AVALIAÇÃO DO DESEMPENHO DE AGENTES MÓVEIS NO GERENCIAMENTO DE REDES

Marcelo Gonçalves Rubinstein

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Otto Carlos Muniz Bandeira Duarte, Dr.Ing.

Prof. Julius Cesar Barreto Leite, Ph.D.

Prof. Carlos Becker Westphall, Dr.

Prof. José Ferreira de Rezende, Dr.

Prof. Jorge Lopes de Souza Leão, Dr.Ing.

RIO DE JANEIRO, RJ - BRASIL

MARÇO DE 2001

RUBINSTEIN, MARCELO GONÇALVES

Avaliação do Desempenho de Agentes Móveis no Gerenciamento de Redes [Rio de Janeiro] 2001

XIV, 87 p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia Elétrica, 2001)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Agentes Móveis
2. Gerenciamento de Redes
3. Avaliação de Desempenho

I. COPPE/UFRJ II. Título (série)

Agradecimentos

Aos meus pais Irineu e Luiza, pelo amor e apoio em todas as minhas decisões. Ao meu irmão Maurício e ao meu tio Samuel, pelo incentivo. À minha esposa Cláudia, pelo amor e apoio durante toda a tese.

Ao Prof. Otto Carlos Muniz Bandeira Duarte, por seu incentivo, orientação, reconhecimento e amizade. Ao Prof. Guy Pujolle, por ter me recebido muito bem na França. Ao Prof. José Ferreira de Rezende, pela amizade, ajuda e paciência na tentativa de resolução de problemas de simulação. Aos Profs. Carlos Becker Westphall e Edmundo Madeira, pelas sugestões dadas no Exame de Qualificação. Aos Profs. Julius Leite e Leão, pela presença na banca e contribuição à tese.

Aos meus amigos Flávio, Renata e Paulo, pela ajuda na solução de diversos problemas surgidos durante a simulação. Ao Pedro Braconnot Velloso, pela contribuição na realização do protótipo do agente móvel. Ao "Baiano" e ao Luís Henrique, pela utilização de suas máquinas durante a fase de testes na França.

Aos Profs. Richard, Mauros, Aloysio, Roosevelt e Gelson; a Marcos, Vidal, Belém, Artur, Aline, Kleber, Saulo, Valentim, Gardel, Fagundes, Marcial, Marcio, Alexandre, Granato, Eric, Bernardo, Rafael, Bicudo, Juliana, Bia, Solange, Evelyn, Wilson, Fidel e Wanderley, pela amizade. A Nadjib, Yacine, Khaldoun, Laurent, Thamer, André-Luc, Lila, Mourad, Khaled, Faten, Salima, Guillaume, Mauro, Anelise, Roberto, Larissa, Adolfo, Christine, Pascale, Luciano, Patrick, Christelle e Gil, pelo convívio na França. A Isabela, Carlos, Isabel, Carlos Eduardo, António, Pillar, Alexandra, Paula, Paulo, Spíndola, Rogério, Ildemar, José Eduardo, "Luri", "Ziza", Carlos, Daniela, Ana Luiza, Alessandro, Andrea, Victor, Mauricio, Murilo, Mariana, "Long", Tilara, Fernando, Edmond e Adet, pelos momentos de descontração.

Ao PEE/COPPE, pelas instalações e equipamentos utilizados. Ao CNPq, pelo financiamento da pesquisa no Brasil e na França.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

AVALIAÇÃO DO DESEMPENHO DE AGENTES MÓVEIS NO GERENCIAMENTO DE REDES

Marcelo Gonçalves Rubinstein

Março/2001

Orientador: Otto Carlos Muniz Bandeira Duarte

Programa: Engenharia Elétrica

Este trabalho analisa o desempenho de agentes móveis no gerenciamento de redes, comparando-o com o do modelo cliente-servidor do SNMP (*Simple Network Management Protocol*). As medidas de desempenho são obtidas de implementações e de simulações. Protótipos de uma aplicação que obtém variáveis da MIB-II (*Management Information Base-II*) foram realizados e testados em uma rede local. A partir da obtenção de parâmetros relativos ao gerenciamento de redes e à infra-estrutura de agentes utilizada, novos resultados foram obtidos para topologias semelhantes à da *Internet*, com um grande número de nós. A análise dos resultados do tempo de resposta indica que o agente móvel possui um desempenho melhor do que o do SNMP quando o número de elementos gerenciados se encontra entre dois limites, o inferior e o superior, que são determinados, respectivamente, pelo número de mensagens que passam pelo *backbone* e pelo tamanho do agente que cresce com as variáveis recolhidas em cada elemento gerenciado. Os resultados também mostram que uma melhora significativa do desempenho é obtida quando o agente móvel retorna ou envia seus dados à estação de gerenciamento, após visitar um número fixo de nós.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

PERFORMANCE EVALUATION OF MOBILE AGENTS IN NETWORK MANAGEMENT

Marcelo Gonçalves Rubinstein

March/2001

Advisor: Otto Carlos Muniz Bandeira Duarte

Department: Electrical Engineering

This work analyzes mobile agent performance in network management, comparing it with the client-server model used by the SNMP (Simple Network Management Protocol). Performance measurements are obtained from implementation and simulation. Prototypes of an application that gathers MIB-II (Management Information Base-II) variables were created and tested on a LAN. By obtaining parameters from the network management and from the agent infrastructure, new results were obtained on large topologies similar in shape to the Internet. Response time results show that the mobile agent performs better than the SNMP when the number of managed elements ranges between two limits, an inferior and a superior one, respectively determined by the number of messages that pass through a backbone and the size of mobile agent that grows with the variables collected on network elements. The results also show that a significant improvement is achieved when the mobile agent returns or sends data to the management station after visiting a fixed number of nodes.

Lista de Acrônimos

API :	interface de programação de aplicação (*Application Program Interface*);
ASN.1:	*Abstract Syntax Notation One*;
ATM :	*Asynchronous Transfer Mode*;
CMIP:	*Common Management Information Protocol*;
CORBA:	*Common Object Request Broker Architecture*;
FIPA:	*Foundation for Intelligent Physical Agents*;
JDK:	*Java Development Kit*;
IETF:	*Internet Engineering Task Force*;
ISO :	*International Organization for Standardization*;
MASIF :	*Mobile Agent System Interoperability Facility*;
MIB :	*Management Information Base*;
MSS :	tamanho máximo de segmentação (*Maximum Segmentation Size*);
ns :	*network simulator*;
OMG :	*Object Management Group*;
OSI :	*Open Systems Interconnection*;
PDU :	unidade de dados do protocolo (*Protocol Data Unit*);
RMI :	invocação remota de método (*Remote Method Invocation*);
RMON:	monitoração remota (*Remote MONitoring*);
RPC :	chamada remota de procedimentos (*Remote Procedure Call)*;
SMI :	*Structure and identification of Management Information*;
SNMP :	*Simple Network Management Protocol*;
TCP :	*Transmission Control Protocol*;
TMN :	*Telecommunication Management Network*;

UDP : *User Datagram Protocol*;

URL : *Universal Resource Locator*;

WAN : rede de longa distância (*Wide Area Network*).

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

COM o extraordinário desenvolvimento da *Internet*, o volume de informações disponíveis pelo mundo, o custo de movimentação de dados e a pequena experiência em computação de muitos usuários têm sido motivo de preocupação para a comunidade de redes e para os desenvolvedores de aplicativos. O volume de informações disponíveis em diferentes sítios por todo o mundo não tem sido acompanhado pela capacidade de um usuário de procurar, selecionar e obter essas informações. Além disso, a movimentação de grandes volumes de dados através da rede possui um custo significativo e o número de usuários com pouco conhecimento em computação tem crescido e deverá continuar a aumentar rapidamente.

Uma possível solução para estes problemas consiste na utilização de agentes móveis, que são programas que ajudam um usuário a realizar tarefas na rede, agindo em interesse deste usuário. Esses agentes podem mover-se para o lugar onde os dados estão armazenados e selecionar as informações que o usuário necessita; economizando-se banda passante, tempo e dinheiro.

O modelo cliente-servidor tem sido utilizado há anos em sistemas distribuídos. Na maioria dos casos, o cliente e o servidor são executados em máquinas diferentes (Figura 1.1), por conseqüência, aumentando os esforços e os custos de desenvolvimento e de manutenção de clientes em diversas plataformas. A chamada remota de procedimentos (*Remote Procedure Call* - RPC) é o mecanismo mais utilizado para a

comunicação entre clientes e servidores. O modelo de programação remota é utilizado pelos agentes a fim de melhorar o desempenho através da eliminação do tráfego intermediário na rede. Neste modelo, um agente representando um cliente pode ser transportado para um hospedeiro junto com o procedimento a ser executado a fim de interagir com o servidor de agentes deste hospedeiro, realizando suas tarefas localmente [1] (Figura 1.2). A principal vantagem de se usar o modelo de programação remota no lugar da RPC consiste no fato de que somente resultados selecionados de pedidos serão enviados, diminuindo o tráfego na rede pois não há a necessidade de se transmitir comandos, todos os dados e resultados intermediários.

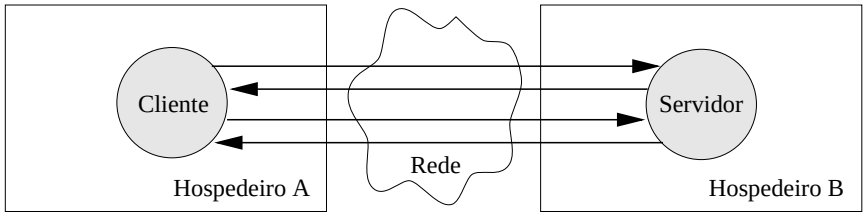

Figura 1.1: O modelo cliente-servidor.

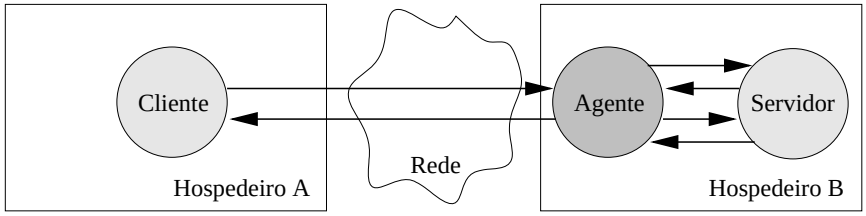

Figura 1.2: O modelo de programação remota.

A mobilidade de agentes conduz a uma extensão do modelo cliente-servidor: os clientes enviam partes deles para o(s) servidor(es) para execução [2]. Além disso, os agentes móveis fazem parte de um conceito mais amplo de migração de código que lida com a capacidade de mover dinamicamente os componentes de uma aplicação distribuída entre os nós de uma rede [3].

1.1 Os Modelos de Códigos Móveis

Supõe-se que um componente computacional A, localizado no $lugar_A$, precisa do resultado da computação de um serviço e que o $lugar_B$ estará envolvido na entrega do serviço. A execução do serviço envolve um conjunto de recursos, o conhecimento do serviço (código de execução) e um componente computacional responsável pela execução do código. Para que o serviço seja realizado, estes elementos devem estar disponíveis em um lugar ao mesmo tempo [4].

Os códigos móveis podem ser classificados em avaliação remota, código sob demanda e agentes móveis [4]. A classificação é feita em função da localização dos diferentes componentes antes e depois da execução do serviço, o componente computacional que é responsável por executar o código e onde a computação ocorre de fato. Na Tabela 1.1, o componente computacional em negrito é aquele que executa o código, o componente sublinhado corresponde àquele que foi movido, enquanto o recurso e o conhecimento são representados por r e c.

Tabela 1.1: Os modelos de códigos.

Modelo	Antes		Depois	
	$Lugar_A$	$Lugar_B$	$Lugar_A$	$Lugar_B$
Cliente-servidor	A	c r B	A	c r **B**
Avaliação remota	c A	r B	A	c r **B**
Código sob demanda	r A	c B	r c **A**	B
Agente móvel	c A	r	-	c r **A**

No modelo cliente-servidor, o servidor (B) oferece um conjunto de serviços. Os recursos (CPU, memória e informações locais) e o conhecimento (código) necessários para a execução do serviço estão no $lugar_B$. O cliente demanda a realização de um serviço através da interação com o servidor. Como resposta, B realiza o serviço executando o código e acessando outros recursos também disponíveis no $lugar_B$. Em geral, o serviço produz algum resultado que será entregue ao cliente através de outra interação.

Na avaliação remota, o componente A tem o conhecimento mas não possui os recursos necessários que estão no $lugar_B$. Então A envia o conhecimento para o componente B que executa o código usando os recursos disponíveis. Os resultados são entregues a A através de uma outra interação.

O paradigma código sob demanda possui um componente que é capaz de acessar os recursos que precisa, no entanto não há no $lugar_A$ a informação de como processar estes recursos. A interage com B para demandar o conhecimento para a realização do serviço e B envia o código para A, que pode então executá-lo.

No paradigma agente móvel, o conhecimento do serviço está em A, o qual está inicialmente no $lugar_A$, mas alguns dos recursos necessários (CPU, memória e informações locais) estão em B. Logo, A migra para o $lugar_B$, carregando o conhecimento (código) e possivelmente alguns resultados intermediários. Após chegar no $lugar_B$, A completa o serviço.

O agente móvel é diferente dos outros modelos de códigos móveis pois envolve a migração de um componente computacional já existente. A avaliação remota e o código sob demanda utilizam a transferência entre componentes, enquanto que no agente móvel um componente computacional inteiro é migrado, junto com o seu estado, código e alguns recursos necessários à realização do serviço.

Os modelos de códigos móveis apresentados anteriormente surgiram da análise das linguagens atuais de códigos móveis, que provêem dois tipos de migração de código [3]. Se uma migração forte é provida por uma linguagem forte de códigos móveis, os agentes codificados na linguagem de códigos móveis podem mover seus códigos e estados para um hospedeiro diferente. O estado de um agente consiste em um estado relativo aos dados do agente, que contém variáveis globais e instanciadas, e em um estado de execução relativo a variáveis locais e parâmetros e aos *threads* de execução [5]. No caso de uma migração fraca ser provida por uma linguagem fraca de códigos móveis, somente o estado relativo aos dados do agente é transferido quando da migração do agente [5]. Conseqüentemente, o programador é responsável por codificar o estado de execução do agente em variáveis do programa.

1.2 O Gerenciamento de Redes

As redes de computadores atuais com um grande número de nós e de equipamentos de diferentes fabricantes não podem mais ser gerenciadas somente através do esforço humano. A complexidade destes sistemas determina o uso de sistemas automatizados de gerenciamento de redes [6].

O gerenciamento de redes é uma das aplicações que podem se beneficiar do uso de agentes móveis, por estar baseado de um modo geral em um paradigma centralizado. Os protocolos de gerenciamento SNMP (*Simple Network Management Protocol*) [7] e CMIP (*Common Management Information Protocol*) [8] baseiam-se no modelo cliente-servidor, no qual o gerente centraliza as informações e fornece a ordem para a execução de ações corretivas e o agente[1] interage com a MIB (*Management Information Base*) e executa as ordens do gerente. Nestes protocolos, as operações disponíveis à estação de gerenciamento para o acesso à MIB são de baixo nível.

O gerenciamento de desempenho é uma das áreas funcionais do modelo de gerenciamento OSI (*Open Systems Interconnection*). Este tipo de gerenciamento se relaciona à disponibilidade de informações de gerenciamento, do modo a poder-se determinar a carga da rede [9]. É necessário o acesso dinâmico a uma quantidade enorme de dados da rede, a qual é obtida através de varredura.

A interação de granularidade fina e a varredura periódica geram um tráfego intenso que sobrecarrega a estação de gerenciamento, resultando em problemas de escalabilidade. Neste sentido, os agentes móveis podem distribuir e escalar o gerenciamento. Os agentes móveis descentralizam o processamento e o controle da estação de gerenciamento e, por conseqüência, reduzem o tráfego ao redor da estação de gerenciamento, tornam assíncrona a comunicação com a estação de gerenciamento (ótimo para enlaces não confiáveis ou com perdas) e aumentam a flexibilidade do comportamento dos agentes de gerenciamento.

[1] Quando não especificado, o termo agente irá se referir neste trabalho aos agentes 'inteligentes/móveis'. Para marcar a diferença, quando necessário, o agente de um sistema de gerenciamento será doravante chamado agente de gerenciamento.

As pesquisas na área de códigos móveis para o gerenciamento de redes são recentes [10]. Diversos trabalhos têm sido publicados, sendo que a maioria trata da implementação de arquiteturas de gerenciamento utilizando agentes móveis.

1.3 Os Trabalhos Relacionados

Magedanz et al. [11, 12] foram os primeiros a propor a utilização de agentes móveis no gerenciamento de redes de computadores e de telecomunicações. Nesses trabalhos, a funcionalidade do agente e do gerente é aumentada através de um ambiente de execução de agentes que habilita a realização de "aplicações móveis de gerenciamento", implementadas por *scripts* de agentes móveis.

Labetoulle et al. [13] apresentam um estado da arte de agentes inteligentes no gerenciamento de redes, do paradigma gerenciamento por delegação [14] ao modelo de agentes móveis. Karmouch e Pham [10] descrevem o uso atual de agentes móveis em telecomunicações e no gerenciamento de redes.

Várias arquiteturas para o gerenciamento de redes utilizando agentes móveis e inteligentes foram propostas na literatura [14, 3, 15, 16, 17, 18, 19, 20, 21].

O gerenciamento por delegação (*Management by Delegation*) [14, 22, 23] foi o primeiro paradigma a visar a descentralização e a automação de tarefas de gerenciamento através da delegação dinâmica de funções de gerenciamento aos agentes. Kooijman [24] estende o gerenciamento por delegação através do uso de agentes responsáveis pelo gerenciamento de uma área, sendo que a execução destes agentes está baseada na ocorrência de eventos específicos. Neuman et al. [25] implementam um protótipo de gerenciamento por delegação utilizando agentes móveis e CORBA (*Common Object Request Broker Architecture*). Oliveira et al. [21] utilizam agentes móveis para prover mobilidade na plataforma Disman do grupo de trabalho *Distributed Management* do IETF.

Alguns autores [3, 15, 26, 27, 16, 28, 29, 30] desenvolvem arquiteturas que utilizam ao mesmo tempo agentes móveis e SNMP, provendo o acesso a agentes legados.

Baldi et al. [3] desenvolvem um protótipo de uma infra-estrutura de gerenciamento com agentes móveis e SNMP. A interação dos agentes móveis com os agentes SNMP ocorre porque nem todas as estações possuem a infra-estrutura necessária para a execução dos agentes móveis; porém os autores citam que o melhor seria utilizar os agentes móveis diretamente [3]. Morin et al. [26] definem uma arquitetura de gerenciamento na qual um servidor e vários gerentes se comunicam com agentes SNMP localizados nos elementos gerenciados e analisam o desempenho dessa arquitetura através de medidas em uma rede *Ethernet* comparando os agentes móveis com o SNMP. Pagurek et al. [16, 31] implementam uma infra-estrutura de mobilidade de código e um conjunto de tarefas simples que interagem com agentes localizados em componentes da rede. Eles também propõem um modelo de rede inteligente no qual o comportamento e o estado da rede fazem parte do modelo, podendo ser atualizados de forma dinâmica [32, 33]. Puliafito et al. [29] utilizam agentes móveis para coletar informações sobre o estado da rede e para realizar um micro-gerenciamento de elementos de rede através de múltiplas variáveis. Silva et al. [30] implementam uma plataforma de agentes móveis para o gerenciamento de redes de telecomunicações e apresentam um protótipo de aplicação de gerenciamento de desempenho para a coleta de dados em TMN. Uma extensa comparação baseada em medidas dessa plataforma com outras é apresentada em [34, 35]. Knight et al. [20] utilizam uma plataforma de agentes móveis voltada para o gerenciamento de redes e apresentam alguns resultados de tempos de resposta de um agente móvel com funcionalidade do aplicativo *ping*.

Vários pesquisadores [17, 36, 37, 38, 39, 18, 16, 40, 41] aplicam agentes inteligentes nas arquiteturas de gerenciamento. Westphall et al. [36] implementam agentes autônomos integrados ao SNMP, com características de flexibilidade e de adaptabilidade para um ambiente de gerenciamento de redes heterogêneas. Protótipos de gerência pró-ativa utilizando sistemas especialistas são implementados por Westphall et al. [42] e Tarouco et al. [37], enquanto Oliveira et al. [38] usam redes neurais para gerenciar de modo pró-ativo redes ATM e Neuman et al. [39] utilizam lógica difusa para realizar um gerenciamento pró-ativo distribuído. A arquitetura inteligente de Ku et al. [18] também suporta agentes móveis e o padrão MIB-II [43], enquanto Ji et

al. [41] descrevem um agente inteligente que processa as informações coletadas por agentes SNMP e utiliza essas informações para detectar anomalias que tipicamente precedem uma falha na rede.

Vários autores [18, 19, 44, 45, 16] utilizam Java [46] como a linguagem de implementação das infra-estruturas de agentes móveis.

CORBA também é utilizada no gerenciamento de redes por diversos pesquisadores [47, 48, 25]. Madeira et al. [48] implementam um modelo de monitoração assíncrono, flexível, genérico e configurável, enquanto que Noemi et al. [49] descrevem uma linguagem interpretada de implementação de aplicações de gerência [50] que, junto com CORBA, flexibiliza o papel do agente de gerenciamento, tornado-o mais dinâmico.

Em relação à avaliação do desempenho dos agentes móveis, vários trabalhos vêm sendo realizados. Straβer et al. [51] descrevem um modelo matemático simples de avaliação do desempenho de agentes móveis, no qual os agentes podem utilizar chamadas remotas de procedimento ou migração. Alguns resultados de implementação também são apresentados. Ismail et al. [52] apresentam uma avaliação de desempenho dos agentes móveis e do modelo cliente-servidor. Vários custos das etapas da migração de um agente móvel são apresentados em uma série de resultados de implementação. Tendo como aplicação o gerenciamento de redes, o desempenho dos agentes móveis tem sido estudado por vários pesquisadores [53, 54, 55, 56, 57, 58, 59, 9, 60, 15]. Geihs et al. [53], El-Darieby e Bieszcad [54] e Outtgarts et al. [55] avaliam quantitativamente de maneira bem simples os agentes móveis e o modelo cliente-servidor. Costa [56] apresenta uma avaliação analítica comparativa do desempenho dos agentes móveis e do SNMP em diferentes topologias simples de rede. Puliafito et al. [57] comparam analiticamente os modelos cliente-servidor, avaliação remota e agentes móveis através de Redes de Petri. Baldi et al. [58, 61] avaliam paradigmas de códigos móveis em aplicações de gerenciamento de redes. Liotta et al. [59, 62] apresentam modelos matemáticos para várias configurações de agentes móveis, nos quais são analisados o número e a localização inicial dos agentes, que gerenciam redes hierárquicas como as redes de telecomunicações.

