diversos (ANÁLISE, 1999), os mais comuns consistem no fato de uma empresa possuir um parque de máquinas muito grande para realizar certas tarefas, como atualização de *software*, monitoração do sistema, e suporte ao usuário para resolução de problemas.

Ainda em ANÁLISE (1999), outro problema trivial do gerenciamento é a atualização de inventários detalhados, pois o administrador de sistemas deve possuir as características de cada máquina existente em seu ambiente. Por causa das incompatibilidades entre *hardware* e *software,* seria ideal ter sempre em mãos o tipo de placa de vídeo que cada máquina possui, quanto de memória, a velocidade do processador, ou seja, um pequeno banco de dados com informações do sistema.

Neste contexto, a utilização de uma MIB (*Management Information Base*), uma base de informações contendo dados a respeito dos recursos gerenciados, é proposta. Muitos fabricantes incorporam em seus produtos, uma MIB para que estes sejam administrados, monitorados e controlados.

Este projeto apresenta a especificação de uma MIB XML para o gerenciamento de sistemas, onde é elaborada uma aplicação Web para o validação do XML (*Extended Markup Language*) em aplicações de gerência.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo principal deste projeto é elaborar e apresentar a utilização da XML como ferramenta de armazenamento e manipulação de informações, ou seja, uma MIB XML.

1.1.2 Objetivos Específicos

Para a elaboração deste projeto, constituiu-se os seguintes objetivos específicos:

- Definição das funções de gerenciamento;
- Elaborar um arquitetura de gerenciamento usando XML;
- Definir informações para a criação de uma MIB;
- Estudar o processo de criação de uma MIB Padronizada;
- Elaborar um agente para mapeamento MIB - XML;
- Definir regras para o mapeamento MIB - XML;
- Elaborar uma aplicação Web usando XML;
- Identificação de sugestões e perspectivas futuras;

1.2 ORGANIZAÇÃO DO TRABALHO

O trabalho está organizado da seguinte forma:

Capítulo 1: Neste capítulo é apresentado o tema de pesquisa deste projeto enfatizando a justificativa e os objetivos propostos;

Capítulo 2: Aborda uma revisão sobre gerência de sistemas. Apresentando a divisão funcional ISO (*Internet Standardization for Organization*) e as funções de gerenciamento;

Capítulo 3: Este capítulo descreve a arquitetura de gerenciamento baseada em agentes SNMP (*Simple Network Management Protocol*) e XML;

Capítulo 4: Identifica as informações a serem coletadas para a elaboração de uma MIB, bem como os passos necessários;

Capítulo 5: Aborda os conceitos e características do XML, bem como a criação dos arquivos DTD (*Document Type Definition*) e XML. Também é apresentado um esquema elaborado para mapeamento da MIB;

Capítulo 6: Este capítulo apresenta a aplicação Web criada para manipular o arquivo XML, onde são descritas as funções JavaScript de validação do documento XML, bem como os de busca e modificação de valores;

Capítulo 7: Apresenta as conclusões alcançadas no desenvolvimento deste projeto, e os trabalhos futuros.

2 GERÊNCIA DE SISTEMAS

Dentre as atividades de gerência e mais especificamente dos objetos gerenciados em uma rede de computadores, pode-se citar as estações de trabalho, que demandam uma atenção especial por estarem sendo operadas pelos usuários. Ao contrário de alguns objetos gerenciados, como um *Hub*, que tem uma função específica e é geralmente gerenciado sob um mesmo enfoque, tem a estação de trabalho, que permite diversas utilizações, pois dependem das necessidades dos usuários e consequentemente apresentam diversos enfoques de gerência. Desta forma, pode-se analisar a utilização de uma estação, verificar quais atividades podem influenciar o andamento das operações da rede.

Para HEGERING (1994), o termo gerenciamento de redes e sistemas é definido como a soma total de todos os procedimentos e produtos para planejar, configurar, controlar, monitorar e gerenciar redes de computadores e sistemas distribuídos. HEGERING (1994) ainda sugere que um esquema de gerenciamento depende de alguns fatores, como:

- O objetivo da rede, que pode ser determinada pela análise das aplicações, levando em conta os serviços de rede que podem ser garantidos e os parâmetros de qualidade de serviço podem ser contabilizados;
- As características de comunicação, ou seja, a distribuição do tráfego da rede em termos de tempo e volume;
- A estrutura física da rede, determinada pelo cabeamento estruturado e a topologia;
- A estrutura lógica da rede, determinada pela arquitetura de comunicação e o resultado da hierarquia de protocolos;
- A distribuição de provisão de serviços pelos recursos;
- A organização estrutural e operacional dos usuários da rede, incluindo a distribuição dos recursos de rede;

- A estrutura e a organização operacional do provedor da rede, determinando a distribuição de conhecimentos e responsabilidades das tarefas de gerenciamento.

Segundo uma pesquisa realizada pela revista *Network Comp*uting Brasil (Edição: 20 / Ano:2000), espera-se uma média de crescimento anual no mercado de gerenciamento de sistemas, de 19% até 2004. Os produtos para gerenciamento de redes e serviços crescem a uma média de 45% ao ano, e as ferramentas para administração de *desktop* crescem a uma média de 33%.

Estes dados mostram uma constante preocupação das empresas em controlarem e monitorarem suas redes e seus computadores. E revelam um futuro promissor para as atividades de gerenciamento de redes e sistemas.

O uso do sistema operacional Linux, vem crescendo junto à empresas e usuários domésticos, como é possível comprovar com os argumentos apresentados a seguir. Com isto, espera-se o desenvolvimento de aplicativos em diversas áreas de conhecimento. Consequentemente com esta grande adoção do sistema operacional Linux, espera-se também uma maior preocupação com os computadores que utilizam este sistema, surgindo então a necessidade de controlá-los e monitorá-los através de atividades de gerenciamento de sistemas.

Segundo GUIDALI (2000), a taxa de crescimento da distribuição do Linux em 2000 no Brasil, em relação a 1999, chegou a 162,7%. Esta marca coloca o país como o quinto mercado do mundo com o maior aumento do uso do sistema operacional. Hoje já são mais de 70 distribuições do sistema operacional Linux.

A primeira edição da Linux Expo Brasil (www.linuxexpobrasil.com.br), realizada em 2000, trouxe bons resultados e perspectivas para o Linux. A seguir são apresentados alguns comentários feitos durante a Seção *Keynote* do Linux Expo Brasil/2000:

- Linux é o sistema que tem se alastrado mais rapidamente no mundo. A previsão do IDC para o mercado global fala num espantoso crescimento de 183% ao ano (Revista INFOEXAME).
- A América Latina é a área que vem registrando maior crescimento de Linux. Nos últimos dois anos, o uso desse sistema aumentou 915%. Enquanto o

mercado mundial vem alcançando índice anual de 93% (Sandro Nunes, Executivo da CONECTIVA).

- Entre os usuários que estão adotando o sistema no Brasil constam Bradesco, Banrisul, Prefeitura de Foz do Iguaçu, Lojas Renner e o provedor de acesso à Internet Terra (Sandro Nunes, Executivo da CONECTIVA).
- A procura deverá aumentar mais ainda após a aprovação do projeto de lei que tramita no Congresso Nacional, determinando que órgãos do governo passem a usar o sistema operacional de código aberto (Sandro Nunes, Executivo da CONECTIVA).
- Todas as pesquisas apontam o mesmo fenômeno: IDC: a base Linux vai passar de 1,3 milhões em 1999 a 4,7 milhões em 2004 (Dirk Hondhel, Executivo da Distribuição Linux, SUSE).

É possível citar algumas manifestações provocadas pela adoção dos sistemas operacional Linux:

- A prefeitura do Recife foi a primeira do mundo a aprovar uma lei que obriga a administração municipal a migrar para programas de código aberto, como o Linux, por exemplo. No Rio Grande do Sul essa migração tem acontecido por determinação administrativa e com bastante sucesso há dois anos (REVISTA DO LINUX, 2000). O Vereador Márcio de Souza do PT/Florianópolis, apresentou no dia 02 de abril de 2001, o Projeto de Lei n.° 9100/2001, que trata do uso do *software* livre pelo Município de Florianópolis e entidades públicas municipais. A proposta apresentada teve como base os Projetos de Lei 059/2000 - ELVINO BOHN GASS, Dep. PT do RS e Projeto de Lei do Dep. Federal Walter Pinheiro- PT-BA (GULINUXFLORIPA, 2001).
- A coordenação de administração urbana da Cidade do México pretende usar o sistema operacional Linux nesta e em outras áreas da administração municipal, e aplicar o dinheiro economizado em programas de combate à pobreza. São dezenas de milhões de pesos pagos em serviços e *software* proprietário (como

o Windows) economizados, e esses fundos extras serão utilizados nos programas de combate à miséria e a marginalização[1].

Para a Conectiva[2], as *interfaces* gráficas, evoluíram espetacularmente nos últimos seis meses; já há milhares de aplicativos para os mais variados fins, inclusive os exigidos pelos usuários comuns; há um contingente razoável de conhecedores de Linux, uma estrutura de suporte já bastante capilarizada e, mais que tudo, há um espírito de solidariedade que não poderia existir na indústria tradicional de *software*, e que certamente vai facilitar a adesão ao Linux e a disseminação do conhecimento sobre ele. A Conectiva ainda afirma que tudo isso faz do Linux hoje, o sistema operacional ideal para servidores corporativos e para *desktops* domésticos também.

2.1 DIMENSÕES DE GERENCIAMENTO

Segundo HEGERING (1994), toda tarefa de gerenciamento pode ser dividida em três sub-áreas, denominadas Dimensões de Gerenciamento. As dimensões são (Figura 1):

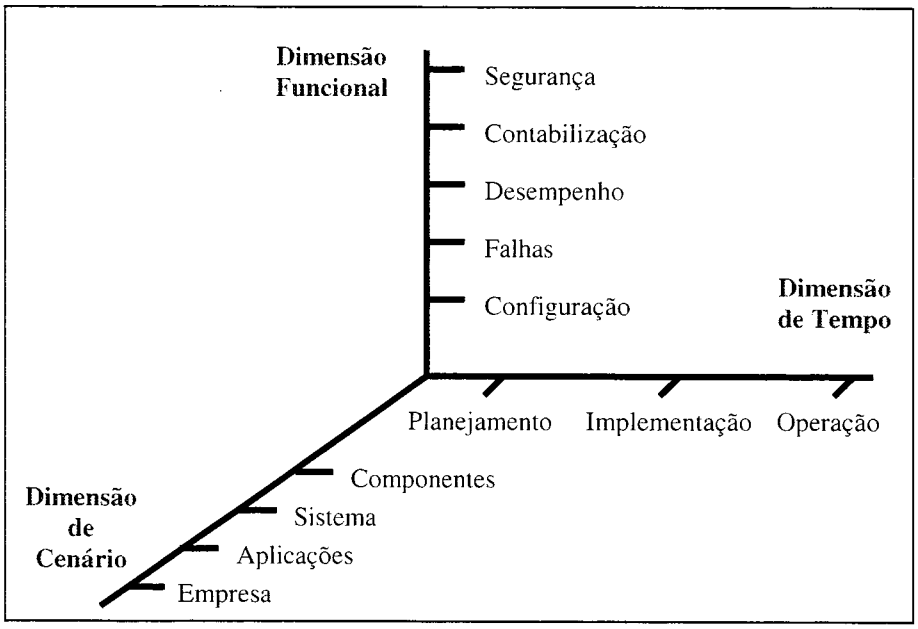

Figura 1: Dimensões de Gerenciamento

[1] José Barberan, em entrevista do site www.reforma.com.mx em 12/03/2001.
[2] Resposta da Conectiva publicado no site PlantãoINFO em fevereiro de 2001, referente às críticas que afirmam que o Linux é difícil de usar, as *interfaces* gráficas não são suficientemente amigáveis e que não é fácil achar aplicativos, no caso do usuário doméstico.

- **Dimensão Funcional**: diz respeito as tarefas de gerenciamento para as áreas funcionais ISO (Configuração, falha, desempenho, contabilização e segurança), apresentadas na seção 2.2;
- **Dimensão de tempo**: divide os processos que implementam as funções de gerenciamento em diferentes fases do ciclo de vida, incluindo planejamento, implementação e operação;
- **Dimensão de cenário**: no gerenciamento clássico de redes com as tarefas centradas em gerência de componentes, identificou-se outros cenários como gerência de sistemas, gerência de aplicações e gerência de atividades da empresa. Estes cenários são distinguidos pelo fato de que os objetos destino são diferentes em cada caso, onde eles tipicamente conduzem para diferentes aplicações de gerenciamento.

2.2 ÁREAS FUNCIONAIS

A ISO fez uma separação funcional (ISO/IEC 7498-4) de necessidades no gerenciamento OSI (*Open System Interconnection*). A seguir são descritas as características das áreas funcionais (HEGERING, 1994) (HELD, 1996) (SPECIALSKI, 1996) (SOARES, 1998):

2.2.1 Gerência de Falhas

Em decorrência de uma falha, deve-se gerar informações e procedimentos para:

- Determinar onde ocorreu a falha;
- Monitorar a rede ou o estado do sistema;
- Restaurar ou reconfigurar a rede ao estado ideal;
- Notificar o gerente, da detecção de uma falha e fornecer informações suficientes para que sejam tomadas as providências cabíveis;
- Procurar e prevenir pontos vulneráveis à falhas;
- Registrar problemas e providenciar o uso de *Help Desk*;

A gerência de falhas contribui para o aumento da confiabilidade da rede, em virtude de suas ferramentas permitirem uma rápida detecção de problemas e inicialização de procedimentos de recuperação: através destas ferramentas, pode-se conhecer as informações necessárias sobre o estado corrente da rede, e os gerentes podem trabalhar na resolução de sua falha mesmo antes deste ser detectada pelos usuários (MELCHIORS, 1999).

Por exemplo, se uma linha de comunicação for cortada fisicamente, nenhum sinal pode trafegar através da mesma.

2.2.2 Gerência de Contabilização

Tem o objetivo de mensurar a utilização de recursos de rede de um usuário ou um grupo de usuários. Provendo procedimentos para:

- Informar os custos envolvidos por recurso de rede e por usuários;
- Informar as atividades dos usuários, permitindo um planejamento de utilização e crescimento da rede;
- Alocar e monitorar quotas;
- Informar a utilização dos recursos em relação aos limites estabelecidos.

Por exemplo, o computador gerente pode elaborar um relatório que indica quais *softwares* são mais utilizados pelos usuários da rede e a freqüência com que são usados.

2.2.3 Gerência de Configuração

Está relacionado as tarefas de inicialização, adição, manutenção e atualização de recursos de rede durante a operação da rede. Conhecer a rede ou os recursos de sistema é um pré-requisito para esta tarefa.

O gerente da rede deve identificar os componentes da rede e definir a conectividade entre eles e manter uma relação atualizada de todos os componentes da rede. Deve também modificar a configuração em resposta a avaliações de desempenho, recuperação de falhas, problemas de segurança ou atualizações da rede.

Por exemplo, o gerente pode elaborar um inventário dos componentes *hardware* e *software* da rede, com suas respectivas configurações.

2.2.4 Gerência de Desempenho

Constitui na monitoração dos recursos e atividades da rede, levantando dados estatísticos referentes ao desempenho da rede. Estas informações permitem determinar tendências de utilização, isolar problemas de performance e resolvê-los mesmo antes que eles causem impacto na rede, determinar os parâmetros de qualidade de serviços e planejamento de desempenho e capacidade. O gerente deve reconhecer as situações indicativas de degradação de desempenho, definindo algumas questões relacionadas ao gerenciamento de desempenho, tais como (SPECIALSKI, 1996):

- Qual o nível de utilização da rede?
- Quais as condições de trafego?
- Existem recursos de rede, degradadores do desempenho?
- Tempo de resposta está aumentando?
- Existem rotas alternativas?

Por exemplo, um agente pode enviar alarmes para o gerente, informando que o percentual de colisões está próximo do limite estabelecido.

2.2.5 Gerência de Segurança

Provê facilidade para proteger recursos da rede e informações dos usuários. O gerenciamento de segurança deve dispor serviços de:

- Criptografia e Autenticação;
- Autorização de acesso e distribuição de senhas;
- Políticas de segurança;
- Manutenção de registros (*logs*) de autoria, acesso e segurança;
- Relato de tentativas de intrusão na rede.

Por exemplo, uma conta de usuário poderá ser desativada por estar sob suspeita de tentativas de intrusão.

