SIMON - Sistema de monitorização de desempenho
em redes UNIX usando JAVA

Christiane Zim Zapelini

Dissertação de Mestrado

Instituto de Computação
Universidade Estadual de Campinas

SIMON - Sistema de monitorização de desempenho em redes UNIX usando JAVA

Christiane Zim Zapelini

março de 1999

Banca Examinadora:

- Prof. Dr. Paulo Lício de Geus (Orientador)
- Prof. Dr. Fábio Queda Bueno da Silva (DI-UFPE)
- Prof. Dr. Luiz Eduardo Buzato (IC-UNICAMP)
- Prof. Dr. Rogério Drummond (IC-UNICAMP)

© Christiane Zim Zapelini, 1999.
Todos os direitos reservados.

Agradecimentos

Gostaria de agradecer ao meu marido José Henrique, pela valiosa ajuda, apoio, carinho e amor.

Agradeço ao meu orientador, Prof. Dr. Paulo Lício de Geus, pela orientação.

Aos meus amigos do IC, do CPqD e a todos que ofereceram apoio e estímulo.

Aos meus pais e irmãos, pelo carinho e apoio apesar da distância.

A minha sogra, pelo estímulo.

Agradeço a Unicamp e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq, pela bolsa de estudos concedida.

E acima de tudo, a Deus, que sempre esteve comigo.

Resumo

Devido às várias versões do sistema operacional UNIX e da crescente diversidade de *hardware*, administrar sistemas em rede é uma tarefa contínua e na maioria das vezes complexa. É imprescindível para o bom funcionamento de um ambiente de rede a existência de um mecanismo para monitorizar o desempenho fornecendo dados confiáveis para uma atuação do administrador. As plataformas de gerência e administração de redes UNIX comerciais e acadêmicas existentes possuem vantagens e deficiências, muitas vezes preteridos por comandos e utilitários isolados devido a sua rapidez e agilidade.

Idealmente, tal ferramenta deve ser simples, fácil de usar, de baixo impacto na máquina e na rede, portátil, capaz de traduzir informação de alta qualidade e de fácil extensibilidade para os requisitos locais, automatizando portanto algumas ações do administrador. SIMON (Sistema de Monitorização), apresentado nesta dissertação, obtém um bom compromisso entre tais qualidades, utilizando uma interface Java para fornecer os dados coletados de maneira mais amigável, otimizando assim o desempenho de ambos, rede e administrador.

Abstract

Due to the existence of several variants of the UNIX operating system and to the increasing hardware diversity, administering networked systems is a continuous, mostly complex, task. For good working conditions in a networked environment, it is a necessary requirement the availability of a mechanism to monitor performance and to supply reliable data for the administrator's actions. Current commercial and academic platforms for UNIX network management and administration have both advantages and drawbacks, at times being overlooked in favor of more simple commands and utilities for their quickness.

The ideal tool should be simple, easy to use, of low impact on both the machine and the network, portable, able to convey high quality information and easily extendable to cover local requirements, thus automating some of the administrator's actions. SIMON, presented in this dissertation, yields a good compromise from the above mentioned features; it uses a Java interface to supply collected data in a more friendly way, thus optimizing the performance of both the network and the administrator.

Lista de Figuras

Lista de Tabelas

Conteúdo

Capítulo 1

Introdução

Nos dicionários as palavras gerenciar e administrar são sinônimas. Contudo em ambientes computacionais, mais precisamente em redes de computadores os termos gerência e administração de redes não são sinônimos.

A gerência de redes tem como principal objetivo garantir a conectividade entre os elementos da rede. Cobrindo os níveis 1, 2 e 3, respectivamente, o nível físico, o de enlace de dados e o de rede do modelo OSI (*Open Systems Interconnection*), seus principais tópicos de interesse são a integridade dos enlaces físicos, o roteamento (no nível de rede) e a monitorização do tráfego nos diversos segmentos constituintes da rede.

A administração de sistemas preocupa-se em controlar a rede em um nível mais alto de abstração, onde o sistema operacional e os serviços de rede serão supervisionados para que funcionem sem problemas e eficientemente. Abrange os níveis superiores do modelo OSI (transporte, sessão, apresentação e aplicação).

A administração de sistemas difere substancialmente da gerência. Assim, administrar redes de sistemas UNIX refere-se a administrar máquinas conectadas à rede que executam o sistema operacional UNIX. Isto compreende configurar, controlar, recuperar, otimizar e principalmente manter serviços de rede de mais alto nível que compreendem as aplicações e o próprio sistema.

