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

Augusto José Venâncio Neto

IMPLEMENTAÇÃO DE UM DISCRIMINADOR DE REPASSE DE EVENTOS PARA O AMBIENTE SNMP

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

Elizabeth Sueli Specialski
Orientadora

Florianópolis, Agosto de 2001

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

Augusto José Venâncio Neto

IMPLEMENTAÇÃO DE UM DISCRIMINADOR DE REPASSE DE EVENTOS PARA O AMBIENTE SNMP

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

Elizabeth Sueli Specialski

Orientadora

Florianópolis, Agosto de 2001

IMPLEMENTAÇÃO DE UM DISCRIMINADOR DE REPASSE DE EVENTOS PARA O AMBIENTE SNMP

Augusto José Venâncio Neto

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. O. Gauthier, Dr.
(Coordenador do Curso)

Banca Examinadora

_____

Elizabeth S. Specialsky, Dra.
(Orientadora)

_____

Alexandre Moraes Ramos, Dr.
(Membro da Banca)

_____

Tereza Cristina Brito de Carvalho, Dra.
(Membro da Banca)

Este trabalho é oferecido a toda minha família, especialmente ao meu pai Augusto José Venâncio Filho, a minha mãe Alice Nemer Venâncio, ao meu irmão Carlos Michel Nemer Venâncio e a minha irmã Érica Bianca Nemer Venâncio.

AGRADECIMENTOS

Primeiramente a Deus, a quem tenho profunda fé. Por todas as minhas orações atendidas e minha saúde plena. Por todas as vezes que me auxiliou na solução dos meus problemas. Por ter colocado no meu caminho pessoas maravilhosas e amigas.

À Nossa Senhora de Nazaré (padroeira dos paraenses) por todas as realizações na minha vida e todas e intercessões perante a Deus pai todo poderoso. Por ter colocado sobre mim todo o poder do ser manto.

Ao meu pai e à minha mãe pelos esforços financeiros em me manter durante o período em que não era ainda bolsista.

À minha eterna orientadora e mãe, Profª Drª Elizabeth Sueli Specialski, pelo excelente trabalho realizado, pelo extremo profissionalismo e capacidade de trabalho, por todas as vezes que puxou minha orelha, por todos os maravilhosos conselhos que muito ajudaram na minha vida, muito obrigado mesmo doutora.

Ao meu eterno padrinho Prof. Dr. Bosco da Mota Alves e à minha eterna madrinha Profª Drª Silvia Modesto Nassar, principais responsáveis pela minha vinda à Florianópolis e à Ufsc.

Um beijo mais que especial às minhas eternas amigas e irmãs Meri, Priscila e Drª Marlene, vocês estarão sempre no meu coração.

À todos os colegas do curso de Mestrado em Ciência da Computação da Ufsc, docentes e discentes, que muito me auxiliaram na aquisição de informações e me deram forças para completar minhas pesquisas. Em especial aos amigos Cleverson, Paulo, Gonçalo e Sandro.

Agradeço também a todos os paraenses residentes em Florianópolis, pelo companheirismo e amizade.

Aos amigos mais chegados, João Santana, Dudu, Peri, Hilton, Iuri, Hugo, Michele, Luizinho, Sobral, Brandão, Baiano, Serginho, Rogério, Cícero, Gaudêncio, João, Paulo e muitos outros mais que não consigo citar agora.

À todos os companheiros de trabalho da Casan, especialmente Adenilson Rodrigues, Paulo Peressoni, Wesly Fuji, Scherry e Joaquim Pacheco.

A todos os companheiros docentes da Unesc e à minha primeira turma de alunos.

RESUMO

Este trabalho apresenta um protótipo, a ser utilizado em ambientes SNMP, o qual implementa as funcionalidades de Objetos Discriminadores de Repasse de Eventos, definidos para ambientes OSI na norma X-734.

Esta aplicação, além de possibilitar a utilização de mais de uma estação de gerenciamento divididas tanto em módulos de gerenciamento quanto em áreas de atuação, também torna possível enfatizar o modelo na recepção de Traps. Desta forma, o modelo assume uma importante característica, a descentralização.

ABSTRACT

This research shows a prototype, to be used in SNMP environments, that implements Event Report Discriminator functionalities defined for OSI environments through X-734 norm.

This application, in addition to enable the use of more than one management station divided in management modules as well as action areas, enables the model to enphasize the reception of Traps. Thus, the model assume an important feature, the decentralization.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ASN.1 – Abstract Syntax Notation. 1

CCITT – International Consultative Committee on Telegraphy and Telephony

CLNS - OSI Connectionless Network Service

CMIP – Common Management Information Protocol

CMOT – CMIP OVER TCP/IP

DDP - Apple Talk Datagram - Delivery Protocol

DOD – Departamento de Defesa dos Estados Unidos

EGP - External Gateway Protocol

IBM – International Business Machine Corporation

ICMP – Internet Control Message Protocol

IEEE – Institute of Electrical and Electronics Engineers

IETF – Internet Engineering Task Force

IP – Internet Protocol

IPX - Novell Internet Packet Exchange

ISOC – Internet Society

ISO – International Organization for Standardization

MIB – Management Information Base

NMS – Network Management Server

OID – Object Identifier

OSI – Open Systems Interconnection

PDU – Protocol Data Unit

RFC – Request for Comments

SMI – Structure of Management Information

SMP – Simple Management Protocol

SNMP – Simple Network Management Protocol

TCP – Transmission Control Protocol

UDP – User Datagram Protocol

CAPÍTULO I

1. Introdução

Atualmente as redes de computadores estão cada vez mais presentes em soluções de tecnologia da informação. Já é comum encontrar microcomputadores interligados por hubs, cabo coaxial ou até mesmo switches, em pequenas empresas, escritórios, condomínios e em alguns casos residências, haja vista os custos estarem cada vez mais compatíveis com as possibilidades individuais de estudantes, empresários e outros. É um cenário diferente de outrora, quando verdadeiros monopólios eram criados por empresas que investiam muito dinheiro em pesquisas, criando soluções proprietárias entre softwares e dispositivos de conectividade.

Entretanto, as redes de computadores não são perfeitas, estando à mercê de diversos fatores que podem ameaçar sua integridade, sejam eles físicos (cabeamento ou interferências, por exemplo), sejam eles lógicos, com os serviços oferecidos pela rede (exemplificando: violação de contas, sobrecarga de servidores ou excesso de tráfego).

Para garantir a seus usuários a disponibilidade dos serviços a um nível de desempenho aceitável, por menor e mais simples que seja a rede de computadores, torna-se necessário seu gerenciamento. À medida que o ambiente cresce, aumenta também a possibilidade da ocorrência de problemas (falhas de dispositivos por exemplo), e se estes problemas não forem resolvidos em tempo hábil, trarão prejuízos aos usuários diretamente envolvidos com os serviços disponibilizados, e conseqüentemente a toda organização. Aumenta também a complexidade do gerenciamento da rede, forçando com isso a adoção de ferramentas automatizadas para a sua monitoração e controle.

Foram então criados os sistemas de gerenciamento com o objetivo de habilitar, os administradores de redes, a monitorar e controlar diversos equipamentos do ambiente,

como servidores e roteadores, por exemplo. Dentre muitas outras funcionalidades, tornou-se possível gerenciar o desempenho de seus ambientes, encontrar possíveis problemas e solucioná-los.

Inicialmente proprietários, estes sistemas de gerenciamento monitoravam e controlavam apenas dispositivos fabricados pelo mesmo fornecedor do sistema, ou seja, um sistema de gerenciamento desenvolvido pela IBM tinha como objetivo gerenciar apenas equipamentos desenvolvidos pela própria IBM, por exemplo. Com isso, o administrador tinha a sua disponibilidade pelo menos um número de sistemas na mesma quantidade do número de fornecedores com equipamentos envolvidos neste ambiente.
Com tantos sistemas distintos, diversos esforços começaram a surgir para que houvesse interoperabilidade entre estes sistemas. Com isso iniciou-se os trabalhos de pesquisas para definição de arquiteturas padronizadas de gerenciamento de redes.

A exemplo de padronizações internacionais de arquiteturas de redes não proprietárias, as arquiteturas de gerenciamento mais amplamente utilizadas são a Arquitetura de Gerenciamento Internet e a Arquitetura de Gerenciamento OSI da ISO.

Dentre as duas arquiteturas de gerência de redes mais difundidas atualmente, a que mais se destaca, no que diz respeito à quantidade de usuários, é a Arquitetura de Gerenciamento Internet, dada a larga utilização de redes de computadores com esta arquitetura. O protocolo de gerenciamento utilizado para monitorar e controlar redes com arquitetura Internet é o SNMP, detalhadamente descrito no capítulo 3.

O SNMP é o protocolo recomendado para redes que utilizam os protocolos TCP/IP. Definido ao nível de aplicação, o SNMP utiliza os serviços do protocolo UDP para realizar trocas de mensagens entre os componentes do modelo de gerenciamento: agente e gerente [CASE 90].

Para se gerenciar uma rede de computadores através do SNMP, são necessários diversos nodos, cada um com uma entidade processo denominada agente, a qual tem acesso a informações de gerenciamento e pelo menos uma estação de gerenciamento.

As estações de gerenciamento executam aplicações de gerenciamento, os gerentes, responsáveis pelo monitoramento e controle dos elementos gerenciados. Estes elementos são dispositivos (como: hosts, roteadores, hubs ou terminais servidores) que são monitorados e controlados através de um conjunto de softwares residentes nestes equipamentos, os agentes, destinados às tarefas de coletar informações sobre as atividades relacionadas com a rede, armazenar estatísticas localmente e responder aos comandos do centro de controle da rede.

Estas informações são como uma coleção de objetos gerenciados, residentes em um repositório de informações de gerenciamento, denominado de Base de Informações de Gerenciamento (*MIB – Management Information Base*) [CASE 96e].

O envio das informações de gerenciamento pode ser feito de duas maneiras:

- Através da solicitação do gerente, o qual realiza um comando específico para obtenção de uma informação de gerenciamento;
- Esporadicamente, o agente envia uma notificação da ocorrência de um determinado evento na rede, sem a solicitação do gerente.

O SNMP é utilizado basicamente para visualização de estados, ou seja, monitoração de equipamentos gerenciáveis. Neste contexto, é aplicada maior ênfase à técnica de polling, dando pouca importância às notificações de eventos. O modelo SNMP é considerado muito simples e limitado, comparado ao modelo de gerência OSI.

O modelo SNMP é centralizado, sendo basicamente adotada apenas uma estação de gerenciamento. Já o modelo OSI é distribuído, onde é comum a adoção de mais de uma estação de gerenciamento divididas em áreas de atuação de gerência, isto é, Domínios de Gerenciamento. Neste caso, torna-se evidente a necessidade de comunicação entre os diversos gerentes. Além disso, uma vez que os agentes deste modelo apresentam maior funcionalidade, a ênfase do modelo é dada aos relatórios de eventos.

1.1 SNMP Descentralizado

Para ajudar no trabalho dos administradores, minimizando a problemática do aparecimento de novos problemas advindos de novas tecnologias, os fornecedores investem cada vez mais em novas funcionalidades, implementando-as tanto nos gerentes quanto nos agentes, sejam em forma de relatórios de eventos ou informações de gerenciamento (novas MIBs), ambos proprietários.

Hoje em dia, as aplicações de gerenciamento ainda requerem bastante a intervenção do administrador para controlar uma rede de computadores. Isto implica em um nível muito baixo de tratamento de informações.

Quanto mais automatizada, melhor a aplicação de gerenciamento. Para aumentar o nível de um sistema de gerenciamento torna-se necessário a implementação de novas funcionalidades como, por exemplo, a adoção de técnicas de inteligência artificial como auxílio na tomada de decisões. Com isso, cada vez mais haverá a necessidade de maior processamento, memória e capacidade de armazenamento.

Para evitar esta sobrecarga de processamento, uma saída seria distribuir as aplicações de gerenciamento, dividindo estas em módulos de gerenciamento localizados em máquinas distintas, sendo diferenciados por atividades de gerência ou por domínios de gerenciamento.

1.2 A problemática da descentralização no SNMP

O modelo tradicional da gerência SNMP é centralizado, implicando uma carga alta na estação de gerenciamento. Esta carga é agravada pelo fato do modelo ser orientado a polling, isto é, a atualização das informações de gerenciamento é, quase que exclusivamente, responsabilidade do gerente. Apesar da adoção do RMON, que permite uma descentralização nas tarefas de cálculo de médias e outros dados estatísticos, a carga maior de processamento de aplicações de gerenciamento continua sendo sobre a estação gerente.

À medida que novas aplicações automatizadas sejam instaladas, esta carga tende a se tornar inviável para ser suportada por uma única estação. Isto é facilmente observável pelo crescente aparecimento de aplicações de gerenciamento que visam auxiliar o processo de tomada de decisões.

Uma maneira bastante eficaz para diminuir a carga da estação de gerenciamento no modelo SNMP seria aumentar a ênfase na recepção de relatórios de eventos, principalmente os Traps proprietários que são muito mais específicos e completos do que os Traps padronizados.

Com uma ênfase maior em Traps, o gerente pode ser liberado da consulta de diversas variáveis, diminuindo a carga de polling. Mesmo assim, o problema não seria solucionado completamente, pois uma única estação gerente ainda seria responsável pelo processamento de todas as informações obtidas da rede. Além disso, o aumento da ênfase em Traps pode causar uma sobrecarga na rede, comprometendo o seu tráfego.

Ainda, se for considerado o modelo centralizado, o acúmulo de relatórios de eventos em uma determinada estação de gerenciamento, pode prejudicar o seu desempenho, comprometendo a solução de problemas.

A solução apontada neste trabalho combina a ênfase maior em traps com a descentralização das atividades de gerenciamento, isto é, a adoção de mais de uma estação gerente, como forma de diminuir a carga de processamento por máquina.

Entretanto, a problemática desta solução se dá no momento da definição do(s) destinatário(s) do(s) Trap(s), já que o modelo considera que um agente, na identificação da ocorrência de um evento pré-definido, encaminha um trap para o gerente ao qual ele está associado.

O modelo de gerenciamento da arquitetura SNMP não disponibiliza nenhum mecanismo para tratamento de relatórios de eventos, no aspecto de destinatários de Traps. A maneira pelo qual os agentes identificam estes destinatários é através de endereços IP

incluídos através de linhas de comandos, sendo assim, o SNMP não possui inteligência computacional para identificar quais estações deveriam ser notificadas e quais não deveriam. A necessidade de que um agente possa encaminhar traps específicos para diferentes destinatários implica em uma modificação do código dos agentes existentes na rede, o que se constitui de uma solução inviável.

Considerando o problema exposto, buscou-se a solução junto ao modelo de gerenciamento OSI, uma vez que este modelo, da mesma forma que o RM-OSI, constitui-se em um modelo de referência que vem sendo utilizado em soluções parciais de questões de gerenciamento.

Para otimizar a tarefa do gerenciamento de ambientes distribuídos, o modelo de referência ISO/OSI subdividiu a gerência de redes em cinco grandes áreas funcionais: Gerência de Falhas, de Configuração, de Desempenho, de Segurança e de Contabilização [BRISA 93]. Além disso, as atividades de gerência podem estar distribuídas por diversos equipamentos na rede. Uma estação gerente pode atender a somente um ou a vários aspectos de gerenciamento. Caso as atividades de gerenciamento sejam distribuídas por vários gerentes, é prevista a comunicação entre eles e, ainda, a funcionalidade nos agentes para identificar os destinatários para cada tipo de relatório de evento. Um possível mapeamento deste modelo para o ambiente SNMP, é ilustrado na figura 1.1.