Bohoris et al. [9, 63] apresentam uma comparação de desempenho entre agentes móveis, CORBA e JAVA-RMI, utilizando um elemento em uma rede ATM. Um conjunto de objetos (dados fictícios) é utilizado para obter-se o tempo de resposta e a utilização da banda passante. Uma plataforma otimizada de agentes e outra avaliação do desempenho em uma *Ethernet* de 100 Mbps são apresentadas em [64]. Gavalas et al. [60, 65] analisam a utilização da banda passante para o SNMP e para o agente móvel. Resultados experimentais de implementação são apresentados em termos de consumo de banda passante e tempo de resposta para a obtenção de uma agregação de múltiplas variáveis em uma rede local de alguns nós. Duas outras aplicações utilizando agentes móveis, a aquisição de atômica de tabelas SNMP e a obtenção de objetos de tabelas SNMP que satisfaçam determinados critérios, são propostas em [66, 67]. Sahai e Morin [15, 68] realizam medidas de tempos de resposta e de utilização de banda passante de agentes móveis e de aplicações cliente-servidor em uma rede *Ethernet* de alguns nós.

1.4 O Direcionamento do Trabalho

Nenhum dos trabalhos descritos anteriormente trata do problema da escalabilidade dos agentes móveis no gerenciamento de uma rede complexa de muitos nós, com uma topologia semelhante à da *Internet*. Por isso, neste trabalho compara-se a escalabilidade dos agentes móveis versus agentes SNMP no gerenciamento de redes, através da análise dos resultados de implementações [69] e de simulações [70, 71, 72, 73, 74, 75, 76]. Dois protótipos de uma aplicação que obtém variáveis da MIB-II, um com agentes móveis e outro somente utilizando o SNMP, são implementados e testados em uma rede local *Ethernet*. A partir da obtenção de parâmetros relativos ao gerenciamento e à infra-estrutura de agentes, novos resultados são obtidos para topologias semelhantes à da *Internet*. Além disso, várias análises são realizadas de modo a avaliar os efeitos de diversas características dos agentes móveis.

Este trabalho está organizado da seguinte forma. No Capítulo 2, conceitos bá-

sicos dos agentes são apresentados. No Capítulo 3 são descritos os principais sistemas de gerenciamento de redes utilizados atualmente, assim como características do gerenciamento de redes com agentes móveis. No Capítulo 4 são apresentados os protótipos implementados e os resultados das medições. No Capítulo 5 são apresentados os resultados das simulações. Por último, no Capítulo 6 são apresentadas as conclusões e as sugestões de trabalhos futuros.

Capítulo 2

Os Agentes Móveis

NESTE capítulo, os conceitos básicos relativos aos agentes móveis são apresentados. As principais características dos agentes são descritas na Seção 2.1. Na Seção 2.2, a estrutura dos agentes móveis é apresentada. Na Seção 2.3, é apresentado um ambiente de desenvolvimento de sistemas baseados em agentes móveis que provê serviços genéricos de migração, comunicação, execução, segurança, tolerância a falhas e localização de serviço.

2.1 As Características de Agentes

Os agentes são encarregados de realizar tarefas e podem operar sobre requisitos fornecidos. Para executar essas tarefas, os agentes podem utilizar as seguintes capacidades: autonomia, inteligência e mobilidade [77].

O grau de autonomia e autoridade investida no agente é medido através da interatividade do agente. Existem três níveis: interatividade com o usuário, interatividade com a aplicação e interatividade com o agente, classificados de acordo com o grau de interatividade permitido, da baixa (usuário) à alta interatividade (agentes). Na interatividade com a aplicação, é permitida a interatividade com o usuário e com a aplicação, enquanto que na interatividade com o agente, há interação entre o usuário, a aplicação e outros agentes.

A inteligência é o grau de raciocínio e comportamento aprendido: a habilidade do agente em aceitar as metas do usuário e realizar a tarefa a ele conferida. Existem quatro níveis de inteligência: preferência, raciocínio, planejamento e aprendizado. No mínimo, deve existir alguma declaração de preferências, talvez na forma de regras, com uma máquina de inferência ou algum outro mecanismo de raciocínio para agir sobre essas preferências. Níveis maiores de inteligência incluem um modelo do usuário ou alguma outra forma de entender e raciocinar a respeito do que o usuário quer que seja feito, podendo o agente planejar os meios de alcançar esse objetivo. Níveis mais altos de inteligência podem ser obtidos através de sistemas que aprendem a se adaptar ao seu ambiente, em termos dos objetivos do usuário e dos recursos disponíveis ao agente.

A mobilidade corresponde ao grau de movimento dos agentes através da rede. Um agente pode ser fixo, residindo na máquina cliente ou instanciado no servidor, ou móvel, criado em uma máquina e enviado para outra, por exemplo para a execução em um ambiente seguro. Um agente móvel pode ser transportado de uma máquina para outra no meio de sua execução e carregar com ele dados acumulados sobre o seu estado.

2.2 A Estrutura dos Agentes Móveis

Os agentes móveis consistem em um código, um estado e em atributos [78]. O código de um agente móvel é um programa que define o comportamento do agente. Obviamente um código de um agente pode ser escrito em qualquer linguagem, porém um agente deve poder ser executado de uma mesma maneira em qualquer hospedeiro para o qual possa se mover [79]. Para isso, as linguagens mais adequadas são as interpretadas diretamente ou as compiladas para uma linguagem intermediária baseada em interpretador que possa ser facilmente transportada e executada sem recompilação. Como exemplos dessas linguagens, têm-se: Tcl, Perl e Java, sendo que o atual interesse em sistemas de agentes móveis é devido quase que inteiramente à grande adoção de Java [80].

Além do código, um agente carrega um estado que permite que suas ações subseqüentes sejam baseadas em suas ações passadas. Com isso, um agente pode retomar o trabalho de onde parou, após mudar para outro hospedeiro. O estado de um agente contém o ponto de execução no novo hospedeiro e as variáveis necessárias para a sua execução que incluem os dados coletados anteriormente.

Os atributos de um agente descrevem o agente, os seus requisitos e o seu histórico para a infra-estrutura. Os principais atributos incluem um identificador único de agente, o dono do agente como um endereço destinatário para resultados intermediários, mensagens de erro ou reclamações quanto a um mau comportamento de um agente, o tempo e o lugar de origem e o histórico do movimento.

2.3 A Infra-estrutura de Agentes

Os agentes devem interagir com os hospedeiros para utilizar os serviços disponíveis nesses hospedeiros e negociar serviços com outros agentes. Os agentes que são móveis, além das capacidades dos agentes fixos, devem ainda poder se mover em redes de computadores. O ambiente que permite essas ações é denominado infra-estrutura de agentes e a entidade que provê suporte para os agentes em um hospedeiro particular é chamada servidor de agentes.

A criação de uma arquitetura de agentes de grande aceitação facilita um grande número de serviços e aplicações em redes [81]. Uma infra-estrutura de agentes visa prover serviços genéricos, tais como: migração, comunicação, execução, segurança e tolerância a falhas, permitindo a um desenvolvedor de aplicações baseadas em agentes focar-se nos detalhes das aplicações, sem se preocupar com a infra-estrutura. Além disso, funcionalidades específicas de cada sistema podem ser adicionadas de acordo com a necessidade [82].

Um extraordinário aumento nas atividades de pesquisa relacionadas a agentes móveis surgiu no início dos anos 90. Um grande número de fabricantes estavam envolvidos com o desenvolvimento de diversas plataformas, construídas sobre diferentes sistemas operacionais e baseadas em diferentes linguagens e tecnologias [83],

tais como: o TACOMA (*Tromsφ And COrnell Moving Agents*) [84], o Agent Tcl [85], o Odyssey [86] (antigo Telescript [87]) e o Aglets [88]. Nos últimos anos, esforços de padronização começaram a ser realizados pela FIPA (*Foundation for Intelligent Physical Agents*) [89], pela Agent Society [90] e pela OMG (*Object Management Group*) [91]. A FIPA foi criada para produzir especificações de tecnologias genéricas de agentes. Para a produção desses padrões, a FIPA utiliza os seus membros e os seus colaboradores da área de agentes para criar especificações que podem ser utilizadas para alcançar uma interoperabilidade entre sistemas baseados em agentes desenvolvidos por diferentes companhias e organizações [92]. Várias plataformas de agentes seguem essa especificação [93], tais como a ZEUS [94], a JADE [95] e a FIPA-OS [96]. A Agent Society visa facilitar a colaboração e a transferência de tecnologia entre desenvolvedores de sistemas de agentes, assim como incentiva a interoperabilidade entre padrões abertos de agentes. A OMG está trabalhando de modo a estabelecer padrões industriais para a tecnologia de agentes móveis e possibilitar a interoperabilidade entre sistemas de agentes. Magedanz et al. [82] também trabalham na interoperabilidade de sistemas de agentes móveis (*Mobile Agent System Interoperability Facility* - MASIF). A especificação MASIF visa um ambiente de objetos móveis distribuídos e unificados que permita interações transparentes em relação à tecnologia e à localização entre objetos estáticos e móveis.

A maioria das infra-estruturas existentes, embora diferindo consideravelmente na realização, utiliza a mesma solução básica para portabilidade e segurança: não permite que os agentes sejam executados na máquina real de processador, memória e sistema operacional. Ao invés disso, os agentes são executados em uma máquina virtual, usualmente através de um interpretador e de um ambiente em tempo de execução, os quais escondem os detalhes da arquitetura do sistema hospedeiro e confinam as ações dos agentes a um ambiente restrito [97].

A base do modelo de arquitetura utilizado em uma infra-estrutura de agentes (Figura 2.1) é o servidor de agentes [78], que está encarregado de prover diversas funções como o transporte de agentes, a segurança dos agentes e dos hospedeiros, as comunicações de agentes com o hospedeiro, com outros agentes e com os seus donos, e o armazenamento persistente. Um ambiente apropriado para a execução de um

agente, denominado ambiente em tempo de execução, é a interface entre o agente e o seu hospedeiro e disponibiliza recursos ao agente de modo controlado. Suas principais funções são proteger o hospedeiro de agentes, capturar o estado de um agente e interagir com o servidor. Para cada agente sendo executado em um servidor, existe um ambiente em tempo de execução dedicado, criado pelo servidor. Com isso, um agente não tem acesso direto a outros agentes que estão sendo executados no mesmo hospedeiro. O usuário (dono) de um agente interage com a infra-estrutura através de um cliente.

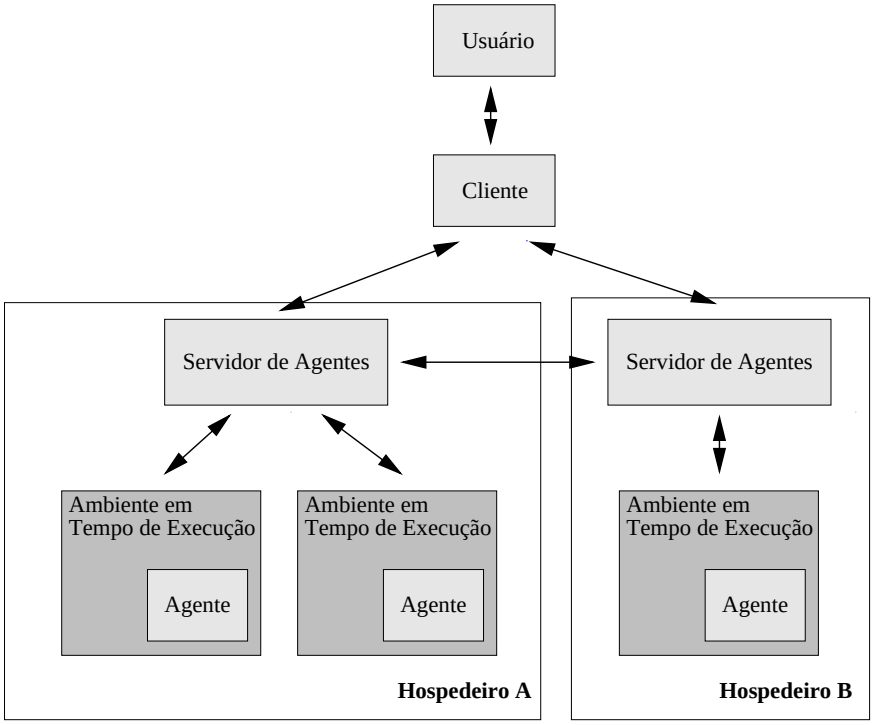

Figura 2.1: O modelo de arquitetura.

O servidor de agentes é um programa (como um servidor de correio eletrônico ou um servidor de FTP) executado em cada computador que será acessível aos agentes e está encarregado dos agentes que estão sendo executados nesse computador. Suas tarefas incluem aceitar agentes, criar o ambiente de execução apropriado, supervisionar as execuções dos agentes, responder a pedidos sobre os estados dos agentes e terminar as execuções dos agentes quando necessário. O servidor de agentes também deve organizar a transferência de agentes para outros hospedeiros, gerenciar as comunicações entre agentes e entre os agentes e os seus donos, autenticar e controlar

o acesso a todas as operações de agentes e recuperar os agentes e as informações carregadas por eles no caso de falhas de computadores e de redes.

As funções do servidor de agentes podem ser divididas em segurança, tolerância a falhas, comunicação, transferência e localização de serviço.

Talvez o tópico mais importante para a completa aceitação de sistemas baseados em agentes seja a segurança. Em um sistema de redes de computadores, nem o agente nem as máquinas são "confiáveis" [98]. Muitas pessoas associam agentes móveis a ameaças impostas por programas maliciosos como o vírus e os cavalos de Tróia, pois um agente pode atacar uma máquina e obter acesso a recursos locais [98]. Deve-se evitar que um servidor de agentes adultere os agentes a seu cargo ou mesmo inspecione o código de um agente, que por exemplo pode implementar uma estratégia de negociação de um competidor.

Mecanismos de recuperação de agentes e das informações que carregam são utilizados no caso de falhas de computadores e de redes, de modo que a execução possa voltar ao normal o mais rápido possível [99]. Persistência é utilizada, armazenando os agentes, os seus dados e os seus estados de tempos em tempos em memórias não voláteis. Esse armazenamento baseia-se em pontos de verificação, que normalmente são pontos críticos na execução de um agente. Quando um desses pontos é alcançado, as informações descritas anteriormente são salvas na memória não volátil.

A maioria dos sistemas de agentes permite que os agentes móveis somente acessem recursos disponibilizados através da infra-estrutura. Com isso, há a necessidade de comunicação entre um agente móvel e um agente fixo que geralmente pode acessar quaisquer tipos de recursos. Além dessa comunicação entre agentes, os agentes devem poder se comunicar com o sistema hospedeiro e com os seus donos.

Uma das características principais de agentes móveis é seu poder de se mover de um hospedeiro para outro. A diferença entre esse tipo de mobilidade e a encontrada em, por exemplo, sistemas de balanceamento de carga que permitem a realocação de processos é que o movimento acontece de acordo com um pedido explícito do agente, ao invés de ocorrer por decisão do sistema operacional (sem o agente no-

tar) [79]. Isto é, a linguagem de implementação do agente inclui uma sentença ou chamada de biblioteca da forma `MOVA-PARA hospedeiro`. Quando essa chamada é executada, o agente é suspenso, codificado para a transmissão, transmitido para o novo hospedeiro, decodificado e instalado para retomar a execução nesse novo local, preservando-se o estado e a integridade do agente.

A funcionalidade de localização de serviço ajuda um agente a realizar a sua tarefa, apresentando a ele algum hospedeiro que provê o serviço especificado [100]. Com isso, não existe a necessidade do usuário saber o nome ou a URL (*Universal Resource Locator*) do hospedeiro. Em cada hospedeiro, existe uma lista dos hospedeiros localizadores de serviço. Um agente móvel migra para um desses hospedeiros específicos e se comunica com um agente estacionário, o agente descobridor, que provê o serviço de localização de serviço. O agente usa palavras-chave e parâmetros específicos para descrever o serviço especificado. Os serviços são divulgados através de um registro junto a um agente fixo. Uma lista de todas as palavras-chave e os parâmetros específicos que descrevem os serviços são utilizados para a divulgação.

Capítulo 3

Os Sistemas de Gerenciamento de Redes

NESTE capítulo são descritos os principais sistemas de gerenciamento de redes utilizados atualmente. Na Seção 3.1, são apresentados os sistemas de gerenciamento centralizado de redes e na Seção 3.2 são descritas as principais modificações feitas nos sistemas de gerenciamento visando a descentralização do processamento e do controle. Por último, as características do gerenciamento de redes com agentes móveis são apresentadas na Seção 3.3.

Basicamente, um sistema de gerenciamento de redes contém quatro tipos de componentes: estações de gerenciamento, agentes de gerenciamento sendo executados em nós gerenciados, protocolos de gerenciamento e informações de gerenciamento [101]. Os agentes de gerenciamento são entidades cujo propósito é prover uma interface padronizada para o acesso a informações sobre o equipamento de rede no qual esses agentes residem [3]. Uma estação de gerenciamento utiliza o protocolo de gerenciamento para comunicar-se com os agentes de gerenciamento. As informações trocadas entre a estação de gerenciamento e os agentes de gerenciamento são armazenadas em uma base de informações de gerenciamento, a MIB (*Management Information Base*).

Os sistemas de gerenciamento de redes são classificados em centralizados e des-

centralizados, de acordo com a localização do processamento e do controle do gerenciamento. No gerenciamento centralizado, o processamento das informações obtidas e o controle das ações de gerenciamento estão concentrados na estação de gerenciamento. Atualmente, a maioria dos sistemas de gerenciamento é centralizada, mas alguns passos visando a descentralização já foram tomados [102].

3.1 O Gerenciamento Centralizado

Os protocolos de gerenciamento SNMP (*Simple Network Management Protocol*) [103, 7, 104] e CMIP (*Common Management Information Protocol*) [8, 105], propostos respectivamente pelo IETF (*Internet Engineering Task Force*) e pela ISO (*International Organization for Standardization*), baseiam-se no modelo cliente-servidor. Nesses protocolos, a estação de gerenciamento age como um cliente que provê uma interface para o gerente da rede e interage com os agentes de gerenciamento, que são servidores que gerenciam o acesso remoto às suas informações locais que são as variáveis da MIB (Figura 3.1).

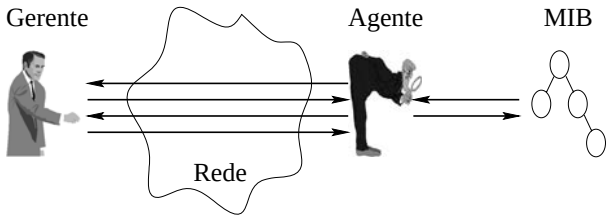

Figura 3.1: Os componentes do gerenciamento via SNMP.

As funções e bases de dados do SNMP são definidas através de três especificações:

- *Structure and Identification of Management Information for TCP/IP-based Networks* (SMI): descreve como os objetos gerenciados contidos na MIB são definidos;
- *Management Information Base for Network Management of TCP/IP-based Internet*: descreve os objetos gerenciados contidos na MIB;

- *Simple Network Management Protocol*: define o protocolo utilizado para gerenciar esses objetos.

A SMI [106] descreve as estruturas comuns e o esquema de identificação de informações de gerenciamento para redes TCP/IP. Descrições formais das estruturas são feitas através da *Abstract Syntax Notation One* (ASN.1).

Todos os objetos da MIB do SNMP estão dispostos em uma estrutura em árvore. Os objetos folha da árvore são os objetos gerenciados reais, cada qual representando algum recurso, atividade ou informação relacionada que deva ser gerenciada. A estrutura em árvore define um agrupamento de objetos em grupos logicamente relacionados.

Esses objetos são definidos segundo um determinado tipo (*object type*), utilizando-se a ASN.1. Os tipos de objetos possuem os seguintes campos:

- objeto: nome textual, denominado descritor de objeto, para o tipo de objeto. Esse nome corresponde a um identificador de objeto. Por exemplo, um dos objetos da MIB-II chama-se udpInDatagrams;
- sintaxe: o tipo de dado que modela o objeto. A sintaxe pode ser simples, como um inteiro, uma *string* de octetos, um identificador de objetos ou nulo, ou composta, caso utilize tipos básicos (para criar uma seqüência) ou uma sintaxe de aplicação (por exemplo: o tipo contador);
- acesso: identifica que tipo de operações pode ser feito pelo gerente sobre o objeto: leitura, escrita, leitura e escrita e não acessível;
- status: informa se a implementação do objeto é obrigatória, opcional ou obsoleta;
- descrição: texto da semântica de um tipo de objeto.

O identificador de objeto corresponde a uma seqüência de inteiros separados por pontos que identificam os ramos da estrutura em árvore (Figura 3.2), de modo similar ao sistema de arquivos do Unix ou ao DNS.

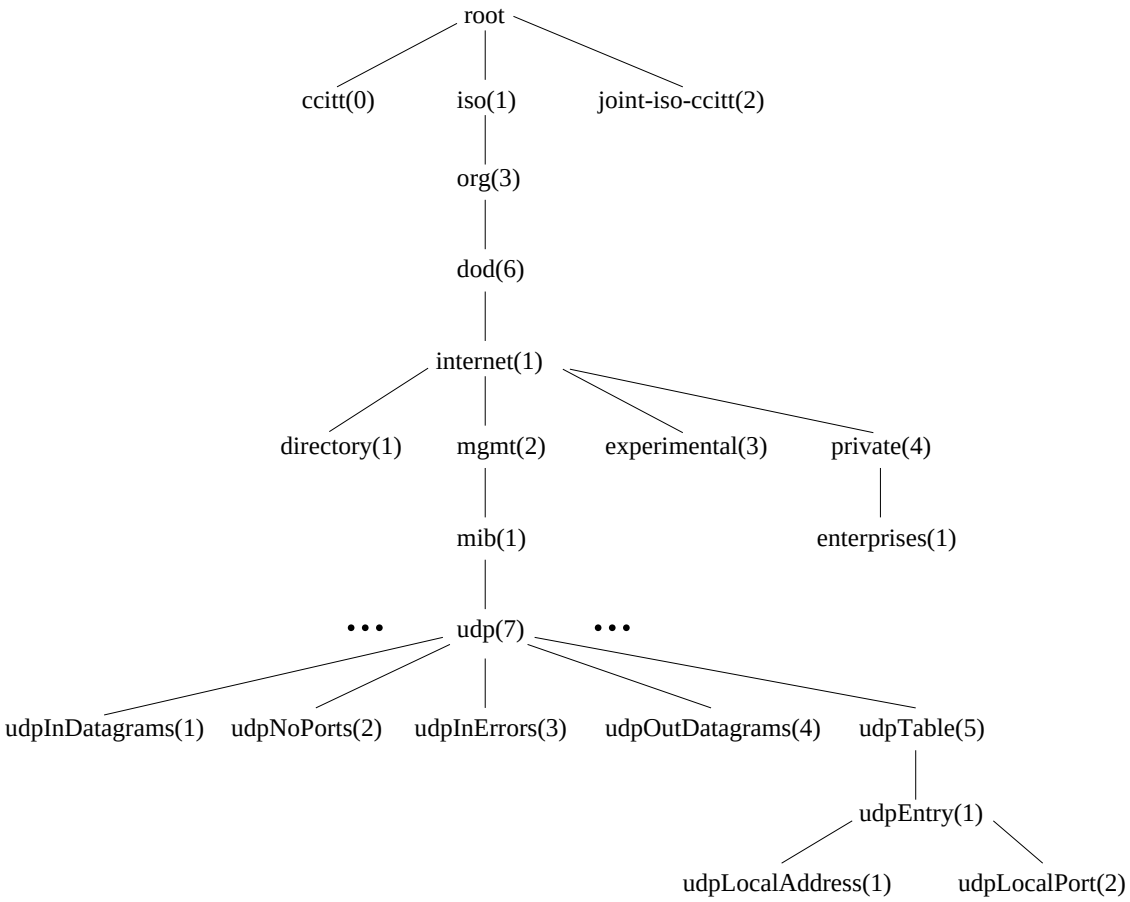

Figura 3.2: Os identificadores de objetos do grupo UDP na MIB-II.