Devido às várias versões do UNIX e da crescente diversidade de *hardware*, o ato de dirigir a rede exige um serviço contínuo e na maioria das vezes complexo, surgindo com isto a necessidade de ferramentas de *software* para auxiliar as pessoas envolvidas na tarefa de gerir os equipamentos de rede e administrar seus sistemas.

As ferramentas comerciais ou acadêmicas na área de gerência e administração de sistemas de rede existentes atualmente possuem vantagens e desvantagens. Como vantagens pode-se citar a abrangência de funcionalidades em atender as necessidades dos usuários e administradores, interfaces gráficas interativas, uso em redes heterogêneas de pequeno a grande porte. E a exigência de grande poder computacional causando o uso dedicado da máquina pela ferramenta e complexidade na instalação, configuração e manutenção são algumas das desvantagens.

A necessidade cada vez maior por serviços rápidos e que atendam melhor às necessidades dos usuários torna difícil a atividade de administrar os serviços no nível de aplicação. Tarefas como verificar a funcionalidade do NFS (*Network File System*), monitorizar processos que estejam dominando o uso de CPU e memória, montar e configurar sistemas de arquivos, entre outros, precisam ser eficientes para que os sistemas de rede não sejam afetados, ou ainda que a rede como um todo não sofra graves quedas de desempenho a ponto de "travar" a interação com o usuário.

É imprescindível para o bom funcionamento de um ambiente de rede a existência de um mecanismo para monitorizar o desempenho e que este mecanismo forneça dados confiáveis para uma atuação do administrador.

Atualmente a tarefa de administrar fica centrada em uma única pessoa ou num grupo de pessoas, o(s) administrador(es) de redes, que devem solucionar os imprevistos que ocorram nos sistemas bem como manter a rede operando eficiente e rapidamente.

Objetivando uma rápida e eficiente tomada de decisão pelo(s) administrador(es) e conseqüente restauração dos serviços, deseja-se automatizar algumas ações do administrador, com uma ferramenta de auxílio, oferecendo dados graficamente e até mesmo evitando que alguns problemas efetivamente ocorram. Muitas vezes, por serem mais rápidos e ágeis, utiliza-se comandos e utilitários do UNIX para encontrar e solucionar os problemas e "gargalos" dos sistemas. A ferramenta de administração de sistemas UNIX procurada deve ser simples e fácil de usar, e não deve sobrecarregar a rede ou a máquina onde será instalada. Além disso, sua extensão e manutenção devem ser tão simples quanto possível, porque cada ambiente de rede tem inúmeras particularidades que necessitarão de personalizações por parte do administrador.

Dentro deste contexto é que se propõe nesta dissertação de mestrado a implementação de uma ferramenta para a administração de sistemas chamada de SIMON (SIstema de MONitorização) com o objetivo de auxiliar a administração dos sistemas de rede de pequeno porte, automatizando algumas ações do administrador e fornecendo estes dados coletados de uma forma mais amigável, usando-se gráficos e tabelas, otimizando assim o desempenho de ambos, administrador e rede.

A seguir descreve-se como será apresentado este trabalho. Como o enfoque baseia-se no desempenho tanto do administrador como da rede e seus sistemas, no capítulo 2 descreve-se alguns dos fatores de desempenho que afetam as máquinas UNIX. Um

levantamento dos fatores críticos de desempenho foi feito nas cinco áreas consideradas mais preocupantes, que são: memória, CPU, sistema de discos, rede e sistemas com grande atividade de entrada/saída (I/O) e de rede como o NFS.

Atualmente existem muitos pacotes no mercado que oferecem soluções de gerência de redes e poucos para administração de sistemas. No capítulo 3 apresenta-se alguns dos sistemas de monitorização, gerência e administração existentes no mercado, dando uma visão geral sobre o que está disponível além de algumas ferramentas propostas academicamente dentro deste contexto. Os conceitos de gerência de redes e administração de sistemas são diferentes, refletindo-se nos *software* existentes.

A proposta do sistema de monitorização, a ferramenta denominada SIMON, é apresentada no capítulo 4. Os utilitários e comandos UNIX utilizados denominados de coletores, a interface gráfica escolhida e detalhes de implementação serão abordados.

Este trabalho encerra-se tecendo considerações sobre a implementação desta ferramenta de monitorização e os sucessos obtidos tendo em vista os objetivos inicialmente propostos.