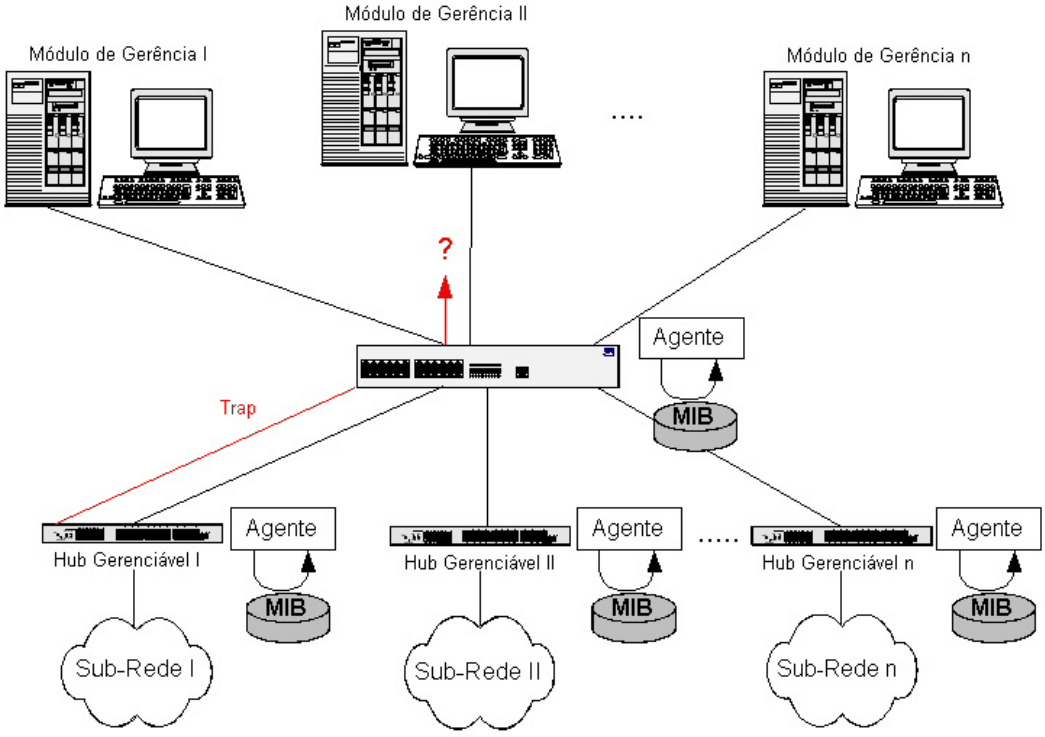

FIGURA 1.1 – Envio de um Trap

A fig. 1.1 mostra um possível cenário de uma rede contendo N gerentes possuindo diferentes atividades de gerenciamento. Supondo-se a ocorrência de um Trap em um dos objetos monitorados pelo agente I no hub gerenciável I, é necessário que este alarme seja encaminhado ao(s) gerente(s) adequado(s). Para resolver este problema, o agente necessita possuir mecanismos de auxílio na escolha do(s) destinatário(s) para o(s) qual(is) o alarme deva ser enviado. A implantação de tais mecanismos em todos os agentes de uma rede é uma tarefa onerosa quando não impossível, pois os agentes implementados nos dispositivos são geralmente feitos pelos próprios fornecedores, tornando seus códigos inacessíveis para os usuários.

Através de políticas de gerenciamento são identificadas quais estações devem, ou não, ser notificadas por determinados tipos de Traps. Esta identificação se dá de acordo com o conceito de cada Trap e a área de atuação de cada módulo de gerenciamento. Por exemplo, um módulo para de desempenho além de ter a necessidade de ser notificado

por um Trap relativo à sua área, também poderia ser notificado por Traps relativos à falhas e/ou segurança.

Na Arquitetura de Gerenciamento OSI esta problemática pode ser solucionada através da utilização de objetos Discriminadores que serão descritos no capítulo 4.

1.3 Trabalhos Correlatos

Uma pesquisa foi realizada para descobrir a existência de sistemas de tratamento de relatórios de eventos no modelo de gerenciamento SNMP, sendo encontradas algumas implementações que são descritas resumidamente nos itens de 1.3.1 a 1.3.3.

1.3.1 Trap Console

Desenvolvido pela CS Software, trata-se de uma aplicação servidora para gerenciamento de Traps. Atualmente na versão 1.3, este *Applet* Java possui independência de plataforma sendo capaz de encaminhar os relatórios de eventos para determinados endereços IP e também notificar um ou mais usuários através de e-mails e mensagens em aparelhos celular. Possui ainda a capacidade de compilação de *MIBs* proprietárias, criação de ações e regras para encaminhamento de *Traps* [TRAPCONSOLE 99].

1.3.2 SunNet Manager

SunNet/Domain Manager trata-se de um sistema de gerenciamento desenvolvido pela *SunSoft*. Atualmente na versão 2.3, este ambiente de gerenciamento disponibiliza uma poderosa ferramenta para tratamento de relatório de eventos. Denominado Snmp *Trap Deamon*, este traduz os *Traps* recebidos para o formato *SunNet Manager Trap*, encaminhando-os posteriormente para um despachante de eventos em uma ou mais estações de gerenciamento [SOLSTICE 96].

Quando o *SunNet Manager Snmp Trap Deamon* recebe um *Trap*, é executada a seguinte seqüência de operações [SOLSTICE 96]:

- Conversão do *Trap* para *strings* ASCII;

- Determinação se o *Trap* deve ou não ser descartado;

- Caso não deva ser descartado, determinação do tipo de prioridade de encaminhamento do *Trap* (*high, medium* e *low*);

- Determinação do elemento da base de dados (dispositivo de rede denominado *Glyph*) para o qual o *Trap* deve ser atribuído (no caso de elementos que representem pseudo-dispositivos – componentes que não possuem seus próprios endereços IP, contudo compartilham o endereço de rede do dispositivo o qual fazem parte ou são anexados).

1.3.3 IBM Tivoli Net view

É o sistema desenvolvido pela IBM para gerenciamento de hosts e sistemas distribuídos. Oferece um novo paradigma para gerenciamento de aplicações, chamado de gerência de sistemas de negócio (*Business Systems Management*). Este paradigma de gerência possibilita, aos seus clientes, gerenciar aplicações de negócios críticos como, por exemplo, ERP, CRM ou ambientes *e-business*.

O filtro de alerta do *Tivoli NetView* é capaz de controlar quais alertas serão encaminhados a determinados hosts. Existe ainda a possibilidade de implementação de um sistema de correlação de eventos o que torna esta ferramenta bastante poderosa e eficaz. É importante ressaltar que a alteração dos códigos deste filtro acarreta em significantes acréscimos de processamento extra da CPU, assim como gargalos na rede.

1.4 Objetivo e Organização do Trabalho

Este trabalho tem por objetivo desenvolver um protótipo capaz de implementar algumas funções inclusas nos objetos Discriminador de Repasse de Eventos definidos no modelo de gerência OSI, aplicável em ambientes SNMP. Também propõe um modelo funcional para implementação destes.

Este trabalho está organizado em 7 capítulos. No capitulo 2 apresenta-se a evolução dos sistemas de gerenciamento de redes e os conceitos básicos do modelo de gerenciamento Internet. No capitulo 3 é apresentado um estudo mais aprofundado do protocolo SNMP - *Simple Network Management Protocol,* considerando-se as versões 1 e 2 e apresentando-se a estrutura da informação de gerenciamento do modelo SNMP. O capitulo 4 aprofunda o conceito de discriminadores de repasse de eventos e tece considerações sobre a aplicação deste conceito em sistemas de gerenciamento baseados no protocolo SNMP. A solução proposta para o problema descrito na introdução é apresentada no capitulo 5. No capitulo 6 apresenta-se considerações sobre a solução proposta, as conclusões obtidas e algumas indicações sobre a continuidade deste trabalho. O capitulo 7 apresenta a bibliografia consultada e as referências bibliográficas. O anexo contém a especificação formal das unidades de dados do protocolo SNMP em ASN.1.

CAPÍTULO II

2. Gerenciamento Internet

2.1 Introdução

Este capítulo apresenta um histórico da gerência de redes definindo datas e momentos importantes. Em seguida, o capítulo mostra as padronizações, enfatizando o protocolo padronizado para gerência de redes internet.

2.2 Histórico do Gerenciamento Internet

A Internet nasceu através do projeto ARPANET, iniciado em 1969, objetivando estudar e construir, no território norte americano, uma rede de computadores que atendesse aos interesses do Departamento de Defesa (DoD) dos Estados Unidos. Como resultado, bastante conhecido por sinal, a ARPANET acabou transformando-se em Internet, a qual hoje em dia está difundida por todo o mundo, possuindo milhões de computadores interconectados.

Quando a arquitetura Internet foi definida, adotando-se de forma padronizada os protocolos TCP/IP para comunicação, pouco se pensou em seu gerenciamento. Os usuários da ARPANET estavam em geral envolvidos em algum aspecto do projeto ARPANET, e deixaram que os *experts* em protocolos de redes cuidassem do assunto [STALLINGS 93].

O fato é que até o final da década de 70 não existiam ferramentas de gerenciamento para a arquitetura Internet. A única ferramenta efetivamente utilizada na época era o protocolo ICMP [STALLINGS 93]. Este protocolo é capaz de transmitir mensagens entre roteadores ou hosts para um determinado *host* da rede, e detectar problemas no

caminho. O ICMP está disponível em todos os dispositivos que suportam o protocolo IP.

Apesar da grande utilidade oferecida pelo ICMP, desde seu uso simples através de comandos como ´ping´ até a construção de ferramentas que o utilizem, o ICMP passou a não ser mais suficiente para todas as situações de gerenciamento de rede Internet. A Internet deixou de ser uma rede com dezenas de computadores para ser uma rede mundial com um número exorbitante de computadores a ela ligados.

Surgiram então três propostas para o gerenciamento Internet [STALLINGS 93]:

1. *High-level Entity Management System* (HEMS): Esta solução era uma generalização do que tenha sido talvez o primeiro protocolo de gerenciamento da Internet, o protocolo de monitoração de hosts (*Host Monitoring Protocol* – HMP).

2. *Simple Network Management Protocol* (SNMP): Uma versão avançada do SGMP – *Simple Gateway Monitoring Protocol* (Protocolo de Monitoramento de Gateways Simples).

3. *CMIP over TCP/IP* (CMOT): Esta foi uma tentativa de incorporar, na máxima extensão possível, o protocolo CMIP, serviços e estrutura da base de dados dos padrões de gerenciamento que estavam sendo padronizados pela ISO.

No início de 1988, o IAO, que regulamenta os padrões na Internet, adotou o SNMP como uma solução imediata, devido à sua simplicidade, e o CMOT como uma solução que, em longo prazo, substituiria a solução adotada.

Para facilitar uma futura transição do SNMP para o CMOT, o IAB definiu que ambos os protocolos deveriam utilizar uma mesma estrutura da informação de gerenciamento (SMI), bem como a mesma base de informação de gerenciamento (MIB).

Logo se percebeu que era impraticável que os dois protocolos utilizassem as mesmas SMI e MIB. Enquanto o gerenciamento OSI trata os objetos gerenciados como entidades com alto grau de inteligência, que possuem atributos, procedimentos associados, e capacidade de notificações, além de outras características sofisticadas da tecnologia de orientação a objetos, o gerenciamento Internet assume que os objetos gerenciados são entidades simples, que possuem um conjunto de variáveis com algumas poucas características, capazes de serem apenas lidas ou escritas. Assim, o IAB permitiu que o desenvolvimento do CMOT utilizasse SMI e MIB distintas do SNMP.

2.3 Padronização da Arquitetura de Gerenciamento SNMP

O IETF é a organização que desenvolve toda a padronização dentro da Internet. A supervisão técnica e de processos do IETF fica a cargo do IAB. Este por sua vez, está hierarquicamente abaixo do ISOC, que é o órgão administrativo de mais alto nível na Internet. O direcionamento técnico e a administração de processos do IETF ficam a cargo do IESG, que é composto pelo coordenador geral do IETF e pelos diretores de suas áreas funcionais.

Para compreender a evolução das padronizações referentes ao modelo de gerenciamento SNMP, é preciso entender o funcionamento do sistema de padronização do IETF.

Em cada área funcional das atividades do IETF, grupos de trabalho (WGs) são criados para completar tarefas específicas, que após seu término, são dissolvidos. Qualquer pessoa está apta a participar como membro de um grupo de trabalho, podendo com isso acompanhar as discussões nas listas de discussão dos grupos de trabalho. Os membros devem cooperar para que sejam atingidos os objetivos do grupo de trabalho.

O resultado do trabalho de um WG é composto de um ou mais documentos. Um documento que geralmente é disponibilizado para revisões, mas ainda não publicado, recebe o nome de *internet-draft* (ID), o qual representa na verdade um trabalho em andamento. Um ID pode variar de um "*raugh graft*" até um "*final Draft* " antes de ser publicado pelo IETF. Quando publicado, o documento pode estar no processo de

padronização (*standards track*) ou pode receber o status *Informational* ou *Experimental*. O primeiro passo de um documento no processo de padronização é o status *Proposed*. Este é seguido pelo status *Draft* e finalmente pelo status *Standard* ou *Full Standard*. Um documento que é o trocado por uma versão atualizada recebe o status *Obsolete*. Um documento que é "aposentado" recebe o status *Historic* [MEIRELLES 97].

Todo documento publicado pelo IETF recebe um número de RFC e é disponibilizado em um meio *on-line*. Aproximadamente de quatro em quatro meses, o IETF publica um documento chamado *INTERNET OFFICIAL PROTOCOL STANDARDS*, que especifica o status de todos os documentos publicados até aquele momento.

É importante ressaltar a diferença entre o status dos documentos *Draft* e *Internet-Draft*. Enquanto o primeiro significa um documento oficialmente publicado pelo IETF, com um número de RFC associado e, portanto, no processo de padronização, o último significa apenas um documento que informa sobre a situação de algum trabalho em andamento.

2.4 Conclusões

A padronização dos sistemas de gerenciamento junto às arquiteturas de rede foi de suma importância para globalizar sua utilização. A utilização de sistemas de gerenciamento proprietários limitava a eficiência da administração do ambiente, já que estes eram apenas capazes de interagir somente com equipamentos produzidos pelo mesmo fornecedor que o desenvolveu. A arquitetura de gerenciamento internet possibilitou a padronização do modelo de informação de gerenciamento disponibilizado pelos recursos a serem gerenciados e é a mais difundida no mundo. O protocolo padrão especificado para seu gerenciamento é o SNMP, que será descrito no próximo capítulo.

CAPÍTULO III

3. Simple Network Management Protocol – SNMP

3.1 Introdução

Este capítulo conceitua o protocolo SNMP de uma maneira bem mais detalhada em relação aos conceitos apresentados na introdução deste trabalho.

Inicialmente o capítulo apresenta uma descrição do protocolo e as versões existentes. Em seguida é conceituada a arquitetura do protocolo, definindo seus componentes básicos e a forma em que a comunicação acontece.

Para esclarecer a maneira como as entidades SNMP se comunicam, o capítulo conceitua as operações SNMP, dando mais ênfase aos objetos principais do modelo proposto neste trabalho: os Traps. São também discutidos aspectos de segurança em SNMP e a coexistência das duas versões.

Por fim o capítulo conceitua a MIB, uma base que contém as informações de gerenciamento que auxiliam o administrador na tomada de decisões.

3.2 SNMP - Conceito

O SNMP é um protocolo, definido ao nível de camadas de aplicação, que facilita a troca de informações de gerenciamento entre dispositivos de rede. O SNMP habilita administradores de rede a gerenciar a performance da rede, encontrar e solucionar problemas e planejar seu crescimento de maneira eficaz [CASE 90].

Atualmente existem 3 versões do protocolo SNMP: SNMP versão 1(SNMP v1), SNMP versão 2 (SNMP v2) e SNMP versão 3 (SNMP v3). As 3 versões possuem um grande

número de ferramentas em comum, contudo o SNMP v3 e o SNMP v2 disponibilizam grandes melhorias em relação ao SNMP v1, tais como operações adicionais do protocolo. Apesar da terceira versão conter bem mais funcionalidades, em relação às duas primeiras versões, esta ainda está longe de ser utilizada tanto quanto a primeira. A fig. 3.1 ilustra uma rede básica de gerência SNMP.

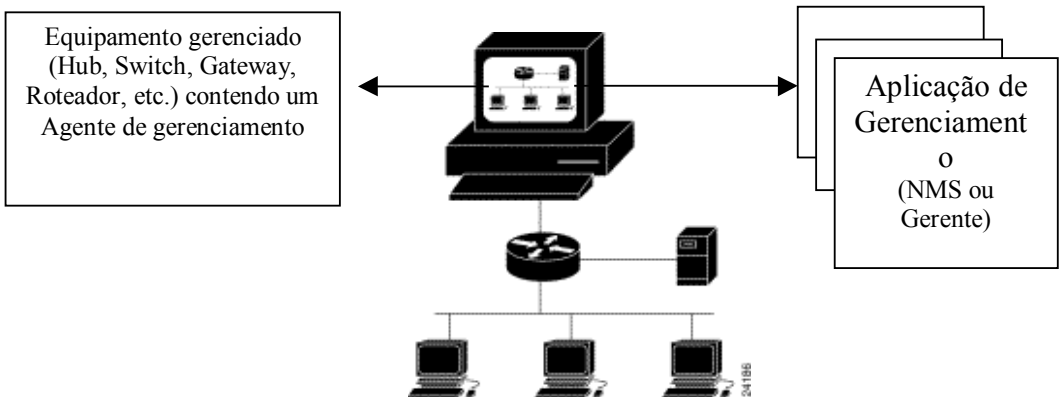

FIGURA 3.1 – Rede básica de gerência SNMP

O modelo SNMP é definido em uma coleção de documentos denominados *Request for Comments – RFC'S*. As *RFC'S* definem uma estrutura de gerenciamento que compreende: um protocolo de gerenciamento, a definição de gerenciamento e eventos, um núcleo de informações de gerenciamento e eventos, e um mecanismo para gerenciar o uso do protocolo. A tab. 3.2 relaciona os documentos que definem a estrutura do modelo SNMP nas versões 1 e 2.