2.2.6 Outras Funcionalidades

Duas áreas funcionais chaves que são parcialmente cobertas pelo padrão OSI e que são importantes o suficiente para justificar suas identificações como entidades separadas segundo HELD (1996) são:

Gerência de bens: Configura as tarefas associadas com o desenvolvimento e recuperação de registros de equipamentos, serviços e pessoal.

- Registros de Equipamentos, podem incluir um ou mais banco de dados de configurações de *software* e *hardware*, estatísticas de uso, dados dos produtos.
- Registro de Serviços, podem simplesmente incluir número de circuitos e pontos de contato da portadora, ou podem conter algumas informações adicionais como parâmetros do Serviço de *FrameRelay*.
- Registro de Pessoal, devem indicar a experiência de trabalho dos funcionários, educação e treinamentos. Estas informações podem ser usadas para requerer junto a organização, projetos de desenvolvimento individual com treinamentos apropriados.

Gerência de planejamento e suporte: Incluem aquelas tarefas que habilitam os administradores da rede há proverem suporte para os usuários e planejamentos futuros. Pode incluir ajuste de serviços de rede para acomodar alterações no uso de tais serviços, colocando equipamentos e serviços para um novo suporte ou expandindo aplicações, e reuniões com usuários finais para determinar seus graus de satisfação com os métodos de comunicação. Talvez a chave para a efetiva gerência de planejamento e suporte seja a análise de tendências. Traçando o resultado da utilização da rede por um período de tempo, você pode evitar uma variedade de problemas de saturação da rede, obtendo a habilidade de iniciar operações pró-ativas.

2.3 FUNÇÕES DE GERENCIAMENTO DE SISTEMAS

Estas funções são citadas como de grande importância no apoio as atividades do administrador de sistemas (MICROSOFT, 1999), em relação ao gerenciamento de sistemas:

- **Inventário de *hardware* e *software***: prover funcionalidades para adquirir informações sobre o *hardware* e o *software* instalado nos computadores clientes;
- **Instalação e distribuição de *software***: prover funcionalidades para distribuir e instalar *softwares* nos computadores clientes;
- **Medição de *software***: prover funcionalidades para medir a utilização de programas e gerenciar licenças;
- **Registro de problemas de *hardware* e *software***: prover funcionalidades para diagnosticar problemas de *hardware*, *software* ou rede. Em adicional, utilitários permitem controle remoto;
- **Utilitário de diagnósticos (monitor do sistema)**: podem ser executados a partir do computador dos técnicos para obter informações em tempo real sobre os computadores clientes;
- **Ferramentas remotas**: permitem que os técnicos tomem controle do computador cliente, através de *chat* com o usuário, copia de arquivos para o computador cliente, reiniciar o computador ou iniciar programas e serviços;
- **Monitor de rede**: permite um administrador capturar e seguir o tráfego, para o computador cliente ou a partir do computador cliente. Em adicional o monitor de rede inclui monitoramento de eventos e análise dos dados capturados.

A seguir estas funções serão descritas com mais detalhes. As tarefas que não são supervisionadas por humanos, devem ser realizadas pelos *softwares* agentes, ou seja por programas específicos.

2.3.1 Inventário de Hardware e Software

Método utilizado para coletar e registrar o *hardware* e *software* e suas configurações contidas nos computadores clientes. A coleta deverá ser executada por um agente específico (Figura 2) que poderá ser acionado: manualmente, durante a inicialização do computador, através de agendamento para execução de tempos em tempos ou através de monitoramento de mudanças.

O agente deverá coletar informações de: CPU (*Central Process Unit*), Memória, Pontos de Montagem, Interrupções, Porta I/O (*Input/Output*), PCI (*Peripheral Component Interconnect*), Canais DMA (*Direct Memory Access*), Placa de Vídeo, Monitor, Teclado, Mouse, Impressora, Unidades de Disco, Protocolos, Placa de Rede, IP (*Internet Protocol*), Hostname, *Cd-Rom*, *Floppy*, Console, Portas COM / LPT, Modem, Placa de Som, etc.

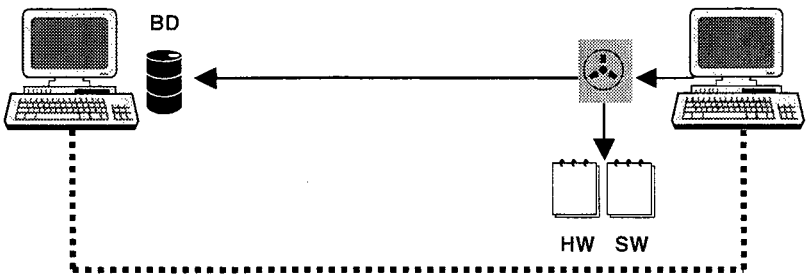

Figura 2: Esquema do agente de Inventário de *Hardware* e *Software*

Como resultado da coleta de *Hardware* e *software*, são emitidos dois relatórios contendo as informações coletadas. Estas informações são enviadas ao Banco de Dados do sistema, localizado na máquina gerente.

2.3.2 Distribuição de Software

Esta funcionalidade deve prover mecanismos para:

- Instalar e remover software;
- Atualizar o Sistema Operacional;
- Executar tarefas administrativas como por exemplo verificação de disco;

- Criar status da execução;
- Registro das instalações (data, hora, etc.);
- Os clientes não podem modificar as configurações dos agentes;
- Verificação de espaço em disco.

Para a realização desta função, deverá ser criado um *software* agente que executará as instalações remotas de programas previamente programadas pelo administrador de sistemas.

2.3.3 Medição de Software

A utilização de funcionalidade de medição de *software*, possibilita o controle mais intenso sobre os programas utilizados nos computadores clientes, tais como:

- Número de licenças usadas,
- Registro da execução de programas (rede, disco ou disquete),
- Autorização de uso de programas,
- Percentual de utilização de programas,
- Tempo de utilização dos programas,
- Emissão de alerta para a utilização de *software* sem licença,
- Controle de atualização de *software*.

Esta funcionalidade, de alguma forma é contrária ao projeto e as características de *Software* Livre defendidas pelos difusores do Linux.

Segundo CONECTIVA (2000), *Software* Livre refere-se à liberdade dos usuários em executar, copiar, distribuir, estudar, modificar e melhorar o programa. Mais precisamente, este termo refere-se a 4 tipos de liberdade, para os usuários do programa:

- Liberdade para executar o programa, com qualquer propósito;
- Liberdade para estudar como o programa funciona e adaptá-lo às suas necessidades. O acesso ao código fonte é um pré-requisito para que isto possa acontecer;

- A liberdade para redistribuir cópias do programa, para que se possa ajudar os amigos, conhecidos, parentes, etc.;
- A liberdade para melhorar o programa e distribuir suas melhorias para o público em geral, de maneira que toda a comunidade possa se beneficiar disto. Acesso ao código fonte é um pré-requisito para que isto aconteça.

A Licença Pública Geral (GPL – *General Public License*) é a licença utilizada pelo projeto GNU (www.gnu.org), que define os termos de distribuição utilizando o conceito do *copyleft*, ou seja, é a garantia de que qualquer usuário tenha direito de copiar, modificar e redistribuir o código de um programa, ou qualquer trabalho derivado do mesmo.

2.3.4 Descoberta / Reconhecimento da rede

Esta função baseia-se na utilização de métodos de descoberta de elementos de rede com localização e coleta de informações, como por exemplo:

- **Método baseado em endereço IP**, onde o agente de descoberta envia uma mensagem em *broadcast* e a partir da resposta, vai mapeando os elementos;
- **Método baseado em *logon***, onde após o usuário efetuar e validar o *logon*, o servidor envia uma mensagem para o cliente informando da sua existência na rede;
- **Método de reconhecimento,** onde o agente de inventário de *hardware* e *software* após executar a coleta de dados terá as seguintes informações: endereço IP, nome da máquina, rota padrão, sistema operacional, versão do sistema, etc.

As informações coletadas são transferidas para o Gerente para armazenamento em Banco de Dados.

O processo de descoberta, além de identificar os elementos de rede, deveria também identificar: nome, Sistema Operacional, (versão do sistema operacional e plataforma).

2.3.5 Ferramentas Remotas

A utilização de ferramentas que possibilitem ao administrador interferir e gerenciar os computadores clientes à distância, como se estivessem no local, tem colaborado na eficiência de diagnósticos e na tomada de decisão, antes mesmo que o usuário tomasse conhecimento do problema ocorrido. Entre as ferramentas de controle remoto, é possível citar:

- Salas de discussão para conversar com o usuário;
- Controle de teclado, vídeo e mouse do computador cliente;
- Diagnóstico do sistema;
- Transferência de arquivos;
- Reinicialização do sistema;
- Execução de comandos, programas e serviços;
- Monitoramento da rede;
- Agendamento de tarefas a serem executadas.

Algumas destas ferramentas já são contempladas junto à outras funções de gerenciamento de sistemas citadas.

2.3.6 Monitor da Rede

Permite que o administrador capture e analise o tráfego de rede para o computador cliente ou a partir do computador cliente. Algumas atividades do monitor são:

- Captura, filtra e mostra *frames* (pacotes),
- Analisador dos dados capturados,
- Protocolo e usuário que mais ocupa banda,
- Monitorar eventos específicos,
- Calcular o tráfego na rede,
- Calcular o tempo de resposta das requisições,

- Detectar o servidor DHCP (*Dynamic Host Configuration Protocol*),
- Monitorar faixas de endereço IP,
- Monitorar sinais de ataque.

2.3.7 Monitor do Sistema

Constitui-se de um agente que coleta de tempos em tempos, informações referentes ao sistema (Figura 3).

As informações coletadas são:

- **CPU**: Ocupada com o Usuário, Ocupada com o Sistema
- **Memória**: Usada e Livre
- *Swap*: Usado e Livre
- **Disco**: Usado e Livre
- **Processos**: Dormindo, Rodando, Parado
- **Rede**: Pacotes Recebidos e Transmitidos

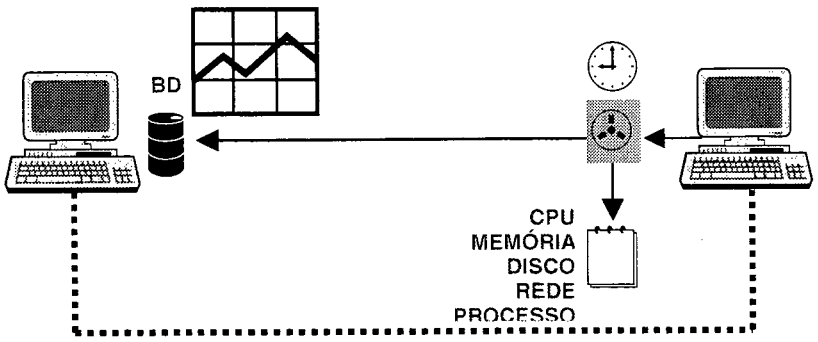

Figura 3: Esquema de monitoramento do sistema

2.3.8 Resolução de Problemas

Na gerência de sistemas, muitos problemas se repetem, surgindo então a necessidade de se conhecer como um determinado problema foi resolvido. Para isto deve-se criar um sistema que possibilite armazenar os problemas e suas respectivas soluções, constituindo-se assim uma memória histórica composta das experiências e conhecimentos utilizados na solução dos problemas (Figura 4).

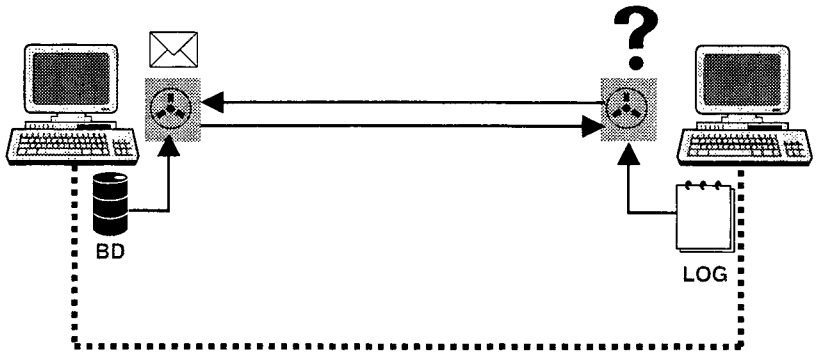

Figura 4: Esquema de resolução de problemas

Ao longo do tempo esta base de conhecimento possuirá uma quantidade considerável de informações, o que acarretará em uma grande quantidade de comparações a serem executadas para encontrar os problemas mais similares ao problema corrente. Uma sugestão a ser seguida é a de dividir os problemas em sub-áreas de conhecimento (*Hardware*, *Software*, Sistemas, etc.), com esta categorização, pode-se fazer menos comparações de problemas (otimizando o processo), pois se tem uma base mais específica de conhecimento.

Os sistemas de registros de problemas tem as seguintes funções (JOHNSON, 1992 *apud* MELCHIORS, 1999):

- Servir como uma memória dos problemas específicos ocorridos no domínio de gerência, mantendo a história completa do problema. Com isso, é possível a comunicação entre os gerentes envolvidos na resolução do problema, de modo que os problemas que se estendem por diversos turnos de trabalho possam ser analisados e trabalhados imediatamente por um novo gerente quando necessário;
- Servir como um visualizador da lista de problemas correntes em um domínio que deve preferencialmente ser fornecida em ordem de prioridade. Eles auxiliam, assim, o escalonamento do fluxo de trabalho dos diversos gerentes de um domínio;
- Enviar atribuições de tarefas ou consultas através de sistemas de correio eletrônico que estejam integrados ao sistema;

- Atribuir temporizadores para cada registro de problema, que quando decorridos gerem um alerta para o registro associado, lembrando sobre o problema;
- Enviar relatórios eletronicamente para os representantes de cada rede controlada pelo domínio de gerência, com resumos dos problemas associados a essa rede, de modo a informar sobre o estado corrente de cada ocorrência ainda não solucionada;
- Permitir a análise estatísticas dos equipamentos da rede e da produtividade do domínio de gerência através do processamento dos campos fixos dos registros. Permitem que sejam gerados relatórios de Tempo Médio entre falhas e Tempo Médio de Reparo, além de relatórios estatísticos de controle de qualidade, que propiciam a detecção de equipamentos defeituosos antes de uma falha efetiva;
- Atuar como filtro dos alertas que sejam relacionados a um registro de problemas em aberto;
- Permitir aos usuários e administradores da rede a visualização das atividades desenvolvidas pelo centro de operações de gerência para a resolução de falhas, indicando assim os esforços empregados para a resolução destes problemas.

A resolução de problemas pode ser feita a partir da verificação dos arquivos de *Log* (Figura 5) para verificar quais erros, alertas ou condições críticas ocorreram e procurar na base de casos os casos mais similares. Para isto, sugere-se uma técnica de IA (Inteligência Artificial), conhecida como RBC (Raciocínio Baseado em Casos) (AAMODT e PLAZA, 1994) (KOLODNER, 1996) (LEAKE, 1996) (WATSON, 1997).

```
LOG DE ERROS

Jan 10 10:46:23 cerebro modprobe: can´t locate module sound-slot-0
Jan 10 17:39:22 cerebro PAM_pwd[751]: auth cold not identify password for [root]

Jan 10 17:52:02 cerebro ifdown: operation failed
Jan 10 17:52:02 cerebro insmod: /lib/modules/2.2.14-14cl/net/net2k-pci.o: invalid
parameter irq
Jan 10 17:52:02 cerebro ifdown: eth0: interface desconhecida: dispositivo
inexistente
Jan 11 01:30:33 cerebro /sbin/mingetty[895]: tty2: invalid character ^[ in login
name
```

Figura 5: Arquivo Log de erros

Os sistemas Raciocínio Baseado em Casos utilizam um processo interativo constituído genericamente por: identificação da situação atual, busca da experiência mais semelhante na memória e aplicação do conhecimento desta experiência na situação atual (THIRY, 1999).

2.4 CONCLUSÕES

Inicialmente, o objetivo do projeto era desenvolver estas funcionalidades em um ambiente com sistema operacional Linux, porém, verificou-se que as informações manipuladas e que são sugeridas para serem armazenadas em uma Base de Dados, constituem uma MIB Não Padronizada. Isto ocasionou um novo objetivo ao projeto, onde agora o foco era a utilização de uma MIB.

Escolheu-se então a função **Monitor de Sistema**, que será implementada para a criação de um agente de coleta de informações e consequentemente para a criação de uma MIB Não Padronizada. A função Monitor de Sistema foi escolhida por apresentar informações atualizadas de desempenho e utilização do sistema. Estas informações permitem que o administrador do sistema mantenha diagnósticos a respeito da "saúde" do sistema.