Capítulo 2

Fatores de Desempenho em Máquinas UNIX

O UNIX originou-se de um projeto de pesquisa dos Laboratórios Bell da AT&T em 1969. A versão V6 foi distribuída gratuitamente em 1976 para universidades. A V7 foi lançada três anos depois e foi a primeira versão amplamente distribuída custando $100 (cem dólares) para universidades e $21,000 (vinte e um mil dólares) para outras entidades. Depois da V7 a AT&T criou o grupo USG (*UNIX Support Group*) para tornar o UNIX um produto comercial. Os Laboratórios Bell e o USG, que mais tarde tornou-se o USL (*UNIX System Laboratories*) continuaram a desenvolver o UNIX, mas os dois grupos divergiram. As versões USL (System III e System V) foram distribuídas e são a base de muitos sistemas modernos.

O UNIX de Berkeley começou em 1977 quando o CSRG (*Computer Systems Research Group*) da Universidade da Califórnia licenciou o código da V6 da AT&T. As versões BSD (*Berkeley Software Distribution*) necessitavam de licença da AT&T para seu uso. Houve uma disputa judicial entre a AT&T e a Universidade da Califórnia sobre a utilização e distribuição do código do UNIX. Em fevereiro de 1994, AT&T e Universidade da Califórnia fizeram um acordo e as versões BSD são distribuídas gratuitamente sem o pagamento de licença [1].

Como o UNIX foi um sistema muito bem aceito comercialmente, foi adotado por várias empresas que adicionaram extensões e inovações em suas versões. Mas basicamente o UNIX descende ou do padrão AT&T ou do BSD.

Com esta diversidade de versões do UNIX, a administração torna-se uma tarefa complexa e o desempenho é um setor importantíssimo para ser analisado e melhorado. Gerenciar o desempenho não é uma tarefa trivial, necessita de uma monitorização contínua dos recursos do sistema. Utiliza-se, em vários casos, ferramentas, utilitários e comandos do próprio sistema que diferem de versão para versão em termos de localização física de diretórios, significados dos campos, comandos com mesmo nome porém funções diferentes e comandos existentes em determinadas versões e em outras não. Por possuírem particularidades nos comandos e facilidades nas diferentes variantes de UNIX e devido ao ambiente predominantemente Solaris 2.X da Fundação CPqD, onde a ferramenta foi implementada, e também devido ao fato do Solaris 2.X ser o UNIX comercial de maior

presença em estações de trabalho tanto em empresas como em universidades, o Solaris 2.X será adotado como ambiente padrão neste trabalho. Ele é um UNIX no padrão da AT&T com várias extensões.

Gerência de desempenho compreende monitorizar, analisar, otimizar e fazer ajustes nos recursos computacionais para fornecer ao usuário um grau de satisfação no uso dos serviços da rede. Estes recursos computacionais são memória, CPU, sistema de discos, rede e aplicações relacionadas. É um processo que pode ser reativo (capacidade de reagir quando um determinado problema ocorre) ou pró-ativo (capacidade de antecipar a ocorrência de um determinado problema) dependendo da atuação nos sistemas de rede e na otimização pretendida.

Dentro deste contexto e segundo Cockcroft [2], os fatores que afetam o desempenho das máquinas e sistemas UNIX podem ser agrupados da seguinte forma:

1) memória

2) CPU

3) sistema de discos

4) tráfego em e entre subredes e

5) sistemas com grande atividade de entrada/saída (I/O) e de rede como o NFS.

Estes fatores podem ser monitorizados separadamente ou agrupados, pois o desempenho pode degradar baseado não apenas em um fator e sim em um conjunto de fatores que contribuem direta ou indiretamente na deterioração dos serviços do sistema.

Para otimizar o sistema é necessário, em primeiro lugar, entender seu funcionamento, evitando fazer mudanças desnecessárias. Em princípio, realizar as tarefas mais simples e sempre fazer uma mudança de cada vez para perceber melhor qual o problema de desempenho.

Como uma regra geral, otimizar o desempenho consiste em realizar algumas tarefas [3], tais como:

1) Analisar a carga de trabalho do sistema e determinar quais componentes necessitam de ajuste.

2) Fazer melhorias nos sistemas de I/O, que compreende configurar uso da rede, sistemas de arquivos como o NFS, entre outros.

3) Otimizar o uso da CPU.

4) Otimizar o uso de sistema de discos.

5) Ajustar o uso e o compartilhamento de memória.

O sistema operacional possui vários parâmetros e atributos de configuração que podem afetar o sistema de I/O, por exemplo, e se forem ajustados de maneira correta ajudam a melhorar o desempenho do sistema. Contudo, segundo Nemeth [1], se qualquer parâmetro ou atributo do sistema for alterado incorretamente o desempenho pode piorar.

