PLATAFORMA DINÂMICA DE GESTÃO DE REDES

Paulo Pereira[1], Jorge Sousa[2], Pedro Teixeira[2]
Instituto Superior Técnico
INESC - R. Alves Redol, 9. 1000 LISBOA
Tel.: +351-1-3100345, Fax: +351-1-3145843.
E-mail: prbp@inesc.pt, jdso@mariel.inesc.pt,
pedro.g.teixeira@inesc.pt

Paulo Pinto[3]
Faculdade de Ciências e Tecnologia.
Universidade Nova de Lisboa.
UNINOVA, Quinta da Torre. 2825 Monte da Caparica.
Tel: +351-1-2948515, Fax: +351-1-2941253.
E-mail: pfp@uninova.pt

Sumário
As redes de computadores têm crescido em número, dimensão, complexidade e variedade de equipamentos. Para controlar tais redes apresenta-se uma plataforma dinâmica de gestão baseada num sistema hierárquico distribuído de gestores intermédios onde se pode delegar *scripts* de gestão. Permite-se assim automatizar as tarefas de gestão, descentralizá-las, e fazer face ao dinamismo existente nas redes.

1. INTRODUÇÃO

Os sistemas baseados em computadores têm vindo a ser progressivamente interligados em rede, e as aplicações tomam partido da distribuição para oferecer mais e melhores serviços, tornando as redes e os recursos associados indispensáveis. À medida que as redes e os sistemas distribuídos aumentam de dimensão e importância, torna-se cada vez mais necessário recorrer à gestão de redes para garantir que a rede funciona correctamente e providencia os serviços esperados pelos utilizadores.

A gestão de redes tem como funções supervisionar e controlar as redes e os serviços por elas oferecidos, bem como planear modificações na rede. Para serem geridos, os equipamentos têm uma base de dados de informação de gestão (MIB) mantida por um agente de gestão, à qual as aplicações de gestão acedem através de um protocolo de gestão de redes (SNMP na comunidade Internet, e CMIP na comunidade OSI).

Este modelo gestor-agente é muito simples, sendo adequado para a gestão de um pequeno número de equipamentos. Para redes de grande dimensão ou complexidade, geograficamente dispersas, ou que necessitem de muito tráfego de gestão, as plataformas actuais de gestão de redes apresentam algumas limitações que se pretende colmatar com a plataforma desenvolvida. Assim, foi objectivo deste trabalho desenvolver uma plataforma de gestão de redes que permitisse suportar redes de dimensão e complexidade arbitrária, além de automatizar e descentralizar as tarefas de gestão.

Na secção 2 analisam-se alguns conceitos propostos pela comunidade científica para lidar com os problemas enunciados, comparando-se com a forma como foram utilizados neste trabalho. Na secção 3 apresenta-se a arquitectura da plataforma dinâmica de gestão de redes desenvolvida, e descreve-se o funcionamento dos vários componentes. Na secção 4 apresenta-se a aplicação de gestão que permite monitorizar e controlar a plataforma de gestão, oferecendo a interface com o utilizador. Na secção 5 apresentam-se alguns exemplos de *scripts* de gestão que permitem automatizar algumas funções de gestão. Finalmente, na secção 6 apresentam-se algumas conclusões e propostas de trabalho futuro.

2. CONCEITOS

O primeiro conceito a analisar é a **arquitectura** de gestão. A arquitectura mais simples é a centralizada com um gestor para muitos agentes, que sofre de problemas de não ser escalável, ser vulnerável a falhas, ser estática, necessitar de muito tráfego de gestão, ter falta de autonomia dos agentes para resolver qualquer situação e de ter uma manutenção cara. Destas desvantagens, que se tornam mais significativas com o crescimento da complexidade das redes, conclui-se haver uma crescente necessidade de soluções de gestão não centralizadas [1]. Dentro das soluções não centralizadas incluem-se as arquitecturas hierárquica e distribuída. A hierárquica tem a vantagem de poder ser facilmente mapeada na topologia de rede existente. Na solução distribuída (*peer-like*), um conjunto de agentes cooperam para realizar as tarefas de gestão, com a vantagem de ser mais fácil replicar agentes para obter tolerância a falhas ou uma melhor distribuição de carga [2].

É ainda possível recorrer a combinações das arquitecturas hierárquicas e distribuídas. Foi esta a opção tomada neste trabalho, pois pretendia-se combinar as vantagens de ambas e ter a possibilidade de ter uma arquitectura com um maior dinamismo na sua estrutura para melhor lidar com variações nas características da rede gerida.