TABELA 3.2: Documentos que definem a estrutura do modelo SNMP v1 e v2.

ESCOPO	DESCRIÇÃO	PUBLICAÇÃO	RFC
Protocolo de Gerenciamento	SNMP v1 Protocol	Maio 1990	1157
	SNMP v2 Protocol Operations	Janeiro 1996	1905
	SNMP v2 Transport Mappings	Janeiro 1996	1906
Estrutura de Informações de Gerenciamento	SMI v1 with typs fixed	Maio 1990	1155
	SMI v1 Concise MIB format	Março 1991	1212
	SMI v1 Traps formats	Março 1991	1215
	SMI v2	Janeiro 1996	1902
	SMI v2 Textual Conventions	Janeiro 1996	1903

	SMI v2 Conformances	Janeiro 1996	1904
Núcleo de Informações de Gerenciamento	MIB II SNMP v2 Core	Março 1991 Janeiro 1996	1213 1907

3.3 Objetivos da Arquitetura

O modelo proposto busca minimizar o número e a complexidade de funções de gerenciamento realizadas pelos agentes de gerenciamento. As razões que tornam este objetivo atrativo são [STALLINGS 96]:

- O custo de desenvolvimento do software de agente de gerenciamento, necessário para suportar o protocolo, é significativamente reduzido;

- O grau e a quantidade das funções de gerenciamento é gradativamente aumentado na medida em que surgem novos recursos tecnológicos internet. Estas novas funcionalidades vêm sob a forma e sofisticação das ferramentas de gerenciamento;

- Conjuntos simplificados de funções de gerenciamento são facilmente entendidos e utilizados pelos desenvolvedores de ferramentas de gerenciamento de redes.

O segundo objetivo do protocolo é que o paradigma funcional para monitoração e controle deve ser suficientemente extensível para acomodar aspectos adicionais, e possivelmente não previstos da operação e gerenciamento de redes.

O terceiro objetivo é que a arquitetura deve ser, tanto quanto possível, independente da arquitetura e dos mecanismos de hospedeiros e gateways particulares, ou seja, não proprietários, podendo ser utilizado por qualquer equipamento.

3.4 Componentes Básicos da Arquitetura SNMP

Uma rede de computadores, gerenciada segundo o modelo SNMP, deve possuir os seguintes elementos [CASE 90]:

- Um ou mais nós gerenciados, cada um contendo uma entidade de processamento denominada agente;

- Pelo menos uma estação de gerenciamento contendo uma ou mais entidades de processamento denominadas aplicações de gerenciamento (gerentes);

- Opcionalmente entidades de processamento capazes de agir tanto no papel de gerente como de agente, chamadas de entidades de duplo comportamento;

- Informação de gerenciamento em cada nó gerenciado, que descreve a configuração, estado, estatística e que controla as ações do nó gerenciado;

- Um protocolo de gerenciamento, o qual os gerentes e agentes utilizam para trocar mensagens de gerenciamento, aqui no caso o SNMP.

A fig. 3.2 descreve os elementos supracitados, existentes em uma rede de computadores que utiliza o protocolo de gerenciamento de rede SNMP.

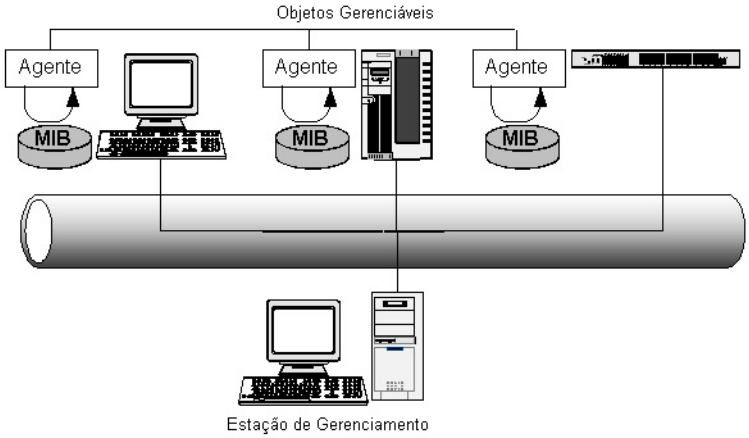

FIGURA 3.2: Elementos básicos da arquitetura SNMP.

A estação de gerenciamento é tipicamente uma máquina dedicada a este fim, mas é comum encontrar-se máquinas oferecendo outros serviços na rede. A estação de gerenciamento serve como a interface para o gerente da rede.

A estação de gerenciamento deve possuir no mínimo [STALLINGS 93]:

- Um conjunto de aplicações para análise de dados, recuperação de falhas, etc.;

- Uma interface pela qual o gerente da rede possa monitorar e controlar a rede de computadores;

- Capacidade de traduzir os requisitos de gerente de redes em monitoração e controle efetivos de elementos na rede;

- Uma base de dados de informações extraídas das MIBs de todas as entidades gerenciada na rede.

No modelo SNMP a interconexão de rede consiste de uma coleção de uma ou mais redes que utilizam protocolos comuns e são conectadas por *gateways*. Qualquer um dos dois pontos finais pode se comunicar através da implementação de um esquema de endereçamento global e do uso de protocolos padronizados como os que formam a pilha *TCP/IP* (*Transmission Control Protocol/ Internet Protocol*). Nos protocolos existem regras as quais permitem que a comunicação entre redes seja possível.

A comunicação de redes que utilizam a pilha *TCP/IP* é realizada por diferentes protocolos, os quais operam ao longo das camadas que formam a arquitetura. Um *host*, por exemplo, deverá possuir a implementação de ao menos um protocolo em cada uma das camadas (camada de acesso à rede, camada de rede, camada de transporte e camada de aplicação (superior)).

Outro componente ativo no gerenciamento Internet é o agente de gerenciamento. Normalmente esta entidade é implementada em elementos-chave da rede, como hosts,

pontes, roteadores, switches, hubs, etc. Assim, cada um destes elementos pode ser gerenciado pela estação de gerenciamento.

O agente responde às requisições enviadas pelo gerente. Estas requisições podem ser um simples pedido de leitura de uma informação, como também podem ser uma requisição para tomada de uma ação. Eventualmente, quando da ocorrência de determinados eventos, o agente pode notificar o gerente deste fato, sem que tenha sido solicitado para tal [STALLINGS 96].

Os recursos da rede são representados como objetos. Cada objeto, na verdade, é uma variável que representa uma informação sobre o recurso gerenciado. O conjunto de todos os objetos de um recurso gerenciado forma uma visão da base de informações de gerenciamento (MIB) que é acessada pelo agente com o intuito de enviar informações ao gerente.

As variáveis da MIB podem ser vistas como pontos de acesso pelos quais um gerente pode atingir um agente. Normalmente, recursos gerenciados semelhantes (como roteadores) possuem o mesmo conjunto básico de variáveis (mas cada fabricante pode incluir outras informações adicionais sobre seu produto).

A comunicação existente entre o gerente e o agente se dá através de um protocolo específico de gerenciamento, neste caso o SNMP. Para possibilitar essa comunicação, o SNMP disponibiliza unidades de dados de protocolo (PDU), descritas mais adiante quando forem conceituadas as duas primeiras versões do SNMP [STALLINGS 96].

A fig. 3.3 demonstra graficamente a comunicação entre os dois módulos de aplicações básicas existentes no modelo de gerenciamento de redes Internet, o gerente e o agente.

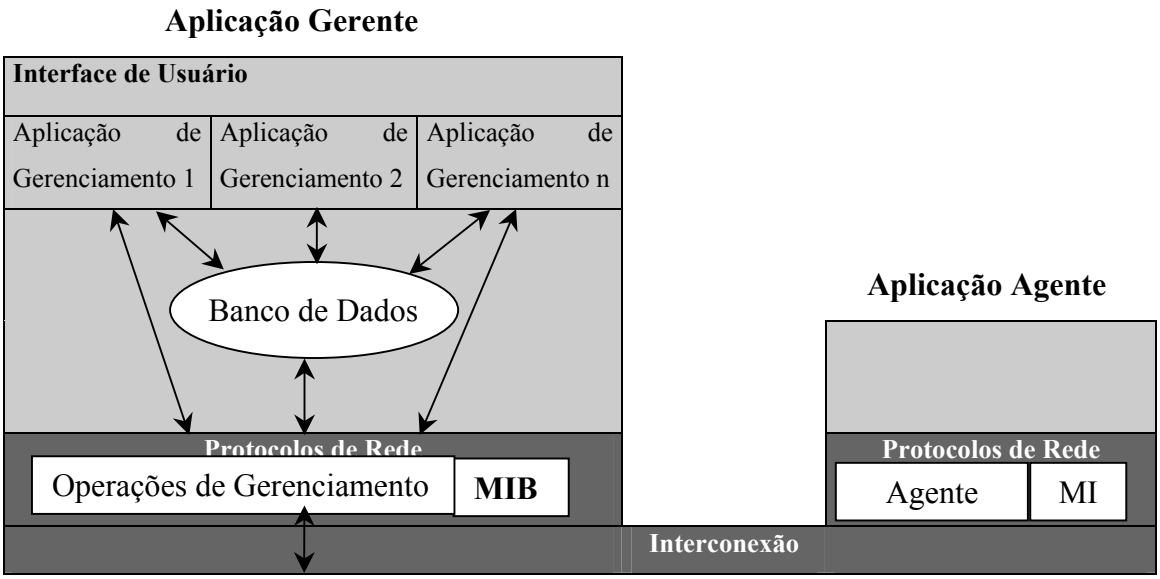

FIGURA 3.3: Tipos de aplicações de gerenciamento e forma de comunicação

3.5 Modelo de Informação

Dois documentos definem a informação de gerenciamento do modelo SNMP: o RFC 1065, Estrutura da Informação de Gerenciamento; e o RFC 1066, Base de Informação de Gerenciamento. Os dois documentos foram projetados para serem compatíveis tanto com os modelos de gerenciamento SNMP quanto OSI, porém, o modelo de informação de gerenciamento OSI utiliza definições mais complexas [ROSE 88].

As funções de gerenciamento de rede podem ser agrupadas em duas categorias: monitoração de rede e controle de rede. A monitoração da rede está relacionada com a tarefa de observação e análise do estado e configuração de seus componentes; é uma função de "leitura". O controle da rede é uma função de escrita e está relacionada com a tarefa de alteração de valores de parâmetros e execução de determinadas ações.

3.5.1 Monitoração da Rede

A monitoração consiste na observação de informações relevantes ao gerenciamento. Estas informações podem ser classificadas em três categorias:

- Estática: caracteriza a configuração atual e os elementos na atual configuração, tais como o número e identificação de portas em um roteador;

- Dinâmica: relacionada com os eventos na rede, tais como a transmissão de um pacote na rede;

- Estatística: pode ser derivada de informações dinâmicas; ex. média de pacotes transmitidos por unidade de tempo em um determinado sistema.

A informação de gerenciamento é coletada e armazenada por agentes e repassada para um ou mais gerentes. Duas técnicas podem ser utilizadas na comunicação entre agentes e gerentes: *polling* e *event-reporting*.

A técnica de *polling* consiste em uma interação do tipo *request/response* entre um gerente e um agente. O gerente pode solicitar a um agente (para o qual ele tenha autorização), o envio de valores de diversos elementos de informação. O agente responde com os valores constantes em sua MIB [HARNEDY 97].

Na técnica de *event-reporting* a iniciativa é do agente. O gerente fica na escuta, aguardando pela chegada de informações. Um agente pode gerar um relatório periodicamente para fornecer ao gerente o seu estado atual. A periodicidade do relatório pode ser configurada previamente pelo gerente. Um agente também pode enviar um relatório quando ocorrer um evento significativo ou não usual.

Tanto o *polling* quanto o *event-reporting* são utilizados nos sistemas de gerenciamento, porém a ênfase dada a cada um dos métodos difere muito entre os sistemas. O modelo de gerenciamento OSI contempla os dois métodos, sem nenhuma ênfase em qualquer deles. A arquitetura usada na gerência de redes de telecomunicações é baseada no modelo de gerenciamento OSI, entretanto dá ênfase para o método de relatório de eventos. Em contraste, o modelo SNMP é fortemente voltado para o método de polling, dando pouca importância para os Traps.

A escolha da ênfase depende de um número de fatores, incluindo os seguintes [HARNEDY 97]:

- A quantidade de tráfego gerada por cada método;
- Robustez em situações críticas;
- O tempo entre a ocorrência do evento e a notificação ao gerente;
- A quantidade de processamento nos equipamentos gerenciados;
- A problemática referente à transferência confiável versus transferência não confiável;
- As considerações referentes ao caso em que um equipamento falhe antes de enviar um relatório.

3.5.2 Controle de Rede

Esta parte do gerenciamento de rede diz respeito à modificação de parâmetros e à execução de ações em um sistema remoto. Todas as cinco áreas funcionais de gerenciamento (falhas, desempenho, contabilização, configuração e segurança), envolvem monitoração e controle. Tradicionalmente, no entanto, a ênfase nas três primeiras destas áreas, tem sido na monitoração, enquanto que nas duas últimas, o controle tem sido mais enfatizado. Alguns aspectos de controle na gerência de configuração e de segurança são apresentados a seguir.

O controle de configuração inclui as seguintes funções [HARNEDY 97]:

- Definição da informação de configuração – recursos e atributos dos recursos sujeitos ao gerenciamento;
- Atribuição e modificação de valores de atributos;

- Definição e modificação de relacionamentos entre recursos ou componentes da rede;

- Inicialização e terminação de operações de rede;

- Distribuição de software;

- Exame de valores e relacionamentos;

- Relatórios de status de configuração.

O controle de segurança é relativo à segurança dos recursos sob gerenciamento, incluindo o próprio sistema de gerenciamento. Os principais objetivos em termos de segurança, são relativos à confidencialidade, integridade e disponibilidade. As principais ameaças à segurança referem-se à interrupção, modificação e mascaramento. As funções de gerenciamento de segurança podem ser agrupadas em três categorias [BRISA 93]:

- Manutenção da informação de segurança;

- Controle de acesso aos recursos;

- Controle do processo de criptografia.

3.6 SNMP e Representação de Dados

SNMP deve responsabilizar-se por fazer o ajuste nas incompatibilidades entre dispositivos gerenciados. Máquinas diferentes utilizam diferentes técnicas de representação de dados as quais podem comprometer a habilidade do SNMP para troca de informações entre os dispositivos gerenciados. O SNMP utiliza a notação ASN.1 para acomodar as comunicações entre os sistemas variados [CASE 90].

Para facilitar a compreensão deste protocolo de gerenciamento descreve-se, de forma mais completa, as duas versões mais utilizadas do SNMP, a versão 1 e a versão 2. Devido a pouca disponibilidade e utilização da terceira versão, esta não será conceituada nem utilizada neste trabalho de pesquisa.

3.6.1 SNMP Versão 1 (SNMP v1) e SNMP Versão 2 (SNMPv2)

A versão 1 do SNMP está minuciosamente descrita no RFC 1157. Seu funcionamento está dentro das especificações da Estrutura de Informação de Gerenciamento (SMI - *Structure of Management Information*). Esta primeira versão efetua suas operações utilizando outros protocolos, como por exemplo: o UDP (*User Datagram Protocol*); IP (Internet Protocol); CLNS (OSI *Connectionless Network Service*); DDP (*Apple Talk Datagram-Delivery Protocol*) e IPX (*Novell Internet Packet Exchenge*). O SNMPv1 é largamente utilizado e é, de fato o protocolo para gerência de redes na comunidade internet.