O `kernel` é o núcleo do sistema operacional. No sistema Solaris 2.X ele é autoconfigurável, seguindo a tendência de vários sistemas operacionais modernos. Assim, o `kernel` não precisa ser reconstruído para adição de novos dispositivos ou para redimensionar as tabelas de parâmetros do sistema operacional. Sendo autoconfigurado dinamicamente toda vez que a máquina é reinicializada, aloca o espaço para suas tabelas de parâmetros de acordo com a necessidade. O `kernel` localiza-se no diretório `/kernel/genunix` e o arquivo `/etc/system` contém alguns parâmetros de configuração.

A seguir são relacionados alguns dos fatores que afetam o desempenho dos sistemas Solaris 2.X. Esses fatores são divididos nas áreas de memória, CPU, sistema de discos, rede e o sistema de arquivo NFS.

2.1. Memória

O sistema de memória torna-se um fator limitante para o desempenho quando os programas em execução necessitam de mais memória física do que a disponível na máquina [4]. É um importante recurso que deve ser cuidadosamente gerenciado, sendo um dos pontos mais críticos nos sistemas.

O sistema Solaris 2.X [5] pode utilizar regiões de discos para armazenamento temporário e não apenas arquivos dentro de sistemas de arquivos. Estas regiões são denominadas partições de *swap* e são utilizadas como áreas de armazenamento de memória virtual quando o sistema não tem memória física suficiente para manipular os processos correntes.

O sistema de memória virtual do Solaris 2.X mapeia cópias físicas de arquivos em disco para endereços virtuais na memória. Páginas de memória física que contenham os dados para este mapeamento podem ser recuperadas para arquivos no sistema de arquivos ou para a partição de *swap*. Se a memória é recuperada para a partição de *swap* ela é referenciada como uma memória anônima, porque o usuário não conhece os nomes dos arquivos que retornam da memória.

Um sistema de memória virtual é igual à soma de todo o espaço de *swap* físico (retornado do disco) mais a porção de memória física disponível correntemente.

O sistema de arquivos TMPFS[1] armazena arquivos (diretório `/tmp`) e as informações associadas na memória ao invés do disco, sempre que possível. O TMPFS aloca espaço no diretório `/tmp` para os recursos de *swap* do sistema. É ativado automaticamente no Solaris 2.X por uma entrada no arquivo `/etc/vfstab`. Na realidade, a somatória da memória física e das partições de *swap* constitui um único recurso, compartilhado tanto para atividades de memória virtual quanto para o sistema de arquivos TMPFS no `/tmp`.

A partição de *swap* é alocada na instalação do Solaris 2.X em regiões *default* de acordo com a quantidade de memória física. Depois de instalado o *swap* é listado no arquivo `/etc/vfstab` e ativado pelo *script* `/sbin/swapadd` quando o sistema é iniciado.

O ideal é não realizar *swap*, pois o acesso a disco é várias vezes mais lento do que acesso a memória física. Sistemas de memória SDRAM em computadores simples, hoje em dia, têm 64 bits/palavra. O primeiro acesso leva 50ns, mas acessos a posições subsequentes de memória levam apenas 5ns. Em um disco rígido atual, de bom desempenho, o tempo de acesso médio é da ordem de 12ms, e após o posicionamento da cabeça os dados são lidos tipicamente a 10 MB/s. Para exemplificar: gasta-se 112ms para ler-se 1 MB de dados do disco. Se este mesmo 1 MB estiver na SDRAM e como a transferência é feita a 64

[1] TMPFS é um sistema de arquivos baseado na memória que utiliza os recursos do `kernel` relacionados ao sistema de memória virtual e *cache* de páginas como um sistema de arquivos temporário. [24]

bits/palavra, e somando-se o tempo de acesso de 50ns, gasta-se 0,65ms. Portanto o acesso a SDRAM é 172 vezes mais rápido do que o acesso a disco, para esta quantidade de dados.

O desempenho deteriora quando o sistema começa a realizar uma quantidade enorme de *swapping* e *paging*. *Swapping* é o mecanismo de transferir a imagem de um processo para disco de forma a liberar memória física para outro processo e o de trazé-la de volta mais tarde. *Paging* é o mecanismo usado para implementar memória virtual e consiste em dividir um ou mais blocos de disco em páginas de tamanho fixo e mapeá-las em páginas de RAM sob demanda.

O termo *swapping* é usado de maneira genérica sempre que houver atividade/necessidade em disco do sistema de memória virtual.