Para o SNMP, existem duas MIBs padrões: a MIB-I [107] e a MIB-II [43]. A MIB-II é uma extensão da MIB-I, com grupos e objetos adicionais, e é utilizada atualmente com o SNMP. Os objetos gerenciáveis da MIB-II estão subdivididos em dez grupos com funcionalidades diferentes: *system*, *interfaces*, *at*, *ip*, *icmp*, *tcp*, *udp*, *egp*, *transmission* e *snmp*. Por exemplo, o grupo udp contém informações relativas à implementação e à operação do UDP em um nó. Esse grupo possui algumas variáveis e uma única tabela (Figura 3.2). A variável *udpInDatagrams* (*1.3.6.1.2.1.7.1.0*) corresponde ao número de datagramas UDP entregues a processos de usuários. A forma numérica desse objeto termina com zero, pois o objeto é a única instância existente. A definição do objeto *udpInDatagrams* da MIB-II é apresentada na Tabela 3.1.

O protocolo de gerenciamento SNMP [103] define cinco tipos de mensagens que são trocadas entre o gerente e o agente (Figura 3.3):

Tabela 3.1: O objeto udpInDatagrams da MIB-II.

```
udpInDatagrams OBJECT-TYPE
    SYNTAX  Counter
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
            "The total number of UDP datagrams delivered to
            UDP users."
    ::= { udp 1 }
```

- GetRequest: pede o valor de uma ou mais variáveis;
- GetNextRequest: pede o valor da próxima variável na ordem lexicográfica após uma ou mais variáveis especificadas. Esse operador é muito utilizado na busca em tabelas;
- SetRequest: atualiza o valor de uma ou mais variáveis;
- GetResponse: retorna o valor de uma ou mais variáveis;
- Trap: notifica sobre eventos assíncronos.

As PDUs do protocolo são construídas utilizando-se uma estrutura em ASN.1 e são codificadas através de regras básicas de codificação (*basic encoding rules*).

As cinco operações disponíveis à estação de gerenciamento para o acesso à MIB são de baixo nível, logo a interação agente-gerente de granularidade fina, chamada microgerenciamento, não possui uma boa escalabilidade devido ao tráfego intenso e à sobrecarga computacional na estação de gerenciamento [58].

Alguns passos visando a descentralização já foram tomados pelo IETF e pela ISO [102].

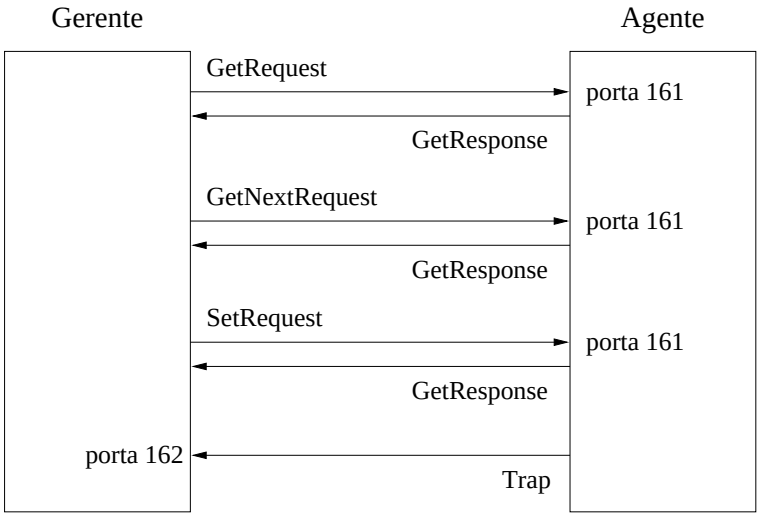

Figura 3.3: As operações do SNMP.

3.2 As Modificações Visando a Descentralização

Na notificação de eventos, os agentes SNMP notificam a estação de gerenciamento quando da ocorrência de alguns eventos significativos. Os agentes de gerenciamento utilizam mensagens chamadas *traps*, enviadas sem um pedido explícito da estação de gerenciamento, para diminuir o uso intensivo da varredura. A proposta da ISO utiliza agentes de gerenciamento mais complexos com maior capacidade de processamento. Em ambos os casos, o agente de gerenciamento somente é responsável pela notificação do evento.

Um gerenciamento de redes mais descentralizado é adotado no SNMPv2 [7, 108] (SNMP versão 2), no qual podem existir múltiplas estações de gerenciamento no mais alto nível de hierarquia, chamadas servidores de gerenciamento. Cada servidor é responsável por gerenciar agentes, porém responsabilidades podem ser delegadas a um gerente intermediário. Esse gerente, também chamado agente procurador, tem funcionalidades de gerente para monitorar e controlar os agentes de gerenciamento sob a sua responsabilidade e também age como um agente de gerenciamento a fim de prover informações e ser controlado por um servidor de gerenciamento (Figura 3.4). A versão 3 do SNMP, o SNMPv3, incorpora um novo esquema de segurança [7, 109] e foi criada para ser utilizada com as versões anteriores do protocolo SNMP.

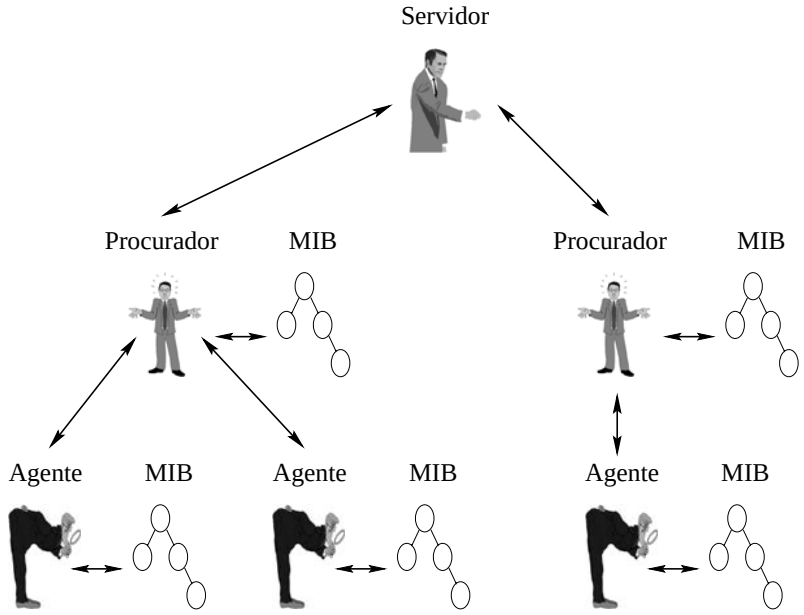

Figura 3.4: Os componentes do gerenciamento via SNMPv2.

O IETF também propôs o RMON (*Remote MONitoring*) [110] que pode utilizar equipamentos para realizar o monitoramento pró-ativo de redes locais em segmentos locais ou remotos [13]. Esses monitores provêem informações sobre enlaces, conexões entre estações, modelos de tráfego e status dos nós da rede (Figura 3.5). Além disso, os monitores também detectam falhas e comportamentos errôneos, geram alarmes quando um limiar predefinido é atingido e identificam eventos complexos, mesmo não estando em contato com a estação de gerenciamento.

Essas propostas de descentralização levam a uma redução do tráfego ao redor da estação de gerenciamento. Porém, como o poder de cálculo dos nós da rede tem aumentado, torna-se possível delegar funções de gerenciamento ainda mais complexas aos nós [22]. Além disso, de modo a satisfazer as diversas necessidades das redes atuais, novos sistemas de gerenciamento de redes que podem analisar dados, tomar decisões e obter medidas pró-ativas para manter a qualidade de serviço da rede devem ser desenvolvidos [40].

Os agentes móveis podem ser utilizados para descentralizar o processamento e o controle da estação de gerenciamento [71].

Tanto o RMON quanto o SNMPv2 possuem algumas desvantagens se compara-

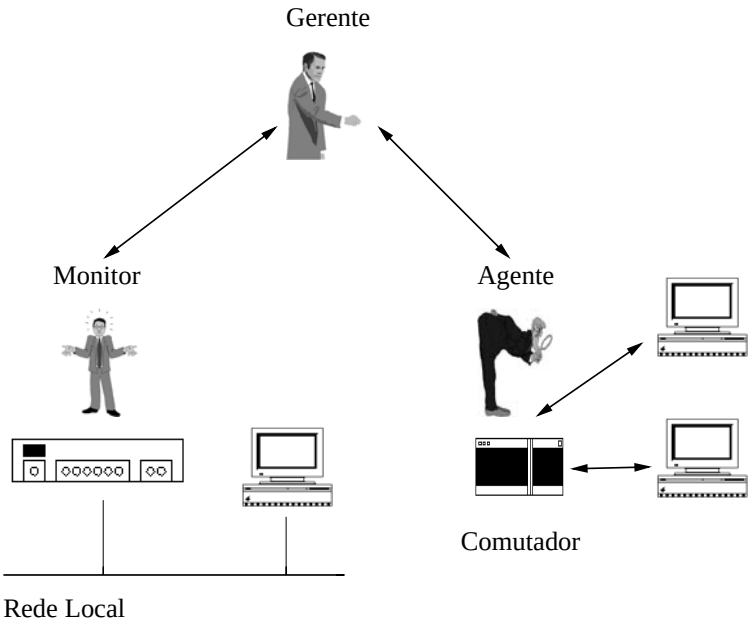

Figura 3.5: Os componentes do gerenciamento via RMON.

dos a um gerenciamento de redes que utiliza agentes móveis. O RMON tipicamente utiliza um monitor para cada segmento de rede, logo o gerenciamento de múltiplos segmentos gera um crescimento considerável do custo [66]. Além disso, tanto o RMON quanto o SNMPv2 só permitem que as operações de controle sejam criadas ou alteradas quando da configuração dos agentes, dos gerentes, dos procuradores ou dos monitores.

3.3 O Gerenciamento Descentralizado com Agentes Móveis

As principais vantagens que podem justificar a utilização de agentes móveis no gerenciamento de redes são:

- redução de custo: através de uma compressão semântica a nível global ou de domínio, o agente pode filtrar e selecionar somente informações relevantes e transferi-las para a estação de gerenciamento;
- processamento assíncrono: um agente pode ser enviado através da rede, reali-

zando suas tarefas em outros nós. Enquanto o agente se encontra fora de seu nó de origem, pode não ser necessário que esse nó permaneça em operação;

- flexibilidade: um novo comportamento dos agentes de gerenciamento pode ser introduzido pela estação de gerenciamento através do envio de um agente móvel com um código de execução diferente, substituindo em tempo real o código anterior;
- autonomia: o agente pode tomar decisões, realizando um gerenciamento reativo com delegação de tarefas.

Conforme apresentado anteriormente, um agente móvel pode migrar de uma máquina para outra no meio de sua execução e carregar com ele dados acumulados sobre o seu estado que incluem as informações obtidas na execução das tarefas. Conforme o número de nós visitados aumenta, o tamanho do agente também cresce, fazendo com que a migração para outro nó seja realizada com maior dificuldade. Uma possível solução para esse problema é percorrer um número fixo de nós e, após isso, retornar ao nó de origem do agente ou enviá-lo os dados (reduzindo o tamanho do agente) e recomeçar a tarefa nos nós restantes (Seção 5.2.2).

Como o SNMPv2 não está amplamente difundido como o SNMP e como o gerenciamento de redes no SNMP não é escalável quando o tamanho ou a complexidade da rede cresce, devido à centralização do processamento e do controle, os agentes móveis podem ser utilizados para aumentar a escalabilidade do gerenciamento de redes, sendo importante descobrir sob que condições esses agentes aumentam a eficiência do gerenciamento.

Capítulo 4

As Implementações de Protótipos de uma Aplicação de Gerenciamento

NESTE capítulo são descritas duas implementações de protótipos de uma aplicação que obtém variáveis da MIB-II [43]: uma baseada em agentes móveis e outra somente baseada no SNMP. Na Seção 4.1, são descritas as principais características da infra-estrutura de agentes móveis utilizada em uma das implementações. Na Seção 4.2, são apresentados dois aplicativos utilizados nos protótipos. As duas implementações de modo a obter variáveis da MIB-II são apresentadas na Seção 4.3. O tempo total relativo a um agente móvel é descrito na Seção 4.4. Por último, o estudo experimental e os seus resultados são apresentados nas Seções 4.5 e 4.6.

A infra-estrutura Mole [5, 111, 112] foi utilizada na implementação do protótipo com agentes.

4.1 A Infra-estrutura Mole

A infra-estrutura Mole foi desenvolvida pelo Grupo de Sistemas Distribuídos da Universidade de Stuttgart, Alemanha, de 1995 a 1998, e foi a primeira infra-estrutura a utilizar a linguagem Java [113]. A versão 3.0 dessa plataforma de agentes foi escolhida por ser gratuita, implementada em Java e possuir código fonte aberto.

Neste sistema, os agentes podem mover, se comunicar entre si e interagir com o subsistema de comunicação.

Dois tipos diferentes de agentes são providos: agentes do sistema e agentes do usuário. Os agentes do sistema são geralmente interfaces para recursos fora da infra-estrutura. Esses agentes possuem mais direitos do que os agentes que não pertencem ao sistema, porém não podem migrar. Os agentes do usuário só podem ter acesso ao recursos disponibilizados através da infra-estrutura, mas podem migrar.

A infra-estrutura Mole utiliza o protocolo TCP para transferir os agentes móveis, que são implementados na linguagem Java. Na plataforma Mole, uma migração fraca é provida (Seção 1.1), logo o programador codifica o estado de execução do agente em variáveis. A escolha da migração fraca pelos desenvolvedores da plataforma, deve-se à inexistência de um mecanismo de captura do estado de um *thread* em uma máquina virtual Java comum. Essa migração é implementada através do uso da serialização de objetos do Java, obtida usando-se uma parte do pacote de invocação remota de método (*Remote Method Invocation* - RMI). Depois que um *thread* do agente chama o método *migrateTo()*, todos os *threads* pertencentes ao agente são suspensos. A partir desse momento, não são mais aceitas novas mensagens ou chamadas remotas. Após a entrega ao agente de todas as mensagens pendentes, o agente é removido da lista de agentes ativos. É feita então a serialização do agente, na qual uma representação independente do sistema é criada e enviada para o destino que reinstancia o agente. Um novo *thread* é iniciado e logo que esse *thread* assume o controle do agente, uma mensagem de sucesso é enviada à fonte que finaliza todos os *threads* pertencentes ao agente e remove-o do sistema. Se ocorrer um erro em qualquer fase, a migração é interrompida e os *threads* do agente na fonte voltam a ser executados.

4.2 Os Aplicativos SNMP Utilizados nos Protótipos

Os dois protótipos implementados, um com agentes móveis e o outro sem, utilizam o protocolo SNMP para obter variáveis da MIB-II. A biblioteca SNMP da AdventNet [114] contém interfaces de programação de aplicações (APIs) que tor-

nam mais simples a implementação de aplicações que utilizam o protocolo SNMP. Foi utilizada a versão 2.2 do AdventNet SNMPv1 que é disponível gratuitamente.

O *daemon* snmpd do pacote ucd-snmp [115], originado na Universidade da Califórnia em Davis, que vem com o Linux Red Hat, também foi utilizado. Esse agente SNMP responde a pedidos de gerentes SNMP ou de outras entidades com comportamento análogo. As versões utilizadas do pacote foram a 3.5.3 (para as máquinas com Red Hat 5.2) e a 4.0.1 (para o Red Hat 6.x). O projeto ucd-snmp, a partir de outubro de 2000, passou a chamar-se net-snmp [116].

4.3 As Duas Implementações

A implementação baseada em agentes móveis (Figura 4.1) consiste em um agente móvel, que migra para todos os elementos de rede a serem gerenciados, em um agente SNMP, que acessa as variáveis da MIB-II, e em um agente tradutor, que converte o pedido do agente móvel para o formato do protocolo SNMP. O agente móvel migra para um elemento de rede (arco 1 da Figura 4.1) e se comunica via chamada remota de procedimentos com o agente tradutor (arco 2). Este agente tradutor envia um pedido (PDU GetRequest) para o agente SNMP (arco 3) e obtém a resposta (arco 4) que é repassada ao agente móvel (arco 5). O agente móvel então segue para o próximo elemento (arco 6) e recomeça a sua execução. Ao terminar sua tarefa, que consiste na visita de todos os elementos de rede, o agente móvel retorna à estação de origem (arco n). É importante destacar que essa implementação utiliza o protocolo SNMP com as suas facilidades. O agente móvel apenas elimina a troca de mensagens pela rede entre o gerente e o agente, que foi substituída pela migração do agente móvel e pela troca local de mensagens.

Na implementação que só utiliza o SNMP, o modelo tradicional deste protocolo foi utilizado. O gerente envia um pedido para um agente SNMP (arco 1 da Figura 4.2) que responde a esse gerente (arco 2). São enviados seqüencialmente pedidos a todos os elementos da rede, um após o outro, ou seja, um novo pedido é iniciado após receber a resposta do anterior, até que o último elemento de rede receba um

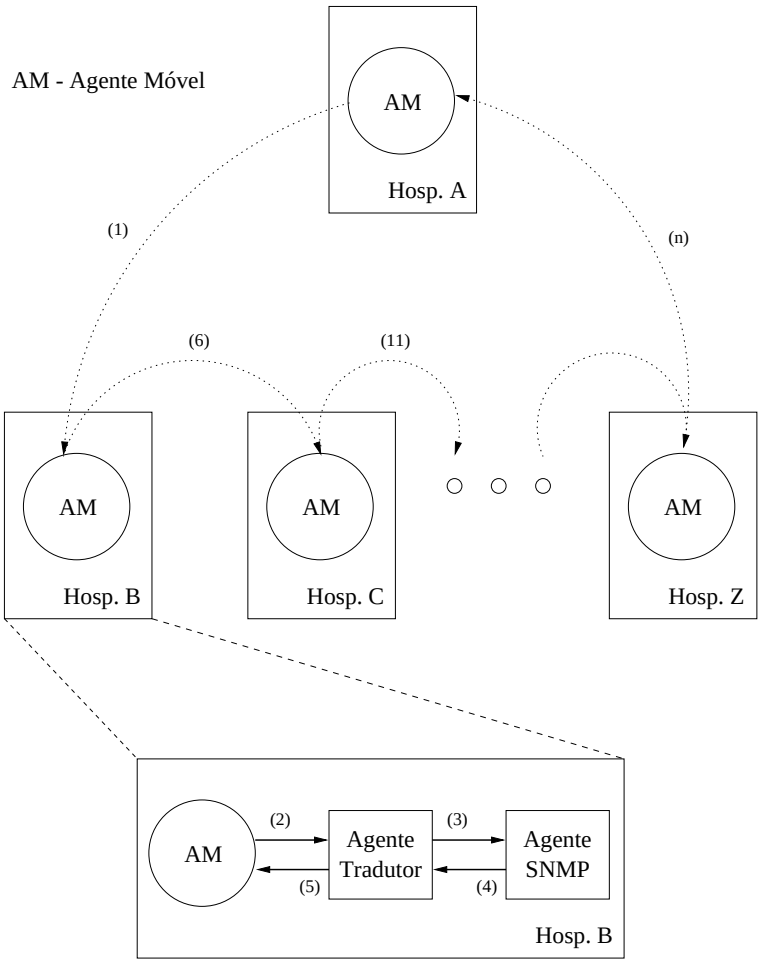

Figura 4.1: O gerenciamento por agente móvel.

pedido (arco n-1) e envie a resposta ao gerente (arco n). Esse gerente foi implementado na linguagem Java, fora da infra-estrutura Mole, a não ser em um caso especial descrito na Seção 4.6.

4.4 O Tempo Total de um Agente Móvel

As migrações de um agente de um lugar para outro estão representadas na Figura 4.1 pelo arcos 1, 6 e n. De um modo geral, uma migração de um agente consiste nos seguintes passos:

a) a serialização dos dados do agente e a construção de uma mensagem que inclui os dados do agente;

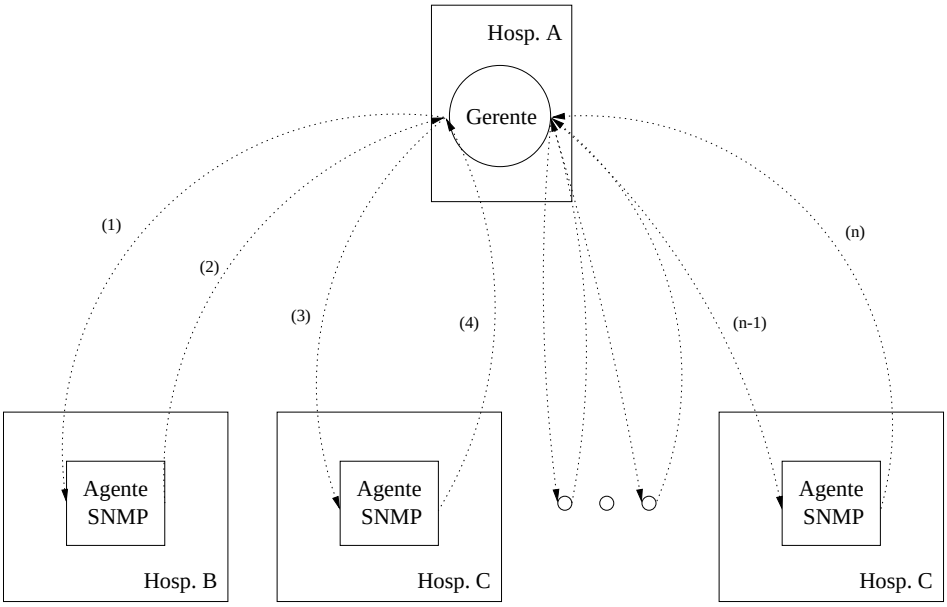

Figura 4.2: O gerenciamento tradicional usando o SNMP.

b) o envio da mensagem ao servidor de agentes no destino;

c) a recepção da mensagem pelo servidor de agentes;

d) a criação de um novo *thread* para a execução do agente;

e) a desserialização do agente;

f) o começo da execução do agente;

g) o envio de uma mensagem de sucesso à origem do agente, para que esse seja retirado do sistema.

De modo simplificado, pode-se dizer que o tempo total de um agente consiste no tempo de transferência na rede (tempo decorrido entre **b)** e **c)**), no tempo relativo à infra-estrutura (tempo decorrido entre **a)** e **f)**, excluindo-se o tempo de transferência na rede) e no tempo de execução da aplicação.