O SNMPv2 é uma evolução da versão inicial descrita anteriormente. Originalmente, o SNMPv2 foi publicado como um grupo de padrões Internet propostos em 1993. Como na versão 1, a versão 2 funciona de acordo com as especificações descritas na SMI. A segunda versão do protocolo SNMP oferece algumas melhorias com relação à primeira versão, incluindo operações adicionais suportadas pelo protocolo [CASE 96a].

3.6.2 Estrutura de Informação de Gerenciamento (SMI)

A SMI define as regras para descrever informações de gerenciamento, utilizando a notação ASN.1. A SMIv1 encontra-se descrita no RFC 1155, enquanto que a SMIv2 no RFC 1902. A SMIv1 criou três especificações chave [CASE 96a]:

- ASN.1 *Data Types* - A SMIv1 descreve que todos os objetos gerenciados possuem um certo subgrupo de tipos de dados ASN.1 associados a eles. Três tipos de dados ASN.1 são requeridos: nome, *syntax* e codificação. O nome tem função de identificador de objeto (*object ID*). A *syntax* define o tipo de dados do objeto (por exemplo, inteiro ou string). A codificação de dados descreve como informações associadas com um objeto

gerenciado é formatada como uma série de itens de dados para transmissões sobre a rede;

- SMI-Specific Data Types - A SMIv1 especifica o uso de um número de SMI-*specific data types*, os quais são divididos em duas categorias [CASE 96a]:

a) *simple data types* - Três *simple data types* são definidos na SMIv1, sendo todos valores únicos: *inteiros*, *string de octetos* e *object IDs*. O tipo de dado *inteiro* compreende valores entre –2.147.483.648 até 2.147.483.647. *String de Octetos* são seqüências ordenadas de zero a 65.535 octetos. *Object IDs* vem do grupo de todos os objetos identificadores alocados de acordo com as regras especificadas em ASN.1.

b) *application-wide data types* - Sete tipos existem na SMIv1: *network addresses*, *counters*, *gauges*, *time ticks*, *opaques*, *integers* e *unsigned integers*. O endereço de rede representa um endereço de uma família de protocolos particulares. O SNMPv1 suporta apenas endereços IP de 32 bits. Contadores são inteiros não negativos o qual aumentam até que eles alcancem um valor máximo e depois retornem a zero. No SNMPv1, uma medida de contador de 32 bits é especificada. Medidores são inteiros não negativos o qual podem aumentar ou decrescer, mas retendo o valor máximo alcançado. Um *time tick* representa um centésimo de segundo desde a ocorrência de um evento. Um *opaque* representa um codificador arbitrário que é utilizado pela SMI. Um *integer* representa uma informação de valor inteiro assinado. Este tipo de dado redefine o tipo de dado inteiro, o qual possuem uma precisão arbitrária em ASN.1 mas precisão limitada na SMI. Um *unsigned integer* representa informações de valores inteiros não assinados e é útil quando valores são sempre não negativos. Este tipo de dado redefine o tipo de dado inteiro, o qual tem precisão arbitrária em ASN.1, mas precisão limitada para o SMI.

- SNMP *MIB-Tables* - A SMIv1 define tabelas altamente estruturadas que são utilizadas por grupos de instâncias de objetos arrumas em tabelas (um objeto que

contém múltiplas variáveis). Tabelas são compostas de zero ou mais linhas, os quais são indexadas de maneira a possibilitar o SNMP retornar ou alterar uma linha inteira com um simples comando Get, GetNext ou Set.

Na SMIv2 existem melhorias em cima da SMIv1-specific data types, assim como a inclusão de *bit strings*, *endereços de redes* e *contadores*. *Bit strings* são definidos apenas no SNMPv2 e compreendem zero ou mais bits nomeados que especificam um valor. Endereços de Rede representam um endereço de uma família particular de protocolo. O SNMPv1 suporta apenas endereços IP de 32 bits, já no SNMPv2 suporta outros tipos de endereços. Contadores são inteiros não negativos que aumentam até alcançarem seus valores máximos, retornando a zero depois. No SNMPv1, apenas é especificado um contador de 32 bits. Já na versão 2 são definidos contadores de 32 e 64 bits [CASE 96a].

3.6.3 Mensagens SNMP

O protocolo SNMP é uma aplicação para gerenciamento de redes pela qual as variáveis (instâncias de objetos) contidas na MIB do(s) agente(s) devem ser inspecionadas ou alteradas [STALLINGS 96].

A comunicação entre as entidades envolvidas no protocolo é efetuada através da troca de mensagens. Cada mensagem é definida completa e independentemente através de um datagrama UDP, utilizando as regras básicas da notação ASN.1. Uma mensagem consiste de um identificador de versão, um nome de comunidade SNMP e uma unidade de dados do protocolo (PDU – *Protocol Data Unit*) [STALLINGS 96].

Uma entidade do protocolo residente no *host* recebe mensagens de resposta através da porta UDP 161 e mensagens que reportam *Traps*, na porta UDP 162. Esta separação de portas de recepção é devido ao fato de que *Traps* são mensagens que devem ser identificadas para posterior processamento [STALLINGS 96].

Uma implementação deste protocolo não necessita aceitar mensagens as quais excedem 484 octetos, entretanto, é recomendado que as implementações suportem datagramas com o maior tamanho possível.

As mensagens do SNMP, tanto na versão 1 quanto na versão 2, contém duas partes: um cabeçalho de mensagem e a unidade de dados do protocolo (PDU). A fig. 3.4 apresenta o formato básico de uma mensagem SNMPv1 [STALLINGS 96].

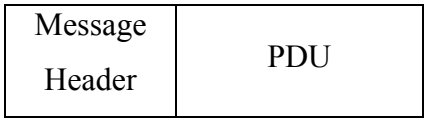

FIGURA 3.4: formato básico de uma mensagem SNMPv1

O cabeçalho da mensagem SNMPv1 contém dois campos [CASE 90]:

- Número da Versão – Especifica a versão do SNMP utilizado;

- Nome da Comunidade – Define um ambiente de acesso para um grupo de NMS's. NMS's dentro de comunidades são ditos existentes em alguns domínios administrativos. Nomes de comunidades são uma forma frágil de autenticação já que dispositivos que desconhecem suas próprias comunidades são impedidos de executar operações pelo SNMP.

Uma PDU é uma denominação genérica para uma unidade de dados suportada por um protocolo. Tratanto-se de um ambiente SNMP, uma PDU consiste em uma mensagem trocada entre as entidades envolvidas no modelo. As PDUs suportadas pelo protocolo SNMP estão descritas a seguir.

3.6.4 Mensagens suportadas pelos protocolos SNMPv1 e SNMPv2

É obrigatório que todas as implementações do SNMP suportem as cinco PDUs implementadas na primeira versão: GetRequest-PDU, GetNextRequest-PDU, GetResponse-PDU, SetRequest-PDU e Trap-PDU [CASE 90].

A Fig. 3.5 ilustra os campos definidos para as PDUs SNMPv1 GetRequest, GetNextRequest, GetResponse e SetRequest. A Figura 3.6 ilustra a PDU Trap [CASE 90].

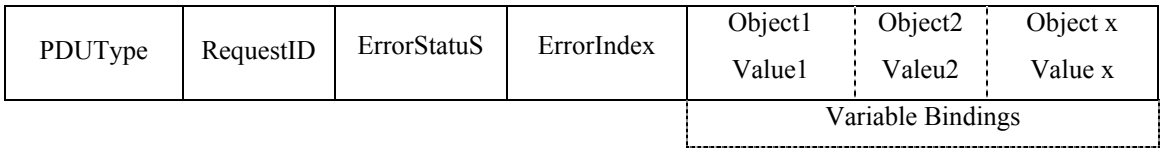

FIGURA 3.5: Formato de uma PDU SNMPv1.

Onde cada campo significa:

- *PDU Type* – Especifica o tipo da PDU transmitida;

- *Request ID* – Associa solicitações SNMP com respostas;

- *Error Status* – Indica um número dentre uma infinidade de tipos de erros. Apenas a operação resposta seta este campo. Outras operações seta este campo a zero;

- *Error Index* – Associa um erro com uma instância de objeto particular. Apenas a operação resposta seta este campo, as demais colocam zero;

- *Variable Bindings* – Serve como um campo de dados da PDU SNMPv1. Cada variável associa uma instância particular de um objeto com seu valor correspondente (com a exceção de Get e GetNext request, pelo qual o valor é ignorado).

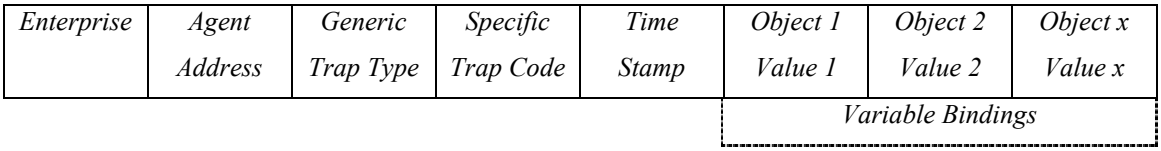

FIGURA 3.6: Formato de uma Trap-PDU SNMPv1.

Onde cada campo significa:

- Enterprise – Identifica o tipo do objeto gerenciado que gerou o *Trap*;
- Agent Address – Fornece o endereço do objeto gerenciado que gerou o *Trap*;
- Generic Trap Type – Indica um número, dentre vários, de tipos genéricos de *Traps*;
- Specific Trap Code – Indica um código específico para o *Trap* gerado;
- Time Stamp – Fornece a quantidade de tempo decorrente entre a última reinicialização da rede até a geração do *Trap*;
- Variable Bindings – O campo de dados da PDU Trap *SNMPv1*. *Cada variable binding associa uma instância particular de um objeto* com seu valor correspondente.

No SNMPv2, as PDUs GetRequest, GetNextRequest, InformRequest, GetResponse e SetRequest têm o mesmo formato das PDUs descritas acima. A principal diferença refere-se ao *Trap*, pois na segunda versão do protocolo a Trap-PDU tem o mesmo formato da PDU GetRequest [CASE 90d].

3.6.4.1 Procedimento para geração de uma PDU

Esta sessão descreve as ações tomadas por uma entidade de protocolo ao implementar uma PDU SNMP. Nos procedimentos a seguir, é mencionado o termo endereço de transporte, que no caso do UDP, consiste de um endereço IP juntamente com uma porta. Outro serviço de transporte pode também ser utilizado para o SNMP, neste caso, a definição de endereço de transporte deve ser feita em conformidade a este serviço. Para gerar uma mensagem, uma entidade de protocolo de nível superior, implementa os seguintes passos:

1. Primeiramente é construída a PDU como um objeto ASN.1, por exemplo a GetRequest-PDU;

2. Em seguida passa este objeto para o serviço que implementa o esquema de autenticação desejado com: um nome de uma comunidade, o endereço de transporte da origem, e o endereço de transporte do destino. Este serviço de autenticação retorna um outro objeto ASN.1;

3. A entidade de protocolo constrói então um objeto mensagem ASN.1 utilizando o nome da comunidade e o objeto ASN.1 resultante;

4. Este novo objeto ASN.1 é então serializado, usando as regras básicas de codificação ASN.1, e então enviado, utilizando um serviço de transporte, para a entidade de protocolo destino.

As ações da entidade de protocolo que recebe a mensagem são as seguintes:

1. Ela executa uma divisão elementar do datagrama chegado para construir um objeto ASN.1 correspondente a um objeto mensagem ASN.1. Se a divisão falhar, ele descarta o datagrama e não executa mais nenhuma ação;

2. Ela então verifica o número da versão da mensagem SNMP; se for incompatível, ele então descarta o datagrama e não faz mais nada;

3. A entidade de protocolo transmite então o nome da comunidade e os dados do usuário, encontrados no objeto mensagem ASN.1, junto com o código de origem do datagrama e o endereço de transporte do destinatário para o serviço o qual implementa o esquema de autenticação desejado. Esta entidade retorna um outro objeto ASN.1, ou sinaliza uma falha de autenticação. Em um segundo caso, a entidade de protocolo registra esta falha, (se possível) gera um *Trap*, descarta o datagrama e não faz mais nada;

4. A entidade de protocolo então executa uma divisão elementar no objeto ASN.1 retornado do serviço de autenticação para construir um objeto ASN.1 correspondente a um objeto PDU ASN.1. Se a divisão falhar, ele descarta o datagrama e não faz mais nada. De outro jeito, utilizando a comunidade SNMP nomeada, o perfil apropriado é selecionado e a PDU é processada conforme solicitado. Se, como resultado deste processo, uma mensagem é retornada, então o endereço de transporte origem que enviou a mensagem deveria ser idêntico ao endereço de transporte do destinatário que enviou a mensagem original solicitante.

3.6.4.2 Construções Comuns

Antes de conceituar as PDUs do SNMPv1 e SNMPv2, é importante ressaltar algumas utilizações constantes do ASN.1:

```
-- request/response information
RequestID ::=
        INTEGER
ErrorStatus ::=
        INTEGER {
           noError(0),
           tooBig(1),
           noSuchName(2),
           badValue(3),
           readOnly(4)
           genErr(5)
        }
ErrorIndex ::=
        INTEGER
-- variable bindings
VarBind ::=
        SEQUENCE {
           name
              ObjectName,
           value
```

```
            ObjectSyntax
        }
VarBindList ::=
        SEQUENCE OF
          VarBind
```

RequestIDs são utilizados para distinção entre solicitações que estão sendo tratadas. Através do uso de *RequestID*, uma entidade de aplicação SNMP pode correlacionar respostas que chegam com solicitações que estão sendo tratadas. Em casos onde um serviço de datagrama não confiável está sendo utilizado, o *RequestID* provê também um meio simples de identificação de mensagens duplicadas na rede.

Uma instância não negativa de *ErrorStatus* é utilizada para indicar que uma exceção ocorreu enquanto estava processando um pedido. Nestes casos, *errorIndex* poderá oferecer informações adicionais indicando quais variáveis, em uma lista, causaram a exceção.

O termo "variáveis" refere-se a uma instância de um objeto gerenciado. Uma *variable binding*, ou *VarBind*, refere-se ao casamento do nome da variável com o seu valor. Uma *VarBindList* é simplesmente uma lista de nomes de variáveis e seus valores correspondentes. Algumas PDUs preocupam-se apenas com o nome da variável e não seu valor, por exemplo a GetRequest-PDU, neste caso o valor é ignorado pela entidade de protocolo. Entretanto, a porção valor deve ainda ter uma syntax ASN.1 válida e codificada. É recomendado que o valor ASN.1 nulo seja usado pela porção valor de tal ligação.

3.6.4.3 A GetRequest-PDU

Esta PDU é enviada de uma estação de gerenciamento para um agente, sendo utilizada para devolver o valor da instância de objeto referenciado pelo OID contido na PDU.

Abaixo estão listados alguns erros que podem ocorrer:

1. Se nenhum objeto na MIB for compatível com aquele cuja leitura foi solicitada na PDU, então o agente devolve um GetResponse-PDU da mesma forma, exceto que o valor do campo error-status é *noSuchName*;

2. Se qualquer objeto, referenciado no campo *variable-bindings* da GetRequest-PDU, for do tipo agregado (como definido na SMI), então o agente envia ao gerente uma mensagem GetResponse-PDU da mesma forma, exceto que o valor do campo error-status é *noSuchName*;

3. Se o tamanho da GetResponse-PDU gerada, como descrita acima, exceder uma limitação local, então o agente envia ao gerente uma mensagem GetResponse-PDU da mesma forma, exceto que o valor do campo error-status é tooBig, e o valor do campo *error-index* é zero.

3.6.4.4 A GetNextRequest-PDU

A forma desta PDU é idêntica a GetRequest-PDU, exceto pela indicação do tipo da PDU [CASE 90].

Esta PDU é enviada de uma estação de gerenciamento para um agente, sendo utilizada para devolver o(s) valor(es) da(s) instância(s) de objeto(s) cujo(s) OID(s) é(são), lexicograficamente, o próximo daquele OID contido na PDU.

Caso não ocorra nenhuma situação de erro, como aquelas descritas para a GetRequest-PDU, então o agente envia ao gerente uma mensagem do tipo GetResponse-PDU, só que, para cada nome especificado no campo variable-bindings da mensagem recebida, o componente correspondente da GetResponse-PDU representa o nome e o valor da variável sucessora imediata na MIB.

3.6.4.5 A GetResponse-PDU

A forma da GetResponse-PDU é idêntica a GetRequest-PDU, exceto a indicação do tipo da PDU [CASE 90].