Considerando-se o *swapping* e *paging* como tarefas intrínsecas do sistema, pode-se melhorar seu funcionamento e reduzir sua atividade excessiva assim [3]:

a) Adicionar mais memória física.

b) Reduzir a demanda de memória no sistema executando menos aplicações simultaneamente. Uma maneira de realizar-se isto é utilizando comandos como `at` ou `batch` para executar aplicações em horários de menor carga do sistema. Os comandos `at` e `batch` executam comandos em horários predeterminados [6].

c) Utilizar múltiplos discos para distribuir a partição de *swap*, desta forma, o tempo médio de acesso tende a ser dividido pelo número de discos.

2.2. CPU

O controle de processos é uma tarefa primordial para melhorar o desempenho da unidade central de processamento. Um processo é a instância de um programa em execução. É uma abstração do UNIX que consiste de um espaço de endereçamento e um conjunto de estruturas de dados dentro do `kernel`. O espaço de endereçamento é um conjunto de páginas de memória que o `kernel` marca para uso do processo. As estruturas de dados são: segmentos para o código do programa que está em execução, variáveis utilizadas pelo processo, a pilha do processo e outras informações necessárias para o `kernel` enquanto o processo está em execução.

O UNIX utiliza uma técnica chamada *time-slicing* que faz o controle de concorrência dos processos, mudando os processos em execução utilizando pedaços de tempo regulares e pequenos da CPU. Esta técnica dá a ilusão de que vários processos são executados paralelamente [4].

Alguns ajustes no modo de executar programas e em alguns parâmetros do sistema podem melhorar o desempenho da CPU [3]. Tais como:

a) Escalonar trabalhos, isto é, distribuir os trabalhos por períodos. Pode ser feito de várias formas:

- Priorizar os trabalhos mais importantes colocando-os para executar primeiro, pode-se utilizar os comandos `nice` e `renice`. O comando `nice` escalona um programa requisitando que ele execute com uma prioridade diferente no sistema (para programas ainda não iniciados) [7]. O comando `renice` altera a prioridade de um processo que já está em execução [8].

- Escalonar trabalhos em tempos distintos (comandos `at` e `cron`) ou quando o nível de carga permitir (comando `batch`). O comando `cron` inicia um processo que executa os comandos especificados em datas e horas predeterminadas [9].

b) Utilizar os comandos `limit` e `ulimit` para alterar os limites do ambiente do sistema. A sintaxe pode variar de acordo com a *shell* utilizada na utilização destes comandos. Os limites são: `cputime` (especifica o número de segundos para ser usado por cada processo na CPU), `stacksize` (especifica o tamanho em KB para a área de pilha de processos), `filesize` (especifica o número de blocos de 512 bytes para arquivos escritos por processos filhos, arquivos de tamanhos variados podem ser lidos), `datasize` (ou `heapsize`, especifica o tamanho em KB para a área de dados de processos, incluindo pilha), `coredumpsize` (especifica o tamanho em blocos de 512 bytes para arquivos `core`), `descriptors` (especifica o número de descritores de arquivo) e `memorysize` (especifica o tamanho em KB de memória virtual por processo) [10].

c) Desabilitar serviços (*daemons*) que não estejam sendo utilizados. Nos arquivos de inicialização `/etc/rc*` e no arquivo `/etc/inetd.conf` pode-se averiguar quais serviços não são utilizados, podendo-se removê-los ou desabilitá-los.

d) Verificar a carga da CPU. Um comando útil seria o `uptime` [11]. Uma carga aceitável depende do tipo de sistema e de quanto ele está sendo utilizado.

e) Verificar o tempo de CPU alocado para o sistema, usuário e o tempo que ficou ociosa. Existem vários comandos que oferecem estes dados, por exemplo, `vmstat` [12] e `iostat` [13]. O objetivo é manter a CPU produtiva, minimizando o tempo que permanece ociosa. Ciclos de CPU ociosa ocorrem quando não existem processos rodando ou quando a CPU está esperando para completar uma requisição de memória ou I/O. Deste modo:

- um tempo de sistema alto e um tempo ocioso baixo pode indicar que há uma sobrecarga de alguns serviços do sistema. Tais sobrecargas de operações podem consistir de altas freqüências de chamadas de sistema, altas taxas de interrupção, grande número de pequenas transferências de I/O ou grandes números de transferências de rede.

- um tempo de sistema alto e um tempo ocioso baixo também pode indicar que o sistema está fazendo muito *paging* e *swapping*.