Para a aplicação de busca de variáveis da MIB, o tempo de execução é a soma dos tempos da comunicação agente móvel-agente tradutor, da comunicação agente tradutor-agente SNMP, do envio da PDU GetRequest, da consulta à MIB, do rece-

bimento da PDU GetResponse, da comunicação agente SNMP-agente tradutor e da comunicação agente tradutor-agente móvel.

4.5 O Estudo Experimental

Um estudo experimental é realizado, de modo a avaliar a escalabilidade das duas abordagens: a implementação baseada nos agentes móveis e a implementação convencional SNMP.

A topologia utilizada no estudo consiste em uma estação de gerenciamento (estação A) e em dois elementos gerenciados de rede (estações B e C), interconectados através de uma rede local *Ethernet* de 10 Mbps (Figura 4.3). As estação A é um Pentium MMX 233 Mhz, com 128 Moctetos de memória e rodando Linux Red Hat versão 6.2. As estações B e C são Pentiuns II 350 MHz, com 64 Moctetos e 128 Moctetos de memória e rodando também Linux Red Hat versões 6.1 e 5.2.

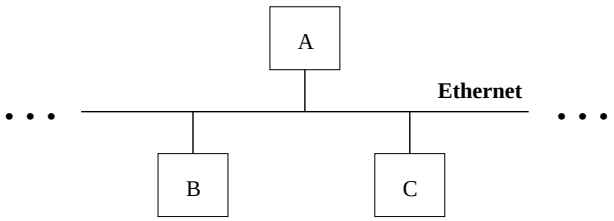

Figura 4.3: A topologia utilizada nos testes.

De modo a avaliar o desempenho do gerenciamento para um grande número de elementos de rede, os dois elementos de rede B e C são repetidos alternadamente; por exemplo, se o número de estações a serem gerenciadas é cinco, o itinerário a ser seguido pelo agente é B, C, B, C , B e A.

Os parâmetros de desempenho considerados são o número de octetos recebidos e transmitidos pela estação de gerenciamento e o tempo de resposta na obtenção da variável da MIB-II [43] *ifInErrors*, a qual representa o número de pacotes recebidos descartados por causa de erros.

O JDK (*Java Development Kit*) 1.1.7 versão 3 foi utilizado em todas as máquinas.

O aplicativo *tcpdump* foi usado para medir os tamanhos dos pacotes dos protocolos UDP e TCP. Todas as medidas foram realizadas cedo pela manhã ou à noite, de modo a limitar as variações de desempenho da rede, evitando retransmissões de pacotes SNMP, que influenciam os resultados de tempo de resposta.

As implementações foram testadas nas mesmas condições, utilizando o mesmo itinerário. Os testes foram realizados com os ambientes de execução dos agentes móveis sendo executados sem interrupções. O número de elementos gerenciados varia entre 1 e 250. Para as medições realizadas, foram calculados intervalos de confiança de 99% relativos à média de dez medidas. Estes intervalos estão representados nos gráficos através de barras verticais.

O agente móvel carrega consigo o nome da variável, o itinerário e as respostas já obtidas, enquanto que o SNMP envia uma PDU GetRequest e recebe uma PDU GetResponse.

4.6 Os Resultados dos Testes

O efeito do número de elementos gerenciados na banda utilizada e no tempo de resposta foi analisado nas medições realizadas. Em todas as figuras são apresentadas as médias das medidas.

As medidas do número de octetos foram feitas sem uma identificação da amostra correspondente, por isso não são apresentados os intervalos de confiança. Para um pequeno número de elementos gerenciados, o SNMP utiliza um menor número de octetos do que o agente móvel para realizar a tarefa (Figura 4.4). Porém, conforme o número de elementos gerenciados aumenta, a sobrecarga devido ao grande número de PDUs GetRequest do SNMP ultrapassa a sobrecarga relativa ao nome da variável, ao itinerário, às respostas já obtidas e a mensagens internas. Com isso, a utilização do agente móvel torna-se mais apropriada, como pode-se prever através de uma extrapolação do gráfico do agente móvel. De acordo a análise das medições realizadas através do *tcpdump*, o tamanho inicial do agente móvel é de aproximadamente 1,5 koctetos.

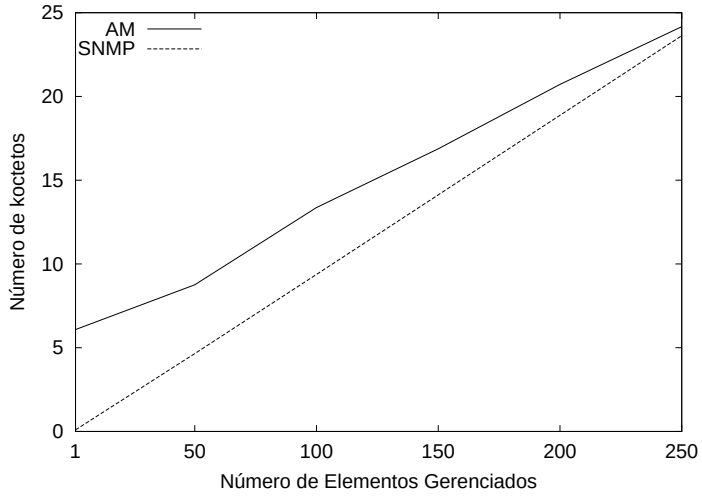

Figura 4.4: Número de octetos para o agente móvel e para o SNMP.

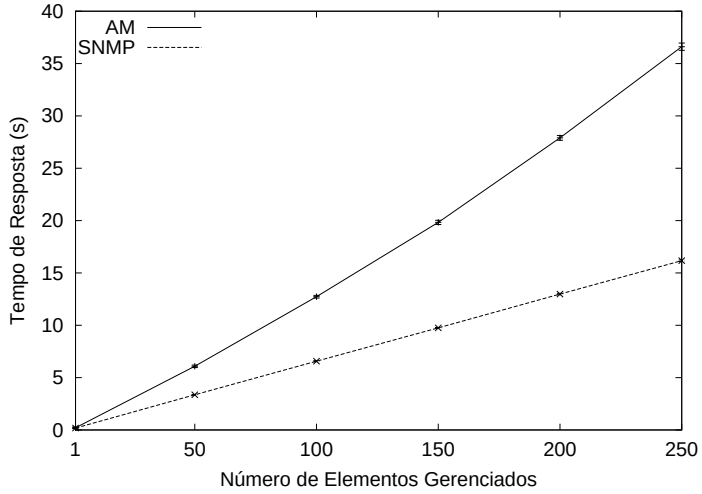

Figura 4.5: Tempo de resposta para o agente móvel e para o SNMP.

Na Figura 4.5, o tempo de resposta é apresentado. O tempo de resposta para o SNMP cresce proporcionalmente ao número de elementos gerenciados, pois o tempo para gerenciar um elemento é praticamente o mesmo para todos os elementos. Para o agente móvel, o tempo de resposta cresce mais rápido conforme o número de elementos aumenta, por causa do tamanho crescente do agente móvel. Na topologia utilizada, o desempenho do SNMP é bem melhor do que o do agente móvel.

De modo a melhor analisar os resultados, foram realizadas medidas dos tempos de acesso às MIBs (envio/recebimento de PDUs GetRequest/GetResponse) e des-

se tempo acrescido aos tempos de comunicação agente móvel-agentes tradutores e agentes tradutores-agente móvel (Figura 4.1).

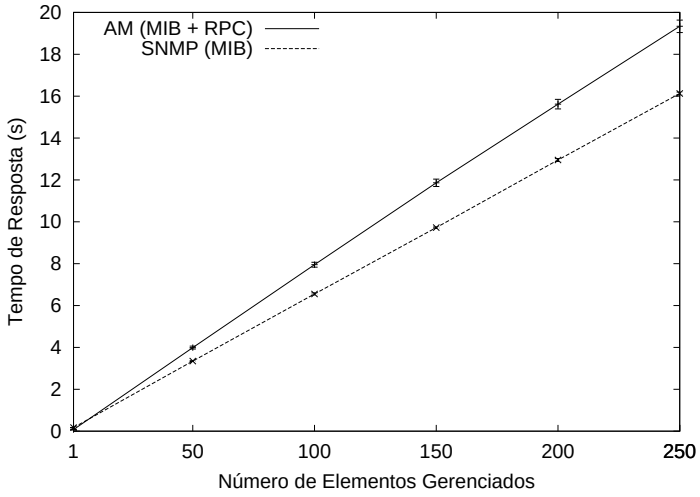

Figura 4.6: Tempo de resposta relativo aos acessos às MIBs.

Na Figura 4.6, são apresentados os tempos de resposta para os acessos às MIBs e para as RPCs relativas às comunicações agente móvel-agentes tradutores. Em relação ao SNMP, pode-se dizer que para um número de estações gerenciadas igual a 250, aproximadamente 99,6 % do tempo total é gasto nos acessos às MIBs. Para o agente móvel, quando o número de estações é 250, os acessos às MIBs e as RPCs ocupam 52,8 % do tempo total. Pode-se afirmar que a consulta às MIBs cresce de modo linear, em função do número de elementos gerenciados e gasta aproximadamente 65 ms para cada elemento para o caso do SNMP. Essa consulta às MIBs acrescida das RPCs agente móvel-agentes tradutores também crescem linearmente e gastam aproximadamente 78 ms para cada elemento.

O tempo restante do agente móvel é calculado através da diferença do tempo total do agente móvel e do tempo das MIBs e das RPCs e é apresentado na Figura 4.7. O cálculo dos intervalos de confiança não foi feito por se tratar da diferença entre amostras do tempo total e do tempo de MIB+RPC. Como o tempo de transmissão do agente é muito pequeno em relação aos demais tempos, o tempo restante corresponde a gastos relativos à infra-estrutura, como a serialização/desserialização, a criação de *threads* e o envio de mensagens internas. Apesar das RPCs também estarem

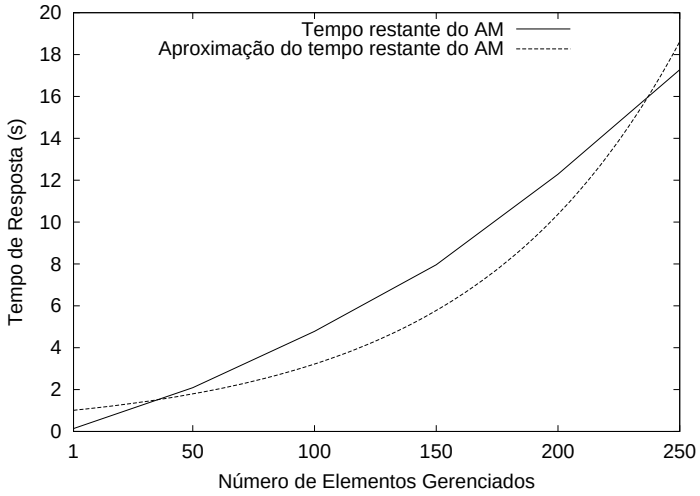

Figura 4.7: Tempo restante do agente móvel.

relacionadas à infra-estrutura, o tempo das chamadas remotas foi contabilizado junto ao tempo das MIBs pois a medição foi realizada no código do agente móvel.

Pode-se dizer que o tempo de resposta cresce exponencialmente com o número de elementos gerenciados, logo a curva da Figura 4.7 pode ser aproximada por:

$$y = a^x, \tag{4.1}$$

onde a = 1,01176 (Figura 4.7). Essa aproximação foi escolhida por permitir, de maneira simples, a sua utilização nas simulações realizadas para topologias mais gerais que são explicadas no Capítulo 5.

O tempo de transmissão na rede para o SNMP também foi medido. O programa *tcpdump* foi novamente utilizado. Esse aplicativo fornece o tempo no qual um pacote é visto pelo núcleo (*kernel*) da máquina pela primeira vez. Se esse tempo for medido em duas máquinas diferentes, tem-se idéia do tempo de transmissão deste pacote. Como o que se procura é o tempo de ida e volta (PDUs GetRequest e GetResponse), supondo-se que não há variação entre os relógios das máquinas durante a medida e sendo t_1 o tempo quando da transmissão pela máquina A de um pacote X, t_2 o tempo quando da recepção pela máquina B do mesmo pacote, t_3 o tempo quando da transmissão pela máquina B de um pacote Y, t_4 o tempo quando da recepção pela máquina A do mesmo pacote e T a diferença entre os relógios das máquinas A e B

(Figura 4.8), tem-se:

$$t = t_4 - (t_3 + T) + (t_2 + T) - t_1 = (t_4 - t_1) - (t_3 - t_2). \tag{4.2}$$

Logo, a diferença entre os relógios é compensada.

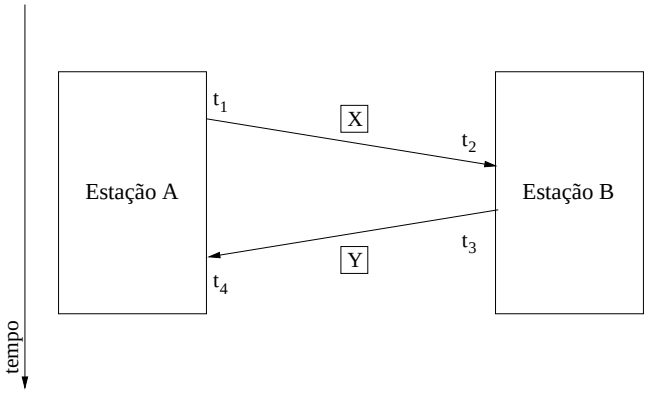

Figura 4.8: Envio e recebimento de pacotes para as máquinas A e B.

Estendendo-se essa análise para o par de máquinas A e C, pode-se medir o tempo total de transmissão. As medidas foram feitas sem uma identificação da amostra correspondente, por isso não são apresentados os intervalos de confiança.

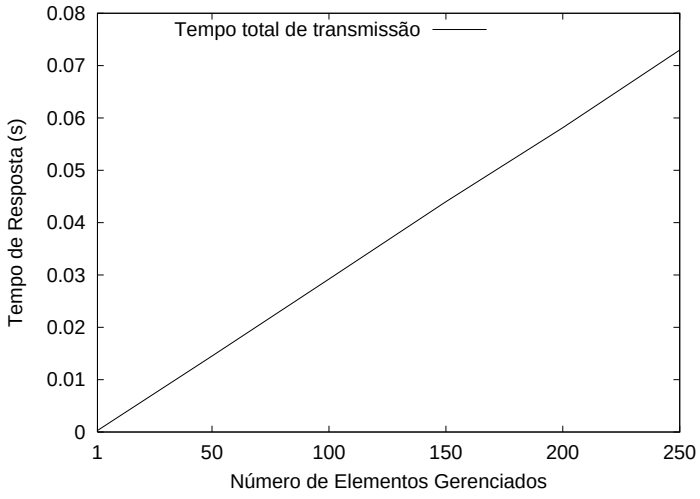

Figura 4.9: Tempo total para a transmissão.

De acordo com a Figura 4.9, o tempo médio de transmissão na rede por variável SNMP é igual a aproximadamente 0,3 ms. Para 250 estações gerenciadas, o tempo total de transmissão corresponde a 0,4 % do tempo total.

Um gerente SNMP também foi implementado na infra-estrutura Mole (neste caso foi utilizado um agente fixo). O objetivo deste experimento é verificar a sobrecarga quando da utilização da plataforma.

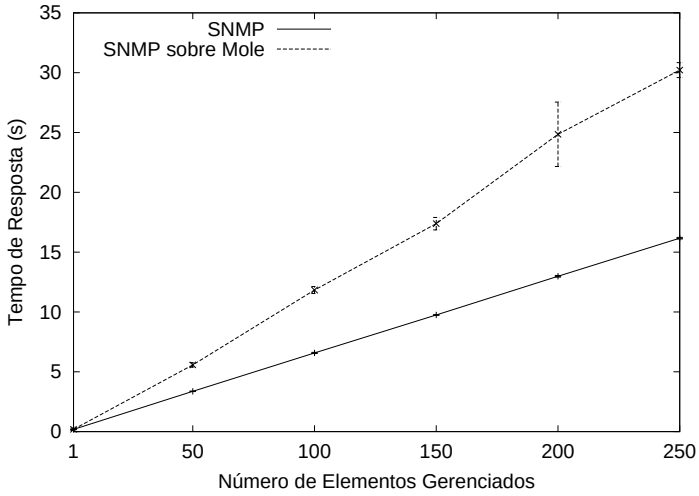

Figura 4.10: Tempo de resposta para o SNMP e para o SNMP sobre a Mole.

De acordo com a Figura 4.10, o tempo de resposta aumenta de aproximadamente 86,9 % (para 250 elementos gerenciados) quando o SNMP é utilizado sobre a infra-estrutura Mole. Isso demonstra que a plataforma Mole realmente exerce uma grande influência no tempo de resposta. A Figura 4.10 indica ainda que para o SNMP sobre a Mole, as medições relativas a 200 elementos gerenciados apresentaram um grande intervalo de confiança causado por variações nos desempenhos da rede ou do processamento das máquinas.

Em relação aos acessos às MIBs, a Figura 4.11 apresenta os tempos para o SNMP e o SNMP sobre a Mole. A consulta às MIBs gasta aproximadamente 120 ms para cada elemento para o caso do SNMP sobre a Mole, contra 65 ms do SNMP; ou seja, há um aumento de aproximadamente 84,6% no tempo. Pode-se observar ainda que a variação nas medidas do SNMP sobre a Mole com 200 elementos gerenciados está relacionada aos acessos às MIBs.

A próxima análise realizada trata da influência do poder de processamento das máquinas utilizadas. A configuração utilizada anteriormente (configuração 1) foi comparada a uma outra (configuração 2), na qual o poder de processamento das

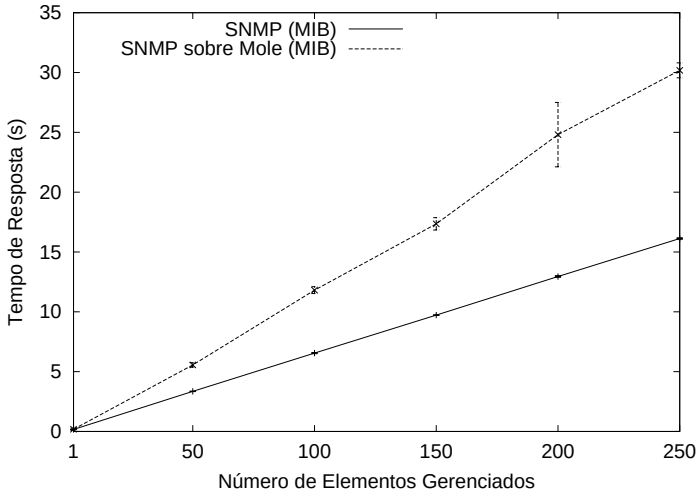

Figura 4.11: Tempo de acessos às MIBs para o SNMP e para o SNMP sobre a Mole.

máquinas é menor. Na configuração 2, a máquina A é a mesma da configuração anterior, a máquina B é um Pentium MMX 233 MHz, com 96 Moctetos de memória e rodando Linux Red Hat versão 5.2, e a máquina C é um Pentium 133Mhz, com 32 Moctetos e Red Hat 6.1.

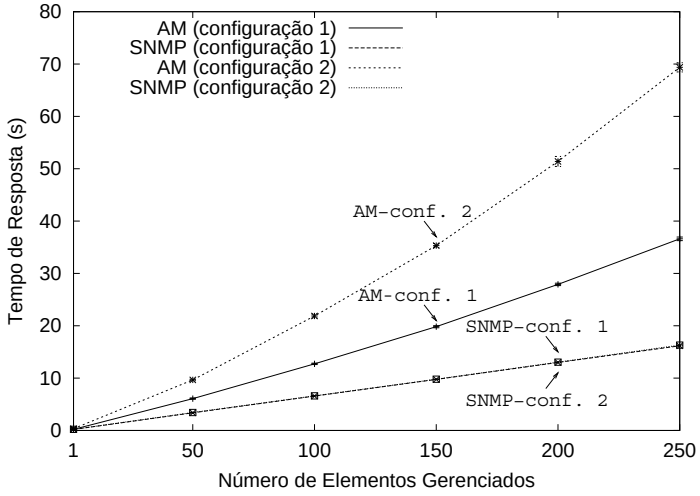

Figura 4.12: Tempo de resposta para as duas configurações.

Pode-se observar pela Figura 4.12, que o comportamento do SNMP praticamente não muda quando diminui-se o poder de processamento das máquinas utilizadas, porém, para o agente móvel, o tempo de resposta aumenta de 89,5 % para 250 nós quando as máquinas mais lentas são utilizadas.

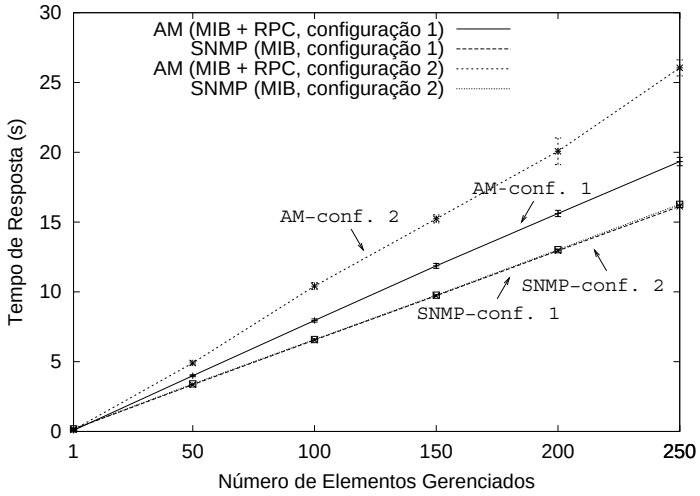

Figura 4.13: Tempo de resposta relativo aos acessos às MIBs para as duas configurações.

Quanto aos tempo de resposta para os acessos às MIBs e para as RPCs relativas às comunicações agente móvel-agentes tradutores, pode-se observar um aumento do tempo de resposta para o agente móvel na configuração 2 (Figura 4.13). Para 250 elementos gerenciados, o tempo de resposta aumenta de 34,7 %.

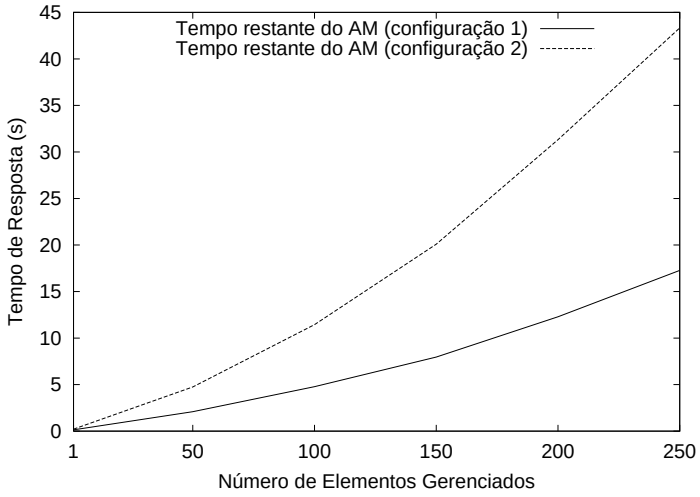

Figura 4.14: Tempo restante do agente móvel para as duas configurações.