Outra forma de lidar com a complexidade de grandes redes é agrupar logicamente equipamentos de acordo com alguma característica comum, como seja a localização, ou o tipo de equipamento. Com esse objectivo, em [3] são definidos objectos **domínio** que permitem criar agrupamentos lógicos de objectos para fins de gestão sem afectar o seu normal funcionamento. O poder de estruturação dos domínios é conseguido através do conceito de sub-domínio, que é um objecto domínio membro de outro domínio, ficando os membros do sub-domínio a pertencer indirectamente ao domínio que o inclui, permitindo-se também a sobreposição de domínios.

Um conceito semelhante aparece em [4] onde se define um serviço de sistemas conhecidos que mantém uma lista de todos os sistemas sobre os quais é possível realizar operações de gestão. É também definido um serviço de domínios de gestão que permite seleccionar subconjuntos dos sistemas conhecidos, de acordo com alguma regra, sobre os quais se executam operações de gestão. Finalmente é definido um serviço de alvos de operações de gestão que fornece, com base em filtros, uma lista de sistemas de um dado domínio sobre os quais faz sentido executar uma dada operação de gestão.

Na plataforma desenvolvida associa-se a cada classe de objectos domínio um conjunto de elementos da rede com características comuns, e ainda um conjunto de serviços específicos desse tipo de elementos da rede. A super-classe domínio apenas implementa os métodos comuns a todos os domínios, que incluem o registo e desregisto de elementos do domínio, sendo os serviços específicos implementados nas sub-classes. Com esta abordagem obtém-se uma ligação mais estreita entre os domínios e os seus membros e uma maior facilidade de implementação.

Outro conceito importante é o de **gestão por delegação** [5] no qual a estação de gestão envia programas (agentes delegados) para servidores elásticos existentes nos equipamentos, cujo tamanho varia consoante os programas carregados. Os programas podem ser escritos em linguagens arbitrárias, interpretadas, ou compiladas, são ligados dinamicamente, e executados sob controlo local, ou remoto. A delegação permite aproximar as funções de gestão dos dados, em vez de enviar os dados para as funções. Assim, reduz-se o tráfego de gestão, descentraliza-se a gestão garantindo escalabilidade, e permite-se que os programas delegados sejam facilmente modificados, permitindo flexibilidade.

[1] Assistente do Instituto Superior Técnico, Investigador do Inesc.
[2] Estudante finalista de Engª Informática e de Computadores, ramo de Sistemas Computacionais, do Instituto Superior Técnico.
[3] Professor Associado do Departamento de Engª Electrotécnica da Universidade Nova de Lisboa.

Para implementar os programas delegados recorre-se a ***scripts*** de gestão, que são funções que permitem automatizar as tarefas de gestão. Em [4] está prevista a utilização de *scripts*, não sendo feita nenhuma suposição sobre a linguagem, podendo-se até distribuir código compilado. No entanto, neste *draft* já estão previstas as seguintes linguagens: Java Virtual Machine, Tcl (Tool Command Language), Perl, Scheme e SNMP Script Language. Outros exemplos de linguagens de *scripts* são: [6] que propõe uma linguagem, que pode ser compilada ou interpretada, baseada em C com comandos SNMP, como sejam snmpopen(), snmpget(), snmpgetnext(), snmpclose(); [7] propõe uma linguagem DEAL baseada em SQL, que é traduzida para um *script* SNMP baseado em Tcl, que por sua vez é interpretado; [8] propõe extensões à linguagem Tcl para aplicações de gestão, cobrindo SNMP e CMIP.

Uma alternativa à delegação é a utilização de agentes móveis [9], em que um programa é injectado na rede, e migra autonomamente pelos vários nós, realizando tarefas de gestão em cada um.

Na plataforma desenvolvida, utilizou-se a linguagem Java [10] e a Java Management API [11], que embora exija mais conhecimentos da parte do programador do que outras linguagens de *scripts*, tem mais poder expressivo, é mais eficiente por ser compilada, oferece a possibilidade de carregamento dinâmico de classes facilitando a gestão por delegação, permite concorrência entre várias *threads*, inclui a possibilidade de implementar uma política de segurança, além de se verificar existirem bons ambientes de desenvolvimento, com extensas bibliotecas de classes.