A GetResponse-PDU é gerada por um agente apenas se receber um GetRequest-PDU, GetNextRequest-PDU ou SetRequest-PDU. Esta PDU é uma resposta à solicitação do gerente, sendo que ao receber um pedido em forma de PDU (GetRequest por exemplo) uma entidade de protocolo encaminha ao solicitante o resultado em forma de uma mensagem resposta, um GetResponse.

3.6.4.6 A SetRequest-PDU

Como a GetResponse-PDU, a forma desta PDU é idêntica a GetRequest-PDU, exceto na indicação do seu tipo de PDU [CASE 90].

SetRequest-PDU é uma operação de escrita, que consiste na solicitação do gerente para a alteração do conteúdo (valor) de uma determinada variável constante na MIB. Esta variável somente será alterada se for do tipo *read-write*, pois se a variável solicitada for do tipo *read-only* então o agente envia um GetResponse-PDU informando da não concretização do pedido. Da mesma forma acontece se não existir tal variável ou qualquer erro. Se nada de errado ocorrer, então a entidade receptora envia um GetResponse-PDU, tendo o valor no campo error-status igual a *noError* e o valor do campo *error-index* zero.

3.6.4.7 A Trap-PDU

Esta PDU é enviada pelo agente a um grupo selecionado de estações de gerenciamento, com o objetivo de notificar um evento fora do comum ocorrido na rede. Estes eventos são padronizados e subdivididos em 7 [CASE 90]:

- **O coldStart Trap** – Um *Trap* do tipo coldStart(0) significa que a entidade de envio de protocolo está se reinicializando para que a configuração do agente ou a implementação da entidade de protocolo possa ser alterada [CASE 90];

- **O warmStart Trap** – Um *Trap* do tipo warmStart(1) significa que a entidade de envio de protocolo está se reinicializando, mas não especificamente para mudança

da configuração do agente ou da implementação da entidade de protocolo [CASE 90];

- **O linkDown Trap** – Um *Trap* do tipo linkDown(2) significa que a entidade de envio de protocolo identifica uma falha em um dos links de comunicação contidos na configuração do agente. Este tipo de *Trap* contém, como primeiro elemento do seu campo *variable-bindings*, o nome e o valor da instância *ifIndex* para a interface afetada [CASE 90];

- **O linkUp Trap** – Um *Trap* do tipo linkUp(3), significa que a entidade de envio de protocolo reconhece que um dos links de comunicação, contidos na configuração do agente, voltou ao ar. Este tipo de *Trap* contém, como primeiro elemento do seu campo *variable-bindings*, o nome e o valor da instância *ifIndex* para a interface afetada [CASE 90];

- **O authenticationFailure Trap** – Um *Trap* do tipo authenticationFailure(4) significa que uma entidade emissora de protocolo é o destinatário de uma mensagem de protocolo que não está corretamente autenticada. Uma implementação SNMP além de ser capaz de gerar um *Trap* deste tipo, ela deve ainda ser capaz de anular a emissão destes através de uma implementação com mecanismo específico [CASE 90];

- **O egpNeighborLoss Trap** - Um *Trap* do tipo egpNeighborLoss(5) significa que um vizinho EGP, o qual possuía como par EGP uma entidade de envio de protocolo, foi marcada como "caído", não mais obtendo seu relacionamento par. Uma PDU *Trap* deste tipo contém como primeiro elemento da sua *variable-bindings*, o nome e valor da instância *egpNeighAddr* ao vizinho afetado [CASE 90];

- **O enterpriseSpecific Trap** – Um *Trap* do tipo enterpriseSpecific(6) significa que a entidade emissora do protocolo reconhece que algum evento proprietário do

fabricante ocorreu. O campo *specific-trap* identifica o *Trap* proprietário ocorrido [CASE 90].

É importante ressaltar que a operação *Trap*, no SNMPv2, satisfaz a mesma função daquela utilizada pela primeira versão, contudo, utiliza um formato diferente de mensagem e é designada para substituir o *Trap* SNMPv1.

Além destas PDUs existentes nas duas primeiras versões do SNMP, a segunda versão trouxe outras PDUs.

3.6.4.8 A GetBulkRequest-PDU

A PDU GetBulkRequest tem o seguinte formato [CASE 96d]:

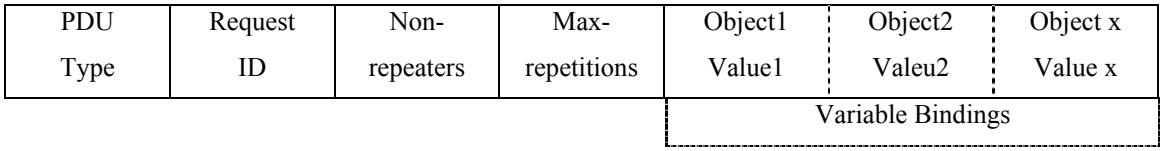

FIGURA 3.7: Formato de uma PDU SNMPv2

Onde cada campo significa:

- PDU Type – Identifica o tipo da PDU transmitida *(Get, GetNext, Inform, Response, Set ou Trap*;.

- Request ID – Associa solicitações *SNMP* com respostas;

- Error Status – Indica um de um número de erros e tipos de erros. Apenas a operação resposta seta este campo, as demais operações preenchem este campo com o valor zero;

- Error Index – Associa um erro com uma instância particular de um objeto. Apenas a operação resposta seta este campo, as demais preenchem este campo com o valor zero;

- Variable Bindings - *Serve como o campo de dados da PDU SNMPv2. Cada variable binding associada a uma instância particular de um objeto com seu valor correspondente (exceto do Get e GetNext requests*, pelos quais os valores são ignorados).

Esta PDU é utilizada pelo NMS para retornar, de forma eficiente, grandes blocos de dados, como, por exemplo, múltiplas linhas em uma tabela. A GetBulkRequest-PDU constrói uma mensagem resposta na mesma medida do dado solicitado como combinarem. Se o agente responder a esta operação que não pode prover valores de todas as variáveis da lista, então ele provê resultados parciais.

3.6.4.9 A InformRequest-PDU

A InformRequest-PDU possui o mesmo formato que a GetRequest-PDU. Esta operação permite a um NMS enviar informações *Trap* para outros NMS e receber uma resposta [CASE 96d].

No SNMP, as notificações podem ser enviadas em forma de *traps* ou *inform-requests*. *Traps* não são confiáveis devido a não confirmação de sua chegado no destino. O emissor não é capaz de determinar a efetivação da operação de envio do *Trap*. Entretanto, uma entidade SNMP ao receber um *inform-request*, confirma seu recebimento junto ao emissor através de um SNMP response PDU. Caso o emissor não receba a mensagem de confirmação da chagada, o *inform-request* pode ser enviado novamente [CISCO 00].

Entretanto, *informs* consomem mais recursos no agente e na rede, diferente de um *Trap*, o qual é descartado tão logo seja enviado. Um *inform-request* fica armazenado em memória até que a resposta seja recebida ou que seja esgotado o tempo de resposta. Além disso, os Traps são enviados apenas uma vez, enquanto que um *inform* pode ser

reenviado quantas vezes for necessário, contudo estes reenvios aumentam o tráfego e contribuem com o overhead na rede [CISCO 00].

3.7 Gerenciamento SNMP

O SNMP é um protocolo de gerenciamento centralizado. Um sistema pode operar exclusivamente em um NMS ou num agente, ou ele pode executar funções em ambos. Quando um sistema opera em ambos, NMS e agente, um outro NMS solicita que o sistema consulte os dispositivos gerenciados e disponibiliza um sumário das informações conhecidas, ou que ele repasse informações de gerenciamento localmente armazenadas.

3.8 Segurança no SNMP

O SNMP é carente de capacidades de autenticação, apenas adotando um esquema de comunidades, resultando com isso em vulnerabilidade em uma variedade de ameaças. Estes incluem disfarces (*masquerading)*, modificações de informação (*modification of information)*, seqüências de mensagens (*message sequence)* e modificações no tempo (*timing modifications)* e revelação (*disclosure)*.

a) Disfarce (*masquerading*) consiste da tentativa de uma entidade não autorizada, executar operações de gerenciamento assumindo a identidade de uma entidade gerenciável autorizada;

b) Modificações de informações (*modification of information*) envolvem a tentativa de uma entidade não autorizada, alterar uma mensagem gerada por uma entidade autorizada de forma que a mensagem resulte na contabilização gerenciável não autorizada ou operação de gerenciamento de configuração;

c) Seqüências de mensagens (*message sequence*) e modificações no tempo (*timing modifications*) ocorrem quando uma entidade não autorizada reordena, atrasa ou copia, repetindo em seguida a mensagem gerada por uma entidade autorizada;

d) Revelação (*disclosure*) resulta quando uma entidade não autorizada extrai valores armazenados em um objeto gerenciável, ou a descoberta de notificações de eventos através da troca de monitoramento entre gerentes e agentes.

Devido ao fato do SNMP não implementar autenticações, muitos fornecedores não implementam a operação Set, reduzindo com isso a facilidade do SNMP em controlar redes.

3.9 Interoperabilidade no SNMP

Como descrito anteriormente, o SNMPv2 é incompatível com o SNMPv1 em duas áreas chave: formatos de mensagens e operações suportadas pelos protocolos. As mensagens na segunda versão do protocolo utilizam cabeçalhos e formatos de unidade de dados do protocolo (PDU) diferentes daquelas implementadas no SNMPv1. O SNMPv2 também utiliza duas operações de protocolo que não são especificadas na versão 1, além disto, o RFC 1908 define duas estratégias de coexistência possíveis para SNMPv1/v2: agentes proxy e sistemas de gerenciamento de redes "bilíngüe" [CASE 96e].

3.9.1 Agentes proxy

Um agente SNMPv2 pode atuar como um *agente proxy* em nome de dispositivos gerenciáveis do SNMPv1, como segue [CASE 96e]:

- Um NMS SNMPv2 implementa uma operação(mensagem) a um agente SNMPv1;
- O NMS envia a mensagem SNMP para o agente proxy SNMPv2;
- O agente proxy encaminha as mensagens Get, GetNext e Set para o agente SNMPv1;
- Mensagens GetBulk são convertidas pelo agente proxy para mensagens GetNext e depois são encaminhadas para os agentes SNMPv1;

- O agente proxy mapeia mensagens Trap SNMPv1 para mensagens Trap SNMPv2 e encaminhando em seguida para o NMS.

3.9.2 Sistemas de Gerenciamento de Redes Bilíngue

Sistemas para gerenciamento de redes SNMPv2 bilíngue suportam tanto o SNMPv1 quanto SNMPv2. Para suportar estes dois ambientes duplos de gerência, uma aplicação de gerenciamento em um NMS bilíngüe deve contactar um agente. O NMS examina então informações armazenadas em uma base de dados local para determinar se o agente suporta SNMPv1 ou SNMPv2. Baseado nas informações da base de dados, o NMS comunica-se com o agente utilizando a versão apropriada do SNMP.

3.10 Base de Informações de Gerenciamento – MIB

A MIB é uma coleção estruturada de objetos gerenciados. Objetos gerenciados representam os recursos sujeitos ao gerenciamento. Cada nodo do sistema de gerenciamento mantém uma MIB que reflete o estado dos recursos gerenciados naquele nodo. Uma entidade de gerenciamento pode monitorar os recursos de um nodo, lendo os valores dos objetos na MIB e pode controlar os recursos de um nodo, modificando estes valores [ROSE 88a].

A informação de gerenciamento é representada de acordo com um subconjunto da linguagem ASN.1, que é especificada para a definição de tipos não agregados na SMI. Também, para efeitos de simplicidade, o SNMP utiliza um subconjunto das regras básicas de codificação ASN.1. Todas as codificações utilizam a forma de tamanho definido. Além disso, quando permitido, são usadas codificações de não construtores, preferencialmente às codificações de construtores. [ROSE 88a].

Os nomes para todos os tipos de objetos contidos na MIB, são definidos explicitamente na MIB padrão Internet ou em outros documentos que seguem as convenções de nomeação definidas na SMI. A SMI requer que todos os protocolos de gerenciamento definam mecanismos para identificar instâncias individuais dos tipos de objetos de um elemento de rede particular [ROSE 88a].

Cada instância de tipo de objeto definido na MIB é identificada nas operações SNMP por um nome único chamado *nome de variável*. Geralmente, o nome de uma variável SNMP é um OBJECT IDENTIFIER da forma x.y, onde x é o nome de um tipo de objeto não agregado definido na MIB e y é um fragmento de OBJECT IDENTIFIER que, de uma forma específica para o tipo de objeto nomeado, identifica a instância desejada.

Esta estratégia de nomeação permite a exploração completa da semântica da PDU GetNext Request, porque ela atribui nomes para variáveis relacionadas, em uma ordem lexicográfica contínua.

A nomeação de tipos específicos de algumas instâncias de objetos, para algumas classes de tipos de objetos, é definida a seguir. Instâncias de um tipo de objeto, para as quais nenhuma das seguintes convenções de nomeação são aplicáveis, são nomeadas por um OBJECT IDENTIFIER da forma x.0, onde x é o nome do tipo de objeto na definição da MIB [ROSE 88a].

Suponha-se, por exemplo, que se deseje identificar uma instância da variável sysDescr. A classe de objeto para sysDescr é:

Isso	Org	Dod	Internet	Mgmt	Mib	system	sysDescr
1	3	6	1	2	1	1	1

Neste caso, o tipo de objeto x deve ser 1.3.6.1.2.1.1.1, para o qual deve ser conectado um sub-identificador 0, isto é, 1.3.6.1.2.1.1.1.0 identifica uma e somente uma instância de sysDescr.

A figura 3.8 mostra a árvore de registro utilizada para nomeação de objetos definidos na MIB. A sub-árvore MGMT contém a definição das bases de informação de gerenciamento que foram aprovadas pelo IAB. Atualmente existem duas versões da MIB: mib-1 e mib-2. A mib-2 é uma extensão da primeira. As duas possuem o mesmo identificador na sub-árvore porque apenas uma das duas estará presente em qualquer configuração.

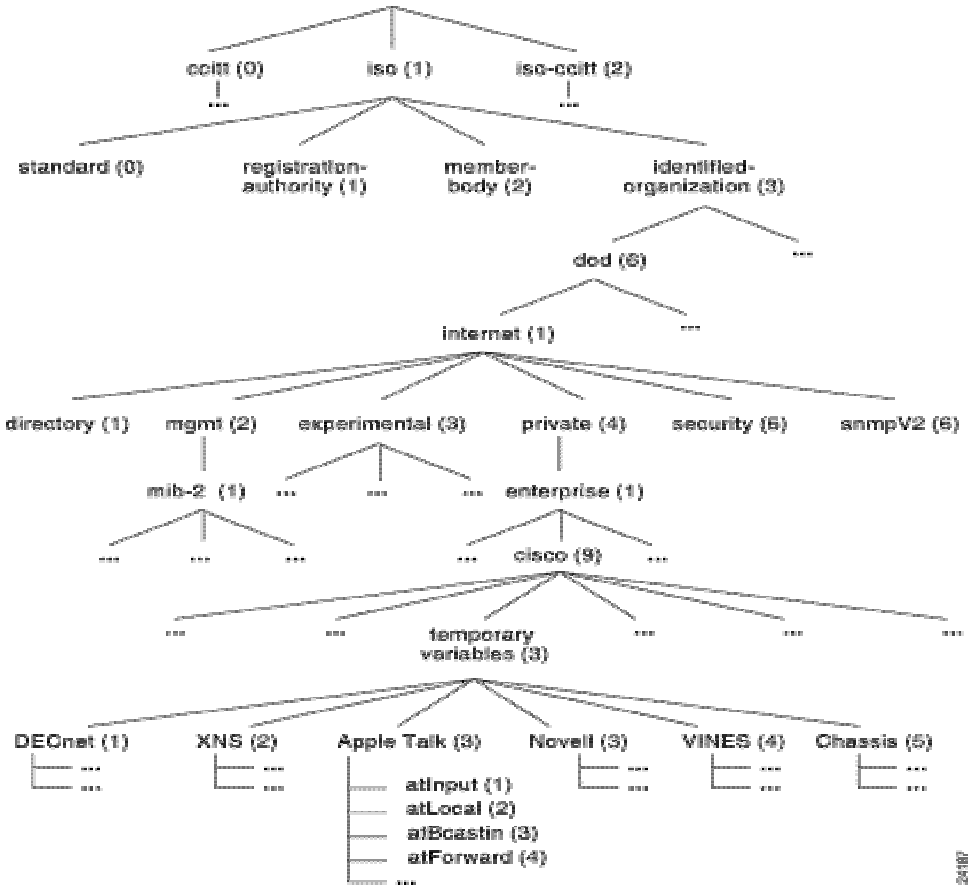

FIGURA 3.8: Árvore de Registro de Tipos de Objetos

Os objetos da mib-2 são subdivididos nos seguintes grupos [CASE 96f]:

- *system*: informações gerais sobre o sistema;

- *interfaces*: informações sobre cada uma das interfaces do sistema para a sub-rede;

- *at (address translation; deprecated)*: descreve a tabela de translação de endereços para mapeamento de endereços internet para endereços de sub-rede;

- *ip*: informação relativa a experiências de implementação e execução do protocolo IP no sistema;

- *icmp*: informação relativa a experiência de implementação e execução do protocolo ICMP no sistema;

- *tcp*: informação relativa a experiência de implementação e execução do protocolo TCP no sistema;

- *udp*: informação relativa a experiência de implementação e execução do protocolo UDP no sistema;

- *egp*: informação relativa a experiência de implementação e execução do protocolo EGP no sistema;

- *transmission*: fornece informações sobre esquemas de implementação e protocolos de acesso em cada interface do sistema;

- *snmp*: informação relativa a experiências de implementação e execução do protocolo SNMP no sistema.