2.3. Sistema de discos

O tempo de acesso ao disco é várias ordens de grandeza maior se comparado ao acesso a CPU ou a memória (conforme mostrado na seção 2.1). Sendo o sistema de discos considerado também "gargalo" no desempenho de máquinas UNIX, é importante configurá-lo e utilizá-lo corretamente para otimizar seu uso [4]. Algumas soluções para tanto são apontadas a seguir [3]:

a) Utilizar discos rápidos, na maioria das vezes, isto significa discos mais modernos.

b) Usar, se possível, mais discos pequenos em vez de poucos discos grandes. Um número maior de discos com tamanhos pequenos usualmente rende um tempo de *seek* médio e latência melhor, conforme citado na seção 2.1.

c) Executar, se possível, simultaneamente poucas aplicações com altas taxas de acesso a disco.

d) Utilizar quotas para limitar o espaço do usuário. É recomendável manter-se 10% do espaço livre, segundo Loukides [4]. O comando `df` fornece as taxas de utilização dos discos [14].

e) Otimizar o *layout* dos sistemas de arquivos organizando as informações através dos múltiplos discos (se houverem) para distribuir a carga de I/O igualmente. Agrupar arquivos similares, projetos e grupos.

f) Isolar aplicativos de desempenho crítico que sobrecarreguem discos e rede.

g) Distribuir as partições de *swap* por múltiplos discos locais.

h) Verificar a fragmentação do disco e verificar o tamanho do fragmento do sistema de arquivo. Utilizando o comando `newfs` pode-se alterar o tamanho do fragmento. Pode-se usar um tamanho de fragmento maior, ao invés do tamanho *default,* se o sistema de arquivo é usado para arquivos executáveis e um tamanho de fragmento *default* se o sistema de arquivos é usado para arquivos pequenos ou para código de desenvolvimento. Um tamanho de fragmento grande pode desperdiçar espaço em disco. Um tamanho de fragmento pequeno usa espaço em disco mais eficientemente do que um tamanho de fragmento grande. O comando `newfs` é um *front-end* do comando `mkfs` que cria sistemas de arquivos UFS (*UNIX File System*) em partições do disco. `newfs` calcula os parâmetros apropriados e chama o `mkfs` [15].

i) Reduzir a densidade de inodes, se o sistema de arquivo possui muitos arquivos grandes. Utilizando o comando `newfs -i nbpi`.

j) Mudar o espaçamento rotacional entre blocos escritos no disco, alocando desta forma blocos contíguos logicamente. Utilizando os comandos `tunefs` ou `newfs`. O comando `tunefs` é projetado para mudar os parâmetros dinâmicos que afetam as políticas de *layout* de um sistema de arquivos [16].

k) Incrementar o número de blocos que são combinados para leitura alterando o valor do atributo `maxcontig,` que especifica o número máximo de blocos contíguos permitidos para leituras. Com isto o sistema de arquivos permitirá pedidos de leitura maiores definido pelo valor de `maxcontig` multiplicado pelo tamanho do bloco. Nos sistemas Solaris 2.X os comandos `tunefs` e `newfs` permitem esta alteração de atributos.

l) Mudar o valor do parâmetro `maxbpg`. Utilizando os comandos `tunefs` ou `newfs` para trocar o valor de `maxbpg`, que é o número máximo de blocos de arquivo que qualquer arquivo pode alocar por grupo de cilindro. Normalmente, este valor é configurado para aproximadamente 1/4 (um quarto) do total de blocos em um grupo de cilindro. O parâmetro `maxbpg` é usado para prevenir que um arquivo utilize todos os blocos em um grupo de cilindro, que poderia degradar o tempo de acesso para todos os arquivos subseqüentemente alocados no grupo de cilindro. Se o sistema de arquivo contém somente arquivos grandes, pode-se configurar o parâmetro `maxbpg` mais alto do que o valor *default*.

m) Diminuir o número de sistemas de arquivos por disco, diminuindo desta forma o acesso a disco.

2.4. Rede

A funcionalidade de rede é imprescindível porque é através dela que as informações e dados fluem entre as máquinas. Porém, melhorar o desempenho de uma rede é uma tarefa complexa pois existem vários fatores, tanto de *hardware* quanto de *software*, envolvidos. Umas das primeiras ações para otimizar seu *hardware* e *software* é monitorizá-la.

A monitorização da rede é uma ação de vital importância, baseando-se em alguns dados coletados, pode-se verificar problemas e solucioná-los, tais como [3]:

a) Verificar o número de retransmissões pela rede ou de pacotes perdidos, com isto pode-se descobrir congestionamentos na rede.

b) Verificar a existência de taxas excessivas de erros de entrada, saída ou colisões pois indicam problemas na rede como, por exemplo, cabos mal conectados, conectores e placas em mau estado ou mesmo uma saturação da rede. O comando `netstat -i` fornece estas informações [17].

c) Verificar más configurações de topologia que podem causar altas taxas de colisões e tráfego intenso.

d) Verificar problemas com roteamento causados por instabilidade em algoritmos de roteamento.

e) Verificar o atraso e a perda no envio dos pacotes, utilizando para isto o comando `ping -s`. O comando `ping` utiliza o datagrama ECHO_REQUEST do protocolo ICMP para extrair uma resposta do tipo ECHO_RESPONSE de uma máquina específica [18].