O próximo capítulo, apresenta as arquiteturas de gerenciamento identificadas com a utilização de agentes SNMP e XML para a manipulação da MIB.

3 ARQUITETURA

A Figura 6 apresenta um esquema da utilização do XML e SNMP para a manipulação de informações de uma MIB, podendo esta MIB ser Padronizada (P) ou Não Padronizada (NP).

Baseado nas afirmações de WEBBER (1997), entende-se como MIB Padronizada (MIB P), aquela MIB, que:

- foi escrita corretamente;
- tenha sido verificada quanto à sintaxe e semântica através de um compilador;
- apresente conformidade com a SMI (*Structure of Management Information*);
- possa ser acessada através de primitivas SNMP.

E entende-se como uma MIB Não Padronizada (MIB NP), aquela MIB criada por uma empresa, com intuito de coletar informações para um determinado fim, sem que se tenha preocupação com a padronização, por exemplo, um programa escrito por um funcionário, para coletar informações de um objeto gerenciável e gravar estas informações em um arquivo texto. Uma MIB NP, tendem a ser uma MIB proprietária.

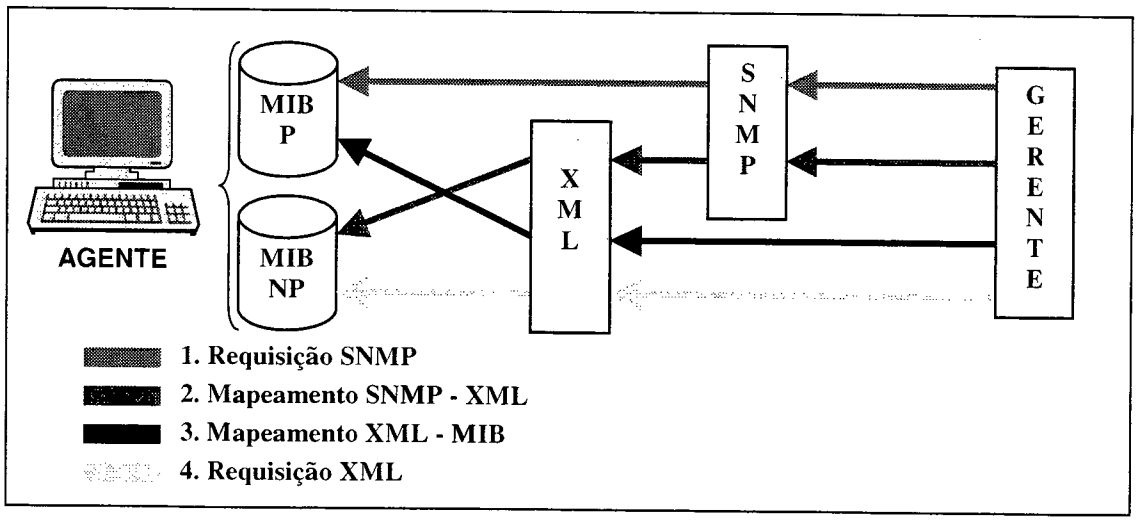

Figura 6: Esquema de mapeamento XML - SNMP

A seguir serão apresentadas as 4 possibilidades identificadas, de se gerenciar uma Base de Informações Gerenciais (MIB):

3.1 REQUISIÇÃO SNMP

É o método tradicional de gerenciamento, onde um *software* de gerenciamento manipula informações da MIB Padronizada, através de primitivas SNMP.

3.2 MAPEAMENTO SNMP-XML

Um *software* de gerenciamento deseja manipular informações de uma MIB Não Padronizada, porém a MIB é manipulada em XML e o *software* de gerenciamento só utiliza primitivas SNMP. Neste caso, o agente SNMP recebe as requisições SNMP provenientes do *software* de gerenciamento e as repassa ao agente XML. Uma vez obtida as informações, o agente XML devolve as informações ao agente SNMP e este responde a solicitação do *software* de gerenciamento.

3.3 MAPEAMENTO XML-MIB

Existe uma MIB Padronizada coletando informações de um equipamento, porém não existe um *software* de gerenciamento que faça requisições SNMP. Neste caso, são feitas requisições XML para o agente XML. O agente XML converte as requisições em primitivas SNMP e faz as requisições para a MIB.

3.4 REQUISIÇÃO XML

Existe uma MIB Não Padronizada e não existe um *software* de gerenciamento. Neste caso, são feitas requisições XML para o agente XML. Este então busca as informações e devolve ao solicitante.

A utilização de uma MIB Não Padronizada, poderia ser o caso de uma empresas que possui um *software* proprietário, criado para coletar algumas informações, sendo que este software não possui qualquer padronização em relação aos padrões MIB SNMP.

3.5 CONCLUSÕES

Este trabalho apresenta o uso de XML no gerenciamento de informações de uma MIB Padronizada ou Não Padronizada.

Deste modo, este trabalho manterá o foco nos itens 3 e 4 apresentados anteriormente, por apresentarem maior ênfase no Agente XML.

O item 1 por tratar-se do método de gerenciamento tradicional, já conhecido pelo meio acadêmico, não será abordado.

O item 2, por abranger a criação e a adaptação de um agente SNMP, também não será abordado, sendo apresentado como trabalhos futuros na seção 7.1.

4 MIB PROPOSTA

Segundo PEITER (2000), embora já existam MIBs prontas com agentes e gerentes implementados que monitoram uma grande parte de informações de interesse de uma arquitetura de rede e até sistemas, é difícil achar uma que se adapte as nossas necessidades, pois existem várias implementações SNMP, sendo difícil conseguir uma aplicação que resolva os problemas específicos de cada empresa. Por este motivo, muitas empresas acabam criando uma MIB própria que colete e armazene as informações desejadas. A MIB criada torna-se então Não Padronizada, pela dificuldade e pelo tempo despendido para a definição, compilação e implementação.

Uma MIB (MILLER, 1999) é uma estrutura que contém as variáveis necessárias para monitorar, gerenciar ou administrar os componentes em redes *Internet*. Basicamente, existem três tipos de MIBs (PEITER, 2000), pois a MIB I se tornou obsoleta quando foi acrescentada de alguns itens, tornando-se assim a MIB II.

- **MIB II**: fornece informações sobre o equipamento gerenciado como por exemplo, o estado da *interface*, informações sobre os protocolos de rede, número de pacotes com erros, etc.
- **MIBs experimentais**: são aquelas que estão em fase de testes para que no futuro possam ser padronizadas.
- **MIBs privadas**: são específicas dos equipamentos gerenciados, como *HUBs*, *Swiches*, Roteadores, etc. Estas fornecem informações particulares de cada um destes equipamentos.

Foi elaborada uma análise das informações disponíveis numa estação de trabalho usando o sistema operacional Linux, para identificar e definir as informações que são relevantes para o gerenciamento da estação.

Várias informações foram identificadas como sendo de importância para o gerenciamento de sistemas, porém serão usadas algumas apenas para validar o projeto proposto. Estas informações foram escolhidas com base no ambiente de testes usado para o projeto. É considerado que cada computador usado só permita um processador,

mas é possível a existência de mais de uma placa de rede. Inclusive um dos computadores possuia duas placas de rede, pois o mesmo funcionava como um *gateway*. Na próxima seção serão descritas estas informações.

4.1 INFORMAÇÕES COLETADAS

Os objetos usados para serem gerenciados e armazenados em uma MIB foram divididos em 4 categorias:

COMPUTADOR: são informações a respeito do computador. Os atributos desta categoria são:

- *HostName*: nome que identifica o computador na rede
- *Kernel*: versão do *kernel* (núcleo do sistema operacional)

CPU: são informações que identificam o processador usado no computador e sua utilização. Os atributos desta categoria são:

- Processador: tipo de processador (ex.: Pentium)
- Velocidade: velocidade em MHz do processador
- Fabricante: nome do fabricante do processador (Ex.: Intel)
- Ocupada Usr: percentual de CPU ocupada pelo usuário
- Ocupada Sis: percentual de CPU usada pelo sistema operacional
- Ociosa: percentual de CPU livre para uso

MEMÓRIA: são informações que mostram a utilização da memória RAM (*Random Access Memory*) e memória de SWAP. Os atributos são:

- Total: quantidade de memória RAM total (em Kbytes)
- Usada: quantidade de memória RAM ocupada (em Kbytes)
- Livre: quantidade de memória RAM livre (em Kbytes)
- *Swap* Total: quantidade de memória SWAP total (em Kbytes)
- *Swap* Livre: quantidade de memória SWAP livre (em Kbytes)

PLACA REDE: são informações que identificam as *interfaces* de rede disponíveis no computador. Os atributos são:

- Dispositivo: nome do dispositivo que identifica a *interface* de rede (Ex.: eth0)
- Pacotes Recebidos: quantidade de pacotes recebidos pela *interface* de rede
- Pacotes Transmitidos: quantidade de pacotes transmitidos pela *interface* de rede
- Endereço IP: endereço IP usado pela *interface* de rede
- IRQ: número da interrupção usada pela *interface* de rede
- ES: endereço de entrada e saída usada pela *interface* de rede

As informações selecionadas, podem ser modeladas através de um método orientado a objetos. Foi adotado para este projeto a UML (*Unified Modeling Language*) (ERIKSSON, 1998) (FOWLER, 2000).

Segundo FOWLER (2000), a UML é uma linguagem de modelagem bem definida, expressiva, poderosa, geralmente aplicável, não proprietária e aberta a todos.

A Figura 7 representa o Diagrama de Classes, na notação UML, onde para cada categoria foi criada uma classe correspondente.

As classes CPU, PLACAREDE e MEMÓRIA possuem um relacionamento de agregação por composição com a classe COMPUTADOR, ou seja, um computador é composto de cpu, memória e placa de rede.

Também foi criada uma classe mais genérica chamada OBJETO, que possui o atributo OID (*Object Identifier*) e os métodos GET e SET, que serão herdados pelas demais classes (classes mais especializadas).

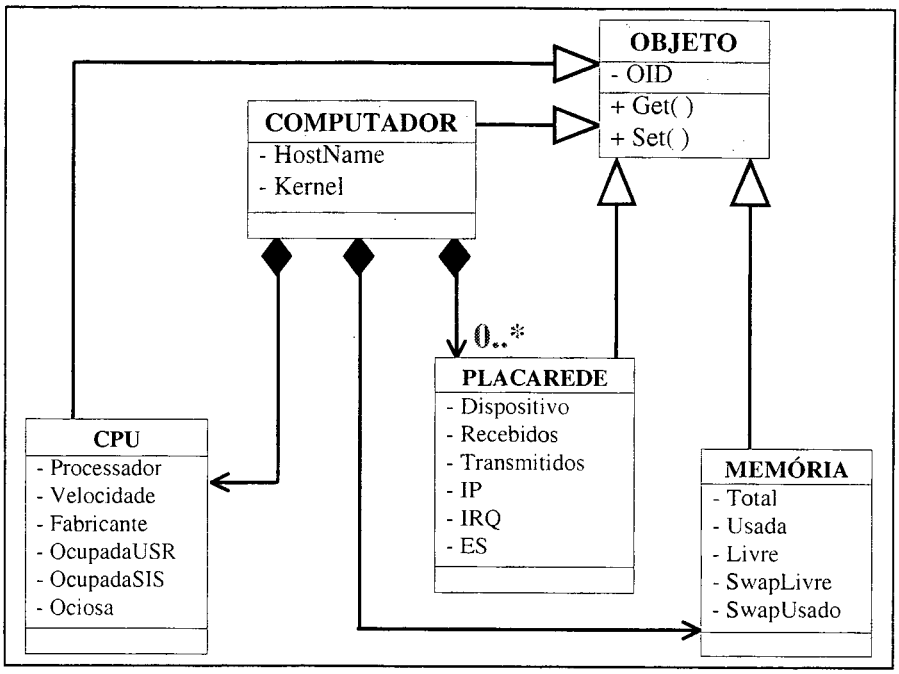

Figura 7: Diagrama de Classes - UML

Para a utilização das informações de gerenciamento identificadas, deve-se criar uma MIB que suporte as informações desejadas ou então utilizar uma MIB já disponível. A MIB a ser criada pode ser Padronizada ou Não Padronizada.

A seguir será apresentado um resumo para a criação de uma MIB Padronizada. Mais informações sobre conceitos e o processo de desenvolvimento de uma MIB podem ser obtidas em WEBBER (1997) e MILLER (1999).

4.2 PROJETO DE UMA MIB

Quando um novo projeto de MIB é abordado, é importante ter em mente algumas categorias de objetos, para pensar no problema de uma forma organizada (PERKINS, 1997 *apud* WEBBER, 1997):

- **Ações**: controlam um sistema. Objetivos do tipo ação são usados para permitir a execução de tarefas bem definidas, como: reinicializar um dispositivo, desativar um serviço de rede, etc.;
- **Estatísticas**: fornecem informação útil sobre o que aconteceu no sistema desde o início de um certo intervalo de tempo. Estatísticas podem incluir

itens como: o número de pacotes transmitidos por uma *interface* de rede, ou o número de vezes que um usuário se conectou a uma máquina.

- **Estados**: indicam o estado corrente de um sistema. Estado podem incluir itens como: uma placa está inicializando, ou um ar-condicionado está com defeito;
- **Componentes**: ajudam a descrever conjuntos de dispositivos físicos e lógicos, ou serviços que estão sob controle do agente SNMP. Por exemplo, as placas presentes em um sistema de chassis com múltiplos *slots*, ou ainda os nomes dos serviços ativos em um servidor de arquivos;
- **Atributos**: são as propriedades de um objeto modelado, que descrevem coisas relativamente estática sobre um dispositivo ou serviço. Exemplo: o número de portas em um *hub Ethernet*, ou a pessoa a chamar em caso de falha de um dispositivo, ou ainda que tipo de CPU está instalada no sistema.

Os passos necessários para projetar uma MIB são apresentados na Figura 8, onde em cada estágio, um conjunto de documentos precisa ser produzido (PERKINS, 1997 *apud* WEBBER, 1997):

- **Primeira fase**: descreve os objetos gerenciáveis que serão modelados na MIB;
- **Segunda fase**: consiste em estabelecer a base geral de informações, e decidir como organizar as definições de objetos em um mais módulos de MIB;
- **Terceira fase**: é onde são criadas as definições de objetos gerenciados.

As duas últimas fases produzem módulos de informação que consistem em respostas à MIB recém-criada.

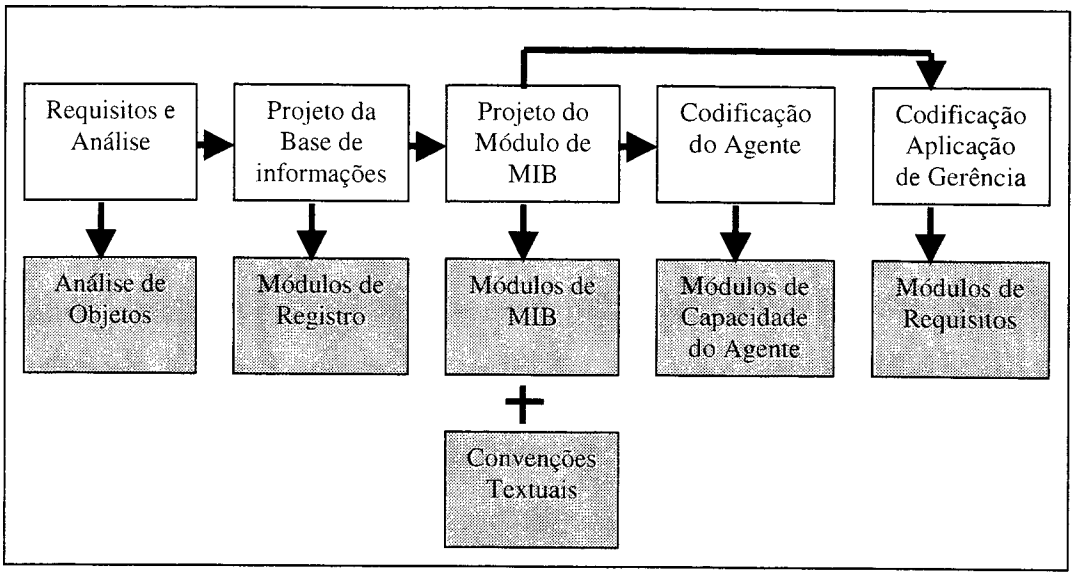

Figura 8: Passos no Desenvolvimento de uma MIB

Cada objeto identificado deverá conter um identificador único chamado OID (*Objetct Identifier*). Em uma MIB, a estrutura de identificadores de objetos (OID) devem ser organizados dentro do espaço de OID sob responsabilidade da empresa. Os ramos superiores da árvore de registro de OID são definidas pelos órgãos de padronização e são fixos (WEBBER, 1997). A Figura 9 ilustra a árvore de objetos da MIB-Linux.