Na próxima secção descreve-se a arquitectura de gestão da plataforma desenvolvida e como os conceitos apresentados nela se integram.

3. ARQUITECTURA DO SISTEMA

Na figura 1 apresenta-se a arquitectura de gestão baseada numa hierarquia de gestores intermédios (GI), onde podem ser delegados *scripts* de gestão. Cada GI constitui uma entidade autónoma de gestão, com a capacidade e responsabilidade de gerir os seus subordinados. A aplicação de gestão tem a interface com o utilizador, e será apresentada na secção seguinte.

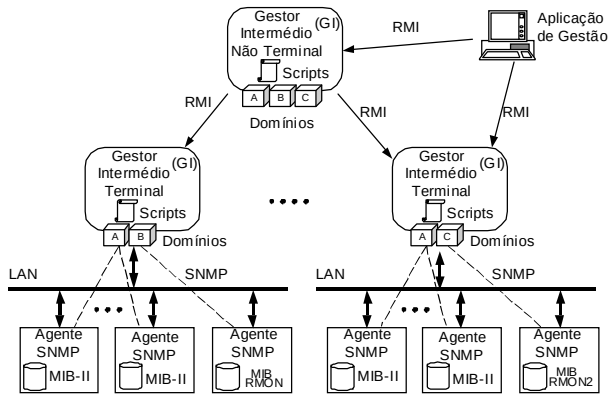

Figura 1 - Arquitectura de Gestão.

Em cada GI podem ser carregados vários domínios e vários *scripts*, havendo diferentes implementações consoante o nível na hierarquia. As implementações não terminais (intermédias) permitem distribuir as operações de gestão pelos vários membros de cada domínio, recorrendo ao protocolo de comunicação da plataforma de gestão (RMI), e fazer algum tratamento da informação retornada por estas operações, através de funções de resumo, análise estatística, soma, etc. As implementações terminais têm, além disso, a função de contactar directamente com os agentes existentes nos equipamentos, através do protocolo de gestão de redes, sendo actualmente suportado apenas o SNMP.

Para tornar o funcionamento da plataforma mais eficiente, minimizando o trafego de gestão, a arquitectura deve adaptar-se à topologia física da rede. Assim, os GIs terminais são distribuídos pelos vários segmentos de rede local, ficando com a responsabilidade de gerir os equipamentos desse segmento, podendo ter-se GIs não terminais responsáveis por cada edifício, ou conjunto de segmentos de rede. Desta forma, cada GI pode automatizar o processamento de gestão sobre os elementos seus subordinados, libertando os níveis superiores dessa carga, e dando uma certa autonomia em termos de gestão aos vários segmentos de rede.

Não se utilizou ainda mais de dois níveis de GIs, mas é possível pensar em arquitecturas com mais níveis, onde os níveis superiores realizam funções com um grau de abstracção mais elevado, como sejam funções de correlação ou combinação dos dados provenientes do nível hierárquico inferior, análise do funcionamento da rede, e auxílio ao planeamento e organização da rede.

A plataforma foi implementada em Java, pelo que é possível colocar um GI em qualquer dispositivo com capacidade de executar uma *Java Virtual Machine*. Por razões de simplicidade de utilização, foi utilizado o Remote Method Invocation (RMI) do Java para comunicação entre GIs, pois oferece uma transparência quase total na invocação de métodos remotos. No entanto, é fácil integrar outros protocolos de comunicação na pilha de comunicação entre processos. Para melhorar a robustez, foi implementado um serviço adicional de recuperação de ligações falhadas que tenta repetidamente estabelecer a ligação durante um certo tempo, permitindo-se assim recuperar de reinicializações de GIs.

Cada GI é composto de um conjunto de módulos de gestão que são descritos em seguida.

O **Gestor de Eventos** fornece um serviço de entrega de eventos originados internamente no GI, ou num GI subordinado, onde cada entidade no sistema interessada em receber determinado tipos de eventos se regista.

O **Gestor de *Scripts*** permite carregar classes de novos *scripts*, instanciá-las para execução, abortar a sua execução, e ainda listar os *scripts* carregados e instâncias de *scripts* em execução. Os *scripts* são executados como *threads* Java, e têm algumas funcionalidades já implementadas na classe abstracta *Script*, da qual todas as outras são derivadas, destacando-se a possibilidade de suspender e reactivar o *script* em determinada altura, e as funções para enviar e receber eventos.