A organização em grupos é conveniente porque os objetos são organizados de acordo com as funções das entidades gerenciadas e também porque ele oferece um guia para os implementadores de agentes, no sentido de identificar quais objetos devem ser implementados. Se a semântica de um grupo for aplicável para uma determinada implementação, então todos os objetos do grupo devem ser implementados. Por exemplo, uma implementação deve incluir todos os objetos do grupo tcp se e somente se ela implementa o protocolo TCP; portanto, uma ponte ou um roteador não necessita implementar os objetos do grupo TCP. Uma exceção a esta regra é o grupo de translação de endereços (at). A fig. 3.9 ilustra a estrutura do grupo system.

3.11 Conclusões

O procolo SNMP possui uma arquitetura bastante simples e com vulnerabilidades quanto à segurança. Trata-se de um protocolo centralizado, onde comumente é adotada

apenas uma estação de gerenciamento utilizada basicamente para monitoração de rede através da técnica de polling.

Como mencionado na introdução deste trabalho, quanto mais funcionalidades automatizadas, maior é o nível de um sistema de gerenciamento, não dependendo completamente do administrador da rede para tomar decisões. Consequentemente, o aplicativo necessitará de uma infra-estrutura mais robusta para o seu funcionamento com níveis de desempenho aceitáveis.

Para não sobrecarregar uma máquina, a solução proposta consiste em dividir o sistema de gerenciamento em módulos de gerência onde cada um tenha incumbências específicas. Estes módulos, estando em máquinas diferentes, proporcionam uma melhoria no desempenho das atividades de gerenciamento. No entanto, a distribuição dos módulos de gerenciamento em várias máquinas acarreta um novo problema a ser resolvido: a entrega dos traps.

O SMNP não possui nenhuma ferramenta nativa que filtre os Traps, determinando destinatários ou descartando alarmes caracterizados desnecessários pelo administrador. No ambiente OSI, existe uma ferramenta capaz de filtrar alarmes sendo detalhadamente descrita no capítulo a seguir.

CAPÍTULO IV

4. Discriminadores de Repasse de Eventos

4.1 Introdução

Este capítulo apresenta uma funcionalidade utilizada em ambientes OSI, o Discriminador de Repasse de Eventos. Embasado na norma X-734 [itu 93], o capítulo traz o escopo da norma, definindo os componentes da arquitetura destes objetos, as necessidades e os objetivos desta ferramenta.

4.2 Conceitos de Discriminadores de Repasse de Eventos

Como mencionado na introdução deste trabalho, o objeto Discriminador de Repasse de Eventos é inexistente em ambientes que utilizam SNMP como protocolo de gerenciamento. O conceito de discriminadores de repasse de eventos é definido e utilizado por ambientes com gerência OSI, sendo minuciosamente conceituado na norma X-734, datada de 1993 [ITU 93].

4.3 Escopo da Norma

Esta proposta/padronização internacional define uma função de gerência de sistemas a qual deve ser utilizada por uma aplicação em um ambiente de gerência centralizado ou não, para interagir com os propósitos de um sistema de gerenciamento, como definido pelo CCITT Rec. X.700/ISO 7498-4. Esta proposta define a função de repasse de eventos, contendo além dos serviços, duas unidades funcionais. Esta norma tem como objetivos [ITU 93]:

- Estabelecer necessidades do usuário para a função de repasse de eventos;

- Estabelecer modelos que relatem os serviços providos pela função para requisitos de usuário;

- Definir os serviços que a função disponibiliza;

- Especificar o protocolo necessário para oferecer tal serviço;

- Definir o relacionamento entre os serviços e as operações e notificações da SMI;

- Definir relacionamentos com outras funções de gerenciamento;

- Especificar requisitos para adaptação.

Estão fora dos objetivos desta norma:

- Definir a natureza de qualquer implementação destinada a oferecer função de repasse de eventos;

- Especificar a maneira pela qual o gerenciamento é consumado pelo usuário da função de repasse de eventos;

- Definir a natureza de qualquer interação a qual resulte no uso da função de repasse de eventos;

- Especificar os serviços necessários para o estabelecimento, comunicado normal ou não de uma associação de gerenciamento;

- Especificar os requisitos para autorização da utilização da função de repasse de eventos para qualquer atividade associada;

- Definir os objetos gerenciáveis destinados ao gerenciamento de protocolos proprietários.

4.4 Conceitos Fundamentais

Para facilitar a compreensão do assunto, serão conceituadas algumas entidades envolvidas no processo da função de gerenciamento de relatório de evento [ITU 93]:

- **Discriminador -** Um objeto de suporte a gerência que permite um sistema selecionar operações de gerenciamento e repasse de relatórios de eventos, para outros objetos gerenciáveis.

- **Discriminador de repasse de eventos -** Um discriminador que age de acordo com repasse de relatório de eventos potenciais.

- **Relatório de evento potencial -** Um relatório de evento potencial é uma notificação, emitida por um objeto, que será submetida a um discriminador. Esta notificação pode ou não virar um relatório de evento (depende se ela satisfaz ou não o critério de discriminação para ser repassada para algum gerente).

- **Função de gerenciamento de repasse de eventos -** Uma função, incluindo a definição de uma classe de objetos de suporte à gerência, que possibilita um gerente controlar a transmissão de relatórios de eventos de objetos gerenciáveis independente da definição de objetos gerenciáveis.

4.5 Necessidades

As necessidades a serem satisfeitas são [ITU 93]:

a) A definição de um serviço de controle de relatório de eventos flexível que possibilite um sistema a selecionar qual(is) relatório(s) de evento(s) devem ser enviados para um sistema de gerência particular;

c) A especificação dos destinatários ao(s) qual(is) os relatórios devam ser encaminhados;

c) A especificação de um mecanismo para controlar o repasse de relatórios de eventos, por exemplo, suspendendo ou reiniciando seu encaminhamento;

d) A possibilidade de um sistema externo de gerência modificar as condições utilizadas no relatório de eventos;

e) A possibilidade da designação de uma localização de backup os quais os relatórios de evento possam ser enviados se a localização primária não estiver acessível.

4.6 Modelo para a função de gerenciamento de relatórios de eventos

As necessidades funcionais anteriormente citadas, referem-se ao comportamento dos sistemas, podendo ser reduzidas a uma necessidade básica no comportamento de um sistema. Esta é a possibilidade de especificar condições a serem satisfeitas por um relatório de eventos potencial emitido por um objeto gerenciável particular para destinatários especificados [ITU 93].

4.7 Modelo de gerenciamento de relatórios de eventos

O modelo de gerenciamento de relatórios de eventos descreve os componentes conceituais que fornecem os relatórios de eventos para relatórios de eventos remotos e processamento local de relatórios de eventos potenciais. O modelo ainda descreve o controle de mensagens, mensagens de relatórios de eventos e mensagens de retorno.

A função conceitual de pré-processamento de eventos recebe notificações locais e cria um relatório de evento potencial. De forma conceptiva, estes relatórios de eventos potenciais são distribuídos para todos os discriminadores de repasse de eventos contidos no sistema aberto local. Um relatório de evento potencial é visto como um objeto de entrada de um discriminador somente com o propósito de ser discriminado pelo discriminador de repasse de eventos, não sendo visível para os demais componentes do sistema.

O discriminador de repasse de eventos é utilizado para determinar quais relatórios de eventos devem ser encaminhados para um destinatário particular durante um período de tempo especificado. Ele ainda deve ser utilizado para especificar o modo (confirmado ou não-confirmado) que o relatório deve ser encaminhado. Cada discriminador de repasse de evento deve conter uma capacidade de sincronização determinando a duração dos intervalos para os quais os relatórios de evento serão selecionados para repasse. Cada discriminador contém um *discriminator construct,* o qual especifica as condições que um relatório de evento potencial deve satisfazer para ser encaminhado. Os relatórios de eventos que tenham sido selecionados são encaminhados para seu(s) destino(s) assim que possível.

Um discriminador de repasse de eventos é por si só um objeto gerenciável podendo com isso emitir notificações. Estas notificações são processadas como relatório de evento potencial por todos os discriminadores de repasse de evento, incluindo o que gerou a notificação.

A fig. 4.1 mostra uma representação sistemática dos componentes envolvidos na geração, processamento e encaminhamento dos relatórios.

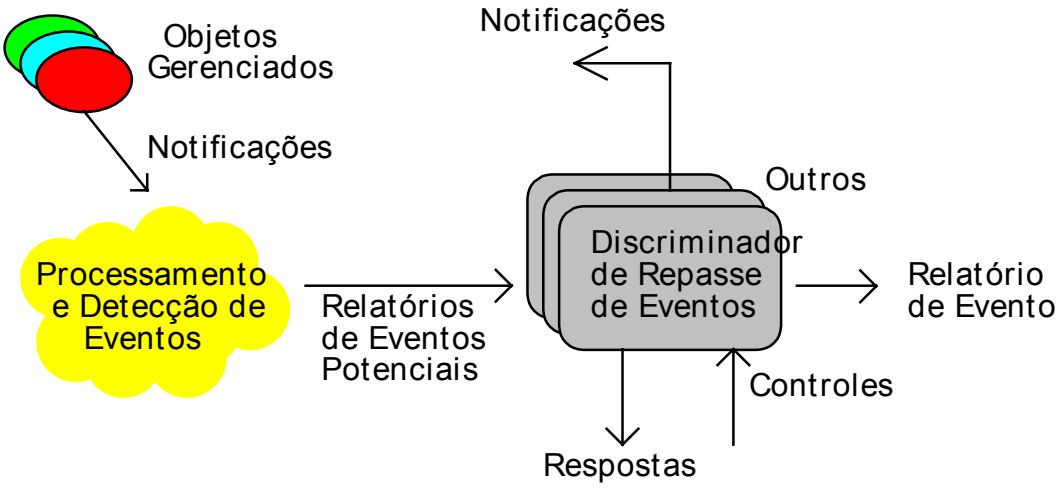

FIGURA 4.1 - Modelo do gerenciamento de relatórios de evento

4.8 Função de gerenciamento de relatório de eventos

Permite a um sistema aberto estabelecer e controlar o discriminador e o encaminhamento de relatórios de evento para outros sistemas abertos. Os relatórios de eventos são gerados como resultado de uma notificação da ocorrência de um evento como, por exemplo, a violação de uma porta ou a mudança do status de uma configuração. A função de gerenciamento de relatórios de evento dá a capacidade de identificação dos destinos para os quais relatórios de eventos selecionados irão ser encaminhados. O gerente de relatórios de evento provê os meios pelos quais, a discriminação e o encaminhamento, podem ser inicializadas, finalizadas, suspensas ou reinicializadas, e através do qual os atributos dos discriminadores de repasse de eventos podem ser lidos e modificados.

Um discriminador de repasse de eventos possui dois estados operacionais, bloqueado (*locked*) e desbloqueado (*unlocked*); então se um discriminador encontrar-se no estado operacional desbloqueado (*unlocked)*, o sistema aberto relatado encaminha relatórios de evento para o destino especificado.

O gerenciamento de relatórios de evento compreende o seguinte [ITU 93]:

a) Inicialização de encaminhamento de eventos;

b) Término de encaminhamento de eventos;

c) Suspensão de encaminhamento de eventos;

d) Reinício de encaminhamento de eventos;

e) Modificação de condições de encaminhamento de eventos;

f) Restauração de condições de encaminhamento de eventos.

Os principais objetivos da função de gerenciamento de relatórios de evento são [ITU 93]:

- Selecionar os relatórios de eventos que devem ser enviados a um sistema de gerenciamento particular;

- Determinar os destinatários para os quais os relatórios de eventos devem ser enviados;

- Controlar (suspender e retomar) o repasse de relatórios de eventos;

- Possibilitar que um sistema de gerenciamento externo modifique as condições de emissão de relatórios de eventos;

- Designar endereços alternativos

4.9 Atributos do Discriminador de Repasse de Eventos

O discriminador de repasse de eventos, além dos atributos herdados da classe discriminador, possui os seguintes atributos:

- *Destination address*: especifica um grupo de endereços primários;

- *Backup address list*: lista ordenada de endereços a serem usados no caso de falha do endereço primário;

- *Active address*: identifica o endereço da Entidade de Aplicação para a qual os eventos são repassados pelo discriminador.

Outros atributos definidos para o objeto Discriminador são os atributos de estado administrativo, operacional e de utilização e o atributo *discriminator Construct*, que define o critério de discriminação a ser empregado.

O atributo *discriminator Construct* tem como valor um conjunto de uma ou mais asserções sobre a presença de valores de atributos. Estas asserções podem ser agrupadas usando operadores lógicos AND e OR.

Um objeto Discriminador pode ser especificado para testar condições específicas de igualdade ou desigualdade de atributos, a presença de atributos e a ausência de qualquer uma destas condições.

A classe de objeto Discriminador de Repasse de Eventos também pode ser derivada para acomodar condições específicas em cada subclasse particular.

4.10 Pacotes de Programação

Permitem que os discriminadores troquem, automaticamente, suas condições de *reporting-on* e *reporting-off*. São definidos três tipos de Pacotes de Programação:

a) Pacote de Programação Diária (*Daily Scheduling Package*) tem um único atributo: *Intvls*

b) Pacote de Programação Semanal (*Weekly Scheduling Package*) tem os atributos: *StartTime StopTime* e *WeekMask (DaysOfWeek* e *IntvlsOfDay)*

c) Pacote de Programação Externa (*External Schedular Scheduling Package*) tem um único atributo que especifica o nome do MO programador (*SchedularName*).

4.11 Serviços

a) Criação de Relatório de Repasse de Eventos;

b) Eliminação de Relatório de Repasse de Eventos;

c) Modificação de valores de atributos;

d) Suspensão e Retomada de atividade de discriminação.

4.12 Conclusões

O objeto Discriminador de Repasse de Eventos é uma ferramenta importantíssima para a arquitetura OSI. A descentralização do modelo é claramente percebida com a utilização desta ferramenta.

A utilização desta funcionalidade no ambiente SNMP permite a descentralização das aplicações de gerenciamento. No caso do trabalho proposto, o valor do atributo *Discriminator Construct* é a regra definida como critério de discriminação.

Os objetivos a serem obtidos com a inserção de um objeto discriminador no modelo SNMP são:

- Selecionar os relatórios de eventos (traps) que devem ser enviados a um sistema de gerenciamento particular;

- Determinar os destinatários para os quais os Traps devem ser enviados.

A especificação e a implementação do protótipo da solução proposta estão detalhadamente descritas no capítulo seguinte, onde são apresentadas a arquitetura do protótipo e as fases de validação e avaliação.

CAPÍTULO V

5. A solução proposta

5.1 Introdução

Este capítulo apresenta uma descrição geral do modelo de solução proposta e as etapas utilizadas para a implementação do protótipo contendo as funcionalidades de objetos discriminadores, apresentados no capítulo 4. Também mostra sua arquitetura e as considerações sobre o desenvolvimento, justificando a adoção da linguagem Java.

Em seguida são apresentadas as instalações e o completo funcionamento do sistema apresentando os diagramas de estados dos dois principais componentes da arquitetura. De forma detalhada é descrito como foi implementado o processo de discriminação dos *Traps* e, exclusivamente para este trabalho, a metodologia utilizada para determinação do(s) destinatário(s).

Por fim, este capítulo traz as etapas para avaliação e validação da ferramenta com todos os resultados obtidos nas fases de testes a que foi submetido.