O tempo restante do agente móvel é apresentado na Figura 4.14. Este tempo também cresce ao se utilizar máquinas menos potentes. Para 250 elementos, o tempo restante aumenta de 150,8 %, logo, o poder de processamento tem uma maior

influência sobre o tempo restante do agente móvel.

Como foi explicado anteriormente, o modelo de gerenciamento com agentes móveis, no que se refere a desempenho, pode diminuir o tráfego na rede às custas da mobilidade do agente e de um maior tempo de processamento devido às interações do agente na infra-estrutura. Assim, o desempenho do gerenciamento com agentes móveis, comparado ao gerenciamento convencional pelo SNMP, será melhor quanto maior for a limitação (tempo de transmissão de mensagens) de enlaces na rede. As medidas práticas realizadas consideraram o caso mais desfavorável para o modelo de gerenciamento com agentes móveis pois em uma rede local *Ethernet* o tempo de transmissão é muito pequeno em relação ao tempo de processamento das máquinas envolvidas. As medidas foram realizadas para comprovar este fato e para obter parâmetros mais perto dos reais que serão utilizados nas simulações usando topologias maiores e mais próximas das encontradas na *Internet*. O Capítulo seguinte descreve essas simulações.

Capítulo 5

A Análise do Desempenho do Gerenciamento Através de Simulações

A ANÁLISE do desempenho do gerenciamento usando agentes móveis em uma topologia com uma rede local e em topologias maiores e mais próximas das encontradas na *Internet* é realizada através de simulações. Na Seção 5.1, é descrito o modelo utilizado nas simulações. Na Seção 5.2, os resultados das simulações são apresentados.

5.1 O Modelo das Simulações

O simulador de redes *ns* (*Network Simulator*) versão 2.1b4 [117] foi utilizado neste trabalho. Esse simulador de eventos discretos oferece uma infra-estrutura para simulações em redes de computadores com abstrações para nós e para enlaces e possui diversos protocolos já implementados; sendo que nas simulações deste trabalho, foram utilizadas as funções de rede local *Ethernet*, de topologias similares à da *Internet* e dos protocolos TCP e UDP. Alguns módulos dos protocolos TCP e do UDP foram alterados (Apêndice A).

O *ns* trata o envio de pacotes através de uma rede e geralmente não considera o tempo de processamento da camada aplicação em cada nó. Por isso, vários parâmetros relativos ao gerenciamento de redes foram adicionados ao modelo de simulação, de modo a tornar os resultados da simulação mais fiéis a uma implementação real.

É evidente que os parâmetros das implementações são extremamente dependentes da infra-estrutura de agentes, do sistema operacional e da configuração das máquinas utilizadas. Além disso, os parâmetros também são fortemente influenciados pelas tarefas que estão sendo executadas pelo sistema operacional e pelo tráfego existente nos enlaces de comunicação. Apesar de todas estas limitações, o uso de parâmetros de uma configuração real permite a obtenção de resultados de simulação mais confiáveis.

A aplicação de gerenciamento usando agentes móveis é bem mais complexa de se modelar do que a aplicação usando SNMP, pois ela envolve todo um sistema de criação de um ambiente protegido para a execução de tarefas do agente móvel na máquina que o hospeda. Por isso, nas simulações foram considerados os parâmetros que puderam ser extraídos das implementações descritas no Capítulo 4 e que são mostrados na Tabela 5.1.

Tabela 5.1: Os parâmetros utilizados nas simulações.

Parâmetro	Valor
Tamanho inicial (código) do agente móvel	1500 octetos
Tamanho da PDU GetRequest da variável *ifInErrors*	42 octetos
Tamanho da PDU GetResponse da variável *ifInErrors*	51 octetos
Tempo da MIB para o agente móvel por nó	78 ms
Tempo da MIB para o SNMP por nó	65 ms
Relativo ao tempo restante do agente móvel	1,01176

Todos os parâmetros estão relacionados a funções lineares exceto o parâmetro do tempo restante do agente móvel que pertence a uma curva exponencial (Seção 4.6).

O modelo das simulações assume que os enlaces e nós não possuem carga e

que os enlaces não tem perdas. O tamanho máximo de segmentação (*Maximum Segmentation Size* - MSS) utilizado nas simulações é igual a 1500 octetos; com isso, não há fragmentação de mensagens SNMP por essas serem pequenas. Já para o agente móvel, como o seu tamanho inicial é de 1500 octetos, logo após a visita ao primeiro nó gerenciável o seu tamanho estará maior do que a MSS, o que fará com que o agente seja fragmentado e enviado em diferentes pacotes, prejudicando o seu desempenho. Cada pedido de uma variável é enviado em uma mensagem diferente. Em todas as simulações, o agente móvel segue um itinerário preestabelecido de lugares [100]. O agente móvel utiliza como protocolo de transporte o TCP-Reno, por esse ser atualmente de grande utilização na *Internet* e o protocolo UDP é usado nas simulações do SNMP. Em relação ao TCP, o tamanho da janela do receptor é 20, porém em nenhuma das simulações realizadas existe um envio de 30 koctetos de uma só vez, e o mecanismo *slow-start* (Seção 5.2.2) foi utilizado. Mais detalhes a respeito das simulações estão descritos no Apêndice A.

Foram utilizados dois tipos de topologias em todas as simulações. O primeiro tipo (Figura 5.1) consiste em elementos dispostos em uma rede local de "menor custo" (maior banda passante e menor atraso) conectada à estação de gerenciamento através de um enlace de gargalo de maior custo (maior latência e menor banda). A rede local é uma *Ethernet* com 250 nós, banda passante igual a 10 Mbps e latência de 10 μs. Os valores para o enlace de gargalo variam com o experimento realizado.

O segundo tipo de topologia utilizada procura ser próximo às estruturas encontradas na *Internet*. Esse tipo de topologia é chamado *transit-stub*. Cada domínio de roteamento na *Internet* pode ser classificado em *transit* ou *stub*. O propósito dos domínios *transit* é interconectar domínios *stubs* de modo eficiente. Um domínio *transit* compreende um conjunto de nós do *backbone*, nós que estão tipicamente conectados entre si. Em um domínio *transit*, cada nó do *backbone* também pode conectar domínios *stubs*.

Foi usado um programa gerador de topologias chamado GT-ITM [118] que gera topologias aleatórias. Nesse programa, para se gerar topologias do tipo *transit-stub*, é necessário fornecer diversos parâmetros dentre os quais destacam-se o número de

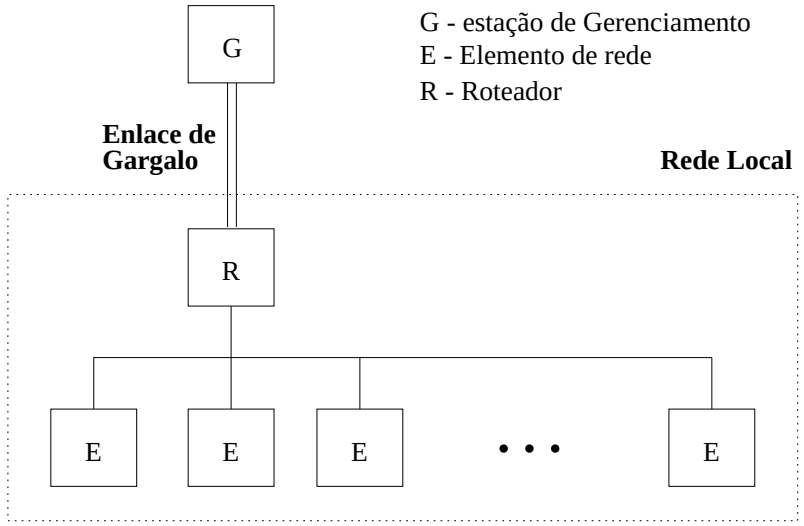

Figura 5.1: A topologia com rede local utilizada nas simulações.

topologias, o número de domínios *transit*, o número médio de nós por *transit*, o número médio de *stubs* por nó *transit* e o número médio de nós por *stub*. Mais detalhes do gerador de topologias são apresentados no Apêndice B.

Esse tipo de topologia pode ser utilizado no gerenciamento de redes de uma organização do tipo matriz-filiais na qual uma matriz quer gerenciar suas filiais espalhadas geograficamente. A estratégia de gerenciamento utilizada neste trabalho para topologias *transit-stubs* considera que a estação de gerenciamento está localizada em um nó de um domínio *stub* e os elementos a serem gerenciados pertencem aos outros domínios *stubs* (Figura 5.2). No caso do conjunto matriz-filiais, a estação de gerenciamento da matriz gerencia os roteadores das filiais, sendo que cada filial (representada por um *stub*) contém vários roteadores.

Os parâmetros de desempenho avaliados foram o número de octetos transmitidos e recebidos pela estação de gerenciamento e o tempo de resposta na obtenção de variáveis SNMP (objetos da MIB-II [43]).

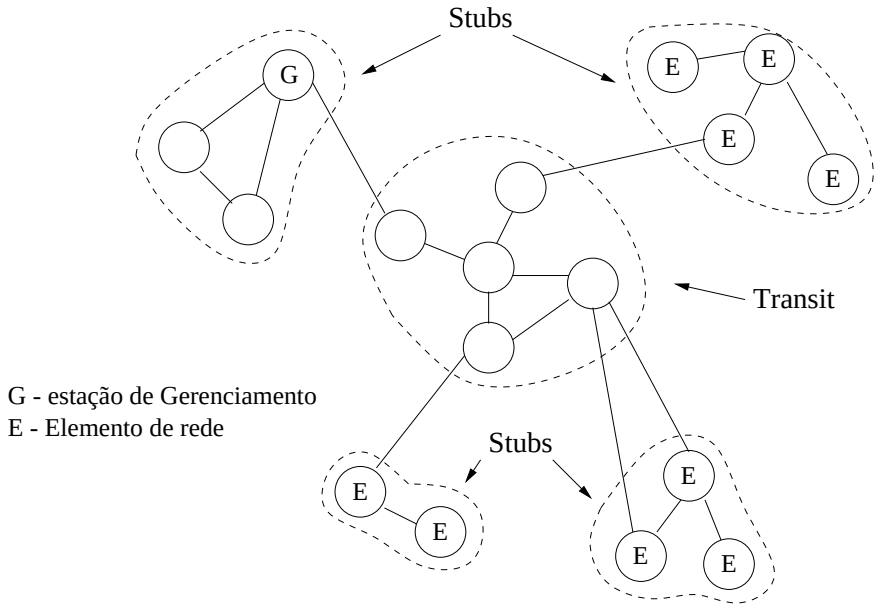

Figura 5.2: O gerenciamento em uma topologia *transit-stub*.

5.2 Os Resultados das Simulações

Resultados para topologias com rede local e para topologias mais complexas, assim como para novas características dos agentes móveis, foram obtidos.

5.2.1 O Gerenciamento de uma Rede Local

A topologia com rede local (Figura 5.1) foi utilizada nessas análises. No primeiro experimento, o enlace de gargalo possui banda passante infinita e latência zero, logo a estação de gerenciamento se comporta como se pertencesse à rede local. Com isso, pode-se comparar os resultados dessa simulação aos resultados das implementações do Capítulo 4.

Na Figura 5.3 são apresentados os tempos de resposta para o agente móvel e para o SNMP, na implementação e na simulação. Observa-se que os modelos simulados reproduzem os modelos implementados, havendo uma pequena diferença no tempo do agente móvel devido à aproximação utilizada para o tempo restante do agente (Seção 4.6).

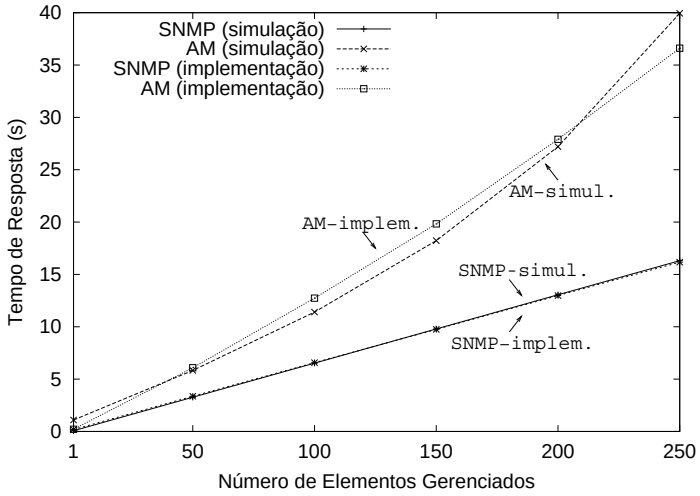

Figura 5.3: Tempo de resposta para o agente móvel e para o SNMP.

Uma situação bastante comum é o gerenciamento remoto de uma rede local. Nesse caso, a estação de gerenciamento se conecta à rede geralmente através de uma rede de longa distância (*Wide Area Network* - WAN) que possui uma maior latência e uma menor banda passante. Esta situação foi representada pela estação de gerenciamento sendo conectada a uma rede local através de um enlace de alto custo (Figura 5.1).

Para essa topologia, foram feitas análises dos efeitos da latência e da banda passante do enlace de alto custo, da variável a ser obtida e do protocolo de transporte utilizado.

O Efeito da Latência e da Banda Passante do Enlace

Três diferentes valores para a latência do enlace são utilizados (Tabela 5.2) e a banda passante do enlace é 2 Mbps.

De acordo com a Figura 5.4, o comportamento do agente móvel praticamente não muda com a latência (todas as cinco curvas estão muito próximas umas das outras). Porém para o SNMP, a diferença de desempenho entre os tempos de resposta para os cinco valores de latência aumenta com o número de elementos gerenciados, pois os pacotes SNMP passam várias vezes pelo enlace de gargalo. Para o enlace e_5, a

Tabela 5.2: Os valores para diferentes latências do enlace de gargalo.

Enlace	Latência (ms)	Banda Passante (Mbps)
e_1	120	2
e_2	90	2
e_3	60	2
e_4	30	2
e_5	1	2

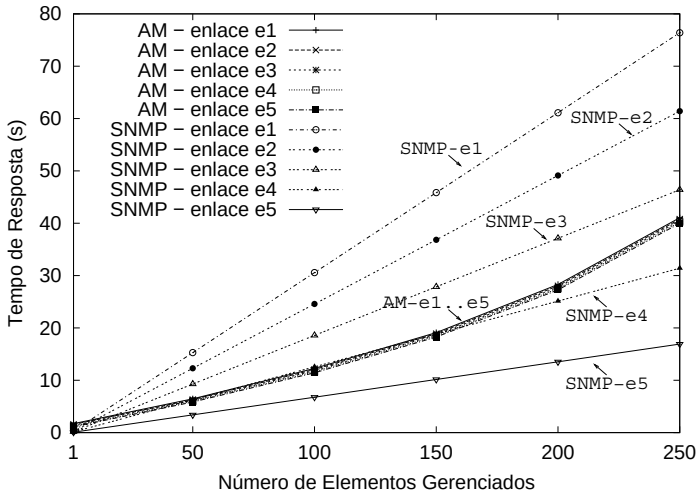

Figura 5.4: Tempo de resposta para diferentes latências do enlace com banda de 2 Mbps.

latência não é suficiente para que o agente móvel apresente um melhor desempenho, enquanto que para os enlaces e_3 a e_1, o agente móvel apresenta um desempenho superior ao do SNMP. Pode-se dizer que o agente móvel se mostra praticamente insensível à latência do enlace de gargalo e que o agente possui um maior desempenho para latências típicas das WANs atuais.

O efeito da banda passante do enlace de gargalo também é avaliado. Quatro diferentes valores para a banda passante são utilizados (Tabela 5.3) e a latência do enlace é 90 ms.

Na Figura 5.5, para a menor banda (enlace e_1 da Tabela 5.3), o agente móvel e

Tabela 5.3: Os valores para diferentes bandas passantes do enlace de gargalo.

Enlace	Banda passante (Mbps)	Latência (ms)
e_1	0,01	90
e_2	0,1	90
e_3	2	90
e_4	10	90

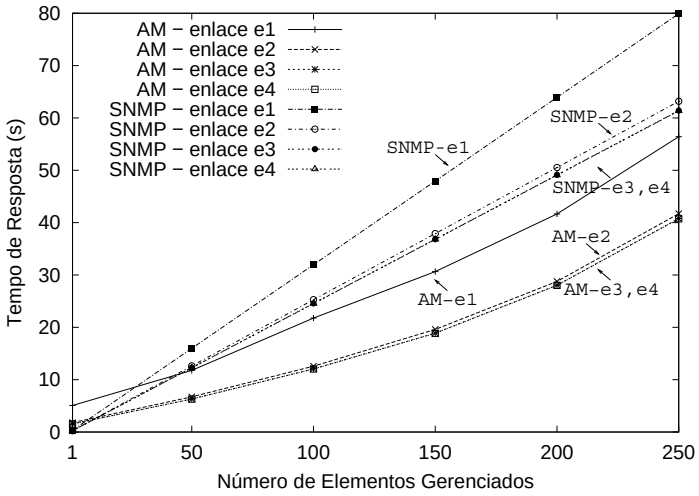

Figura 5.5: Tempo de resposta para diferentes bandas passantes do enlace com latência de 90 ms.

o SNMP apresentam tempos de resposta maiores pois as pequenas bandas fazem com que a transferência de agentes móveis e de pacotes SNMP seja dificultada. Para os outros valores do enlace, os comportamentos do agente móvel e do SNMP praticamente não mudam com a banda passante, pois a soma das parcelas do tempo de resposta relativas à latência e ao tempo da MIB é, para esses valores de banda passante, bem maior do que a parte do tempo de resposta relativa à banda.

O Efeito da Variável a ser Obtida

Uma análise quanto ao comportamento do agente móvel e do SNMP na busca de diferentes variáveis é realizada. Três variáveis são utilizadas: a variável v_1 é a

mesma utilizada anteriormente e as variáveis v_2 e v_3 são respectivamente chamadas *sysORDescr.3* e *sysORDescr.5*. Os tamanhos dos pedidos e das respostas relacionados a essas variáveis estão descritos na Tabela 5.4. A latência do enlace é 90 ms e a banda passante é igual a 2 Mbps.

Tabela 5.4: Os tamanhos dos pacotes de pedido e resposta para as variáveis.

Variável	Pedido (octetos)	Resposta (octetos)
v_1	42	51
v_2	42	87
v_3	42	128

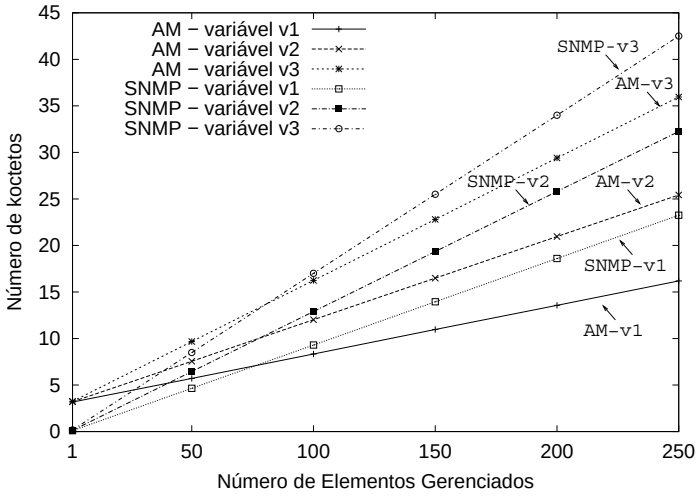

Figura 5.6: Número de octetos para diferentes variáveis.

Na Figura 5.6, pode-se observar o número de octetos para as diferentes variáveis. Quanto maior o tamanho do pedido/resposta, maior o número total de octetos transmitidos e recebidos pela estação de gerenciamento. Para um número de elementos de rede gerenciados maior do que 70, aproximadamente, o agente móvel utiliza menos octetos relacionados à estação de gerenciamento do que o SNMP para realizar a tarefa. Com isso, o gerenciamento de grandes redes locais pode ser feito com um menor custo.

Em relação ao tempo de resposta, de acordo com a Figura 5.7, para o SNMP, a

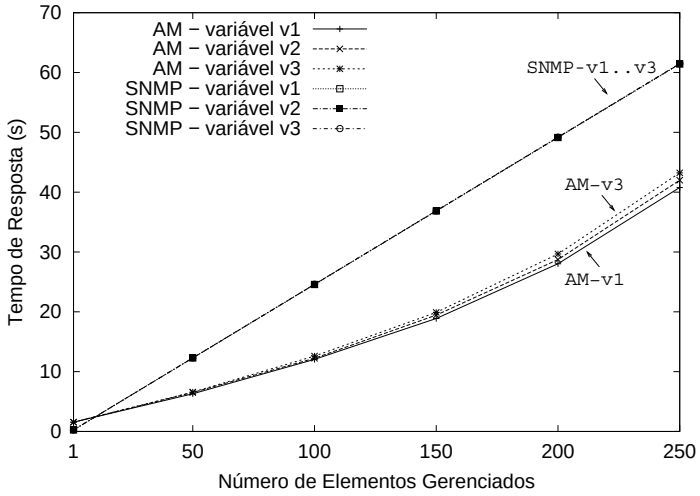

Figura 5.7: Tempo de resposta para diferentes variáveis no enlace de 90 ms e 2 Mbps.

variável obtida não tem grande influência pois o número de octetos trocados entre a estação de gerenciamento e o elementos gerenciados é pequeno. Para o agente móvel, o tempo de resposta é praticamente o mesmo para as diferentes variáveis, nessa configuração de rede local com enlace de gargalo.

O Efeito do Protocolo de Transporte

De modo a se verificar a influência do protocolo de transporte nos resultados, foram realizadas simulações com o agente móvel utilizando o protocolo UDP e o SNMP usando o TCP-Reno. A banda do enlace de gargalo é 2 Mbps e a latência é de 90 ms.

Na Figura 5.8, pode-se notar que o número de octetos no caso do SNMP aumenta bastante quando o TCP é utilizado devido ao envio dos reconhecimentos de tamanho 40 octetos. Esses reconhecimentos a cada pacote não têm grande influência sobre o agente móvel pois devido à mobilidade do agente, os reconhecimentos se distribuem e poucos passam pelo enlace de gargalo. O agente móvel utiliza menos octetos relacionados à estação de gerenciamento do que o SNMP para realizar a tarefa quando o número de elementos gerenciados é maior do que, aproximadamente, 20 com o TCP e 70 com o UDP.