O **Gestor de Domínios** permite que um GI seu subordinado se registe num domínio. Ao fazê-lo, o subordinado passa a estar incluído nas invocações que são feitas a operações desse domínio. O GI contacta o seu superior para que este também se registe nesse domínio. Assim, um GI pertence a um domínio se algum dos seus subordinados também pertencer, como ilustrado na figura 1 para os domínios A, B e C. Sendo os domínios implementados por classes, é possível utilizar herança entre os domínios. Assim, o domínio RMON2 herda do domínio RMON, tendo ainda serviços específicos deste tipo de equipamentos.

O **Gestor de Classes** permite o carregamento e descarregamento dinâmico de classes nos GIs a partir de um repositório de classes. Os seus serviços são utilizados pelos gestores de *scripts* e de domínios locais, para carregar uma nova classe de *script* ou domínio que precise de ser instanciada. Consegue-se assim, uma efectiva implementação do conceito de gestão por delegação.

O **Gestor de Nomes** oferece um serviço que permite saber quais os identificadores dos GIs subordinados, e ainda um serviço de resolução de identificadores de GI em nomes. Na inicialização de um GI é configurado qual o seu superior, onde ele se inscreve, obtendo-se um identificador que é utilizado para identificar os eventos que venha a enviar e para registos

em domínios. Os eventos são identificados por um conjunto de identificadores relativos correspondentes aos GIs por onde passaram na hierarquia, sendo-lhes acrescentado o identificador local sempre que passam por cada GI.
O **Gestor de Persistência** permite guardar e recuperar o estado de um objecto.
O **Gestor de *Logs*** oferece um serviço de *log*, que permite o registo de eventos considerados importantes, e acesso remoto ao registo.
O **Serviço SNMP** é responsável pelas interacções com os agentes SNMP existentes nos equipamentos.
Na secção seguinte apresenta-se a aplicação de gestão que permite controlar a hierarquia de GIs.

4. APLICAÇÃO DE GESTÃO

A aplicação de gestão base é responsável por oferecer ao utilizador uma interface ao sistema, nomeadamente no que diz respeito à gestão de domínios e de *scripts*.
A aplicação permite visualizar a hierarquia de GIs, representando-a como uma árvore do lado esquerdo do écran (figura 2). Os agentes são representados por uma elipse, os GIs não terminais por uma pasta e os terminais por uma pasta com um T dentro. Os nomes dos agentes e GIs são apresentados ao lado dos respectivos ícones.

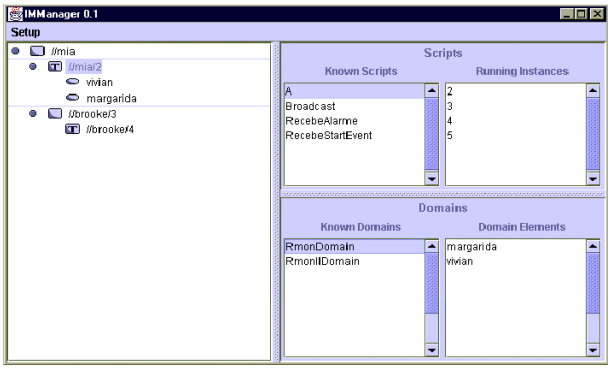

Figura 2 - Aplicação de Gestão.

A selecção de um GI da árvore mostra, do lado direito do écran, o estado actual desse GI no que diz respeito a *scripts* e domínios.
O utilizador pode carregar e remover *scripts* num GI, executar *scripts* e parar instâncias de *scripts* que estejam a ser executadas. Os *scripts* podem ser parametrizados antes de serem executados definindo valores para propriedades específicas de cada *script*. Esta operação só é necessária caso não se pretendam utilizar os valores por omissão das propriedades.
O resultado da execução de uma instância de *script* é apresentado numa janela independente. Para permitir realizar algum processamento dos dados recolhidos para apresentar ao utilizador, podem-se integrar módulos de aplicação na aplicação de gestão. Um exemplo de módulo de aplicação é o módulo gráfico que permite fazer alguns gráficos com os dados obtidos.
A aplicação de gestão permite ao utilizador carregar e remover domínios num GI, e ainda registar e desregistar agentes nos domínios.
Pressupõe-se que pode haver mais do que uma cópia da aplicação de gestão em execução, pelo que para manter a coerência entre o que é apresentado e o estado real do sistema, implementaram-se mecanismos de *polling* que actualizam periodicamente, quer a representação da hierarquia, quer a representação do estado dos *scripts* e domínios.
Na secção seguinte apresentam-se exemplos de *scripts* de gestão.