2.5. NFS

O NFS é um serviço baseado em *Remote Procedure Call* (RPC) que habilita máquinas a compartilhar arquivos através da rede. É constituído de dois serviços básicos: `mountd` e `nfsd`. O `mount` é o comando que habilita o `kernel` a usar hierarquia (disco local ou rede) permitindo ao usuário acessar arquivos remotos e hierarquias transparentemente, como se eles estivessem localmente na máquina do usuário.

Clientes utilizam arquivos montando os diretórios exportados do servidor. Quando um cliente "monta" um diretório, o processo de `mount` usa uma série de RPCs para habilitar seu acesso transparentemente aos diretórios do disco do servidor. Deste modo nenhuma cópia do diretório é feita no cliente.

A informação é processada pelo *daemon* `rpc.mountd` e ao cliente é permitido ou não "montar" a hierarquia de arquivos. Uma vez que o cliente "monta" um diretório remoto, qualquer acesso a este se comporta como um acesso local, exceto por eventuais diferenças de desempenho.

Muitos problemas de desempenho com NFS podem ser atribuídos aos problemas com o sistema de arquivos, rede ou discos. Neste caso pode-se [3]:

a) Trocar os limites de *timeout* de *cache* para melhorar os sistemas de arquivos somente para leitura e conexões de rede lentas. Estes *timeouts* afetam rapidamente as atualizações dos arquivos ou diretórios que tenham sido modificados por outro servidor. Se os arquivos não são compartilhados com usuários em outros servidores, incluindo o servidor do sistema, aumentar estes valores melhorará significativamente o desempenho e reduzirá a taxa de tráfego gerada na rede.

b) Verificar os valores de atributos relacionados ao `mount` nos arquivos de configuração, por exemplo os atributos de tamanho de *buffer* de escrita e leitura.

c) Verificar com o comando `nfsstat -c` [19] (do lado cliente) os valores dos campos `timeouts` e `calls`. A taxa de `timeouts` deve ser inferior a taxa de `calls` basicamente segundo referência [3] o valor de `timeouts` deve ser 1% do valor de `calls`, pois se não for pode indicar queda no servidor ou conexões lentas.

Algumas das soluções mostradas neste capítulo e os comandos `vmstat`, `iostat`, `uptime`, `swap`, `netstat`, `df`, `ps`, `nfstat` serão descritos no Apêndice A pois serão utilizados na ferramenta SIMON. Alguns comandos como `tunefs` e `newfs` somente podem ser utilizados pelo super-usuário. As plataformas para gerência e administração de redes serão o tema do próximo capítulo.

Capítulo 3

Plataformas de Gerência e Administração de Rede

O objetivo deste capítulo é relacionar o que existe em termos de estado da arte em plataformas de gerência, e em alguns casos de administração de redes, de pacotes comerciais e soluções apresentadas academicamente.

Este capítulo está dividido em duas seções, uma para pacotes de administração/gerência de redes de uso comercial e outra para propostas acadêmicas de ferramentas para administração/gerência de redes. Na abordagem dada às seções o enfoque foi dado a características e a produtos e subprodutos de *software* em administração/gerência de redes UNIX. Algumas linhas de produtos de empresas como 3COM, Cisco e Sun abrangem gerência não só de sistemas, enfoque dado a esta pesquisa, mas também de toda uma gama de *hardware* de rede como *hubs*, roteadores e afins.

3.1. Pacotes Acadêmicos:

São descritos a seguir 8 ferramentas propostas academicamente: Igor, Satool, SAFO, Olho Vivo, HLMMIB, Buzzerd, I-DREAM e SAGRES.

3.1.1. Igor

Igor [25] é um `rsh`[2] multiplexado e interativo para distribuição de tarefas com uma interface gráfica controlada por um operador central. A ferramenta Igor surgiu porque a utilização do comando `rsh` para execuções remotas possuía vários problemas como: execução serial, estilo de execução em lote, uso de `rcp` ou NFS para transferência de *scripts*. Foi escrito em Tcl/Tk[3] e Perl 4 e consiste de 2 partes: GUI (*Graphic User Interface*) interativa controlada por operador e um *daemon* que executa os comandos em máquinas UNIX remotas.

Igor aceita comandos via GUI e passa-os para um conjunto de *scripts* em Perl, os quais distribuem tais comandos, que serão listados na tabela 3.1, para os sistemas remotos.