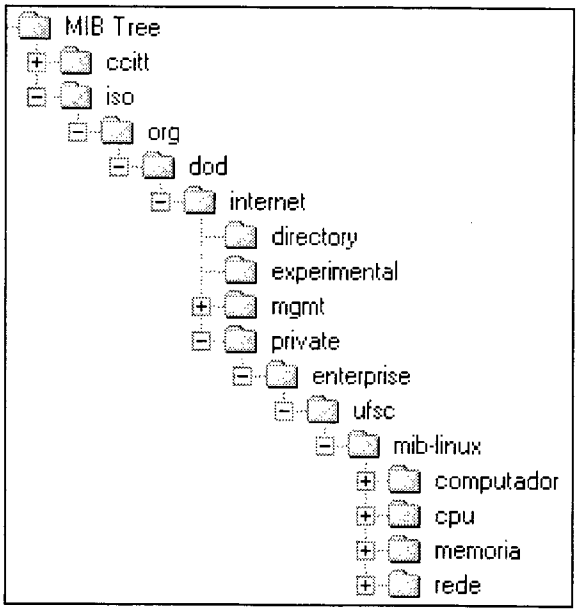

Figura 9: Grupo de objetos da MIB-LINUX

WEBBER (1997) sugere uma MIB, chamada de *Network Client Application MIB* (MIB para Aplicações Cliente de Rede). Esta MIB é definida na árvore, como uma sub-árvore de Private-Enterprise-Ufsc. Seguindo as sugestões de WEBBER (1997), a MIB Linux foi colocada na mesma sub-árvore. Assim, os objetos identificados pela empresa podem ser colocados com uma sub-árvore do OID da empresa. Desta forma o objeto Velocidade do Processador corresponderia ao OID 2.2 dentro da estrutura de objetos da empresa.

Novos objetos podem ser adicionados a uma MIB SNMP através de três maneiras diferentes (STALLING, 1993 *apud* WEBBER, 1997):

- A sub-árvore MIB-II pode ser expandida ou completamente trocada por uma nova revisão (provavelmente seria chamada de MIB-III. Por exemplo a RMON (*Remote Monitoring*) MIB (WALDBUSSER, 1995);
- Uma MIB experimental pode ser construída para uma aplicação particular. Tais objetos podem ser subsequente movidos para a sub-árvore MGMT.
- Extensões privadas podem ser adicionadas à sub-árvore PRIVATE.

Os objetos gerenciados possuem atributos que contém determinadas informações. A descrição dos objetos é dividida em cinco partes (PEITER, 2000):

- nome do objeto,
- sintaxe abstrata do objeto,
- descrição textual do significado do objeto,
- tipo de acesso permitido ao objeto,
- estado do objeto.

O nome do objeto é um nome textual para o tipo de objeto, denominado "Descritor de Objetos" que acompanha o identificador de objeto. A sintaxe abstrata diz respeito ao tipo de valor que o objeto armazena. A descrição textual é vital para que os objetos tenham significados consistentes. É uma descrição da semântica do objeto. O tipo de acesso permitido ao objeto pode ser para leitura, leitura e escrita ou inacessível. E por fim o estado do objeto pode ser obrigatório, opcional ou obsoleto.

A Figura 10 apresenta a descrição de alguns objetos do grupo Memória:

```
memTotal OBJECT-TYPE
       SYNTAX INTEGER
       ACCESS read-only
       STATUS mandatory
       DESCRIPTION
          "Total de memória Física (RAM)."
       ::= { memoria 1}

memUsada OBJECT-TYPE
       SYNTAX INTEGER
       ACCESS read-only
       STATUS mandatory
       DESCRIPTION
          "Quantidade de memoria Fisica usada."
       ::= { memoria 2}

memLivre OBJECT-TYPE
       SYNTAX INTEGER
       ACCESS read-only
       STATUS mandatory
       DESCRIPTION
          "Quantidade de memoria fisica livre."
       ::= { memoria 3}
```

Figura 10: Objetos do grupo Memória

Uma vez definida a MIB, um agente extensível MIB deverá ser elaborado para manipulação e armazenamento dos objetos.

Segundo PEITER (2000), os requisitos para desenvolver um agente extensível são:

- ter instalado o protocolo TCP/IP (*Transmisson Control Protocol / Internet Protocol)*;
- instalar o serviço SNMP, pois é o serviço SNMP que irá monitorar a porta 161 UDP (*User Datagram Protocol*) para verificar a chegada de mensagens SNMP;
- configurar o serviço SNMP para que as mensagens recebidas e enviadas estejam de acordo com as definições do administrador de rede;
- desenvolver uma biblioteca de funções que processe mensagens *Get Request*, *Get-Next Request*, *Set Request* sobre os dados definidos em uma MIB;

- instalar a biblioteca de funções do agente extensível para que na inicialização do serviço SNMP a biblioteca seja carregada para a memória junto com o serviço SNMP.

4.3 CONCLUSÕES

Este capítulo, demonstrou de forma resumida, os passos a serem seguidos para a construção de uma MIB Padronizada, desde a identificação dos objetos até elaboração de um agente que fará a manipulação da MIB. Dois processos que demandam tempo e um conhecimento mais especializado são a definição dos objetos em ASN.1 e a construção do Agente SNMP.

Maiores informações no processo de elaborar uma MIB Padronizada, podem ser obtidos em WEBBER (1997) ou PEITER (2000).

Foi elaborado também uma MIB Não Padronizada, através de um programa escrito em Linguagem C, que tem o objetivo de coletar as informações e gravá-las em um arquivo texto. Como estas informações devem estar atualizadas para serem disponibilizadas para consulta, usou-se um recurso do sistema operacional Linux conhecido como CRONTAB, que tem a função de agendamento de tarefas. Foi então definido que a cada 5 minutos o programa de coleta seria inicializado automaticamente.

No próximo capítulo serão apresentados os conceitos e a estrutura do XML e principalmente, como será feito a conversão das informações da MIB para o formato XML.

5 XML

XML é uma linguagem de conteúdo, definida em 1996 pelo W3C (*World Wide Web Consortium*), derivada da SGML (*Standard Generalized Markup Language*), um padrão internacional ISO/IEC 8879, para a definição de estruturas e conteúdos de documentos eletrônicos (BARBIERI, 2000) (HOLZNER, 2001) (CARLSON, 2001) (BOSAK, 1997). Em fevereiro de 1998, o W3C lançou a especificação 1.0 da XML (HOLZNER, 2001).

O XML foi inicialmente considerado um substituto do HTML (*HiperText Markup Language*), pois este apresentava várias limitações referentes a criação de novos elementos, reutilização de documentos e na relação conteúdo-informação-apresentação. XML difere de HTML por 3 aspectos principais (BOSAK, 1997):

1. Provedores de informação podem definir novas *tags* e nomes de atributos;
2. Estruturas de documentos podem ser aninhadas para qualquer nível de complexidade;
3. Qualquer documento XML pode conter uma descrição opcional de sua gramática para uso de aplicações que necessitam desenvolver uma estrutura de validação.

Os arquivos XML são criados e desenvolvidos, sem que seja necessário conhecer como o conteúdo é processado ou apresentado. XML também permite que um mesmo documento seja produzido em várias mídias e formatos distintos (BARBIERI, 2000).

As aplicações usadas com XML, podem ser divididas em 4 categorias, segundo BOSAK (1997):

- Aplicações que requerem um cliente Web para mediador entre duas ou mais bases de dados heterogêneas;
- Aplicações que tentam fazer algo para distribuir uma significativa proporção de processamento carregadas de um Servidor Web ou um Cliente Web;

- Aplicações que requerem um Cliente Web para apresentar diferentes visões dos mesmos dados, para usuários diferentes;
- Aplicações em que cada Agente Inteligente Web faz algo para descobrir informações sobre medida para as necessidades dos usuários individualmente.

Segundo LEIVA ET. AL. (1999), um documento XML é composto por (Figura 11):

- **Conteúdo**: é o arquivo DTD, que possui a estrutura das informações, especificando a correta sintaxe do documento;
- **Informação**: é o arquivo XML, que possui as informações estruturadas conforme o DTD.
- **Apresentação**: são os arquivos XSL (*Extensible Stylesheet Language*) e CSS (*Cascading Style Sheet*), usados para definir como as informações serão apresentadas. As folhas de estilo, herdadas, permitem que a partir de um arquivo XML

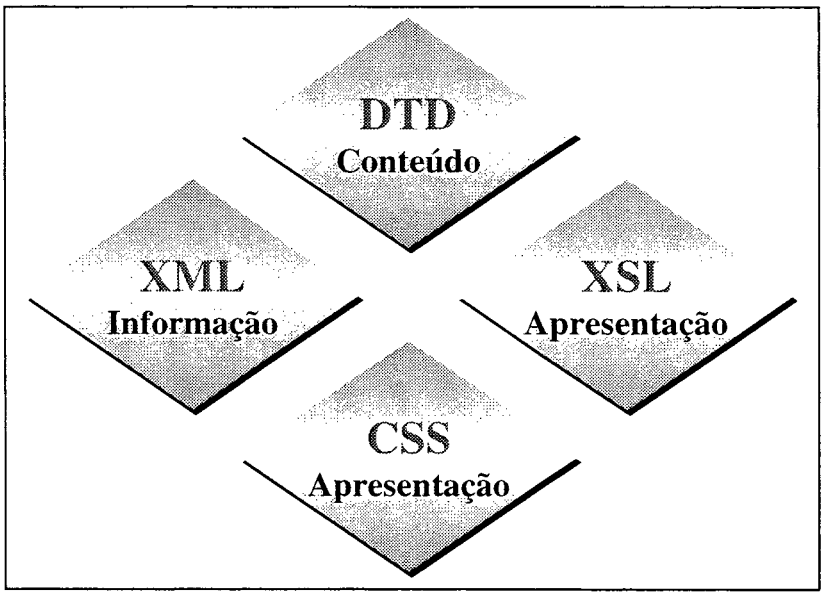

Figura 11: Composição de um documento XML

Os elementos DTD e XML serão discutidos mais a frente neste capítulo, e os elementos XSL e CSS serão sugeridos como trabalhos futuros na seção 7.1. O desenvolvimento deste projeto, foi baseado na especificação 1.0 da XML (BRAY,

1998), que inclui o DTD com linguagem de definição de dados. Mas também é possível usar o XML *Schema*, que segundo HOLZNER (2001), ainda está em estágio de *Draft*, mas brevemente será alterado para o status de recomendação.

O XML *Schema* supre algumas limitações do DTD, como por exemplo o uso de tipos de dados como *Integer*, *Float*, *Boolean*, *Data*, etc, bem como criar tipos de dados. Além de apresentar alguns parâmetros que permitem estabelecer maior segurança às informações.

Para CARLSON (2001), XML *Schema* é uma completa reconceitualização dos tipos de definições em um documento XML, que inclui uma linguagem de definição de dados junto com uma linguagem de especificação estrutural. CARLSON (2001) ainda afirma que um XML *Schema* não é requerido para o padrão XML.

O XML *Schema* não será abordado neste projeto, sendo apresentado como trabalhos futuros na seção 7.1.

5.1 CARACTERÍSTICAS

SILVA (2000), sugere que as principais características da XML, podem ser divididas em:

- **Simplicidade**: A linguagem XML provê tanto a programadores como autores de documentos um ambiente muito mais amigável do ponto de vista da computação. O rígido conjunto de regras da XML ajuda a tornar documentos mais fáceis de serem lidos quer seja por humanos, quer seja por máquinas. A sintaxe de documentos em XML contém um relativamente pequeno conjunto de regras, possibilitando aos desenvolvedores uma simples iniciação na linguagem. DTDs podem ser desenvolvidos através de processos padronizados.

- **Extensibilidade**: A linguagem XML é extensível de dois modos. Primeiro, ela permite aos desenvolvedores a criação de seus próprios DTDs, efetivamente criando conjuntos marcas "extensíveis" que podem ser usadas para múltiplas aplicações. A XML é uma meta-linguagem, ou seja, um conjunto de regras que pode ser utilizado para a criação de conjuntos de regras para documentos..

- **Interoperabilidade**: A XML pode ser utilizada sobre várias plataformas e interpretada por várias ferramentas. Devido ao comportamento consistente das estruturas de documentos, os navegadores que as interpretam podem ser desenvolvidos a um relativamente baixo custo em quaisquer de um grande número de linguagens para este fim. A XML suporta um grande número de padrões para a codificação de caracteres, permitindo que ela possa ser usada em um grande número de diferentes ambientes.
- **Abertura**: Todos os avanços obtidos pelo XML *Working Group* do W3C costumam ser levados a público, possibilitando o acompanhamento de seus progressos. É relativamente fácil a interpretação de um documento XML e sua validação requer apenas que seja providenciado um DTD adequado. Ao criar um documento XML que se comporte de maneira específica em relação as suas aplicações, os dados nele incluídos estarão disponíveis para quaisquer outras aplicações.
- **Experiência**: O desenvolvimento de aplicações de sucesso para a XML vão requerer um conhecimento na modelagem de um novo conjunto de ferramental. As especificações iniciais da XML foram criadas praticamente pelos integrantes do grupo SGML, o que significa que as companhias que trabalham com SGML procurarão se adaptar para o uso de XML.

5.2 APLICAÇÕES

Segundo BARBIERI (2000), XML tem potencial de uso em duas grandes áreas:

- **Primeira**, seria a possibilidade de usar como uma linguagem universal para a definição de documentos, com sintaxes específicas para os mais variados domínios de conhecimento (matemática, química, comércio, etc.);
- **Segunda**, pela utilização como transportador de documentos no emergente segmento de comércio eletrônico, atuando na integração de aplicações e nos sistemas B2B (*Business to Business*).

Para BARBIERI (2000), XML se apresenta como uma nova forma de *middleware*, que habitará um servidor do seu ambiente Web e se relacionará com o

servidor Web clássico, com o servidor de aplicações e com o servidor de banco de dados. Por exemplo, os pedidos que chegam ou as notas fiscais que saem poderão ser dinamicamente produzidas de um lado e plenamente interpretadas do outro, num ambiente clássico de comércio eletrônico, sem que o comprador e o vendedor tenham que conhecer os documentos recíprocos.

BRAY (1996), justifica a razão pela qual o XML será adotado em aplicações Web:

1. XML será francamente utilizável pela Internet;
2. XML suportará uma grande variedade de aplicações;
3. XML será compatível com SGML;
4. Será fácil escrever programas que processem documentos XML;
5. número de características opcionais em XML é para ser mínima, ideal é zero;
6. Os documentos XML serão legíveis para os seres humanos e suficientemente claro;
7. projeto XML será preparado mais rapidamente;
8. projeto XML será formal e conciso;
9. Documentos XML serão fáceis de criar;
10. Abreviações terão importância mínima.

Algumas empresas estão adotando e apostando nesta tecnologia, como é possível perceber pelos exemplos abaixo:

- Os médicos do Instituto do Coração de São Paulo agora contam com um sistema de Prontuário Eletrônico baseado no XML. Este sistema garante maior qualidade e rapidez no fornecimento de informação ao pacientes, onde o médico tem em seu computador o histórico detalhado de consultas e exames já realizados pelo paciente (BARROS, 2000).

- A IBM lançou dois programas gratuitos usando o padrão XML: **XEENA**, um editor XML baseado em java com *interface* gráfica; e o **XML Parser For Java**, que permite criar documentos XML e ao mesmo tempo verificar a correção sintática dos comandos (RAMOS, 1999).