5. EXEMPLOS

Um nível anormal de *broadcasts* numa rede é um problema conhecido por tempestade de *broadcasts*, que pode acontecer quando há uma falha numa interface. Foi concebido um *script* para detectar e, se possível, corrigir este problema, desactivando ou reinicializando a interface.
O *script* utiliza os serviços do domínio RMON para programar em cada sonda RMON [12] um alarme sobre o número de *broadcasts* ocorridos num dado intervalo de tempo. Desta maneira, se o número de *broadcasts* exceder um determinado limiar é enviado um trap SNMP com o alarme.
Quando é recebido o evento de alarme (figura 3), o *script* consulta o TopN de *broadcasts* existente na sonda, e caso encontre um equipamento que esteja de facto a enviar um número significativo de *broadcasts*, procura na sua MIB qual a interface que causa o problema, gera um relatório para *log*, e se possível desliga essa interface.

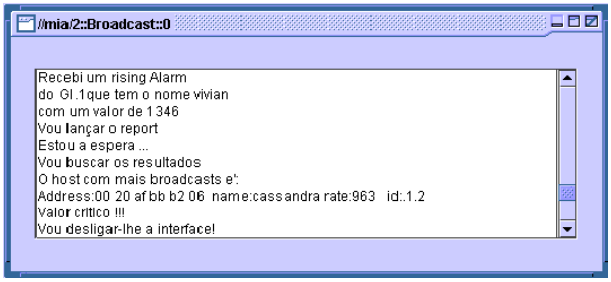

Figura 3 - Resultado produzido pelo *script*.

Este é um exemplo de detecção e correcção de um determinado tipo de falhas que é muito facilmente automatizável através de um *script* de gestão. Além disso, por ser um problema local a cada sub-rede, beneficia da descentralização que esta plataforma oferece.
O segundo exemplo de *script* tem como objectivo contabilizar os recursos de rede utilizados por protocolo de nível aplicação. São utilizados os serviços do domínio RMON2 para recolher informação de sondas RMON2 [13] que mantêm estatísticas da utilização de cada protocolo por cada equipamento, por cada par origem-destino, e ainda os totais do segmento. Assim, é possível definir vários domínios derivados do RMON2 que incluam os equipamentos pretendidos de acordo com algum critério, e recolher estatísticas sobre o tráfego http, ftp, nntp, smtp, snmp, telnet, netbios, x11, rpc, etc., para cada equipamento, podendo os GIs de nível superior na hierarquia agrupar os dados, e estudar a sua evolução. Com a plataforma desenvolvida é fácil implementar diferentes políticas de contabilização de utilização de recursos da rede consoante a posição ou localização na hierarquia de gestão, ou o domínio em causa, sendo também fácil actualizá-las através do processo de carregamento remoto de classes.
Na figura 4 apresenta-se o resultado produzido pelo módulo gráfico para as estatísticas globais de tráfego por protocolo, para um dado período de um segmento de rede Ethernet.

6. CONCLUSÕES E TRABALHO FUTURO

Construiu-se uma plataforma hierárquica distribuída de gestão de redes, que apresenta dinamismo a dois níveis. Não só é possível acrescentar, ou retirar gestores intermédios, adaptando a plataforma às características da estrutura e carga da rede, como também é possível dinamicamente carregar e descarregar domínios e *scripts* de gestão sem afectar o normal funcionamento da rede, permitindo-se assim organizar e automatizar as tarefas de gestão.
Como trabalho futuro, seria útil desenvolver mecanismos de descoberta automática de agentes e sua classificação em domínios de acordo com determinadas regras. Seria ainda

possível estudar métodos automáticos de divisão da hierarquia de gestores intermédios para uma melhor adaptação à carga da rede, desenvolver domínios e *scripts* com novas funcionalidades e para outros tipos de equipamentos, introduzir adaptadores para outros protocolos de gestão tais como o CMIP e o DMI, e estudar formas de incluir um GI em equipamentos. Outra área possível de pesquisa, que está a ser seguida, é o estudo de outras linguagens de *scripts* ou outros conceitos que permitam oferecer ao gestor da rede um ambiente de trabalho a um nível de abstracção mais elevado, no qual não seja necessário programar explicitamente todas as interacções com os agentes.