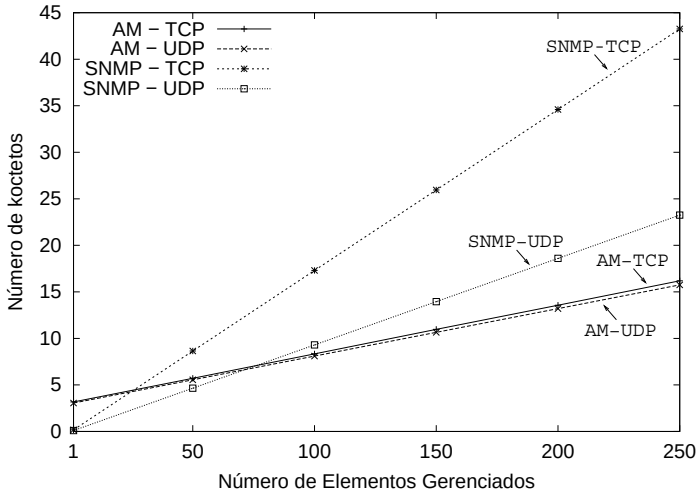

Figura 5.8: Número de octetos para os protocolos de transporte TCP e UDP.

Quanto ao tempo de resposta (Figura 5.9), para esse experimento, o desempenho é praticamente o mesmo para o UDP e para o TCP, pois a topologia utilizada faz com que não haja uma grande influência do mecanismo *slow-start* (Seção 5.2.2) do controle de congestionamento do TCP. Isso ocorre porque a latência do enlace e da rede local não são suficientes para aumentar de modo significativo o tempo de resposta, quando os pacotes segmentados não são enviados de uma só vez.

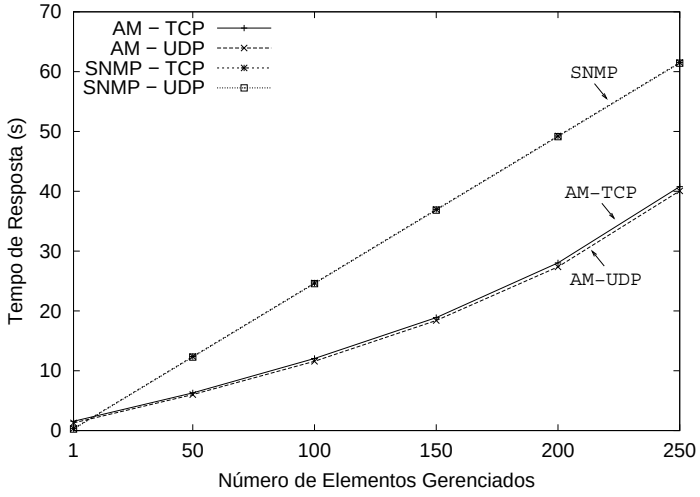

Figura 5.9: Tempo de resposta para os protocolos de transporte TCP e UDP no enlace de 90 ms e 2 Mbps.

5.2.2 O Gerenciamento em uma Topologia *Transit-Stub*

Na seção anterior, fica evidente que ao se evitar um maior tráfego no enlace de gargalo, o agente móvel supera o modelo de gerenciamento pelo SNMP quando o número de elementos gerenciados ultrapassa um valor relacionado à sobrecarga dos vários pedidos do SNMP. Os resultados de simulação obtidos comprovam esse fato, quantificam a melhora de desempenho e mostram os limiares onde o gerenciamento por agentes móveis é melhor do que o gerenciamento pelo SNMP. A situação de gerenciamento remoto de uma rede local através de um enlace de gargalo é um caso muito particular e já abordado pelo RMON.

Nesta seção, procura-se avaliar o desempenho do agente móvel em uma situação mais próxima da encontrada na *Internet*. Para esse fim, foram utilizadas topologias *transit-stubs*. Para garantir que os resultados não sejam específicos a uma determinada topologia, o programa gerador de topologias aleatórias foi utilizado e foram geradas três topologias de 272 nós, banda passante igual a 2 Mbps e latência de algumas dezenas de milisegundos. Nestas simulações, a estação de gerenciamento controla grupos de 16 elementos de rede que é o número de nós de cada um dos *stubs*.

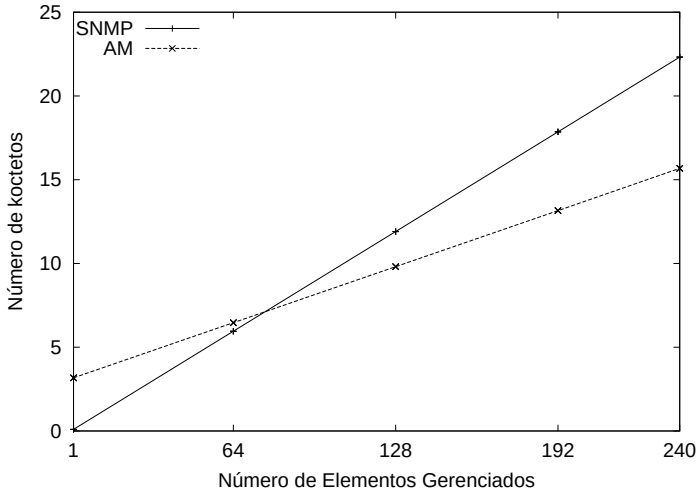

Figura 5.10: Número de octetos para o agente móvel e para o SNMP.

Para um número de elementos gerenciados menor do que aproximadamente 70,

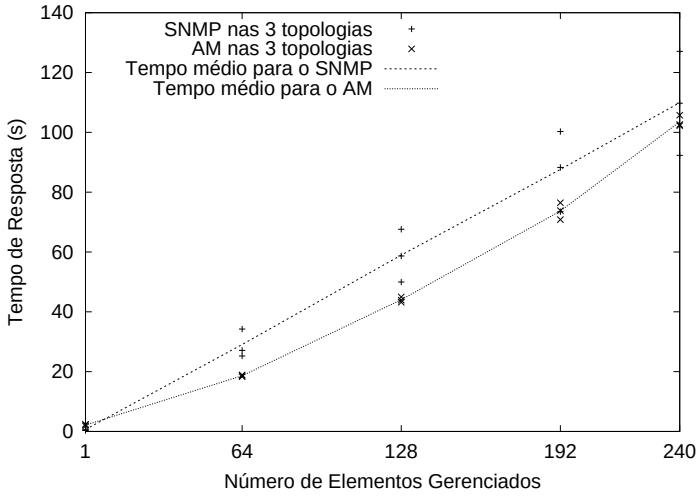

Figura 5.11: Tempo de resposta para o agente móvel e para o SNMP.

o SNMP utiliza menos octetos do que o agente móvel para realizar a tarefa (Figura 5.10). Porém, com o aumento do número de elementos a serem gerenciados, a sobrecarga devido ao grande número de PDUs GetRequest do SNMP faz com que o desempenho do modelo por agentes móveis seja bem melhor.

O comportamento do agente móvel praticamente não muda com a topologia (Figura 5.11), porém para o SNMP, há uma pequena variação nos tempos de resposta para as três topologias. Essa variação se deve ao grande número de pacotes do SNMP que passam pelos enlaces do *backbone* (*transit*) e à configuração dos nós do *backbone* que varia com a topologia. Na Figura 5.11, os tempos médios de resposta também são apresentados. Para um número muito pequeno de elementos gerenciados, o SNMP possui um desempenho melhor pelo fato do tamanho do pacote do SNMP ser menor do que o tamanho inicial do agente móvel. Conforme o número de elementos gerenciados aumenta, o tempo de resposta para o SNMP cresce proporcionalmente, pois o tempo para gerenciar um *stub* é praticamente o mesmo para todos os *stubs*. Para o agente móvel, o tempo de resposta cresce mais rápido com o aumento do número de elementos gerenciados, devido ao tamanho crescente do agente. Extrapolando-se o resultado obtido, pode-se afirmar que o agente móvel possui um desempenho melhor do que o do SNMP quando o número de elementos gerenciados encontra-se entre dois limites, o inferior e o superior, que são determi-

nados, respectivamente, pelo número de mensagens que passam pelo *backbone* e pelo tamanho crescente do agente.

É evidente que se existirem enlaces de gargalo no *backbone*, o desempenho do agente móvel será ainda bem melhor do que o do SNMP. Essa situação de gargalos em *backbone* ainda é muito comum nos dias de hoje, principalmente em países em desenvolvimento.

O Efeito da Variável a ser Obtida

Uma análise quanto ao comportamento do agente móvel e do SNMP na busca de diferentes variáveis também é realizada. As variáveis utilizadas são as mesmas descritas anteriormente.

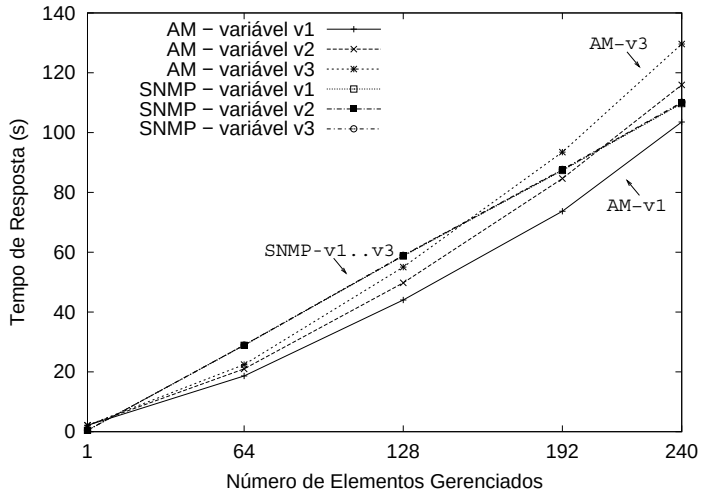

Figura 5.12: Tempo de resposta para diferentes variáveis.

De acordo com a Figura 5.12, para o SNMP, a variável obtida não tem grande influência pois o número de octetos trocados entre a estação de gerenciamento e os elementos gerenciados é pequeno. Para o agente móvel, o tempo de resposta cresce quando o número de octetos trocados aumenta (por exemplo, da variável v_1 para a variável v_3). Nesse experimento, quando o número de elementos gerenciados é 240, o tempo de resposta para o agente móvel aumenta de 25,2%, quando da obtenção da variável v_3 no lugar da variável v_1.

O Efeito do Protocolo de Transporte

De modo a se verificar a influência do protocolo de transporte nos resultados, foram realizadas simulações com o agente móvel utilizando o protocolo UDP e com o SNMP usando o TCP-Reno.

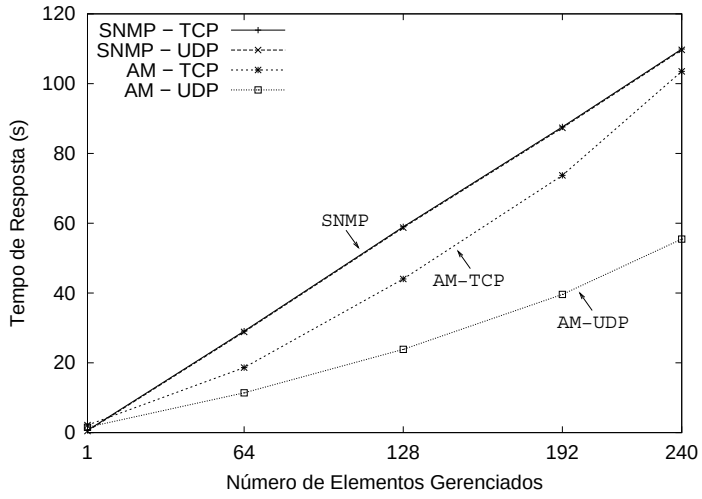

Figura 5.13: Tempo de resposta para os protocolos de transporte TCP e UDP.

No caso desse experimento, o protocolo não tem influência sobre o tempo de resposta do SNMP, porém para o agente móvel, o tempo de resposta diminui bastante ao se utilizar o UDP (Figura 5.13). Isso ocorre devido ao envio dos reconhecimentos e principalmente ao controle de congestionamento do TCP, mais precisamente ao mecanismo *slow-start*. Nesse mecanismo, quando o tamanho da janela de congestionamento é n segmentos, se todos os n segmentos foram reconhecidos em tempo, a janela de congestionamento é aumentada do número de octetos correspondentes a n segmentos. Isso faz com que a janela de congestionamento cresça de maneira exponencial até que haja um estouro de temporizador (*timeout*) ou que a janela do receptor seja alcançada. A idéia é que se rajadas de, por exemplo, 1500, 3000, 4500 e 6000 octetos são recebidas sem problemas pelo receptor, porém uma rajada de 7500 octetos gera um estouro de temporizador, a janela de congestionamento deve ser de 6000 octetos para evitar o congestionamento. Enquanto a janela de congestionamento ficar em 6000 octetos, nenhuma rajada maior será enviada mesmo que a janela do receptor aumente. A utilização desse mecanismo faz com que os pacotes

segmentados do agente móvel não possam ser enviados de uma só vez, o que faz com que o tempo de resposta aumente.

O Efeito do Retorno do Agente Móvel à Estação de Gerenciamento

Como descrito nas seções anteriores, o tamanho do agente móvel cresce com o número de nós visitados e, conseqüentemente, a migração desse agente torna-se mais difícil. Nessa seção, avalia-se o ganho em desempenho da estratégia de fazer o agente móvel retornar à estação de gereciamento, de modo que o tamanho do agente seja reduzido. Esse experimento considera que o agente móvel visita um número fixo de nós "por viagem", ou seja, um número fixo de elementos gerenciados é visitado, o agente retorna à estação de gerenciamento para "descarregar" esses dados já coletados e, depois disso, o agente recomeça a tarefa de coletar variáveis nos nós restantes. As variáveis utilizadas são as três apresentadas anteriormente, porém a busca será feita em todos os 240 elementos gerenciados. O número de estações visitadas por viagem varia de 1 a 240.

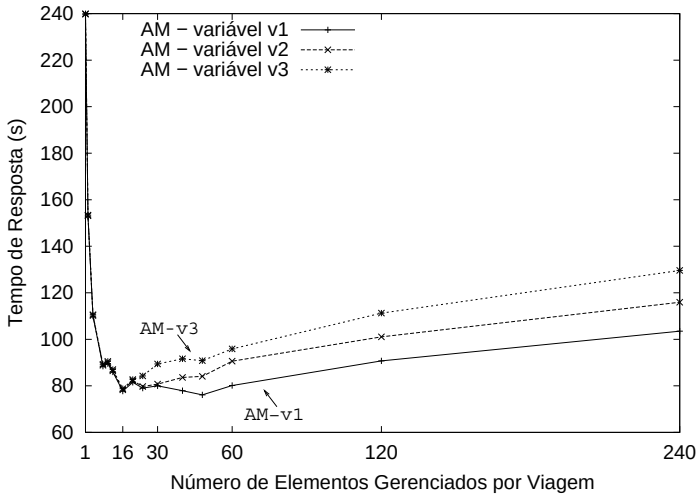

Figura 5.14: Tempo de resposta com o retorno do agente móvel.

Na Figura 5.14, para um número pequeno de elementos gerenciados por viagem, o tempo de resposta diminui acentuadamente quando o número de elementos gerenciados aumenta, pois em uma viagem o agente percorre poucos nós e retorna à estação de gerenciamento. Conforme o número de nós percorridos continua aumen-

Tabela 5.5: Tempo de resposta com o retorno do agente móvel.

No. de Nós	Variável v_1	Variável v_2	Variável v_3
1	239,532 s	239,729 s	239,952 s
2	152,908 s	153,129 s	153,381 s
4	109,974 s	110,244 s	110,552 s
8	88,4694 s	88,8388 s	89,2595 s
10	89,5192 s	89,975 s	90,4941 s
12	85,958 s	86,4579 s	87,0139 s
16	77,7355 s	78,3019 s	78,7644 s
20	81,5078 s	82,1827 s	82,7317 s
24	79,0231 s	79,7112 s	84,2378 s
30	80,0324 s	80,7632 s	89,3899 s
40	77,9024 s	83,625 s	91,6276 s
48	76,0962 s	84,1003 s	90,8335 s
60	80,1638 s	90,5809 s	95,8852 s
120	90,7351 s	101,058 s	111,246 s
240	103,519 s	115,946 s	129,615 s

tando, o tempo de resposta diminui até um ponto no qual o tempo começa a crescer devido à dificuldade de migração do agente por causa do seu tamanho. O formato tipo "dente de serra" da curva deve-se ao fato de que quando o número de estações a serem percorridas de uma vez não é múltiplo do número de estações de um *stub* (16 nesse experimento), o agente móvel atravessa vários *stubs* durante uma viagem e, conseqüentemente, passa mais vezes pelos nós do domínio *transit*, o que aumenta o tempo de resposta. O "ponto ótimo" para esse experimento varia com o tamanho do agente móvel, conseqüentemente com a variável a ser obtida (Tabela 5.5). Por exemplo, para a variável v_1, percorrer 48 nós e retornar à estação de gerenciamento resulta em um melhor desempenho. De acordo com esta tabela, há uma diminuição do tempo de resposta em relação à obtenção das 240 variáveis em uma só viagem de 27%, 32% e 39%, respectivamente, para as variáveis v_1, v_2 e v_3.

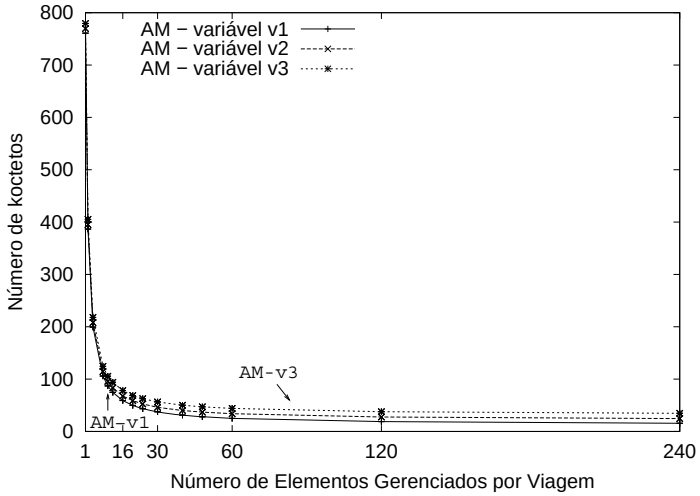

Figura 5.15: Número de octetos com o retorno do agente móvel.

Tabela 5.6: Número de octetos com o retorno do agente móvel.

No. de Nós	Variável v_1	Variável v_2	Variável v_3
1	761,04 koctetos	769,68 koctetos	779,52 koctetos
2	386,64 koctetos	395,28 koctetos	405,12 koctetos
4	199,44 koctetos	208,08 koctetos	217,92 koctetos
8	105,84 koctetos	114,48 koctetos	124,32 koctetos
10	87,12 koctetos	95,76 koctetos	105,6 koctetos
12	74,64 koctetos	83,28 koctetos	93,92 koctetos
16	59,04 koctetos	67,68 koctetos	78,12 koctetos
20	49,68 koctetos	58,8 koctetos	68,64 koctetos
24	43,44 koctetos	52,48 koctetos	62,72 koctetos
30	37,52 koctetos	46,16 koctetos	56,32 koctetos
40	31,2 koctetos	40,08 koctetos	50,16 koctetos
48	28,04 koctetos	36,88 koctetos	47,12 koctetos
60	25,04 koctetos	33,84 koctetos	44 koctetos
120	18,8 koctetos	27,6 koctetos	37,76 koctetos
240	15,68 koctetos	24,52 koctetos	34,64 koctetos

Quanto ao número de octetos, esse diminui quando o número de nós percorridos de uma vez aumenta, conforme a Figura 5.15 e a Tabela 5.6, pois quanto maior o número de vezes que o agente móvel retorna à estação de gerenciamento, maior o número de octetos transmitidos e recebidos por ela.

O Efeito do Envio de Dados à Estação de Gerenciamento

A estratégia de enviar os resultados à estação de gerenciamento, ao invés de retornar à mesma, a cada viagem também é analisada.

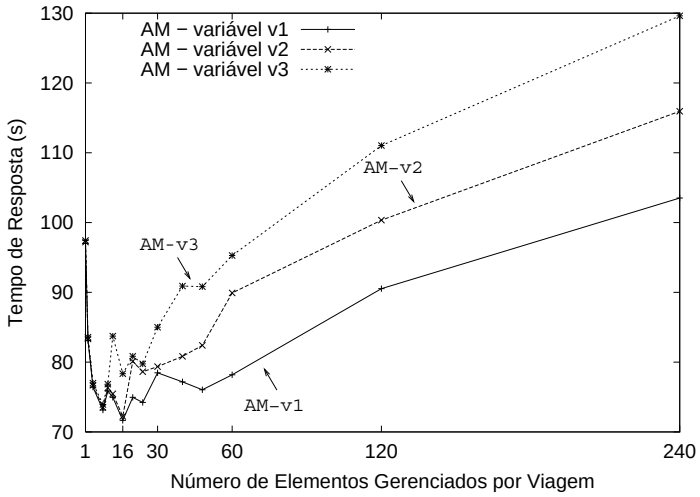

Figura 5.16: Tempo de resposta com o envio dos dados à estação de gerenciamento.

Na Figura 5.16, pode-se observar o mesmo efeito "dente de serra" explicado anteriormente. O tempo de resposta diminui de 31%, 38% e 42%, para as variáveis v_1, v_2 e v_3, comparando-se cada ponto ótimo com a obtenção das 240 variáveis em uma só viagem (Tabela 5.7).

Na Figura 5.17, pode-se observar que a estratégia do envio dos dados tem um desempenho bem melhor do que a do retorno à estação de gerenciamento quando o número de elementos gerenciados é pequeno, pois o código do agente móvel não é enviado para a estação de gerenciamento. Aumentando-se o número de elementos gerenciados por viagem, as estratégias se comportam praticamente da mesma maneira, pois o tamanho total dos dados coletados pelo agente é muito maior do que o

Tabela 5.7: Tempo de resposta com o envio dos dados à estação de gerenciamento.

No. de Nós	Variável v_1	Variável v_2	Variável v_3
1	97,004 s	97,2004 s	97,4241 s
2	83,1018 s	83,3229 s	83,5748 s
4	76,4104 s	76,6809 s	76,9889 s
8	73,1448 s	73,5142 s	73,9349 s
10	75,8676 s	76,3233 s	76,8424 s
12	74,9515 s	75,4514 s	83,7321 s
16	71,6589 s	72,2252 s	78,3445 s
20	74,9619 s	80,1208 s	80,8479 s
24	74,2299 s	78,6249 s	79,7385 s
30	78,4753 s	79,359 s	85,006 s
40	77,168 s	80,8331 s	90,9015 s
48	76,0366 s	82,4054 s	90,8122 s
60	78,1816 s	89,9165 s	95,2698 s
120	90,5236 s	100,339 s	111,035 s
240	103,519 s	115,946 s	129,615 s

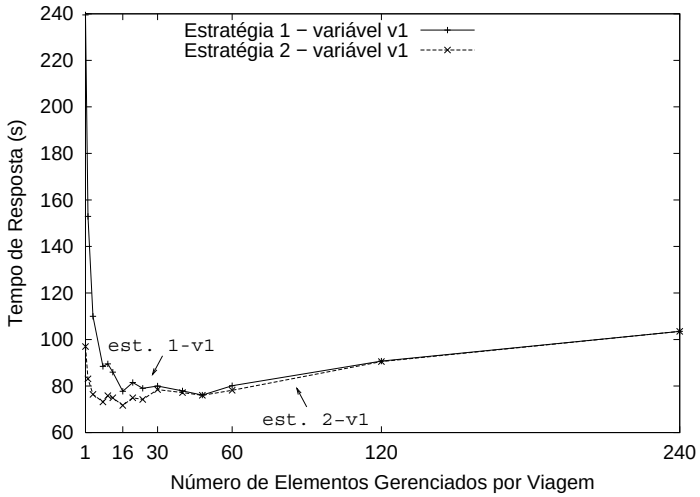

Figura 5.17: Tempo de resposta para as diferentes estratégias.

do código do agente.