Veja na tabela a seguir (Tabela 1), o que significa o uso de XML para alguns profissionais (SUN, 2001):

Executivos	- Permite o intercâmbio de dados em forma baseada em texto que é compreendida pelos seres humanos, não apenas por máquinas. - Aumenta a padronização de formatos e ferramentas básicos, assegurando um conjunto maior e mais coerentemente treinado de técnicos do conhecimento. - Preserva os dados das transações dos negócios eletrônicos em um formato que é mais acessível e fácil de analisar e gerenciar do que os registros atuais de EDI. - Dá aos dados da transação uma possibilidade muito maior de permanecerem processáveis ao longo das sucessivas gerações de equipamentos e das mudanças de software.
Desenvolvedores	- Termina com difíceis discussões sobre intercâmbio de dados ao estabelecer um consenso no setor sobre como certas estruturas comuns de dados são representadas. - Permite que algumas funções básicas de processamento de dados sejam executadas por ferramentas padrão de fonte aberta e especificadas mediante o uso de *interfaces* comuns de programa de aplicativos, as API (*Application Program Interface*)
Designers de Sistemas Empresariais	- Facilita o processamento de dados nos sistemas distribuídos e heterogêneos típicos dos atuais ambientes computacionais baseados na Internet. - Promove a heterogeneidade dos sistemas empresariais. - Propicia que todas as mensagens empresariais sejam texto, tornando as transações individuais mais fáceis de ser depuradas e permitindo o processamento em lote de grandes quantidades de dados por meio de ferramentas simples.
Administradores de Sistemas	- Isola mensagens de processos subjacentes, de modo que os subsistemas são atualizados com mais facilidade e modificados sem afetar outros subsistemas. - Estende a vida útil do sistemas herdados.
Publicadores de Informações	- Responde ao problema de como gerar diferentes apresentações d informação que tem a mesma origem para diversos usuários e diversos tipos de saída: telas de computador, televisão, dispositivos manuais e, naturalmente, papel. - Possibilita a separação de dados e a apresentação que tornam a verdadeira publicação global em mídia cruzada uma realidade prática.
Usuários Finais	- Permite a entrega de dados praticamente feitos sob medida para as necessidades e os desejos dos indivíduos, inclusive a elaboração de dados sob medida para os deficientes visuais. - Leva à plena realização o comércio espontâneo que caracterizará os negócios e o ambiente do consumidor do futuro

Tabela 1: O que significa o uso de XML (SUN, 2001)

O mais importante na XML, segundo SUN (2001), é o processo por meio do qual seus vocabulários e suas formas comuns são determinados. Esse processo deve ser aberto, completamente justificável, livre de controle proprietário e representativo de todas as partes interessadas em cada setor e domínio de problema. A Sun apoia firmemente a XML para o intercâmbio de dados e está incorporando suporte à XML em todos os seus produtos e plataformas relacionadas.

5.3 ARQUIVO DTD

O arquivo DTD é a especificação da estrutura e da sintaxe de um documento XML (HOLZNER, 2001). Este arquivo é usado para validar o arquivo XML, onde são feitas checagem entre o conteúdo e a informação.

5.3.1 Estrutura

Segundo LEIVA ET.AL. (1999), um DTD contém basicamente: elementos e atributos. Uma declaração de um Elemento define o nome, a composição dos elementos filhos com seus conteúdos e opcionalmente os atributos (CARLSON, 2001). Um elemento pode declarar estruturas adicionais ao seu conteúdo. A Tabela 2 apresenta os conteúdos possíveis para um elemento.

Conteúdo	Definição
EMPTY	Nenhum elemento filho ou texto são permitidos
ANY	Qualquer tipo de elemento pode ser usado neste conteúdo
#PCDATA	São os dados caracter em formato livre de texto
Choice	Um elemento de uma lista é incluído no conteúdo (x | y)
Sequence	A sequência de elementos são incluídos na ordem especificada (x, y)
Group	Um grupo de elementos limitados por parênteses podem ser aninhados (x, (y | z)
Mixed	Elementos e texto são permitidos (#PCDATA | x | y)

Tabela 2: Conteúdo do Elementos (CARLSON, 2001)

Os elementos também podem ser especificamente limitados, quanto a restrição de multiplicidade (Tabela 3).

Multiplicidade	Definição
Nenhum	Exatamente uma instância do elemento ou grupo
?	Zero ou uma instância
*	Zero ou mais instâncias
+	Uma ou mais instâncias

Tabela 3: Multiplicidade do elemento (CARLSON, 2001)

A declaração de um elemento não suporta qualquer restrição dos dados caracter incluídos no elemento #PCDATA (*Parser Carchter Data*). Deste modo, a declaração de atributos permite uma restrição, mesmo que limitada (CARLSON, 2001). Não existem atributos do tipo *integer* ou *float*. A Tabela 4 apresenta os tipos de atributos usados em um DTD.

Atributo	Definição
CDATA	Dados formados por *string* de caracteres
ID	Identificador único do documento
IDREF	Referência para um identificador único de um documento
IDREFS	Referência múltiplos identificadores, separados por espaço em branco
NMTOKEN	Um nome composto de CDATA sem os espaços em branco
NMTOKENS	Composição de múltiplos nomes de CDATA separados por espaço em branco

Tabela 4: Tipos de atributos (CARLSON, 2001)

Os atributos podem ter valores padrões, como os apresentado na Tabela 5.

Padrão	Definição
#IMPLIED	Opcional
#REQUIRED	Requerido em cada instância
#FIXED	Não pode ser alterado em uma instância do documento

Tabela 5: Valores *default* dos atributos

Maiores detalhamentos sobre a estrutura de um arquivo DTD pode ser encontrada em HOLZNER (2001) e CARLSON (2001).

5.3.2 Criando o arquivo DTD

A utilização da informações coletadas em um documento XML, começa pela criação do arquivo DTD. Onde, para cada informação ou agrupamentos de informações, será criado um elemento. Neste caso, uma informação será especificada por **<!ELEMENT nome_do_atributo (#PCDATA)>** e um agrupamento de informações será especificada por **<!ELEMENT nome_do_agrupamento (atributo1, atributo2,...)>**.

A primeira declaração é o agrupamento mais genérico, aqui chamado de MIB. Esta declaração pode ser interpretada como: Uma MIB é composta de informações de COMPUTADOR. Em seguida é declarado o atributo COMPUTADOR, este por sua vez é composto das informações HOSTNAME, KERNEL, CPU, MEMÓRIA e PLACAREDE. Para cada atributo de COMPUTADOR, será criada uma declaração específica. Desta forma a próxima declaração é o atributo HOSTNAME, que por não ser uma agrupamento, receberá um #PCDATA, ou seja, refere-se a uma *string* contendo informação pura. Estes procedimento, ou melhor encadeamentos prosseguem até que todas as informações e agrupamentos tenha sido declarados.

As informações escolhidas, contemplam mais de uma placa de rede no computador, sendo assim, é necessário fazer um tratamento especial na declaração do elemento PLACAREDE. Este elemento receberá um símbolo de multiplicidade ao final da declaração, representado pelo asterisco. Isto pode ser interpretado como: O agrupamento PLACAREDE pode conter várias placas de rede, por este motivo um novo agrupamento foi criado e chamado PLACA, para conter as informações específicas de cada placa de rede instalada no computador.

O agrupamento PLACA então será declarado com os atributos DISPOSITIVO, RECEBIDOS, TRANSMITIDOS, IP, IRQ e ES. E estes por sua vez serão declarados com #PCDATA. A partir do Diagrama de Classes - UML é possível chegar à especificação do arquivo DTD, CARLSON (2001) sugere e descreve os caminhos para o mapeamento de CLASSE, ATRIBUTOS, COMPOSIÇÃO, HERANÇA, ASSOCIAÇÃO, etc.

INFORMAÇÕES	DTD
	<!ELEMENT **MIB** (COMPUTADOR)>
COMPUTADOR	<!ELEMENT **COMPUTADOR** (HOSTNAME, KERNEL, CPU, MEMÓRIA, PLACAREDE)>
HostName	<!ELEMENT **HOSTNAME** (#PCDATA)>
Kernel	<!ELEMENT **KERNEL** (#PCDATA)>
CPU	<!ELEMENT **CPU** (PROCESSADOR, VELOCIDADE, FABRICANTE, OCUPADAUSR, OCUPADASIS, OCIAOSA)>
Processador	<!ELEMENT **PROCESSADOR** (#PCDATA)>
Velocidade	<!ELEMENT **VELOCIDADE** (#PCDATA)>
Fabricante	<!ELEMENT **FABRICANTE** (#PCDATA)>
OcupadaUsr	<!ELEMENT **OCUPADAUSR** (#PCDATA)>
OcupadaSis	<!ELEMENT **OCUPADASIS** (#PCDATA)>
Ociosa	<!ELEMENT **OCIOSA** (#PCDATA)>
MEMÓRIA	<!ELEMENT **MEMÓRIA** (TOTAL, USADA, LIVRE, SWAPTOTAL, SWAPLIVRE)>
Total	<!ELEMENT **TOTAL** (#PCDATA)>
Usada	<!ELEMENT **USADA** (#PCDATA)>
Livre	<!ELEMENT **LIVRE** (#PCDATA)>
SwapTotal	<!ELEMENT **SWAPTOTAL** (#PCDATA)>
SwapLivre	<!ELEMENT **SWAPLIVRE** (#PCDATA)>
PLACA DE REDE	<!ELEMENT **PLACAREDE** (PLACA)*>
	<!ELEMENT **PLACA** (DISPOSITIVO, RECEBIDOS, TRANSMITIDOS, IP, IRQ, ES)>
Dispositivo	<!ELEMENT **DISPOSITIVO** (#PCDATA)>
Recebidos	<!ELEMENT **RECEBIDOS** (#PCDATA)>
Transmitidos	<!ELEMENT **TRANSMITIDOS** (#PCDATA)>
IP	<!ELEMENT **IP** (#PCDATA)>
IRQ	<!ELEMENT **IRQ** (#PCDATA)>
ES	<!ELEMENT **ES** (#PCDATA)>

Tabela 6: Criação do arquivo DTD

5.4 ARQUIVO XML

Uma vez definida a estrutura de um documento XML, através do arquivo DTD, o próximo passo é a criação do arquivo XML, ou seja, o arquivo que contém as informações propriamente ditas. Estas informações são coletadas pelos agentes criados para esta tarefa, a partir da MIB Padronizada, nesta caso será necessária utilização de requisições SNMP, ou a partir de MIB Não Padronizada, onde as informações são extraídas de um arquivo binário.

Com a definição da informações descritas no arquivo DTD e as informações armazenadas na MIB (Figura 12), gera-se de forma automatizada o arquivo XML.

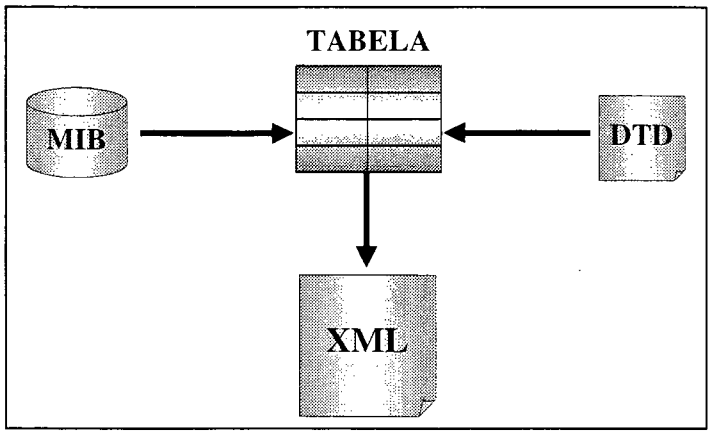

Figura 12: Esquema de geração do arquivo XML

5.4.1 Atribuição de Identificadores

Inicialmente é necessário uma padronização das informações, neste caso, cada informação ou agrupamento de informações receberá um identificador

Caso esteja trabalhando com uma MIB Padronizada, deve-se utilizar o arquivo ASN.1 (*Abstract Syntax Notation One*) relativo a definição da MIB e seguir a mesma estrutura de identificadores OID. Caso esteja trabalhando com uma MIB Não Padronizada, deve-se especificar os identificadores, conforme a estrutura criada (Figura 13).

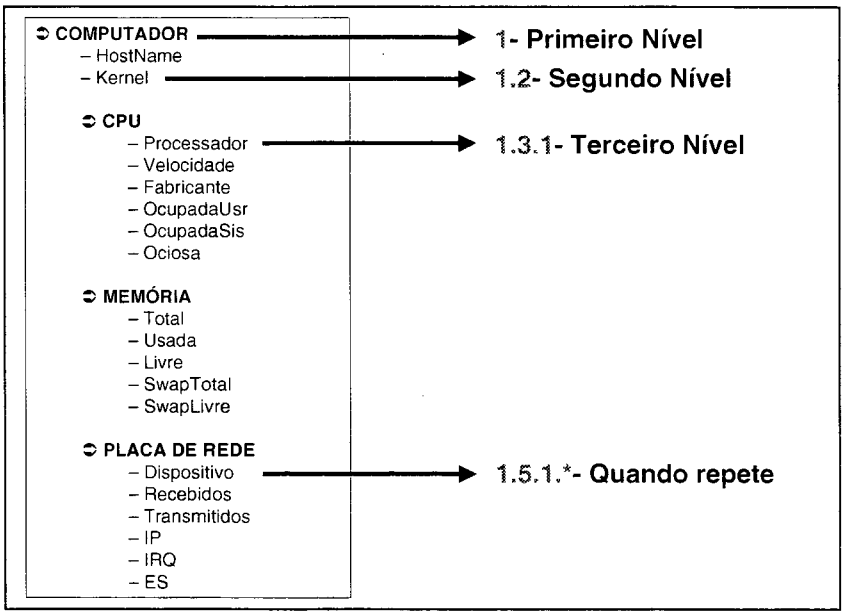

Figura 13: Atribuição de Identificadores

Em ambos os casos, as informações serão divididas em níveis, representados por valores inteiros iniciados em 1. Cada sub-nível será separado por ponto (.) e será recomeçado a contagem em 1.

As informações que repetem, serão representados por asterisco (*), simbolizando a existência de várias ocorrências.

Considerando a MIB criada, estrutura de identificadores a apresentada na Figura 14.

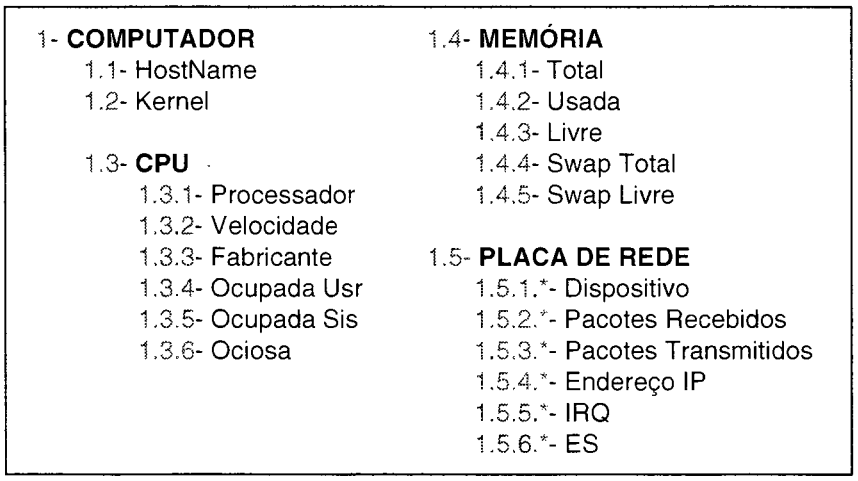

Figura 14: Estrutura de identificadores na MIB Não Padronizada

5.4.2 SCRIPT e COLETA

Após definir a estrutura de identificadores, o próximo passo é criar o arquivo SCRIPT que associará cada elemento #PCDATA do arquivo DTD à um identificador.

Na utilização de uma MIB Não Padronizada, o arquivo de coleta também deverá ser criado com a utilização dos identificadores estabelecidos. No caso de uma MIB Padronizada, a informação será extraída diretamente da MIB através de requisições SNMP.

A Tabela 7 apresenta o conteúdo do arquivo de SCRIPT.

Identificador	Elemento DTD
1.1	HOSTNAME
1.2	KERNEL
1.3.1	PROCESSADOR
1.3.2	VELOCIDADE
1.3.3	FABRICANTE
1.3.4	OCUPADAUSR
1.3.5	OCUPADASIS
1.3.6	OCIOSA
1.4.1	TOTAL
1.4.2	USADA
1.4.3	LIVRE
1.4.4	SWAPTOTAL
1.4.5	SWAPLIVRE
1.5.1.*	DISPOSITIVO
1.5.2.*	RECEBIDOS
1.5.3.*	TRANSMITIDOS
1.5.4.*	IP
1.5.5.*	IRQ
1.5.6.*	ES

Tabela 7: Conteúdo do arquivo SCRIPT

A Tabela 8 apresenta um exemplo do arquivo de coleta.