5.2 O modelo genérico proposto

A fim de atender à crescente demanda por novas aplicações de gerência, propõe-se um modelo de gerenciamento descentralizado, isto é, a possibilidade de se distribuir as tarefas de gerenciamento por mais de uma estação de gerenciamento. O modelo proposto supõe, portanto, a existência de mais de um gerente na rede e a necessidade de um mecanismo que permita determinar os destinatários para os quais um trap particular deve ser encaminhado.

A segunda questão proposta refere-se ao aumento da ênfase da utilização da técnica de relatório de eventos (*Traps*). Esta proposta requer a existência de um mecanismo que permita, entre outras funcionalidades, a discriminação dos traps, isto é, a tomada de decisão referente ao repasse ou não de um determinado tipo de trap, sob determinadas condições.

O objeto discriminador de repasse de eventos, conforme definido na norma X.734, reúne as características necessárias para o atendimento destas duas questões. O modelo proposto contempla, portanto, a inserção de um objeto discriminador de repasse de eventos no modelo de gerenciamento SNMP. A principal dificuldade em se alcançar este objetivo reside na inserção deste objeto como parte do código de um agente. Dada esta dificuldade, optou-se pela inserção de um ou mais objetos discriminadores interceptando a comunicação entre gerentes e agentes. A vantagem desta decisão é evidente, pois ela evita qualquer modificação no código dos agentes e permite a sua utilização em qualquer ambiente de gerenciamento SNMP, mesmo que já tenha sido instalado, com poucas configurações adicionais. É importante ressaltar que, quanto mais próximo fisicamente dos agentes um discriminador estiver, menor será o tráfego de informações desnecessárias na rede. A decisão da localização do discriminador é, portanto, fortemente dependente da configuração topológica da rede. A figura 5.1 ilustra o modelo genérico proposto.

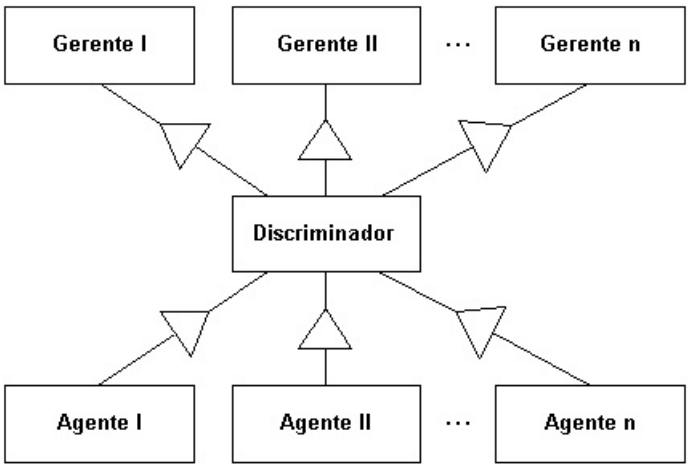

FIGURA 5.1 Modelo genérico de discriminação

Apesar de se propor um modelo de gerência distribuído, a adoção do discriminador mantém a visão de um modelo centralizado para os agentes e, a primeira questão sobre este modelo diz respeito ao desempenho do discriminador. Para resolver este problema, foi adicionado ao sistema um módulo que verifica a eficiência de discriminação obtida, em termos de traps emitidos versus traps processados, conforme será detalhado no capítulo 6. No caso de se verificar situações onde a carga de traps enviados está acima da capacidade de processamento do objeto discriminador, é recomendável a instalação de outro objeto discriminador adicional e a divisão dos agentes em dois grupos. Esta prática, apesar de demandar um esforço extra de configuração dos agentes que devem migrar para o novo discriminador, garante o desempenho e a confiabilidade do sistema de discriminação.

O objeto discriminador de repasse de eventos, proposto neste trabalho, apresenta as seguintes características e funcionalidades:

- Um endereço para o qual os agentes vinculados deverão encaminhar os *Traps*;

- Um critério de discriminação definido através de regras;

- Transparente para os gerentes, isto é, o gerente recebe o *Trap* da mesma forma como receberia do agente que o emitiu, não implicando em nenhuma alteração no código da aplicação de gerenciamento receptora;

- Flexível para a inserção de tantas regras quantas forem necessárias para o processo de discriminação;

- Possibilidade de direcionamento do envio, ou não, de *Traps* para uma quantidade ilimitada de estações de gerenciamento diretamente no discriminador, independente do agente.

Com o objetivo de tornar mais claro o modelo proposto, optou-se pela implementação de um protótipo que permitisse avaliar a viabilidade da proposta, sendo descrito nos itens seguintes deste capítulo.

5.3 Considerações Sobre o Desenvolvimento

Para estudo da viabilidade de utilização de discriminadores de repasse de eventos em ambientes SNMP com mais de uma estação de gerenciamento, foi necessário construir um protótipo que implementasse as funcionalidades requeridas no modelo definido no item 5.2.

Nesta fase do projeto, foi montado um esquema de determinação de destinatários através de regras definidas única e exclusivamente para esta pesquisa. A definição das regras foi feita através da comparação das características da área funcional de gerência de cada módulo gerente com a informação trazida pelo *Trap*, ou seja, de acordo com o conteúdo do *Trap* identifica-se em qual(s) área(s) de gerência ele estaria enquadrado, enviando-o em seguida para o gerente determinado.

A forma de determinação das regras difere de ambiente para ambiente pelo fato de cada um ter suas próprias configurações, equipamentos, arquitetura, etc.

O protótipo foi desenvolvido utilizando a linguagem Java [DACONTA 96] tendo as seguintes justificativas:

- Devido à independência de plataforma, pode ser utilizado em diversos ambientes, do Unix ao Windows, devendo apenas possuir a máquina virtual Java;
- A existência de diversas Api´s abertas SNMP, as quais facilitaram bastante o desenvolvimento deste protótipo;
- Novas classes surgem a todo o momento, sendo disponibilizadas gratuitamente, proporcionando a implementação de aplicações em diversas áreas de IT.
- A possibilidade de diversas melhorias serem implementadas no futuro, entre elas um applet;
- A interface amigável e grande facilidade de implementação.

Para o desenvolvimento dos módulos do sistema, foi utilizada uma API gratuita desenvolvida pela AdventNet [ADVENTNET 00]. Esta API disponibiliza diversas classes para desenvolvimento de operações SNMP, como por exemplo, implementação de GET e SET, construção e manipulação de Traps, etc.

5.3 Arquitetura do Sistema

A fim de facilitar a aplicação do conceito de discriminadores em ambientes SNMP, tal como definido no modelo proposto, fez-se necessário o desenvolvimento de dois módulos com objetivos distintos:

- Discriminador – Responsável pela recepção dos Traps, e aplicação das regras de discriminação com posterior determinação de destinatários (figura 5.2).

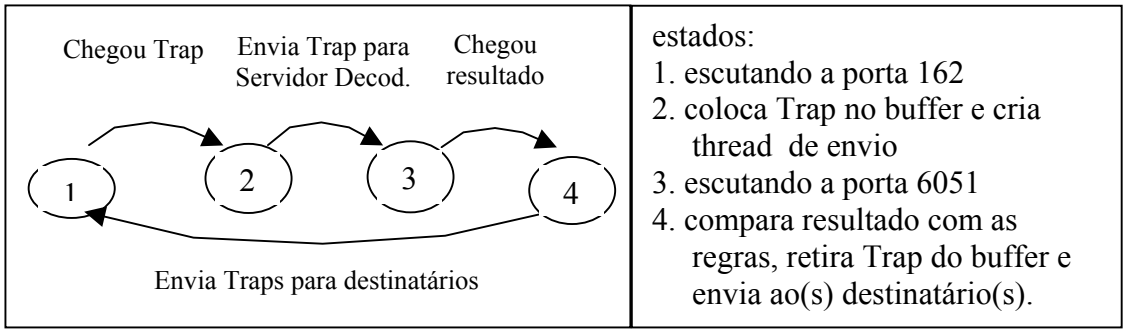

FIGURA 5.2 – diagrama de estados Discriminador

- Servidor Decodificador – Responsável por decodificar a mensagem SNMP devolvendo ao discriminador um pacote contendo o código genérico e específico do Trap.

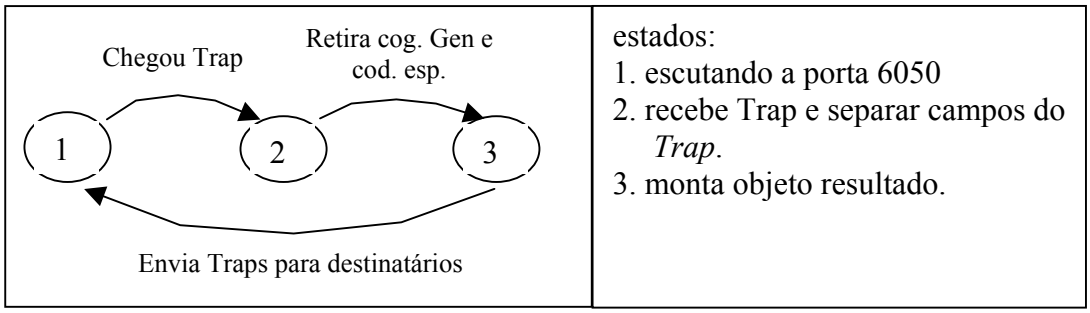

FIGURA 5.3– Diagrama de estados Servidor Decodificador

A figura 5.4 mostra um possível cenário de utilização do objeto discriminador, contemplando os módulos que compõem a arquitetura do sistema proposto.

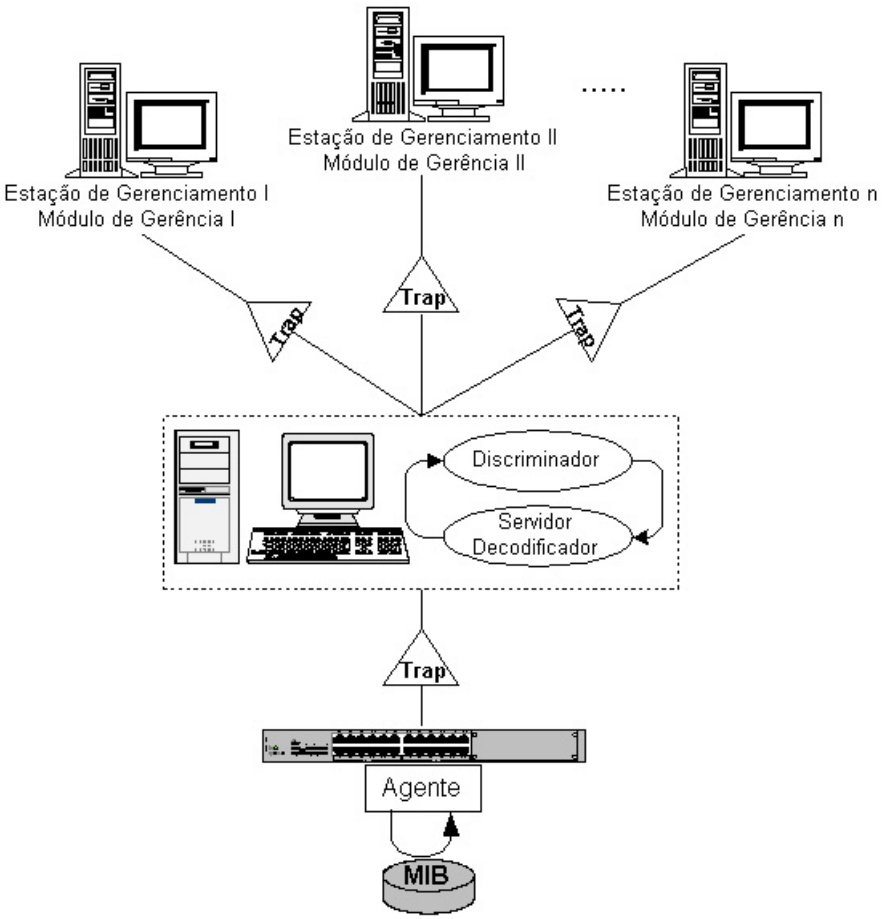

FIGURA 5.4: Arquitetura do Sistema

Para permitir o armazenamento de informações de uma forma mais abrangente, foi utilizado um esquema de *bufferização* dinâmica. A utilização de *buffer* dinâmico é justificada pelo fato de existir a real possibilidade da chegada de diversos Traps ao mesmo tempo, permitindo com isso o livre trabalho das *threads* em alocação e deleção de informações do *buffer,* diferente de um *buffer* estático.

5.4 Funcionamento do Protótipo

A seguir estão descritas, de maneira detalhada, o funcionamento lógico dos módulos componentes do protótipo.

5.4.1 Módulo Discriminador

Ao ser executado, o Discriminador cria duas *threads*, uma que estabelece conexão com a porta 162 local, aguardando a chegada de *Traps* dos agentes, e outra com a porta 6051, aguardando a chegada do objeto *resultado*, ambas através de *socket*.

Ao detectar a chegada de um *Trap*, primeiramente este é colocado temporariamente em um *buffer* e em seguida é criado uma outra *thread* com a função de enviar uma cópia deste *Trap* para o servidor decodificador através de um *socket* para a porta 6050 deste último, onde após o envio, a *thread* é finalizada.

Ao chegar o objeto *resultado* na porta 6051, uma *thread* é então criada para determinação do(s) destinatário(s) do *Trap* de acordo com regras de discriminação. Determinado o(s) destinatário(s), esta mesma *thread* envia o *Trap* para ele(s), finalizando-se ao final. As *threads* com conexão na porta 162 e 6051 são apenas finalizadas ao encerrar o Discriminador, as demais são criadas ao detectarem a chegada do *Trap* e do objeto *resultado*, encerrando-se ao final de suas tarefas específicas.

5.4.2 Módulo Servidor Decodificador

Ao ser iniciado, o Servidor Decodificador cria uma *thread* a qual estabelece uma conexão com a porta 6050 local aguardando a chegada de um *Trap*. Detectada a chegada deste, é então criada uma outra *thread* que tem o objetivo de decodificar a mensagem SNMP. Este processo se dá através da classe *SNMPEncode* a qual implementa as regras de codificação/decodificação BER e ASN.1. Esta classe e disponível através da API utilizada [ADVENTNET 00].

Após a mensagem ser decodificada é então dividida em “pedaços”, através da classe *parser*. O valor de cada “pedaço” do Trap é atribuído a objetos especificamente criados para serem posteriormente lidos através dos métodos descritos na tabela 5.1.

TABELA 5.1: Objetos e métodos extraidos de um *Trap.*

Objeto	Descrição	Método para leitura
Enterprise	Tipo do objeto gerenciado que gerou o *Trap*	GetEnterprise( )
AgentAddress	Endereço do objeto gerenciado que gerou o *Trap*	GetAgentAdress( )
GenericTrapType	Código genérico para o *Trap* gerado	GetGenericType( )
SpecificTrapCode	Código específico para o *Trap* gerado	GetSpecificCode( )
TimeStamp	A quantidade de tempo decorrente entre a última reinicialização da rede até a geração do *Trap*	GetTimeStamp( )
VarBinding	Dados da PDU Trap *SNMPv1.*	GetVarBinding( )

Extraído o código genérico e específico através dos métodos GetGenericType( ) e GetSpecificCode( ), monta-se um objeto denominado *resultado*. Este objeto possui dois campos no formato de *string* contendo o valor do código genérico e especifico do *Trap.*

Código Genérico	Código Específico

FIGURA 5.5: formato do objeto resultado

Depois de montado o objeto de resultado, este é então enviado ao módulo Discriminador através de um socket para a porta 6051 do Discriminador.

Depois de enviar o objeto resultado para o módulo Discriminador, a *thread* é finalizada. A *thread* com conexão à porta 6050 encerra-se apenas ao finalizar o módulo Servidor Decodificador, as demais são criadas ao detectar a chegada de novos *Traps* encerrando-se ao final de suas tarefas específicas.

5.4.3 Discriminação

A discriminação de um Trap é realizada através da implementação de regras de discriminação, que trabalham comparando os valores dos códigos genérico e específico do Trap, fornecidos pelo módulo Discriminador, com as regras definidas pelos administradores do ambiente.

Como exemplo pode-se destacar:

Se o código genérico = 1 (*Link-UP*) e código específico = 0 (porque não se trata de um Trap do tipo *Enterprise Specific*) então enviar Trap para gerente de Falhas e Desempenho.

Exclusivamente para esta pesquisa, a tabela 5.1 mostra as definições para formulação das regras de discriminação.