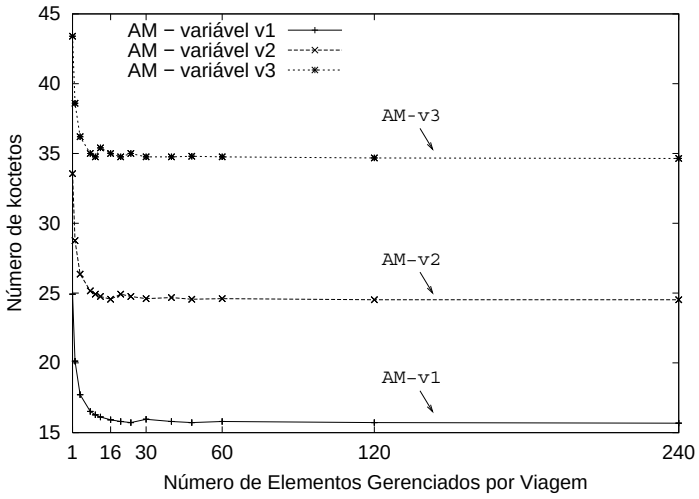

Figura 5.18: Número de octetos com o envio dos dados.

Quanto ao número de octetos, esse diminui quando o número de nós percorridos de uma vez aumenta, conforme a Figura 5.18 e a Tabela 5.8, pois quanto maior o número de vezes que o agente móvel retorna à estação de gerenciamento, maior o número de octetos transmitidos e recebidos por ela.

De um modo geral, conclui-se que os agentes móveis devem ser usados no gerenciamento remoto de diversas sub-redes; principalmente se as ligações entre a estação de gerenciamento e as estações a serem gerenciadas forem de alto custo (pequena banda passante e grande latência).

Tabela 5.8: Número de octetos com o envio dos dados à estação de gerenciamento.

No. de Nós	Variável v_1	Variável v_2	Variável v_3
1	24,92 koctetos	33,56 koctetos	43,4 koctetos
2	20,12 koctetos	28,76 koctetos	38,6 koctetos
4	17,72 koctetos	26,36 koctetos	36,2 koctetos
8	16,52 koctetos	25,16 koctetos	35 koctetos
10	16,28 koctetos	24,92 koctetos	34,76 koctetos
12	16,12 koctetos	24,76 koctetos	35,4 koctetos
16	15,92 koctetos	24,56 koctetos	35 koctetos
20	15,8 koctetos	24,92 koctetos	34,76 koctetos
24	15,72 koctetos	24,76 koctetos	35 koctetos
30	15,96 koctetos	24,6 koctetos	34,76 koctetos
40	15,8 koctetos	24,68 koctetos	34,76 koctetos
48	15,72 koctetos	24,56 koctetos	34,8 koctetos
60	15,8 koctetos	24,6 koctetos	34,76 koctetos
120	15,72 koctetos	24,52 koctetos	34,68 koctetos
240	15,68 koctetos	24,52 koctetos	34,64 koctetos

Capítulo 6

Conclusões

AGENTES móveis é um novo paradigma que vem sendo bastante pesquisado, tanto no meio acadêmico como no meio empresarial. Entre as principais vantagens dos agentes móveis, podem ser citadas as seguintes: a redução de mensagens veiculadas na rede por serem tratadas localmente; o processamento assíncrono que permite um desacoplamento do nó de origem do agente; a distribuição do processamento e a flexibilidade, uma vez que o comportamento do agente pode ser alterado a medida que ele executa suas tarefas nos nós visitados.

Os agentes móveis têm sido utilizados em diversas aplicações distribuídas, tais como: computação móvel, comércio eletrônico, recuperação de informações e gerenciamento de redes.

Este trabalho foca-se na aplicação de gerenciamento de redes, onde o modelo por agentes móveis se contrapõe ao modelo centralizado empregado pelo SNMP (*Simple Network Management Protocol*). Mais especificamente, o trabalho analisa o desempenho do gerenciamento por agentes móveis, comparando-o ao gerenciamento convencional, no que se refere a quantidade de octetos e tempo de resposta.

Foram implementados dois protótipos de uma aplicação que obtém variáveis da MIB-II (*Management Information Base - II*): um baseado em agentes móveis e outro somente baseado no SNMP. Os parâmetros obtidos destas implementações foram usados em simulações, em topologias com um grande número de nós, para se

observar o comportamento dos dois modelos quanto à escalabilidade. O protótipo baseado em agentes móveis utilizou a infra-estrutura de agentes chamada Mole. As duas implementações utilizaram os pacotes SNMP da AdventNet e *ucd-snmp*, e foram testadas em uma rede local *Ethernet*.

Os resultados indicam que os agentes móveis requerem um maior tempo de processamento, enquanto que o SNMP utiliza um maior número de mensagens relativas à estação de gerenciamento quando o número de elementos gerenciados ultrapassa um valor relacionado à sobrecarga dos vários pedidos do SNMP. A infra-estrutura de agentes torna a execução do código em Java muito mais lenta, principalmente devido a serialização/desserialização dos agentes, criação de *threads* e troca de mensagens internas. Para se ter uma idéia do desempenho da infra-estrutura, comparou-se o tempo de resposta do SNMP, diretamente sobre a máquina virtual Java, com o SNMP sobre a infra-estrutura Mole. O tempo de resposta aumentou de 86,9%, ao se utilizar a infra-estrutura Mole.

A topologia utilizada nas medições dos protótipos é bastante desfavorável ao agente móvel, uma vez que a grande disponibilidade de banda passante na rede *Ethernet* torna o tempo de transmissão de mensagens desprezível em relação aos tempos de processamento envolvidos. Nessa topologia, os agente móveis tornam-se bastante sensíveis à capacidade de processamento das máquinas. Para a configuração com maior poder de processamento utilizada nos testes, pode-se afirmar que pouco mais de 50% do tempo de resposta estão relacionados ao tempo de obtenção da variável na MIB e o tempo restante é relativo à infra-estrutura de agentes. A influência do poder de processamento das máquinas utilizadas, nessa configuração de rede local, também foi medida. Para o SNMP, não há praticamente diferença de desempenho ao se utilizar máquinas mais lentas, porém para o agente móvel, o tempo de resposta aumentou de 89,5% quando essas máquinas foram utilizadas.

Das medidas obtidas das implementações realizadas, observou-se que a infra-estrutura Mole, por ser de aplicação genérica, consome muito tempo de processamento. Uma otimização do mecanismo de migração disponibilizado pela Mole faria com que o desempenho dos agentes móveis melhorasse bastante. A aplicação de

gerenciamento por agentes móveis utilizou o protocolo SNMP para a obtenção das variáveis. Nesse caso, um agente fixo se comunicava com o agente SNMP, de modo a obter as variáveis e repassá-las ao agente móvel. Uma melhoria de desempenho seria obtida se o acesso à MIB fosse feito diretamente, através de um agente fixo que faria o papel de um agente SNMP, evitando a troca de mensagens do protocolo SNMP.

Simulações das duas implementações também foram realizadas no simulador de redes *ns*, de modo a se obter resultados em uma topologia com rede local de mais nós e em topologias genéricas maiores, semelhantes às encontradas na *Internet*.

Para uma topologia de rede local, a mesma topologia utilizada nas medições das implementações, pode-se afirmar que os modelos simulados reproduzem fielmente as implementações, exceto pela aproximação utilizada na obtenção do parâmetro do tempo restante do agente móvel.

Uma situação bastante comum é o gerenciamento remoto de uma rede local, no qual a estação de gerenciamento geralmente se conecta à rede através de uma rede de longa distância. Esta rede de longa distância é representada neste trabalho por um enlace de maior latência e menor banda passante. Para esta topologia, os resultados indicam que os agentes móveis são menos sensíveis à latência do enlace de gargalo, pois os pacotes do SNMP passam várias vezes por esse enlace. Além disso, os agentes móveis apresentam um melhor desempenho para latências típicas das WANs atuais. Quanto à banda passante, o agente móvel e o SNMP são bastante afetados quando a banda passante disponível é pequena.

O agentes móveis também sofrem maior influência das variáveis a serem obtidas, pelo fato que, para o SNMP, o número de octetos trocados entre a estação de gerenciamento e os elementos gerenciados é pequeno.

De modo a se avaliar o desempenho dos agentes móveis em uma topologia mais próxima das encontradas na *Internet*, topologias do tipo *transit-stubs* foram geradas através do gerador de topologias GT-ITM. Para essas topologias, o tempo de resposta diminui bastante ao se utilizar o protocolo UDP, pois o mecanismo *slow-start*

do TCP não permite o envio de uma só vez de todos os pacotes relativos ao agente.

Os agentes móveis possuem um melhor desempenho do que o do SNMP quando o número de elementos gerenciados encontra-se entre dois limites, o inferior e o superior, que são determinados, respectivamente, pelo número de mensagens que passam pelo *backbone* e pelo tamanho crescente do agente.

Observa-se que se o agente voltar para a estação de gerenciamento após coletar um número fixo de variáveis, o desempenho melhora pois o tamanho do agente é reduzido a cada viagem. É obtida uma diminuição no tempo de resposta de até 39% para as variáveis utilizadas. Resultados ainda melhores são obtidos quando o agente envia os dados à estação de gerenciamento ao invés de retornar à mesma. Nesse caso, a melhora no resultado é de até 42%.

Em relação ao número de octetos, o agente móvel possui um melhor desempenho do que o do SNMP quando o número de elementos gerenciados ultrapassa um valor relacionado à sobrecarga dos vários pedidos do SNMP.

De um modo geral, conclui-se que o uso de agentes móveis no gerenciamento de redes é uma boa solução no caso de haver diversas sub-redes a serem gerenciadas remotamente; principalmente se as ligações entre a estação de gerenciamento e as estações a serem gerenciadas forem de alto custo (pequena banda passante e grande latência).

Diversas direções podem ser tomadas neste trabalho, a partir dos resultados obtidos:

- colocação de inteligência nos agentes: os agentes móveis podem realizar um gerenciamento reativo com delegação de tarefas, verificando quando determinados limiares são ultrapassados. Com isso, um problema que pode vir a acontecer pode ser diagnosticado ou, na pior das hipóteses, uma anomalia pode ser cadastrada quando esta não puder ser relacionada a nenhum evento conhecido. Além disso, funções de gerenciamento podem ser enviadas a priori e utilizadas quando a rede estiver em situação de congestionamento. Os agentes inteligentes podem ainda avaliar e reagir a efeitos transientes;

- implementação do protótipo em uma infra-estrutura compatível com a padronização da FIPA: a plataforma Mole não está mais em desenvolvimento e a sua documentação técnica é deficiente. Por isso, a medição de tempos de transmissão e de serialização/desserialização do agente móvel não foi possível. Os novos comportamentos de agentes (retorno ou envio à estação de gerenciamento) simulados também podem ser implementados;

- estudo da viabilidade de utilização dos agentes móveis em outras áreas funcionais do modelo de gerenciamento OSI: na área de gerenciamento de detecção de falhas, o uso de agentes móveis e inteligentes deve facilitar a detecção, o isolamento e a correção de funcionamentos anormais do sistema. Na área de gerenciamento de configuração e de nomes, os agentes móveis podem atualizar de maneira simples e eficiente novas versões de programas de gerenciamento.

Referências Bibliográficas

[1] Cockayne, W. R., and Zyda, M. *Mobile Agents*, 1ª ed. Manning Publications Co., 1998. ISBN 1884777368.

[2] Chess, D., Grosof, B., Harrison, C., Levine, D., Parris, C., and Tsudik, G. Itinerant agents for mobile computing. Relatório técnico, IBM T. J. Watson Research Center/IBM Zurich Research Center, Nova Iorque/Rueschlikom, EUA/Suíça, março de 1995.

[3] Baldi, M., Gai, S., and Picco, G. P. Exploiting code mobility in decentralized and flexible network management. In *First International Workshop on Mobile Agents* (Berlim, Alemanha, abril de 1997), pp. 13–26.

[4] Carzaniga, A., Picco, G. P., and Vigna, G. Designing distributed applications with mobile code paradigms. In *19th International Conference on Software Engineering (ICSE'97)* (Boston, EUA, maio de 1997), pp. 22–32.

[5] Baumann, J., Hohl, F., Straβer, M., and Rothermel, K. Mole - concepts of a mobile agent system. *World Wide Web 1*, 3 (1998), 123–137. Baltzer Science Publishers.

[6] Stallings, W. *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2.* Addison-Wesley, 1999. ISBN 0201485346.

[7] Stallings, W. SNMP and SNMPv2: The infrastructure for network management. *IEEE Communications Magazine 36*, 3 (março de 1998), 37–43. The Institute of Electrical and Electronics Engineers (IEEE).

[8] Yemini, Y. The OSI network management model. *IEEE Communications Magazine 31*, 5 (maio de 1993), 20–29. The Institute of Electrical and Electronics Engineers (IEEE).

[9] Bohoris, C., Pavlou, G., and Cruickshank, H. Using mobile agents for network performance management. In *IEEE/IFIP Network Operations and Management Symposium (NOMS'00)* (Honolulu, Havaí, abril de 2000), pp. 637–652.

[10] Pham, V. A., and Karmouch, A. Mobile software agents: An overview. *IEEE Communications Magazine 36*, 7 (julho de 1998), 26–37. The Institute of Electrical and Electronics Engineers (IEEE).

[11] Magedanz, T., and Eckardt, T. Mobile software agents: A new paradigm for telecommunications management. In *IEEE/IFIP Network Operations and Management Symposium (NOMS'96)* (Kyoto, Japão, abril de 1996).

[12] Magedanz, T., Rothermel, K., and Krause, S. Intelligent agents: An emerging technology for next generation telecommunications? In *IEEE INFOCOM'96* (São Francisco, EUA, março de 1996).

[13] Cheikhrouhou, M., Conti, P., and Labetoulle, J. Intelligent agents in network management: a state-of-the-art. *Networking and Information Systems Journal 1*, 1 (1998), 9–38. HERMES Science Publications.

[14] Goldszmidt, G., and Yemini, Y. Delegated agents for network management. *IEEE Communications Magazine 36*, 3 (março de 1998), 66–70. The Institute of Electrical and Electronics Engineers (IEEE).

[15] Sahai, A., and Morin, C. Enabling a Mobile Network Manager (MNM) through mobile agents. In *Second International Workshop on Mobile Agents* (Stuttgart, Alemanha, setembro de 1998).

[16] Bieszczad, A., Pagurek, B., and White, T. Mobile agents for network management. *IEEE Communications Surveys 1*, 1 (setembro de 1998). The Institute of Electrical and Electronics Engineers (IEEE).

[17] Cheikhrouhou, M., Conti, P., Labetoulle, J., and Marcus, K. Intelligent agents for network management: Fault detection experiment. In *Sixth IFIP/IEEE International Symposium on Integrated Network Management (IM'99)* (Boston, EUA, maio de 1999).

[18] Ku, H., Luderer, G. W., and Subbiah, B. An intelligent mobile agent framework for distributed network management. In *IEEE Global Telecommunications Conference (GLOBECOM'97)* (Phoenix, EUA, novembro de 1997).

[19] Puliafito, A., Tomarchio, O., and Vita, L. MAP: Design and implementation of a Mobile Agent Plataform. *Journal of Systems Architecture 46*, 2 (janeiro de 2000), 145–162. Elsevier Science.

[20] Knight, G., and Hazemi, R. Mobile agent based management in the INSERT project. *Journal of Network and Systems Management 7*, 3 (setembro de 1999). Kluwer Academic/Plenum Publishers.

[21] Oliveira, J. L., and Lopes, R. P. Distributed management based on mobile agents. In *First International Workshop on Mobile Agents for Telecommunication Applications (MATA'99)* (Ottawa, Canadá, outubro de 1999), pp. 243–258.

[22] Goldszmidt, G., and Yemini, Y. Distributed management by delegation. In *The 15th International Conference on Distributed Computing Systems* (Vancouver, Canadá, junho de 1995).

[23] Schönwälder, J. Network management by delegation - from research prototypes towards standards. *Computer Networks 29*, 15 (novembro de 1997), 1843–1852. Elsevier Science.

[24] Kooijman, R. Divide and conquer in network management using event-driven network area agents. Relatório técnico, Technical University of Delft, Holanda, maio de 1995.

[25] de Castro e Silva, J. L., da Silva, A. N., and de Souza, J. N. Modelos de implementação de gerência por delegação em sistemas de redes heterogêne-

as, baseados nos conceitos de agentes móveis e objetos remotos cooperantes. In *XVII Simpósio Brasileiro de Redes de Computadores (SBRC'99)* (Salvador, Brasil, maio de 1999), pp. 579–580.

[26] Sahai, A., and Morin, C. Towards distributed and dynamic network management. In *IEEE/IFIP Network Operations and Management Symposium (NOMS'98)* (New Orleans, EUA, fevereiro de 1998).

[27] Sahai, A., Morin, C., and Billiart, S. Intelligent agents for a Mobile Network Manager (MNM). In *IFIP/IEEE International Conference on Intelligent Networks and Intelligence in Networks (2IN'97)* (Paris, França, setembro de 1997).

[28] Pagurek, B., Wang, Y., and White, T. Integration of mobile agent with SNMP: Why and how. In *IEEE/IFIP Network Operations and Management Symposium (NOMS'00)* (Honolulu, Havaí, abril de 2000).

[29] Puliafito, A., and Tomarchio, O. Using mobile agents to implement flexible network management strategies. *Computer Communications 23*, 8 (abril de 2000), 708–719. Elsevier Science.

[30] Silva, L. M., Simões, P., Soares, G., Martins, P., Batista, V., Renato, C., Almeida, L., and Stohr, N. JAMES: A platform of mobile agents for the management of telecommunication networks. In *3rd International Workshop on Intelligent Agents for Telecommunication Applications (IATA'99)* (Estocolmo, Suécia, agosto de 1999).

[31] Schramm, C., Bieszczad, A., and Pagurek, B. Application-oriented network modeling with mobile agents. In *IEEE/IFIP Network Operations and Management Symposium (NOMS'98)* (New Orleans, EUA, fevereiro de 1998).

[32] White, T., Pagurek, B., and Bieszczad, A. Network modeling for management applications using intelligent mobile agents. *Journal of Network and Systems Management 7*, 3 (setembro de 1999). Kluwer Academic/Plenum Publishers.

[33] White, T., Pagurek, B., Bieszczad, A., Sugar, G., and Tran, X. Intelligent network modeling using mobile agents. In *IEEE Global Telecommunications Conference (GLOBECOM'98)* (Sidnei, Austrália, novembro de 1998), pp. 1082–1087.

[34] Silva, L. M., Soares, G., Martins, P., Batista, V., and Santos, L. Comparing the performance of mobile agent systems: a study of benchmarking. *Computer Communications 23*, 8 (abril de 2000), 769–778. Elsevier Science.

[35] Silva, L. M., Soares, G., Martins, P., Batista, V., and Santos, L. The performance of mobile agent platforms. In *First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents (ASA/MA'99)* (Califórnia, EUA, outubro de 1999), pp. 270–271.

[36] Koch, F. L., and Westphall, C. B. Gerenciamento de redes de computadores utilizando inteligência artificial distribuída - uma abordagem operacional. In *XVI Simpósio Brasileiro de Redes de Computadores (SBRC'98)* (Rio de Janeiro, Brasil, maio de 1998), pp. 144–158.

[37] Artola, E. S., and Tarouco, L. M. R. Um sistema especialista para gerência pró-ativa remota. In *XIV Simpósio Brasileiro de Redes de Computadores (SBRC'96)* (Fortaleza, Brasil, maio de 1996), pp. 118–139.

[38] Nascimento, A., Franklin, M., and Oliveira, M. Desenvolvendo agentes inteligentes para a gerência pró-ativa de redes ATM. In *XVII Simpósio Brasileiro de Redes de Computadores (SBRC'99)* (Salvador, Brasil, maio de 1999), pp. 681–696.

[39] Carvalho, E., Belchior, A. D., and de Souza, J. N. Gerenciamento pró-ativo distribuído baseado em lógica difusa. In *XVII Simpósio Brasileiro de Redes de Computadores (SBRC'99)* (Salvador, Brasil, maio de 1999), pp. 649–664.

[40] Gürer, D., Lakshminarayan, V., and Sastry, A. An intelligent-agent-based architecture for the management of heterogeneous networks. In *9th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM'98)* (Delaware, EUA, outubro de 1998).

[41] Hood, C. S., and Ji, C. Intelligent agents for proactive fault detection. *IEEE Internet Computing 2*, 2 (março de 1998), 65–72. The Institute of Electrical and Electronics Engineers (IEEE).

[42] da Rocha, M. A., Fernandez, L. F. N., and Westphall, C. B. Gerência pró-ativa de redes de computadores usando agentes e técnicas de inteligência artificial. In *XIV Simpósio Brasileiro de Redes de Computadores (SBRC'96)* (Fortaleza, Brasil, maio de 1996), pp. 97–117.

[43] McCloghrie, K., and Rose, M. Management information base for network management of TCP/IP-based internets: MIB-II. RFC 1213, março de 1991.

[44] Puliafito, A., Tomarchio, O., and Vita, L. A Java-based distributed network management architecture. In *3rd International Conference on Computer Science and Informatics (CSI'97)* (Durham, EUA, março de 1997).

[45] Puliafito, A., and Tomarchio, O. Advanced network management functionalities through the use of mobile software agents. In *3rd International Workshop on Intelligent Agents for Telecommunication Applications (IATA'99)* (Estocolmo, Suécia, agosto de 1999).

[46] Arnold, K., Gosling, J., and Holmes, D. *The Java Programming Language*. Addison-Wesley, 2000. ISBN 0201704331.

[47] Haggerty, P., and Seetharaman, K. The benefits of CORBA-based network management. *Communications of the ACM 41*, 10 (outubro de 1998), 73–79. Association for Computing Machinery (ACM).

[48] de Queiroz, J. A. G., and Madeira, E. R. M. Facilidade de monitorização assíncrona em um ambiente de gerência CORBA. In *II Seminário Franco-*

Brasileiro em Sistemas Informáticos Distribuídos (Fortaleza, Brasil, novembro de 1997), pp. 135–146.

[49] Rodriguez, N., Catunda, M., and Ierusalimschy, R. Extensão dinâmica de agentes CORBA. In *XVI Simpósio Brasileiro de Redes de Computadores (SBRC'98)* (Rio de Janeiro, Brasil, maio de 1998), p. 768.

[50] de Moura, A. L., Ishikawa, E., Lima, M. E., and Rodriguez, N. Aplicações de gerência extensíveis. In *XVI Simpósio Brasileiro de Redes de Computadores (SBRC'98)* (Rio de Janeiro, Brasil, maio de 1998), pp. 125–143.