Identificador	Informação
1.1	Cerebro.andrey.com.br
1.2	2.40
1.3.1	Pentium 75 - 200
1.3.2	100.228
1.3.3	GenuineIntel
1.3.4	2.9%
1.3.5	1.9%
1.3.6	95.9%
1.4.1	65116K
1.4.2	61576K
1.4.3	3540K
1.4.4	104384K
1.4.5	104384K
1.5.1.1	eth0
1.5.2.1	1661
1.5.3.1	11
1.5.4.1	198.168.0.184
1.5.5.1	11
1.5.6.1	0x6100
1.5.1.2	eth1
1.5.2.2	0
1.5.3.2	0
1.5.4.2	198.168.0.2
1.5.5.2	9
1.5.6.2	0x6200

Tabela 8: Conteúdo do arquivo de COLETA

As informações disponibilizadas no arquivo de coletas, deverá ser acrescido do identificador, para que seja possível o relacionamento da informação do elemento DTD com a informação coletada.

5.4.3 Mapeamento XML-MIB

Agora que os arquivos SCRIPT e COLETA estão devidamente elaborados, o próximo passo é a geração do arquivo XML, a partir da estrutura do arquivo DTD, onde são usados os arquivos SCRIPT e COLETA para o relacionamento entre o conteúdo e a informação, como é mostrado na Figura 15.

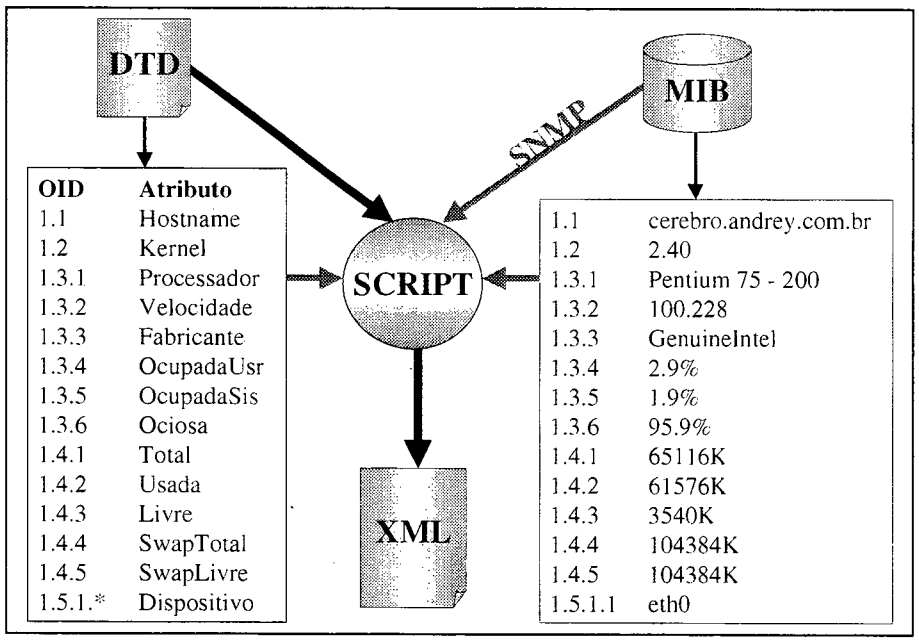

Figura 15: Mapeamento XML - MIB

Basicamente, para cada elemento declarado no arquivo DTD, será criado uma entrada no arquivo XML.

Quando o elemento for #PCDATA, será acionada uma função que buscará o valor da informação na MIB, onde a função procurará pelo nome do elemento no arquivo SCRIPT, e este informará a qual identificador está associado ao elemento. Com o valor do identificador a função então busca o valor do elemento no arquivo de COLETA ou na MIB SNMP através da primitiva GET.

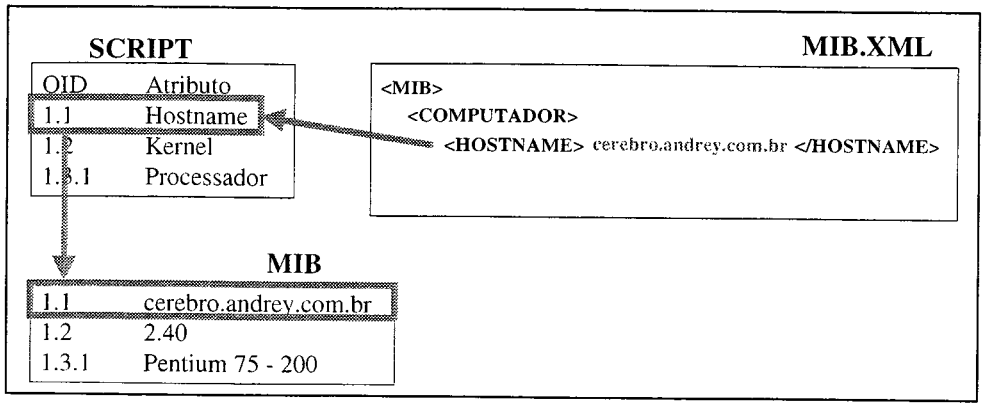

Figura 16: Relacionamento SCRIPT - MIB

Ao final do processo de construção do arquivo XML, tem-se o arquivo com o seguinte conteúdo (Figura 17).

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE MIB SYSTEM "mib.dtd">
<MIB>
        <COMPUTADOR>
                <HOSTNAME>cerebro.andrey.com.br</HOSTNAME>
                <KERNEL>2.40</KERNEL>
                <CPU>
                        <PROCESSADOR>Pentium 75 - 200</PROCESSADOR>
                        <VELOCIDADE>100.228</VELOCIDADE>
                        <FABRICANTE>GenuineIntel</FABRICANTE>
                        <OCUPADAUSR>2.9%</OCUPADAUSR>
                        <OCUPADASIS>1.9%</OCUPADASIS>
                        <OCIOSA>95.9%</OCIOSA>
                </CPU>
                <MEMORIA>
                        <TOTAL>65116K</TOTAL>
                        <USADA>61576K</USADA>
                        <LIVRE>3540K</LIVRE>
                        <SWAPTOTAL>104384K</SWAPTOTAL>
                        <SWAPLIVRE>104384K</SWAPLIVRE>
                </MEMORIA>
                <PLACAREDE>
                        <PLACA>
                                <DISPOSITIVO>eth0</DISPOSITIVO>
                                <RECEBIDOS>1661</RECEBIDOS>
                                <TRANSMITIDOS>11</TRANSMITIDOS>
                                <IP>198.168.0.184</IP>
                                <IRQ>11</IRQ>
                                <ES>0X6100</ES>
                        </PLACA>
                        <PLACA>
                                <DISPOSITIVO>eth1</DISPOSITIVO>
                                <RECEBIDOS>0</RECEBIDOS>
                                <TRANSMITIDOS>0</TRANSMITIDOS>
                                <IP>198.168.0.2</IP>
                                <IRQ>9</IRQ>
                                <ES>0X6200</ES>
                        </PLACA>
                </PLACAREDE>
        </COMPUTADOR>
</MIB>
```

Figura 17: Conteúdo do arquivo MIB.XML

Após o arquivo criado, é possível fazer uma validação de seu conteúdo, através da verificação de consistência entre o arquivo DTD e o arquivo XML. O processo de validação é apresentado na seção 6.2.

5.4.4 Programação e Consultas

Para desenvolver e manipular documentos XML existem vários mecanismos (Tabela 9), apresentados por TERUSKIN (2000).

Linguagem de Consulta (*Query*)	• XML-QL • XQL
Navegação pelo documento	• Xpath • Xlink • Xpointer
Apresentação	• XSL • CSS
Interface de Programação	• SAX (*Simple API for XML*) • DOM (*Document Object Model*)
Programação Distribuída	• WIDL • XML-RPC

Tabela 9: Programação e Consulta com XML

A aplicação proposta neste projeto, descrita no capítulo 6, foi elaborada usando DOM. Outras interfaces de programação como o SAX, poderiam ser usadas, porém não serão abordadas no projeto. Mais informações sobre os objetos, métodos e propriedades podem ser obtidos em HOLZNER (2001).

DOM é uma interface de programação padronizada pela W3C, que fornece meios de manipular um documento como uma árvore de nodos (HOLZNER, 2001). O DOM, permite processar e interpretar documentos XML.

No DOM, todo dado é tratado como um nodo. E segundo HOLZNER (2001), os tipos de nodos possíveis são:

- Elemento
- Atributo
- Texto
- Seção CDATA
- Referência de Entidades
- Entidade
- Instruções de Processamento
- Comentário
- Documento
- Tipo de Documento
- Fragmento de Documento
- Notação

5.5 CONCLUSÕES

O processo de mapeamento de uma MIB para um arquivo XML, pode ser feito de forma automatizada.

Para o caso de uma MIB Padronizada, pode-se converter o arquivo da MIB descrita em ASN.1, para o arquivo DTD, pois já é conhecido os parâmetros dos objetos. A partir do arquivo DTD, é possível construir o arquivo SCRIPT, pois para cada elemento #PCDATA, utiliza-se o OID ASN.1 correspondente. Também a partir do arquivo DTD, constróe-se o arquivo XML, onde busca-se as informações através da primitiva GET. É necessário neste caso, conhecer o valor da informação Community.

Verificou-se neste capítulo a viabilidade de se usar e representar uma MIB, usando arquivos XML e DTD. A seguir no próximo capítulo, é apresentada uma aplicação Web, que tem o objetivo de validar o projeto proposto.

6 APLICAÇÃO

Para a verificação e validação do modelo de gerenciamento proposto, foi utilizado um ambiente de testes composto por dois microcomputadores interligados em rede local TCP/IP, sendo que ambos possuíam *dual boot* com os sistemas operacionais Windows98 / Conectiva Linux 6.0. Os computadores eram compostos de Processador Pentium 200MMX, 64 Mb de memória RAM e aproximadamente 10 Gb de disco rígido.

Inicialmente foi testado o gerenciamento da MIB Não Padronizada, ou seja a MIB criada para manipular e armazenar informações, porém sem nenhuma padronização. Outra característica da MIB Não Padronizada é de não comunicar-se usando SNMP.

Um dos computadores ficou designado para ser o agente XML, e atuará como servidor. Por este motivo, foi configurado no ambiente Linux o serviço HTTP (*HiperText Transfer Protocol*). Este serviço possibilita que outros computadores possam fazer acesso através deste protocolo, usando um browser de navegação pela *Internet*, como por exemplo o Netscape e o Internet Explorer.

Segundo MILLER (1999), o gerenciamento de sistemas baseado na Web - WBEM (*Web-Base Enterprise Management*) consiste de algumas páginas que armazenam informações tipicamente formatadas usando HTML, neste caso os cliente acessam as informações usando um navegador Web. O protocolo de comunicação entre o servidor e o cliente é o HTTP, um protocolo orientado à conexão que usa TCP. MILLER (1999) justifica ainda a vantagem da independência de plataforma em contra partida ao aumento de tráfego na Internet.

Foi instalado então o pacote rpm Apache que acompanha o CD de instalação do Conectiva Linux 6.0, a instalação deste pacote cria os diretórios:

- **/etc/httpd** - para arquivos de configuração do serviço httpd;
- **var/log/httpd** - onde são registrados os logs de utilização do serviço;
- **/home/httpd** - onde são armazenados os conteúdos que serão disponibilizados.

Para que o servidor possa responder as requisições pela porta HTTP (80/tcp) é necessário que o serviço httpd seja inicializado. Isto pode ser feito com o comando **/etc/rc.d/init.d/httpd start** na linha de comando do console.

Testa-se o serviço, para verificar se o mesmo está disponível e respondendo as requisições, é através da URL **http://9.9.9.9**, onde o **9.9.9.9** é o endereço IP do computador que se quer acessar.

Este computador ainda ficou com o agente de coleta de informações em execução, para que de tempos em tempos fossem coletados as informações desejadas do computador e as mesmas fossem disponibilizadas para consultas. A disponibilização das informações ocorre através do arquivo MIB.XML, criado a partir do *Script* de geração de arquivos XML descrito na seção 5.4.2.

Uma vez o arquivo MIB.XML criado, é preciso uma aplicação para manipulá-lo. A seguir serão apresentadas as *home pages* criadas para demonstrar a utilização de informações em arquivos XML.

Foi utilizado o *browser* Internet Explorer 5.5 para utilização da aplicação e a linguagem JavaScript para criação das funções que manipulam o arquivo XML. Também poderia ser usado o browser Netscape 6.0, mas seria necessário verificar quais métodos e propriedades do objeto DOM estão disponíveis para este *browser*.

A página inicial (Figura 18) apresenta o nome do projeto, os desenvolvedores e as opções disponíveis:

- Visualizar MIB.DTD
- Visualizar MIB.XML
- Validação do Documento XML
- Browser do Documento MIB.XML
- Listar Objetos do Documento MIB.XML
- Modificar objetos do documento MIB.XML

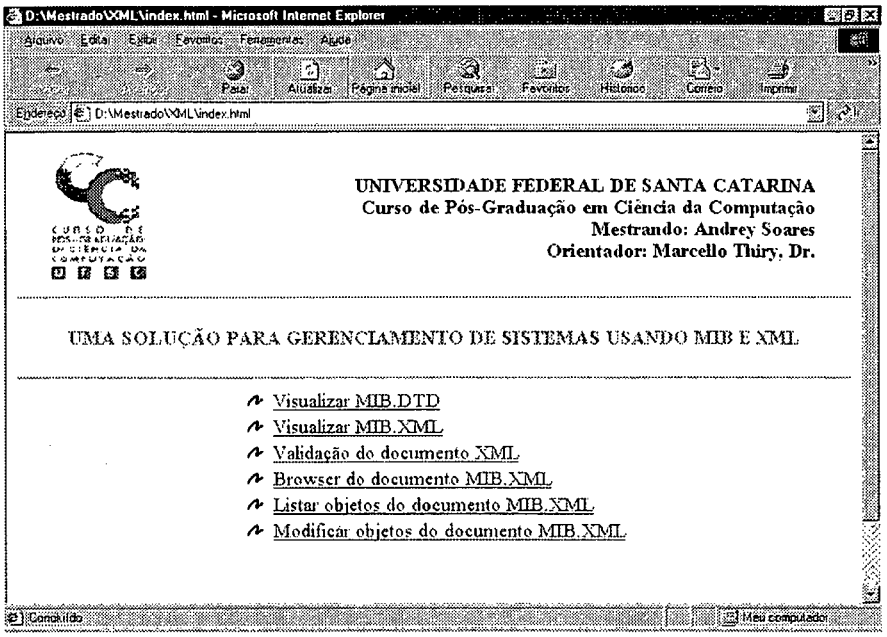

Figura 18: Página Inicial da aplicação

6.1 VISUALIZAR MIB.DTD E MIB.XML

Estas opções apresentam o conteúdo dos arquivos:

- MIB.DTD: arquivo usado para representar as informações a serem disponibilizadas pelo arquivo MIB.XML (Figura 19);

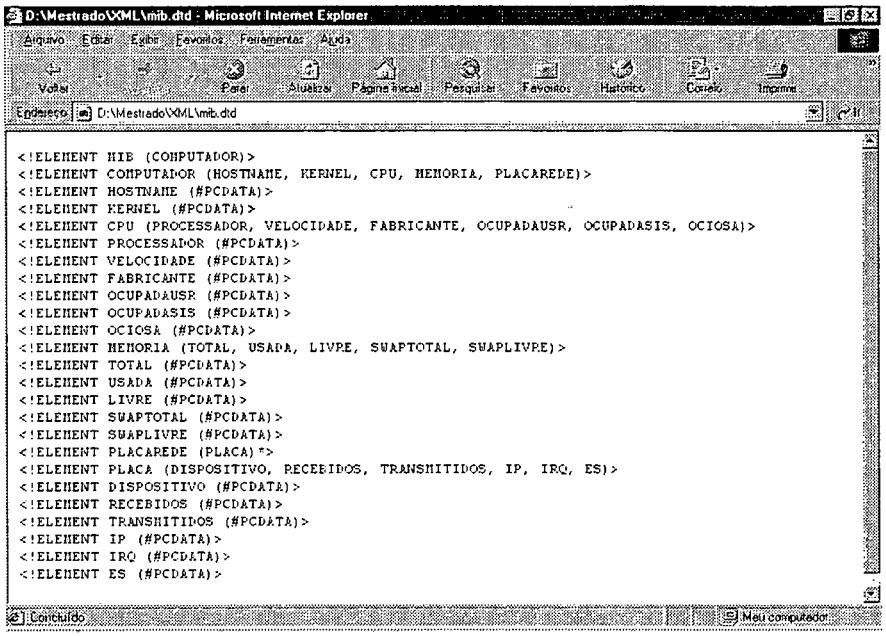

Figura 19: Conteúdo do Arquivo MIB.DTD

- MIB.XML: arquivo que contém as informações a serem disponibilizadas (Figura 20).

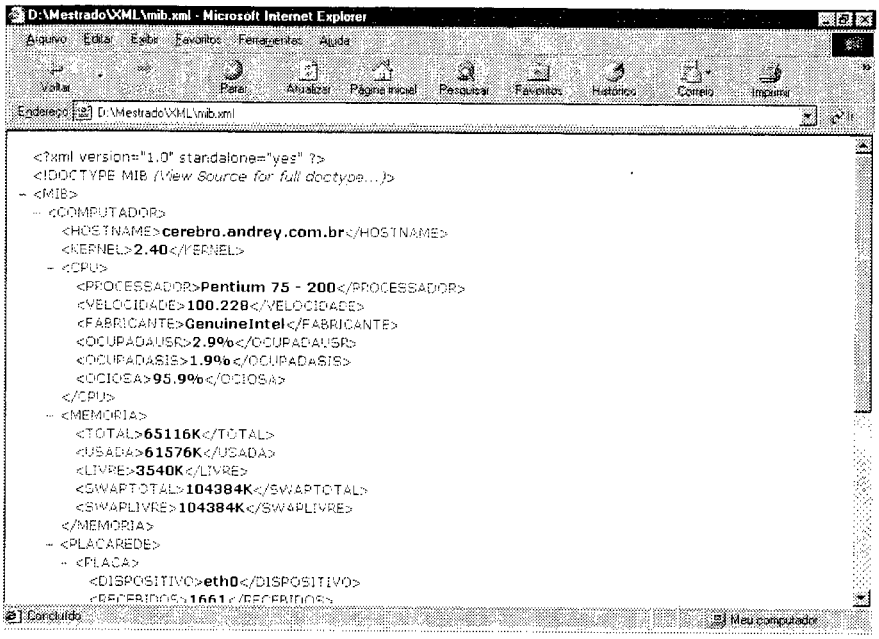

Figura 20: Conteúdo do Arquivo MIB.XML

6.2 VALIDAÇÃO DO DOCUMENTO XML

Este item foi criado para verificar e validar os conteúdos dos arquivos MIB.DTD e MIB.XML.

Como o arquivo MIB.XML é criado a partir das informações constantes na MIB e organizado conforme o esquema apresentado no arquivo MIB.DTD, é importante fazer uma verificação da consistência das informações disponibilizadas.

Para demonstrar a utilização do validador de documentos XML, foi modificado o arquivo MIB.XML, onde foi retirada a linha que apresentava informações do percentual de CPU ociosa:

```
<OCIOSA>95.9%</OCIOSA>
```

Desta forma, as informações sobre o objeto CPU ficaram com uma informação a menos.

```
<CPU>
    <PROCESSADOR>Pentium 75 - 200</PROCESSADOR>
    <VELOCIDADE>100.228</VELOCIDADE>
    <FABRICANTE>GenuineIntel</FABRICANTE>
    <OCUPADAUSR>2.9%</OCUPADAUSR>
    <OCUPADASIS>1.9%</OCUPADASIS>

</CPU>
```

Analisando o arquivo MIB.DTD, é possível verificar que a estrutura de informações esperadas para o objeto CPU é:

```
<!ELEMENT CPU (PROCESSADOR, VELOCIDADE, FABRICANTE,
OCUPADAUSR, OCUPADASIS, OCIOSA)>
```

Onde a ordem das informações deve seguir a mesma ordem apresentada anteriormente, ou seja:

```
<!ELEMENT PROCESSADOR (#PCDATA)>
<!ELEMENT VELOCIDADE (#PCDATA)>
<!ELEMENT FABRICANTE (#PCDATA)>
<!ELEMENT OCUPADAUSR (#PCDATA)>
<!ELEMENT OCUPADASIS (#PCDATA)>
<!ELEMENT OCIOSA (#PCDATA)>
```

Portanto o executar a aplicação de validação de documentos, será apresentada a mensagem de erro mostrada na Figura 21, onde serão indicados:

- **Arquivo**: nome do arquivo .XML, que está sendo analisado;
- **Linha**: linha onde foi encontrado o erro, dentro do arquivo;
- **Posição**: qual a posição do cursor, dentro da linha onde foi encontrado o erro;
- **Origem**: Mensagem informando o motivo pelo qual aconteceu o erro;
- **Erro**: informa o código do erro ocorrido.

Neste exemplo, está sendo informado que o objeto </CPU> está incompleto, segundo o esquema do arquivo DTD, ou seja está sendo esperado o item OCIOSA.

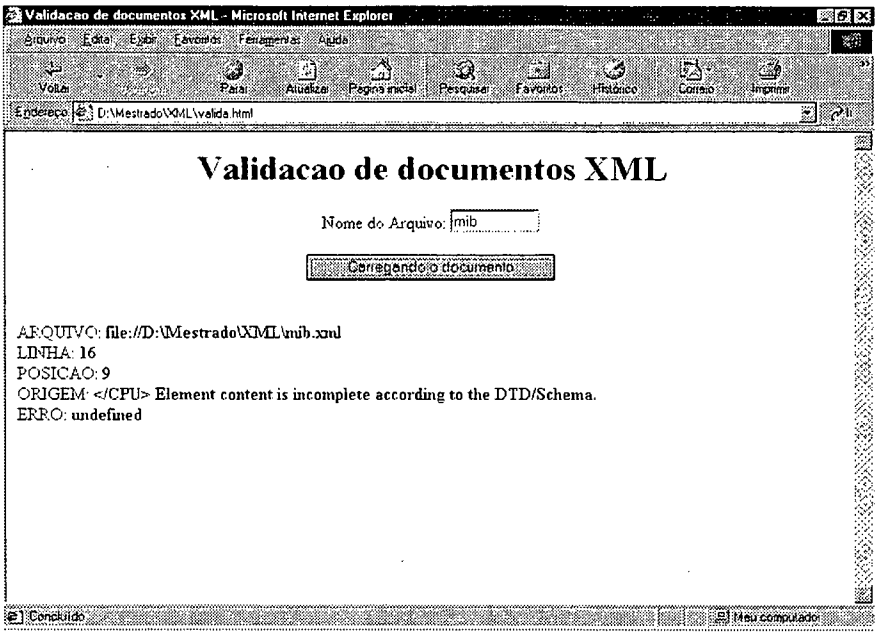

Figura 21: Erro encontrado durante a validação do Arquivo MIB.XML

Quando o arquivo está consistente entre arquivo XML e DTD, a execução do validador apresentará a mensagem que o documento está OK (Figura 22).

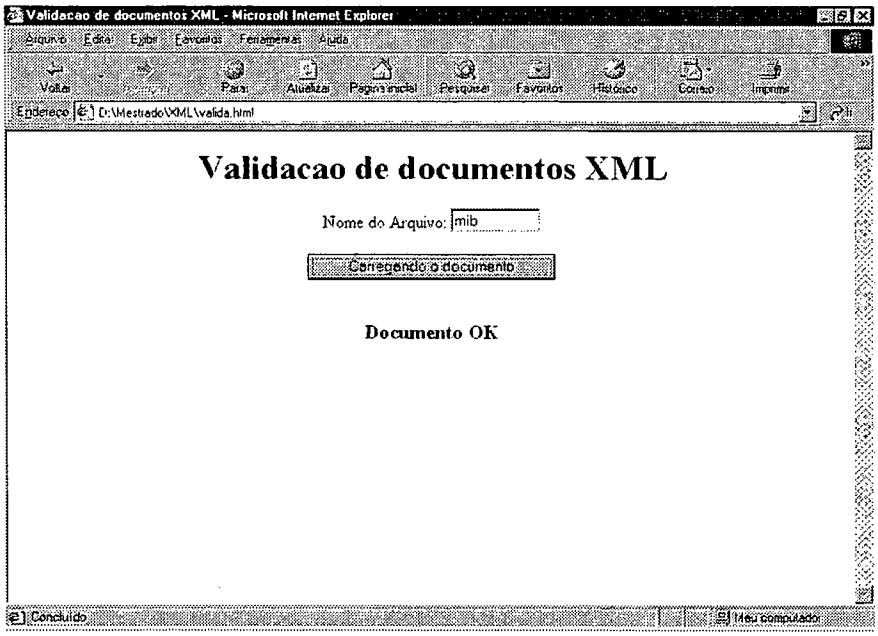

Figura 22: Validação OK do Arquivo MIB.XML

A validação de documentos consiste em checar o objeto **parseError** e suas propriedades **errorCode**, **uri**, **line**, **linepos**, **errorstring** e **reason**. O objeto parseError retorna informações detalhadas do último erro ocorrido. A Figura 23 apresenta função criada para validação do arquivo XML (HOLZNER, 2001):

```
function stateChangeHandler(){
      if(xmldoc.readyState == 4){
            var errorString = xmldoc.parseError.srcText
            errorString =
                  xmldoc.parseError.srcText.replace(/\</g, "&lt;")
            errorString = errorString.replace(/\>/g, "&gt;")

            if(xmldoc.parseError.erroCode != 0){
                  if(xmldoc.parseError.line != 0){
                        messageDIV.innerHTML =
                              " ARQUIVO: <STRONG>" +
                              xmldoc.parseError.url +
                              "</STRONG><BR>" +
                              " LINHA: <STRONG>" +
                              xmldoc.parseError.line +
                              "</STRONG><BR>" +
                              " POSICAO: <STRONG>" +
                              xmldoc.parseError.linepos +
                              "</STRONG><BR>" +
                              " ORIGEM: <STRONG>" +
                              errorString +
                              xmldoc.parseError.reason +
                              "</STRONG><BR>" +
                              " ERRO: <STRONG>" +
                              xmldoc.parseError.erroCode +
                              "</STRONG>"
                  }else{
                        messageDIV.innerHTML =
                        "<H3><CENTER>Documento
                        OK</CENTER></H3><BR>"
                  }
            }
      }
}
```

Figura 23: Função de validação dos arquivos XML e DTD

6.3 BROWSER DO DOCUMENTO MIB.XML

Esta opção foi elaborada para apresentar o valor das variáveis da MIB. Funcionando como uma MIB *Browser*, com a diferença de estar lendo informações de uma arquivo XML.

A Figura 24 mostra a tela do browser criado, onde são apresentados todas as variáveis da MIB e seus respectivos valores.

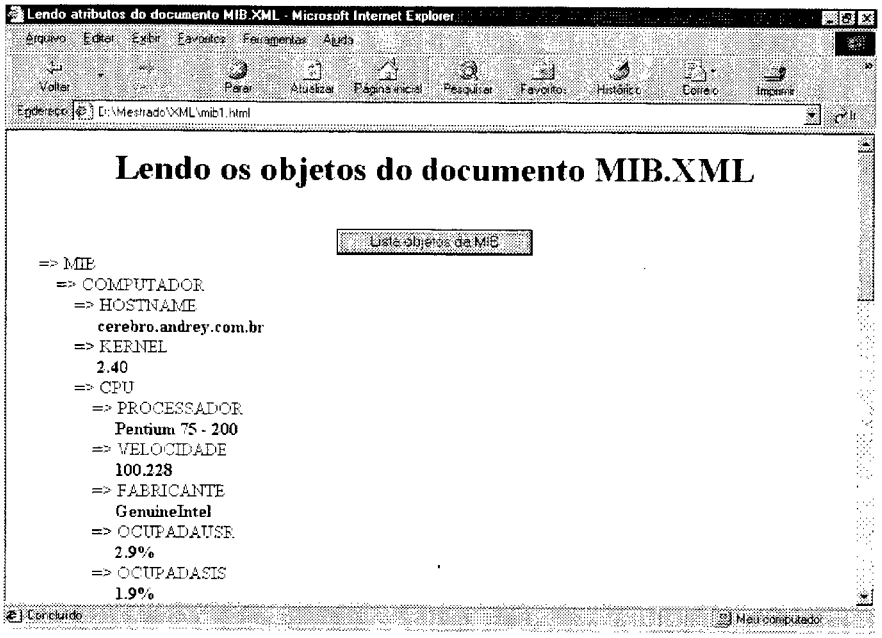

Figura 24: Mostra os objetos do documento MIB.XML

A Figura 25 apresenta a função usada para listar os elementos do documento XML (HOLZNER, 2001). A função usa recursividade para acessar todos os elementos do documento:

```
function listaObjetos(theNode, indentSpacing){
      var text
      if(theNode.nodeType < 7) {
            if(theNode.nodeValue != null) {
                  text = "<STRONG>" + indentSpacing
                        + "  " + theNode.nodeValue
                        + "</STRONG><BR>"
            }else{text = indentSpacing
                        + " => " + theNode.nodeName + "<BR>"
            }
      }else{text = ""
      }
      if(theNode.childNodes.length > 0){
            for(var Index = 0;Index < theNode.childNodes.length;
                  Index++){
                  text+=listaObjetos(theNode.childNodes(Index),
                        indentSpacing+"    ")
            }
            return text
      }
}
```

Figura 25: Função para listagem dos objetos XML

6.4 LISTA OBJETOS DO DOCUMENTO MIB.XML

Segundo WEBBER (1997), o gerente de redes não deseja ter que chamar um MIB Browser para começar a inspecionar as variáveis. Neste caso, seria mais prático o gerente solicitar informações de uma variável específica, pois caso contrário seria necessário navegar pelas variáveis até chegar ao elemento desejado.

Desta forma, foi elaborado a opção para consulta do valor de um elemento específico como mostra a Figura 26.

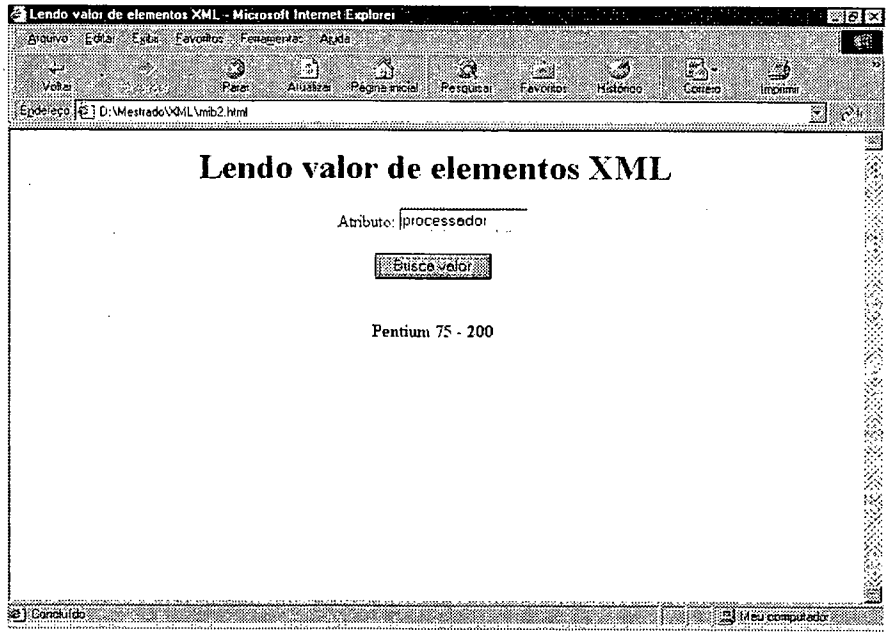

Figura 26: Lendo valor de uma elemento XML

Para a elaboração desta função foi usado o método **getElementByTag** que retorna a lista de objetos do elemento passado como argumento. O argumento é informado pelo usuário.

Em seguida o valor do objeto é retornado com o uso das propriedade **firstChild.nodeValue** e o valor é disponibilizado na tela. Se o objeto não possuir um campo com valor, ou seja se for um campo "pai", será apresentado o valor **null**.

A Figura 27 apresenta a descrição da função usada para ler o valor de um elemento em um documento XML.