TABELA 5.2: Relação *Traps*/Destinatários

Trap	Resultado	Área Funcional	Ação/ Enviar para:
ColdStart	0/0	Config.	Descartar
WarmStart	1/0	Config.	Descartar
LinkDown	2/0	Desemp. Falhas	Ger.Desemp. Ger. Falhas
LinkUp	3/0	Desemp. Falhas	Ger.Desemp. Ger. Falhas
AuthenticaTion-Failure	4/0	Segurança	Ger. Segur.
EgpNeighBorLoss	5/0	Falhas Desemp.	Ger.Desemp. Ger. Falhas
EnterpriseSpecific	6/0		
Syslog	6/1	Segur.	Ger. Segur.

Como já mencionado anteriormente, as definições da tabela 5.1 foram adotadas única e exclusivamente para testar a solução proposta, pois considerou-se que cada ambiente possui suas próprias particularidades, caracterizadas pela sua configuração, arquitetura, políticas e componentes.

A fase seguinte à implementação do protótipo foi a execução de testes para verificação das funcionalidades, desempenho do sistema proposto e para a validação geral do sistema. Esta atividade e os resultados obtidos são descritos no capítulo 6.

CAPÍTULO VI

6. Testes e Validação do Protótipo

Para validação do protótipo, os testes foram realizados em duas etapas, sendo que cada etapa possui duas fases de testes em situações distintas. As etapas estão detalhadas na tabela 6.1

TABELA 6.1: Etapas de testes e suas fases

<table>
<tr><th colspan="2">ETAPA I</th></tr>
<tr><td colspan="2">Local: LISA: Laboratório de Tratamento de Incerteza e Sistemas Adaptativos - UFSC
Dispositivo: Roteador Cisco 7500 acesso dial up
Equipamento: Intel Pentium 233 MMX, 64 Mb de RAM, WNT 4.0 Server.</td></tr>
<tr><th colspan="2">ETAPA II</th></tr>
<tr><td colspan="2">Local: Casan – Gerência de Informática GIN
Dispositivo: 81 roteadores Cisco 2500,1 Cisco 7500
21 swtches catalyst 1900, 1 catalyst 5000
Equipamento: Estação Sun, processador Risc, 128 Mb de RAM, Solaris 7.</td></tr>
<tr><th>FASE I</th><th>FASE II</th></tr>
<tr><td>Gerentes, Discriminador e Serv. Decod. no mesmo micro.</td><td>Gerentes em micros diferentes; discriminador e Serv. Deocd. no mesmo micro.</td></tr>
</table>

Para verificar se a eficácia do funcionamento do protótipo foi alcançada, considerou-se o caso em que a fórmula abaixo fosse verdadeira, dentro de um período de tempo determinado.

$$\sum A = \sum B = \sum C$$

onde: $\sum A$ = número de Traps Enviados pelo(s) Equipamento(s); $\sum B$ = número de Traps Recebidos pelo(s) Discriminador(s); $\sum C$ = número de Traps Discriminados.

A determinação da quantidade de *Traps* originados/recebidos é feita da seguinte forma:

- Antes de iniciar os testes, primeiramente é zerada a variável da MIB do equipamento gerenciado a qual informa o total de Traps enviados por este;
- Após este procedimento, iniciam-se os testes com os valores nulos (*Traps* enviados/recebidos);
- Ao final da fase de testes constata-se o valor total dos Traps enviados pelo equipamento gerenciado em seu IOS e os Traps recebidos pelo discriminador através de um contador implementado neste;
- Confrontam-se os valores para emissão dos resultados da fase de testes.

6.1 Etapa I

Esta etapa teve duração de aproximadamente 3 meses, onde os horários de testes foram variados não seguindo um padrão devido a constante utilização dos equipamentos por estudantes vinculados ao laboratório. Sendo assim, totalizando os dias e horários utilizados para testes possibilitou-se calcular as médias de utilização mostradas na tabela 5.2.

TABELA 6.2: Resultados Apurados na Etapa I

	FASE I	**FASE II**
Média Tempo decorrido	12	12
Média Traps recebidos	300	300
Média Traps Discriminador	300	300
Porcentagem de Acerto	100	100

6.2 Etapa II

Nesta etapa de testes houve a possibilidade de apuração exata do tempo decorrido de testes, 12 horas em 2 dias. Os testes iniciaram às 7 horas da manhã, antes dos funcionários chegarem, e finalizaram às 19 horas, quando todos os funcionários já haviam saído. Os resultados obtidos estão demonstrados na tabela 5.3.

TABELA 6.3: Resultados Apurados na Etapa II

	FASE I	**FASE II**
Média Tempo decorrido	12	12
Média Traps recebidos	1024	983
Média Traps Discriminador	1024	983
Porcentagem de Acerto	100	100

CAPÍTULO VII

7. Conclusões

Este trabalho propôs a inclusão de objetos discriminadores de repasse de eventos na arquitetura de gerenciamento SNMP. O objetivo principal desta inclusão foi possibilitar a existência de mais de um gerente na rede, evitando que os traps emitidos pelos agentes fossem encaminhados para todos os gerentes indiscriminadamente e sem a necessidade de alteração no código nativo dos agentes.

A principal vantagem apresentada por este modelo é a possibilidade de se distribuir a carga de gerenciamento entre diversos gerentes, aumentar a ênfase do modelo original na utilização de traps, sem sobrecarga nos gerentes e evitando a transmissão de traps para destinatários que não irão processá-los.

A implementação de um protótipo permitiu a validação do modelo proposto e a identificação de várias funcionalidades adicionais que podem ser consideradas quando da transformação do protótipo em um produto a ser implantado em um ambiente real.

As fases de testes possibilitaram avaliar e validar o protótipo, constatando-se sua completa aplicabilidade e utilidade para ambientes SNMP.

Nem todos os tipos de *Traps* foram originados pelos equipamentos gerenciados devido a não constatação de situações que exigissem o envio destes. Os *Traps* genéricos detectados foram: *coldStart, warmStart, linkUp, linkDown, authenticationFailure* e *enterpriseSpecific*. Dos *Traps* específicos foram detectados apenas: *sysLog* e *ISDN*.

A não detecção de todos os tipos de *Traps* genéricos não implica em limitação de discriminação dos *Traps* recebidos já que foram implementadas regras de discriminação para todos os tipos de *Traps* genéricos. Devido a grande quantidade de *Traps* específicos desconhecidos, este protótipo limitou-se apenas aqueles conhecidos pelo

autor; entretanto o protótipo está aberto à inclusão de tantas regras quantas forem necessárias, aumentando a gama de *Traps* específicos suportados.

Como se trata de um protótipo, não foi levado em consideração o desempenho na busca das regras para determinação de destinatários implicando em uma limitação considerável devido ao método seqüencial de busca.

7.1. Considerações Finais

Este protótipo demonstrou ser uma ferramenta bastante útil no suporte à administração de redes IP complexas, com mais de uma estação de gerenciamento e diversos dispositivos de conectividade. A filtragem de Traps e o encaminhamento às estações de gerenciamento específicas são uma maneira bastante eficaz para solução de problemas isolados, contribuindo também para diminuição do tráfego de pacotes na rede e diminuição do *overhead* de processamento nas estações de gerenciamento.

Nenhuma incompatibilidade foi identificada nos ambientes e situações de testes. Conseguiram-se excelentes resultados nas discriminações e encaminhamentos dos Traps chegando a uma taxa de 100% de acerto.

Foi constatada que a busca seqüencial, implementada no protótipo, à base de regras é bastante limitada, pois, à medida que a base for crescendo o tempo de busca aumenta proporcionalmente causando diminuição de desempenho do aplicativo.

Para que o protótipo possa ser transformado em um produto comercial, é necessário que esta deficiência seja superada com técnicas de busca mais elaboradas. Um exemplo de solução para este problema é proposto como trabalho futuro no item a seguir.

7.2 Trabalhos Futuros

No decorrer das fases desta pesquisa, foi possível identificar algumas implementações que implicariam em melhorias ao protótipo.

- Atribuição de prioridades aos Traps (baixa, média, alta, por exemplo). Alguns relatórios de eventos, com maior nível de importância, seriam encaminhados de

forma prioritária em relação aqueles com níveis inferiores, com isso, problemas mais críticos seriam tratados mais rapidamente;

- Implementação de um esquema de *Log/Accounting* a fim de totalizar a quantidade de *Traps* recebidos/encaminhados, gerando estatísticas diárias e emissão de relatórios;
- Implementação de interfaces gráficas, possibilitando a configuração do sistema, inclusão de novas regras de discriminação, geração de relatórios, visualização de *logs*, etc;

- Utilização de técnicas de Inteligência Artificial para desenvolvimento de uma base de conhecimento e um motor de inferência para uma busca eficaz, dando a possibilidade de um grande número de regras, não acarretando em perda de desempenho.

Apesar de apresentar várias deficiências em termos de ergonomia de interface e desempenho no processo de busca de regras, o protótipo pode ser utilizado sem apresentar nenhum problema, em sistemas onde o número de regras não seja tão elevado a ponto de comprometer o desempenho do processo de discriminação. Para sistemas que exijam um processo de discriminação mais complexo, as técnicas utilizadas para a implementação do protótipo devem ser revistas e adequadas.

CAPÍTULO VIII

8. Referências Bibliográficas

[ADVENTNET 00] ADVENTNET, the Internet Management Infrastructure Company. 1996 – 2001. http://www.adventnet.com

[BERNHARDT 96] BERNHARDT, M. "Design and implementation of a wed-based tool for ATM connection management". Stuttgart, Aug. 1996. Master´s Thesis – Departament of Computer Science, University of Stuttgart.

[BRISA 93] BRISA. Gerenciamento de Redes – Uma abordagem de Sistemas Abertos, São Paulo: MAKRON Books do Brasil Editora Ltda., 1993.

[CASE 90] Case,j.; Fedor, M.; Schoffstall, M.; Davin, J.; Simple Network Management Protocol, RFC 1157, maio de 1990.

[CASE 96a] Case,j.; McCloghrie, K.; Rose, M.; Waldbusser, S. Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2), RFC 1902, janeiro de 1996.

[CASE 96b] Case,j.; McCloghrie, K.; Rose, M.; Waldbusser, S. Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2), RFC 1903, janeiro de 1996.

[CASE 96c] Case,j.; McCloghrie, K.; Rose, M.; Waldbusser, S. Conformance Statements for Version 2 of the Simple Netowork Management Protocol (SNMPv2), RFC 1904, janeiro de 1996.

[CASE 96d] Case,j.; McCloghrie, K.; Rose, M.; Waldbusser, S. Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2), RFC 1905, janeiro de 1996.

[CASE 96e] Case,j.; McCloghrie, K.; Rose, M.; Waldbusser, S. Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2), RFC 1906, janeiro de 1996.

[CASE 96f] Case,j.; McCloghrie, K.; Rose, M.; Waldbusser, S. Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2), RFC 1907, janeiro de 1996.

[CASE 96g] Case,j.; McCloghrie, K.; Rose, M.; Waldbusser, S. Coexistence between Version 1 and 2 of the Internet Standard Network Management Framework, RFC 1908, janeiro de 1996.

[CISCO 00] CISCO. URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr/fun_r/frprt3/frd3001.htm, agosto de 2000.

[CISCO 00a] CISCO. URL: http://www.cisco.com/univercd/cc/td/doc/product/lan/cat1700/c1700/c17inbnd.htm, setembro de 2000.

[CISCO 99] CISCO. URL: http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/snmp.htm, janeiro de 1999.

[DACONTA 96] DACONTA, M. “Java for C++ programmers”. USA: Ellist, 1996.

[HARNEDY 97] Harnedy, S. Total SNMP: Exploring the Simple Network Management Protocol. Second Edition. New Jersey: Prentice Hall PRT, 1997.

[ITU 93] CCITT; Event Report Management Function, X0734, 1993.

[MEIRELLES 97] Meirelles, Luiz. Uma Proposta para o Gerenciamento de Aplicações em Rede. Dissertação de Mestrado. UFSC. Florianópolis – SC. 1997.

[ROSE 88] Rose, M.; McCloghrie, K.; Structure and Identification of Management Information for TCP/IP-based internets, RFC1065, agosto de 1988.

[ROSE 88a] Rose, M.; McCloghrie, K.; Management Information Base for Network Management of TCP/IP-based internets, RFC1066, agosto de 1988.

[ROSE 95] Rose, M.; McCloghrie, K. How to Manage your Network Using SNMP: The Networking Management Practicum. New Jersey: Prentice Hall PTR, 1995.

[SOLSTICE 96] Sun Microsystems, Inc. Solstice Administration Guide: Site/SunNet/Domain Manager. 1996

[STALLINGS 93] Stallings, William. SNMP, SNMPv2 and CMIP. The Pratical Guide to Network-Management Standards. Reading, Mass: Addison-Wesley, 1993.

[STALLINGS 96] Stallings, W. SNMP, SNMPv2 and RMON: Pratical Network Management. Second Edition. United States of America: Addison Wesley, Inc., 1996.

[TIVOLI 00] IBM. URL: http://www.tivoli.com, dezembro de 2000.

[TRAPCONSOLE 99] CS Software. URL: http://www.cscare.com/TrapConsole/, setembro de 1999.

[VERONEZ 00] Veronez, C. Gerência de Desempenho do Tráfego em Redes Utilizando Baseline Bayesiana. Dissertação de Mestrado. UFSC. Florianópolis – SC. 2000.

[WEBBER 97] Webber, Celso. Uma MIB para Aplicações Internet. Dissertação de Mestrado. UFSC. Florianópolis – SC. 1997.

Anexo 1

Em ASN.1 as PDUs são descritas como segue:

```
RFC1157-SNMP DEFINITIONS ::= BEGIN

    IMPORTS
        ObjectName, ObjectSyntax, NetworkAddress, IpAddress, TimeTicks
            FROM RFC1155-SMI;

    -- top-level message

        Message ::=
            SEQUENCE {
                version       -- version-1 for this RFC
                    INTEGER {
                        version-1(0)
                    },

                community     -- community name
                        OCTET STRING,

                data          -- e.g., PDUs if trivial
                    ANY       -- authentication is being used
            }

    -- protocol data units

            PDUs ::=
                CHOICE {
                    get-request
                        GetRequest-PDU,
```

```
                    get-next-request
                        GetNextRequest-PDU,

                    get-response
                        GetResponse-PDU,

                    set-request
                        SetRequest-PDU,

                    trap
                        Trap-PDU
                    }

        -- the individual PDUs and commonly used
        -- data types will be defined later

        END
```

GetRequest-PDU

```
GetRequest-PDU ::=
            [0]
               IMPLICIT SEQUENCE {
                  request-id
                     RequestID,

                  error-status       -- always 0
                     ErrorStatus,

                  error-index        -- always 0
                     ErrorIndex,
```

```
        variable-bindings
            VarBindList
    }
```

GetNextRequest-PDU

```
GetNextRequest-PDU ::=
        [1]
            IMPLICIT SEQUENCE {
                request-id
                    RequestID,
                error-status        -- always 0
                    ErrorStatus,

                error-index         -- always 0
                    ErrorIndex,

                variable-bindings
                    VarBindList
            }
```

GetResponse-PDU

```
GetResponse-PDU ::=
        [2]
            IMPLICIT SEQUENCE {
                request-id
                    RequestID,

                error-status
                    ErrorStatus,
```

```
        error-index
            ErrorIndex,

        variable-bindings
            VarBindList
    }
```

SetRequest-PDU

```
SetRequest-PDU ::=
        [3]
            IMPLICIT SEQUENCE {
        request-id
            RequestID,

        error-status        -- always 0
            ErrorStatus,

        error-index         -- always 0
            ErrorIndex,
        variable-bindings
            VarBindList
    }
```

Trap-PDU

```
Trap-PDU ::=
    [4]

        IMPLICIT SEQUENCE {
          enterprise          -- type of object generating
                              -- trap, see sysObjectID in [5]
```

```
    OBJECT IDENTIFIER,

  agent-addr          -- address of object generating
    NetworkAddress, -- trap

  generic-trap        -- generic trap type
    INTEGER {
      coldStart(0),
      warmStart(1),
      linkDown(2),
      linkUp(3),
      authenticationFailure(4),
      egpNeighborLoss(5),
      enterpriseSpecific(6)
    },

  specific-trap     -- specific code, present even
    INTEGER,      -- if generic-trap is not
                  -- enterpriseSpecific

  time-stamp        -- time elapsed between the last
   TimeTicks,     -- (re)initialization of the network
                  -- entity and the generation of the
                     trap

  variable-bindings   -- "interesting" information
     VarBindList
}
```