[51] Straβer, M., and Schwehm, M. Performance model for mobile agent systems. In *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'97)* (Las Vegas, EUA, 1997), vol. 2, pp. 1132–1140.

[52] Ismail, L., and Hagimont, D. A performace evaluation of the mobile agent paradigm. In *Int. Conf. on Object-Oriented Programming, Systems and Applications (OOPSLA'99)* (Denver, EUA, novembro de 1999), pp. 306–313.

[53] Zapf, M., Herrmann, K., and Geihs, K. Decentralized SNMP management with mobile agents. In *Sixth IFIP/IEEE International Symposium on Integrated Network Management (IM'99)* (Boston, EUA, maio de 1999), pp. 623–635.

[54] El-Darieby, M., and Bieszczad, A. Intelligent mobile agents: Towards network fault management automation. In *Sixth IFIP/IEEE International Symposium on Integrated Network Management (IM'99)* (Boston, EUA, maio de 1999), pp. 610–622.

[55] Outtagarts, A., Kadoch, M., and Soulhi, S. Client-server and mobile agent: Performances comparative study in the management of MIBS. In *First International Workshop on Mobile Agents for Telecommunication Applications (MATA'99)* (Ottawa, Canadá, outubro de 1999), pp. 69–81.

[56] da Silva Costa, T. F. Avaliação analítica do uso de agentes móveis na gerência de redes. Tese de Mestrado, Ciência da Computação, Universidade Federal de Santa Catarina, outubro de 1999.

[57] Puliafito, A., Riccobene, S., and Scarpa, M. An analytical comparison of the client-server, remote evaluation and mobile agent paradigms. In *First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents (ASA/MA'99)* (Califórnia, EUA, outubro de 1999).

[58] Baldi, M., and Picco, G. P. Evaluating the tradeoffs of mobile code design paradigms in network management applications. In *20th International Conference on Software Engineering (ICSE'98)* (Kyoto, Japão, abril de 1998), pp. 146–155.

[59] Liotta, A., Knight, G., and Pavlou, G. On the performance and scalability of decentralised monitoring using mobile agents. In *10th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM'99)* (Zurich, Switzerland, outubro de 1999), pp. 3–18.

[60] Gavalas, D., Greenwood, D., Ghanbari, M., and O'Mahony, M. Using mobile agents for distributed network performance management. In *3rd International Workshop on Intelligent Agents for Telecommunication Applications (IATA'99)* (Estocolmo, Suécia, agosto de 1999).

[61] Fuggetta, A., Picco, G. P., and Vigna, G. Understanding code mobility. *IEEE Transactions on Software Engineering 24*, 5 (maio de 1998), 342–361. The Institute of Electrical and Electronics Engineers (IEEE).

[62] Liotta, A., Knight, G., and Pavlou, G. Modelling network and system monitoring over the internet with mobile agents. In *IEEE/IFIP Network Operations and Management Symposium (NOMS'98)* (New Orleans, EUA, fevereiro de 1998), vol. 2, pp. 300–312.

[63] Bohoris, C., Liotta, A., and Pavlou, G. Software agent constrained mobility for network performance monitoring. In *6th IFIP Conference on*

Intelligence in Networks (SmartNet 2000) (Viena, Áustria, setembro de 2000), pp. 367–387.

[64] Bohoris, C., Liotta, A., and Pavlou, G. Evaluation of constrained mobility for programmability in network management. In *Services Management in Intelligent Networks, Proceedings of the 11th IEEE/IFIP International Workshop on Distributed Systems: Operations and Management (DSOM'00)* (Austin, Texas, EUA, dezembro de 2000), pp. 243–257.

[65] Gavalas, D., Greenwood, D., Ghanbari, M., and O'Mahony, M. Complimentary polling modes for network performance management employing mobile agents. In *IEEE Global Telecommunications Conference (GLOBECOM'99)* (Rio de Janeiro, Brasil, dezembro de 1999), pp. 401–405.

[66] Gavalas, D., Greenwood, D., Ghanbari, M., and O'Mahony, M. Advanced network monitoring applications based on mobile/intelligent agent technology. *Computer Communications 23*, 8 (abril de 2000), 720–730. Elsevier Science.

[67] Gavalas, D., Ghanbari, M., O'Mahony, M., and Greenwood, D. Enabling mobile agent technology for intelligent bulk management data. In *IEEE/IFIP Network Operations and Management Symposium (NOMS'00)* (Honolulu, Havaí, abril de 2000).

[68] Sahai, A., and Morin, C. Mobile agents enhanced thin client approach to network management. In *IEEE Singapore International Conference on Networks (SICON'98)* (Kent Ridge, Singapura, junho de 1998).

[69] Rubinstein, M. G., Duarte, O. C. M. B., and Pujolle, G. Evaluating the network peformance management based on mobile agents. *Lecture Notes in Computer Science 1931* (setembro de 2000), 95–102. ISSN 03029743, ISBN 3540410694, Springer-Verlag.

[70] Rubinstein, M. G., and Duarte, O. C. M. B. Análise da eficiência de agentes móveis no gerenciamento de redes. In *XIX Congresso Nacional da*

Sociedade Brasileira de Computação, XXVI Seminário Integrado de Software e Hardware (SEMISH'99) (Rio de Janeiro, Brasil, julho de 1999), vol. 1, pp. 167–178.

[71] Rubinstein, M. G., and Duarte, O. C. M. B. Evaluating the performance of mobile agents in network management. In *IEEE Global Telecommunications Conference (GLOBECOM'99)* (Rio de Janeiro, Brasil, dezembro de 1999), pp. 386–390.

[72] Rubinstein, M. G., and Duarte, O. C. M. B. Analyzing mobile agent scalability in network management. In *IEEE Latin American Network Operations and Management Symposium (LANOMS'99)* (Rio de Janeiro, Brasil, dezembro de 1999), pp. 64–74.

[73] Rubinstein, M. G., and Duarte, O. C. M. B. Evaluating tradeoffs of mobile agents in network management. *Networking and Information Systems Journal 2*, 2 (1999), 237–252. HERMES Science Publications.

[74] Rubinstein, M. G., Duarte, O. C. M. B., and Pujolle, G. Improving management performance by using mobile agents. In *ACM Fourth International Conference on Autonomous Agents (Agents 2000)* (Barcelona, Espanha, junho de 2000), pp. 165–166.

[75] Rubinstein, M. G., Duarte, O. C. M. B., and Pujolle, G. Using mobile agent strategies for reducing the response time in network management. In *16th IFIP World Computer Congress, ICCT2000* (Beijing, China, agosto de 2000), pp. 278–281.

[76] Rubinstein, M. G., Duarte, O. C. M. B., and Pujolle, G. *Managing QoS in Multimedia Networks and Services.* ISBN 0792379624. Kluwer Academic Publishers, setembro de 2000, ch. 18: Reducing the Response Time in Network Management by Using Multiple Mobile Agents.

[77] Gilbert, D., and Janca, P. IBM intelligent agents. IBM Corporation, EUA, http://www.networking.ibm.com/iag/iagwp1.html.

[78] Lingnau, A., Drobnik, O., and Dömel, P. An HTTP-based infrastructure for mobile agents. In *Fourth International WWW Conference* (Boston/Massachusets, EUA, dezembro de 1995).

[79] Lingnau, A., and Drobnik, O. An infrastructure for mobile agents: requirements and architecture. In *13th DIS Workshop* (Orlando, Flórida, EUA, setembro de 1995).

[80] Kiniry, J., and Zimmerman, D. A hands-on look at Java mobile agents. *IEEE Internet Computing 1*, 4 (julho de 1997), 21–30.

[81] Harrison, C. G., Chess, D. M., and Kershenbaum, A. Mobile agents: are they a good idea? Relatório técnico, IBM T. J. Watson Research Center, Nova Iorque, EUA, março de 1995.

[82] Breugst, M., and Magedanz, T. On the usage of standard mobile agent platforms in telecommunication environments. In *5th International Conference on Intelligence in Services and Networks (IS&N'98)* (Antuérpia, Bélgica, maio de 1998), pp. 275–286.

[83] Breugst, M., and Magedanz, T. Mobile agents - enabling technology for active intelligent network implementation. *IEEE Network 12*, 3 (maio de 1998). The Institute of Electrical and Electronics Engineers (IEEE).

[84] Johansen, D., van Renesse, R., and Schneider, F. B. An introduction to the TACOMA distributed system. Relatório técnico, University of Tromsϕ/Cornell University, Tromsϕ/Ithaca, Noruega/EUA, junho de 1995.

[85] Gray, R. S. Agent Tcl: a transportable agent system. In *CIKM Workshop on Intelligent Information Agents, Fouth International Conference on Information and Knowledge Management (CIKM'95)* (Baltimore/Maryland, EUA, 1995).

[86] Odyssey. General Magic, http://www.genmagic.com/agents, 1997.

[87] White, J. Mobile agents white paper. Relatório técnico, General Magic, 1996.

[88] Programming mobile agents in Java. IBM Corporation, Japão, http://www.trl.ibm.co.jp/aglets, 1996.

[89] Chiariglione, L. Helping agent technologies get across to the market place. Foundation for Intelligent Physical Agents (FIPA), http://drogo.cselt.stet.it/fipa/general_agents.htm.

[90] The Agent Society. http://www.agent.org.

[91] The Mobile Agent Facility Specification. http://www.omg.org.

[92] FIPA: The Foundation for Intelligent Physical Agents. Foundation for Intelligent Physical Agents, http://www.fipa.org.

[93] FIPA inform: The Newsletter of the Foundation for Intelligent Physical Agents, dezembro de 2000.

[94] The ZEUS Agent Building Toolkit. British Telecommunications, http://www.labs.bt.com/projects/agents/zeus.

[95] Java Agent DEvelopment framework. http://sharon.cselt.it/projects/jade/.

[96] FIPA-OS. http://fipa-os.sourceforge.net/.

[97] Peine, H., and Stolpmann, T. The architecture of the Ara platform for mobile agents. In *1st International Workshop on Mobile Agents* (Berlim, Alemanha, abril de 1997), pp. 50–61.

[98] Abdalla, M., Cirne, W., Franklin, L., and Tabbara, A. Security issues in agent based computing. In *XV Simpósio Brasileiro de Redes de Computadores (SBRC'97)* (São Carlos, SP, Brasil, maio de 1997), pp. 231–244.

[99] de Oliveira, P. C., and Cardozo, E. Mobile agent-based systems: an alternative paradigm for distributed systems development. In *XV Simpósio Brasileiro de Redes de Computadores (SBRC'97)* (São Carlos, SP, Brasil, maio de 1997), pp. 508–524.

[100] Rubinstein, M. G., and Duarte, O. C. M. B. Service location for mobile agent systems. In *IEEE/SBT International Telecommunications Symposium (ITS'98)* (São Paulo, Brasil, agosto de 1998), pp. 623–626.

[101] Meyer, K., Erlinger, M., Betser, J., Sunshine, C., Goldszmidt, G., and Yemini, Y. Decentralizing control and intelligence in network management. In *4th International Symposium on Integrated Network Management* (Santa Bárbara, EUA, maio de 1995).

[102] Kahani, M., and Beadle, H. W. P. Decentralised approaches for network management. *Computer Communications Review 27*, 3 (julho de 1997), 36–47. Association for Computing Machinery (ACM).

[103] Case, J., Fedor, M., Schoffstall, M., and Davin, J. A Simple Network Management Protocol (SNMP). RFC 1157, maio de 1990.

[104] Stallings, W. *SNMP, SNMPv2, and RMON: Pratical Network Management*. Addison-Wesley, 1996. ISBN 0201634791.

[105] Sluman, C. A tutorial on OSI management. *Computer Networks 17*, 4&5 (outubro de 1989), 270–278. Elsevier Science.

[106] Rose, M., and McCloghrie, K. Structure and identification of management information for TCP/IP-based internets. RFC 1155, maio de 1990.

[107] McCloghrie, K., and Rose, M. Management information base for network management of TCP/IP-based internets. RFC 1156, maio de 1990.

[108] Case, J., McCloghrie, K., Rose, M., and Waldbusser, S. Introduction to version 2 of the Internet-standard network management framework. RFC 1441, abril de 1993.

[109] Stallings, W. SNMPv3: A security enhancement for SNMP. *IEEE Communications Surveys 1*, 1 (setembro de 1998). The Institute of Electrical and Electronics Engineers (IEEE).

[110] Waldbusser, S. Remote network monitoring management information base. RFC 1757, fevereiro de 1995.

[111] Mole 3.0 user manual v1.0. Relatório técnico, Universidade College London, Reino Unido, novembro de 1998.

[112] The mole cookbook or how to program a Mole agent. Relatório técnico, Universidade de Stuttgart, Alemanha, outubro de 1998.

[113] The home of Mole. Grupo de Sistemas Distribuídos, Universidade de Stuttgart, http://mole.informatik.uni-stuttgart.de/.

[114] Advent Network Management Inc. AdventNet SNMP release 2.0. http://www.adventnet.com, 1998.

[115] Projeto UCD-SNMP. http://ucd-snmp.ucdavis.edu/.

[116] Projeto NET-SNMP. http://net-snmp.sourceforge.net/.

[117] Fall, K., and Varadhan, K. NS Notes and Documentation. Relatório técnico, The VINT Project, janeiro de 1999.

[118] Zegura, E. W., Calvert, K. L., and Donahoo, M. J. A quantitative comparison of graph-based models for internet topology. *IEEE/ACM Transactions on Networking 5*, 6 (dezembro de 1997), 770–783. The Institute of Electrical and Electronics Engineers (IEEE) e Association for Computing Machinery (ACM).

Apêndice A

O Simulador de Redes *ns*

O *NS* (*Network Simulator*) [117] é um simulador de serviços e de protocolos de rede que vem sendo utilizado principalmente para a *Internet*. Esse simulador encontra-se em desenvolvimento dentro do projeto *Virtual InterNet Testbed* (VINT), uma colaboração entre a Universidade da Califórnia em Berkeley, o *Lawrence Berkeley National Laboratory* (LBL), o *Information Sciences Institute* (ISI) da Universidade da Califórnia do Sul (USC) e o laboratório Xerox PARC.

O *ns* utiliza as linguagens C++ e OTcl (*Object Tool Command Language*), sendo que o seu núcleo é implementado em C++, para permitir um melhor desempenho, e as simulações a serem executadas pelo *ns* são configuradas através de *scripts* OTcl. Nesses *scripts* do usuário são descritos, no mínimo, a topologia, o protocolo e as aplicações a serem simuladas.

As topologias são criadas no *ns* através de nós e da conexão desses nós através de enlaces com características específicas de banda passante e de atraso. As topologias podem ser geradas manualmente ou através do gerador de topologias GT-ITM (*Georgia Tech - Internetwork Topology Models*). O GT-ITM é apresentado no Apêndice B.

A estrutura dos nós no *ns* é apresentada na Figura A.1. Essa estrutura é composta de agentes, um ponto de entrada no nó, um classificador de endereços e um classificador de portas. Os agentes são entidades produtoras ou consumidoras de pa-

cotes e implementam determinados tipos de protocolos. Um pacote gerado por um agente é entregue ao nó ao qual o agente está ligado, através do ponto de entrada, que também recebe pacotes cujo destino é o próprio nó. Após passar pelo ponto de entrada, o pacote é recebido pelo classificador de endereços, que verifica se o pacote deve ser entregue a um agente pertencente ao nó ou deve ser transmitido para um enlace de saída. Caso o pacote seja destinado a um agente do próprio nó, o pacote é então repassado ao classificador de porta que, de acordo com o endereço de destino, entrega o pacote ao respectivo agente.

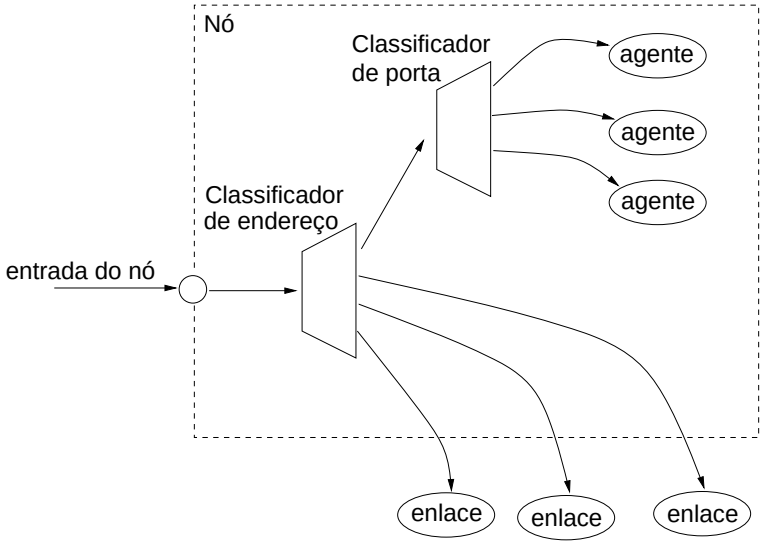

Figura A.1: A estrutura dos nós no *ns*.

Existem vários agentes já implementados no *ns*, tais como o UDP e diversos tipos de TCP (Reno, Vega, NewReno, Tahoe e outros). Neste trabalho, os agentes *UDP*, *TCP-Reno* e *TCP-Sink* foram utilizados. O agente produtor de pacotes *TCP-Reno* foi escolhido por ser atualmente de grande utilização na *Internet*. O agente *TCP-Sink* é o receptor de pacotes dos agentes produtores. Novos agentes foram criados e outros alterados para que as simulações fossem realizadas. Um agente chamado *UDP-Sink* foi criado de modo que o UDP recebesse efetivamente um pacote, pois pela implementação original, não havia uma função de recebimento e de envio de um pacote UDP para a respectiva aplicação. Alterações que permitiram a remontagem de pacotes fragmentados foram feitas no *UDP*, no *TCP-Reno* e no *TCP-Sink*.

As aplicações são executadas em cima dos agentes de transporte. Existem al-

gumas aplicações já implementadas no *ns*, tais como FTP (*File Transfer Protocol*), telnet, distribuição *on-off* exponencial, distribuição *on-off* pareto, tráfego do tipo CBR (*Constant Bit Rate*) e tráfego de traços, que é gerado a partir de um arquivo de entrada. Diversas aplicações foram criadas neste trabalho e são apresentadas a seguir.

A.1 Detalhes das Simulações

As simulações realizadas estão baseadas, de um modo geral, em um modelo que depende da topologia e do protocolo utilizados. Serão descritos os detalhes relativos a uma topologia genérica do tipo *transit-stub* e ao protocolo TCP.

Para o SNMP, várias instâncias de *gerente* e de *agente* foram criadas, cada uma responsável pelo envio e o recebimento de um pacote de pedido de uma variável. O *gerente* envia o pedido de uma variável para um *agente* que recebe o pedido, consulta uma tabela que contém um identificador da variável a ser obtida e o seu respectivo tamanho em octetos e chama um procedimento que cria uma aplicação chamada *agente_ envia*. Esta aplicação envia a resposta do pedido após o tempo da MIB (Seção 5.1) a uma outra aplicação que se chama *gerente_ recebe*. Ao receber a resposta, o *gerente_ recebe* dispara a execução do próximo *gerente*. As aplicações *agente_ envia* e *gerente_ recebe* foram criadas para permitir o envio dos reconhecimentos do protocolo TCP, pois só é permitido o uso de uma aplicação por agente do *ns*; logo, uma aplicação não pode enviar reconhecimentos relativos a um agente móvel que chegou e ainda enviar o agente móvel para o próximo nó.

Nas simulações do agente móvel, em cada estação de gerenciamento é criada uma aplicação *am-sink*. O *am* é transferido para a estação e recebido pela *am-sink* que consulta a tabela de variáveis, aumenta o tamanho do agente móvel e chama um procedimento que cria novos agentes e uma aplicação *am* que será iniciada após o tempo correspondente à soma do tempo da MIB e do tempo restante (Seção 5.1) e migrará para o próximo nó.

Apêndice B

O Gerador de Topologias GT-ITM

O GERADOR de topologias GT-ITM (*Georgia Tech - Internetwork Topology Model*), criado no *College of Computing* do *Georgia Institute of Technology*, é utilizado para a geração de diversos tipos de topologias complexas. Essas topologias são baseadas em um modelo padrão de geração de grafos que distribui os vértices em posições aleatórias de um plano, havendo uma probabilidade p de uma aresta ser adicionada entre um par de vértices. Tal modelo de geração de grafos, se aplicado sozinho, não reflete a estrutura das redes reais, sendo por isso, propostos outros modelos que alteram a função de probabilidade de um vértice para representarem de forma mais fiel a estrutura dessas redes [118]. Em relação a esses novos modelos, o GT-ITM pode gerar topologias planas, hierárquicas de nível n e *transit-stub*.

As topologias *transit-stub* são grafos hierárquicos construídos através da composição de domínios *transit* e *stub*. Primeiramente, é construído um grafo aleatório conexo com cada nó representando um domínio *transit* completo. Em seguida, cada nó é substituído por outro grafo aleatório conexo que representa o *backbone* de um domínio *transit*. Para cada nó desse domínio é gerado um número de grafos aleatórios conexos que representam domínios do tipo *stub* conectados ao nó. Por último, são feitas ligações adicionais entre nós de um domínio *transit* e um domínio *stub*, ou de dois domínios *stub* diferentes.

O *script* a seguir apresenta os dados utilizados para a geração das topologias

transit-stub simuladas. A primeira linha após os comentários indica, respectivamente, o tipo de grafo a ser gerado, o número de grafos e a semente a ser utilizada para a geração dos números aleatórios. A linha *1 0 0 1.0* significa que existirá somente um domínio *stub* conectado a um nó do domínio *transit*, não devendo haver arestas adicionais *transit-stub* ou *stub-stub*. A linha com os parâmetros *1 20 3 1.0* fornece os valores para a geração dos domínios *transit*. O primeiro parâmetro informa a existência de somente um domínio *transit*. Os outros parâmetros, comuns para as três últimas linhas, indicam respectivamente a escala, o modelo de probabilidade de arestas a ser utilizado e um parâmetro desse modelo. A escala corresponde ao número de subdivisões em cada nível da hierarquia. Este número serve para o cálculo do tamanho das arestas. O parâmetro para o modelo de probabilidades de arestas indica a probabilidade de haver um enlace conectando dois nós de um mesmo domínio. As duas linhas seguintes se referem às características dos domínios do tipo *transit* e do tipo *stub*. Desse modo, a linha *16 20 3 0.6* indica que cada domínio *transit* possuirá, em média, 16 nós, existindo uma probabilidade de 0,6% de dois nós estarem conectados. A linha *16 10 3 0.6* indica que todo domínio *stub* terá, em média, 16 nós, havendo uma probabilidade de 0,6% de dois nós estarem conectados.

```
## Comments :
## <#method keyword> <#number of graphs> [<#initial seed>]
## <#stubs/xit> <#t-s edges> <#s-s edges>
## <#n> <#scale> <#edgemethod> <#alpha> [<#beta>] [<#gamma>]
## number of nodes = 1*16* (1 + 1*16) = 272
ts 3 47
1 0 0 1.0
1 20 3 1.0
16 20 3 0.6
16 10 3 0.6
```