```
function loadDocument(){
      var xmldoc, listNodes

      xmldoc = new ActiveXObject("Microsoft.XMLDOM")
      xmldoc.load("mib.xml")

      if(document.form1.T1.value != ""){
            listNodes =
                  xmldoc.getElementsByTagName(document.form1.T1.va
            lue.toUpperCase())

            outputText = listNodes.item(0).firstChild.nodeValue
               messageDIV.innerHTML = "<STRONG>" +
                  outputText + "</STRONG>"
      }else{
            messageDIV.innerHTML =
            "<STRONG>Informe o atributo desejado</STRONG>"
      }
}
```

Figura 27: Função para ler o valor de um elemento em um documento XML

6.5 MODIFICA OBJETOS DO DOCUMENTO MIB.XML

Algumas variáveis da MIB podem ser usadas como controle, desta forma, o usuário poderia fechar uma aplicação, desabilitar a porta de um *Hub*, etc.

Para isto, é necessário modificar o valor das variáveis. A Figura 28 mostra a tela que possibilita a modificação do valor de uma variável.

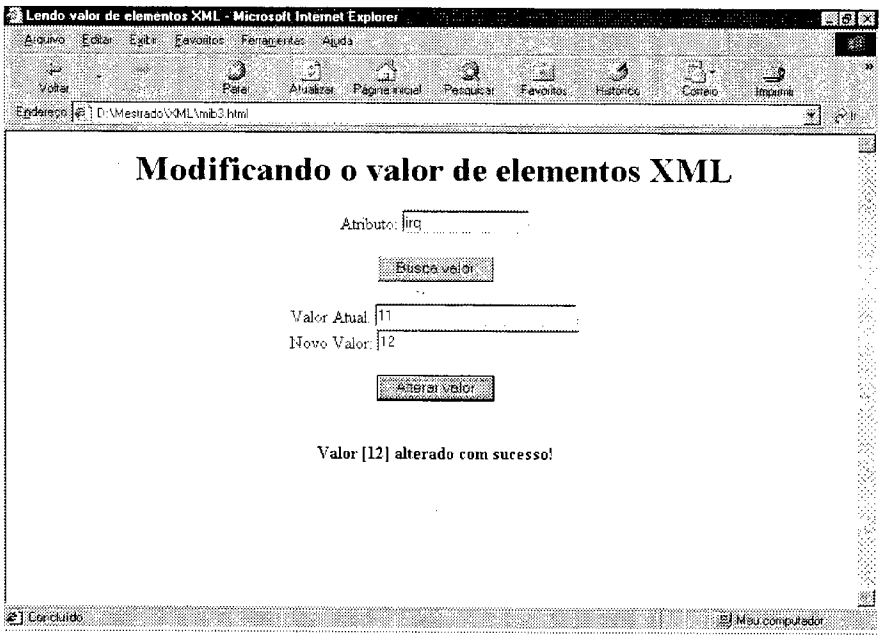

Figura 28: Modificando o valor de um elemento XML

A Figura 29 apresenta a função que modifica o valor de um elemento XML (HOLZNER, 2001). O método **getElementByTag** com já foi visto, retorna a lista de objetos do elemento passado como argumento, o método **replaceData** efetua a modificação do valor do elemento (nó atual) e o método **save** grava o documento com as novas alterações.

O método **replaceData** possui 3 parâmetros, sendo o primeiro para a posição inicial onde será efetuada a modificação, o segundo indica a quantidade de valores substituídos e o terceiro informa o novo valor do elemento.

```
function ChangeDocument()
{
      var listNodes, oNode, tam

      listNodes =
      xmldoc.getElementsByTagName(document.form1.T1.value.toUpper
      Case())

      tam = listNodes.item(0).firstChild.nodeValue.length
      oNode = listNodes.item(0).firstChild
      oNode.replaceData(0,tam,document.form1.T3.value)

      xmldoc.save("mib.xml")

      messageDIV.innerHTML = "<STRONG>Valor [" +
            oNode.data +
            "] alterado com sucesso!</STRONG><BR>"
}
```

Figura 29: Função que modifica o valor de um elemento XML

6.6 CONCLUSÕES

Com a elaboração desta aplicação, verificou-se que a utilização de um documento XML é viável para as atividades de gerenciamento de sistemas.

Foi possível demonstrar a utilização de uma MIB *Browser* e das funções para buscar e alterar uma variável no documento XML, como se fossem operações GET e SET.

A seguir na seção 7 serão apresentadas as conclusões finais do desenvolvimento deste projeto, bem como as sugestões de trabalhos futuros identificadas no decorrer do mesmo.

7 CONCLUSÕES

Inicialmente, este projeto tinha o objetivo de elaborar e apresentar o protótipo de uma ferramenta de gerenciamento de sistemas usando o sistema operacional Linux. Pretendia-se implementar algumas funcionalidades de gerenciamento, onde a funcionalidade de Resolução de Problemas tinha um destaque especial, pela proposta de utilização da técnica de Inteligência Artificial - Raciocínio Baseado em Casos para a otimização das tarefas de resolução de problemas.

No decorrer do projeto, verificou-se a complexidade de cada funcionalidade apresentada. E também a necessidade de se elaborar uma MIB que armazenasse as informações manipuladas pelo sistema.

Deste modo, ficou entendido a importância da MIB no contexto do sistema e deu-se início então a uma nova etapa do projeto, que consistia na elaboração de uma MIB e na forma como esta MIB seria disponibilizada.

A procura por um sistema aberto e multiplataforma levou ao conhecimento da XML. Pois esta apresentou boas condições para o gerenciamento de sistemas heterogêneos.

Comprovadamente, a XML apresentou um grande potencial para o gerenciamento de sistemas, sendo destacados a transparência entre a consulta e a implementação das informações disponibilizadas, onde um usuário ou aplicação não precisa conhecer como a informação é armazenada, sendo suficiente, que se conheça a descrição da informação que está disponibilizada.

Outro ponto importante, foi a compatibilidade com os sistemas de gerenciamento já existentes, permitindo que através de ferramentas automatizadas, uma MIB já definida e Padronizada, fosse rapidamente convertida para documentos XML, com o mínimo esforço.

Ainda com o intuito de otimização, foi proposto com sucesso um mapeamento de MIB para XML, onde foi elaborado um conjunto de regras, identificadas no projeto como SCRIPT, que teria a função de gerar os arquivos XML.

Uma vez convertida a MIB, foi elaborada uma aplicação Web, para demonstrar a utilização e manipulação de arquivos XML. Para isto foram elaboradas páginas HTML, contendo código JavaScript. Foi possível verificar as vantagens na utilização do HTTP para transferência de informações, principalmente com relação ao compartilhamento de informações.

Com os teste realizados, a arquitetura de gerenciamento baseada em XML mostrou-se apta tanto como ferramenta final como ferramenta de apoio as atividades de gerenciamento.

Verificou-se que a utilização de arquivos DTD para a definição da estrutura das informações é bastante limitado, onde cita-se a limitação das informações só serem apresentados em formato *string*, não permitindo a utilização de tipos de dados como *integer*, *float*, *boolean*, etc. O DTD também não apresenta aspectos de segurança, como por exemplo, um atributo não pode ser definido como *read-only*.

Fica então como proposta futura, a verificação e testes do XML *Schema*.

No decorrer do projeto, várias novas linhas de pesquisa foram identificadas, tendo como ponto de partida a utilização da MIB e da XML. A seguir serão apresentadas estas perspectivas.

7.1 TRABALHOS FUTUROS

Apesar deste trabalho ter alcançado os objetivos propostos, verificou-se que várias melhorias e acréscimos podem ser elaborados a partir dos resultados obtidos.

Foram elaboradas as operações equivalentes ao GET e SET, ficando como próximo passo a elaboração da operação TRAP, pela importância de se receber aviso dos eventos importantes que tenham ocorrido.

As páginas apresentadas na aplicação são experimentais, devendo-se numa próxima etapa dar ênfase aos aspectos visuais de como as informações são apresentas. Sugere-se o estudo de XSL e CSS.

Ainda são destacados outros ramos de pesquisa a partir do projeto proposto:

- **Gerenciamento de outros sistemas**: além do gerenciamento direcionado à redes e sistemas operacionais abordados, pode-se expandir a utilização deste

projeto para o gerenciamento de banco de dados, equipamentos específicos, etc. Onde o usuário faria o mapeamento das informações que deseja gerenciar, usando o *script* de geração de arquivos XML proposto neste trabalho;

- **Conversor MIB para DTD**: a partir do arquivo MIB descrito em ASN.1, pode-se elaborar um conversor que gere automaticamente o arquivo DTD com as informações disponíveis na MIB. Refinando um pouco mais, pode-se chegar à um gerador de arquivos XML;

- **Agente SNMP - XML**: Esta etapa envolveria a criação de um agente MIB SNMP que analise a porta SNMP 161/tcp, verificando a chegada de requisições. Deve-se dotar o agente, com um mecanismo que verifique a existência da MIB SNMP correspondente à solicitação ou à existência de um agente XML que consiga responder as solicitações SNMP. Este agente MIB seria então um *proxy*, repassando as primitivas SNMP;

- **Gerador de SCRIPT:** pode-se elaborar uma ferramenta que, a partir do arquivo DTD, seja gerado o arquivo SCRIPT. Automaticamente a numeração dos identificadores seria criada, segundo a posição dos elementos #PCDATA no arquivo DTD;

- **Definição dos objetos a partir do padrão GDMO**: Seria possível usar o padrão GDMO (*Guidelines for Definition of Managed Objects*), um guia para definição de objetos gerenciáveis, especificado na ISO/IEC 10.165-4, para estas definições de objetos, de modo consistente. Usando o GDMO, é possível descrever a classe ou as classes dos objetos, como o objeto se comporta, seus atributos e as classes que podem herdar (JITAO, 2001).

8 REFERÊNCIAS BIBLIOGRAFICAS

AAMODT, A. e PLAZA, E., Case-Based Reasoning: Foundational Issues, Methodological Variations, and Systems Approaches. Artificial Intelligence Communications, Vol. 7, No. 1, 1994.

ANÁLISE, Informática Brasileira em. Gerenciamento Corporativo de Rede. Uma Publicação do CESAR (Centro de Estudos e Sistemas Avançados do Recife). Ano II - Número 29 - Novembro de 1999.

BARBIERI, Carlos. XML: A linguagem do Futuro. Computerworld. Junho, 2000. URL: http://www.uol.com.br/computerworld/communication/0006/0006xml.htm

BARROS, Fábio. INCOR utiliza XML em Prontuários. Computerworld. Agosto, 2000. URL: http://www.sit.com.br/separateneto53.htm

BOSAK, John. XML, Java and the future of the Web. Sun Microsystems. March, 1997 URL: http://www.iblibio.org/pub/sun_info/stabdards/xml/wity/xmlapps.html

BRAY, Tim. SPERBERG-MCQUEEN, C.M, PAOLI, jean. Extensible Markup Language (XML) 1.0. W3C: Recomendation. February, 1998. URL: www.w3.org/tr/rec-xml-19980210.html

BRAY, Tim. SPERBERG-MCQUEEN, C.M. Extensible Markup Language (XML). W3C *Working Draft*, November, 1996. URL: http://www.w3.org/tr/wd-xml-961114.html

BRISA, Gerenciamento de Redes: Uma Abordagem de Sistemas Abertos. Makron Books, São Paulo, 1992.

CARLSON, David. Modeling XML applications with UML: Pratical e-Business Applications. Addison-Wesley. 2001.

CARVALHO, Sandra. Plantão INFO – Vendas de Linux disparam. Revista InfoExame. Abril/2000.

CONECTIVA, Treinamento Avançado Administração do Linux I. Apostila da Conectiva S.A. 2000.

ERIKSSON, Hans-Erik. PENKER, Magnus. UML ToolKit. Wiley. 1998.

FOWLER, M. SCOTT, K. UML Essencial: Um breve guia para a linguagem-padrão de modelagem de objetos. Bookman, 2000.

GUIDALLI, Sandro. Linux: Brasil fecha 2000 como o quinto mercado que mais cresce. Canal Web Digital. Novembro/2000. URL: http://www.canalweb.com.br/noticias.

GULINUXFLORIPA, Câmara de Florianópolis vota lei. Grupo de Usuários Linux. Abril/2001. URL: http://www.gulinuxfloripa.cjb.net.

HEGERING, Heinz-Gerd, ABECK, Sebastian. Integrated Network and System Management. Addison-Wesley Publishing Company. Munich, 1994.

HELD, Gilbert. Lan Management with SNMP and RMON. Wiley Computer Publishing. New York. 1996.

HOLZNER, Steven. Inside XML. New Riders, 2001.

ISO/IEC 7498-4. Information processing systems; Open Systems Interconnection; basic reference model; Part 4: Management framework. 1989. URL: http://www.iso.ch/isso/en

ISO/IEC 8879. Information processing, Text and Office Systems; Standard Generalized Markup Language (SGML). 1986. URL: http://www.iso.ch/isso/en

ISO/IEC 10165-4. Information Technology; Open System Interconnection; Structured of Management Information; Part 4: Guidelines for the definition of managed objects. 1992. URL: http://www.iso.ch/isso/en

JITAO, Hou. GDMO - Guidelilnes for Definition of Managed Objects. July, 2001. URL: http://www.whatis.techtarget.com/definition

JOHNSON, D. NOC Internal Integrated Trouble Ticket System: Funcional Specification Wishlist. RFC 1297. 1992.

KOLODNER, J. e LEAKE, D. A Tutorial Introduction to Case-Based Reasoning. Case-Based Reasoning: Experiences, Lessons, & Future Directions. Ed. David B. Leake, AAAI Press, 1996.

LEAKE, D. B. Case-Based Reasoning: Experiences, Lessons, & Future Directions. AAAI Press, 1996.

LEIVA, Willie. VICENTE, William. Aprendendo XML por meio de um Estudo de Caso. UNICAMP. Campinas, 1999. URL: http://www.dca.fee.unicamp.br/projects/sapiens/seminars/atas/register/sem9911/index.htm

MELCHIORS, Cristina. Raciocínio Baseado em Casos Aplicado ao Gerenciamento de Falhas em Redes de Computadores. Porto Alegre, Dissertação de Mestrado PPGC-UFRGS, 1999.

MICROSOFT. Administering Microsoft Systems Management Server 2.0. Microsoft Official Curriculum - Course Number: 827A. 1999.

MILLER, Mark. Managing Internetworks with SNMP. IETF Network Management Documents. M&T Books, 1999.

MOLTA, Dave. Premio Well-Connected Award. Revista Network Computing Brasil Ano:2. Número:17. Julho/2001. URL:http://www.networkcomputing.com.br. 2001.

PEITER, Rui C. Um modelo de um agente SNMP para gerenciamento de rede. TCC-Univali-Ciência da Computação, 2000.

PERKINS, David; McGinnis, Evan. Undestanding SNMP MIBs. Prentice-Hall PTR, 1997.

RAMOS. Tagil. IBM quer falar a linguagem XML. InfoOnLine. Março, 1999. URL: http://ww2.uol.com.br/info/infonews.031999/09031999-10.shl

REVISTA DO LINUX, Portal promove debate sobre uso software livre. Notícias On-Line. Março/2001. URL: http://www.revistadolinux.com.br/noticias/2001/03/27/1602_det.html.

SILVA, Luiz Fernando. XML-Uma nova linguagem para uma nova geração Web. EPUSP. Junho, 2000. URL: http://www.lem.ep.usp.br/pef411/Luiz%20Fernando%20da%20Silva/xml.htm

SPOHN, Marco Aurélio. Network Operational Control. Porto Alegre, 1995. URL: http://penta2.ufrgs.br/gr952/trab1/noc.html.

SOARES, Andrey. Proposta de Análise de Gerenciamento de Redes usando Ambiente CORBA. São José, Monografia de Graduação em Ciência da Computação-UNIVALI, 1998.

SPECIALSKI, Elizabeth S. Gerência de Redes. INF-UFSC-Grupo de Redes de Computadores, 1996.

STALLINGS, W., SNMP, SNMPv2, and CMIP. The practical Guide to Network-Management Standards. Addison Wesley, 1993.

SUN. A XML abre o caminho para o comércio eletrônico. SunMicrosystems Brasil: O que é .COM? Três grandes apostas. 2001. URL: http://www.sun.com.br/ponto-com/xml.html

TERUSKIN, Rafael. XML - padronização e Aplicação. 2000. URL: http://www.marlin.com.br/~rafaelt/xml.htm

THIRY, Marcello. Uma arquitetura baseada em agentes para suporte ao ensino à distância. Florianópolis, Tese de Doutorado PGEP-UFSC, 1999.

TIBET, Chuck, LINUX - Administração e Suporte, Ed. Novatec, 2001

WALDBUSSER, S. Remote Network Monitoring Management Information Base. RFC 1757. Carnegie Mellon University, February, 1995.

WATSON, I. Applying Case-Based Reasoning: Techniques for Enterprise Systems. Morgan Kaufmann, 1997.

WEBBER, Celso K. Uma Mib para aplicações Internet. Dissertação de Mestrado. CPGCC-UFSC, 1997