AGRADECIMENTO

Este trabalho foi parcialmente suportado pelo PRAXIS XXI, sob o contrato 2/2.1/TIT/1633/95.

7. REFERÊNCIAS

[1] Kraig Meyer, Mike Erlinger, Joe Betser, Carl Sunshine, Germán Goldszmidt, Yechiam Yemini. Decentralizing Control and Intelligence in Network Management. *Proceedings 4th International Symposium on Integrated Network Management*, Santa Barbara, CA, Maio de 1995.

[2] Jürgen Schönwälder. Using Multicast-SNMP to Coordinate Distributed Management Agents. *2nd IEEE Workshop on Systems Management*, Toronto, 20 de Junho de 1996.

[3] Morris Sloman, Kevin Twindle. Domains: A Framework for Structuring Management Policy. Capítulo 16 de Network and Distributed Systems Management. Addison Wesley. 1994. ISBN: 0-201-62745-0.

[4] Andy Bierman, Maria Greene, Bob Stewart, Steve Waldbusser. Distributed Management Framework. Internet Draft <draft-ietf-disman-framework-02.txt>, Agosto de 1998.

[5] Germán Goldszmidt, Yechiam Yemini. Distributed Management by Delegation. *Proceedings 15th International Conference on Distributed Computing Systems*, Junho de 1995.

[6] David J. Hughes. ESL - A Script Language for SNMP (and then some!). Bond University, Outubro de 1993. http://wwwsnmp.cs.utwente.nl/Docs/research/ARTICLES/GENERAL/hug9310.ps

[7] Simon Znaty, Michel Lion, Jean-Pierre Hubaux. DEAL: A DElegated Agent Language for Developing Network Management Functions. *PAAM'96 The first International Conference and Exhibition on the Pratical Application of Intelligent Agents and Multi-Agent Technology*, Londres, Abril de 1996.

[8] Jürgen Schönwälder, H. Langendörfer. Tcl Extensions for Network Management Applications. *3rd Tcl/Tk Workshop*, Toronto, Julho de 1995, pp.279-288.

[9] Mario Baldi, Silvano Gai, Gian Pietro Picco. Exploiting Code Mobility in Decentralized and Flexible Network Management. *MA'97 First International Workshop on Mobile Agents*, Abril de 1997, pp.13-26. ISBN: 3-540-62803-7.

[10] James Gosling, Bill Joy, Guy Steele. The Java Language Specification. Addison Wesley, 1996. ISBN: 0-201-63451-1. http://java.sun.com/docs/books/jls/

[11] Javasoft - Java Management API Overview. http://java.sun.com/products/JavaManagement/

[12] S. Waldbusser. Remote Network Monitoring Management Information Base. IETF RFC 1757. Fevereiro de 1995.

[13] S. Waldbusser. Remote Network Monitoring Management Information Base Version 2 using SMIv2. IETF RFC 2021. Janeiro de 1997.

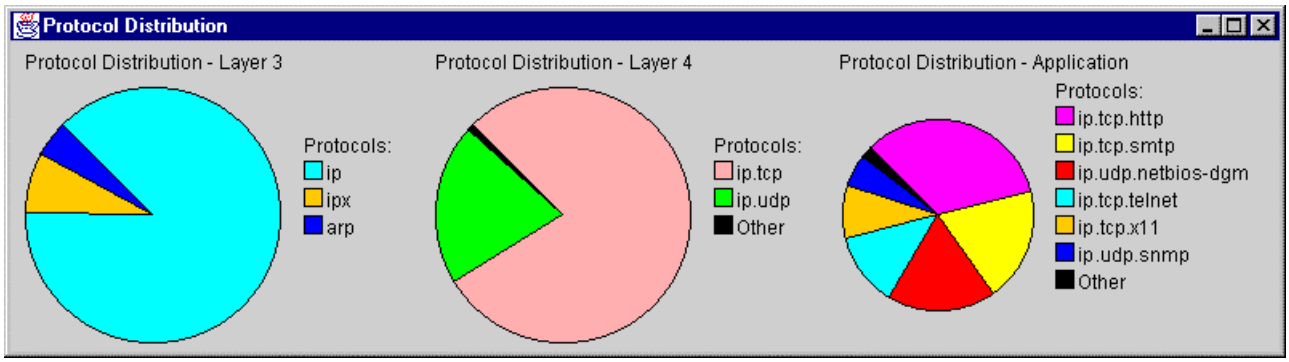

Figura 4 - Estatísticas por protocolo.