[2] O comando `rsh` conecta-se na máquina remota especificada e executa o comando especificado. [23]
[3] Tcl/tk *Tool Command Language*/*Toolkit* X11 acessível pelo Tcl

Os *scripts* Perl mantém várias conexões simultâneas e manipulam os problemas como *timeouts* e problemas de conexão na rede, coletando as informações e retornando-as à GUI.

Igor roda como um *daemon* monitorizando uma porta TCP/IP predeterminada; este *daemon* pode ser inicializado usando a função `spawn` ou por métodos convencionais como *scripts* rc.*. O *script* se "daemoniza"[4] e roda em segundo plano, ouve a porta e quando a conexão é feita, divide-se e seus processos filhos recebem comandos pela GUI. Estes comandos podem ser *shell scripts*, *scripts* Perl ou funções especiais. As saídas e erros são coletadas e enviadas ao console central para análise e os processos filhos são terminados.

Na máquina central, a lista de máquinas está em arquivos ASCII, `spawn` é usado para iniciar o `Igord` nas máquinas remotas. Toda tarefa realizada é interativa e visível pela GUI. Os dados coletados são armazenados no subdiretório `./Idata` em cada máquina. Várias tarefas podem ser executadas ao mesmo tempo.

A máquina central é UNIX e deve ser uma máquina confiável em termos de execução remota de comandos, além de ter mais poder de CPU e rede para fazer várias conexões simultâneas. Os recursos usados pela GUI são ajustáveis, mas dependem do tipo de sistema operacional. A segurança é baseada no mecanismo `rexec` do BSD com os arquivos `~/.rhosts` ou `/etc/hosts.equiv` (lista de máquinas e usuários confiáveis). O *daemon* verifica se o máquina é confiável, caso não seja, nenhum comando é aceito e uma mensagem é escrita no *socket* de comunicação com a GUI.

Os *scripts* do Igor contém embutidos *shell scripts*, arquivos `tar` e comandos de linha para execução remota. Os comandos são listados na tabela 3.1:

Comando	**Descrição**
do *args*	Executa *args* como um comando *shell* (`bin/sh`).
EVAL *args*	Executa *args* como comando Perl.
openfile *file mode*	Abre arquivo (*file*) como um arquivo local com o *mode* especificado.
closefile	Fim do bloco openfile.
id	Relata número da versão, máquina local, máquina remoto, data, hora, tipo de arquitetura local e outras informações relevantes.
quit	Termina o `Igord` remoto e transmite os resultados da coleta.

Tabela 3.1 - Comandos da Ferramenta Igor

[4] Ou seja, desconecta-se do processo controlador do terminal virtual.

3.1.2. Satool

Satool [26] é uma ferramenta com o objetivo de fornecer um modo para monitorizar grupos de máquinas e encontrar problemas em potencial, rápida e eficientemente.

Foi projetada para ser flexível, escalável, fácil de gerenciar e reusar ferramentas existentes. Flexibilidade foi obtida através do uso de arquivos de configuração, módulos e itens configuráveis. Escalabilidade para redes grandes e pequenas, além de ter baixo impacto na coleta de dados e interação do usuário com máquinas de maneira hierárquica. Gerenciável por apresentar parâmetros iniciais padrão, sem mudanças drásticas na adição de novas máquinas, o único arquivo de configuração a ser alterado está no lado do servidor. Reusabilidade conseguida com a utilização de comandos UNIX.

É composta por três partes independentes: agente SNMP (*Simple Network Management Protocol*) que roda em cada máquina monitorizada, um banco de dados no servidor que coleta dados de cada máquina cliente e uma GUI que atua como interface entre o usuário e o banco de dados.

O agente SNMP é baseado no snmp1.1b da Universidade Carnegie-Mellon (CMU) e usa uma MIB (*Management Information Base*) satool. Esta versão é compatível com a MIB-I. Existem duas grandes diferenças entre as versões CMU e satool: o arquivo de configuração e o suporte para variáveis satool na MIB. O agente também possui *scripts* que manipulam os dados de comandos UNIX.

O arquivo de configuração `/usr/local/etc/satool-agent.conf` é usado somente para especificar a máquina no envio de *traps*[5]. Se o arquivo de configuração não existir usará o que for especificado em tempo de compilação (`/usr/local/etc/satoold.conf` na variável preload).

O *daemon* satoold possui três funções principais: coletar dados dos clientes, armazenar no banco de dados e servir pedidos da GUI. Ele também escreve, por conveniência o `id` (identificação) do processo no arquivo `/etc/satoold.pid`. Satoold interroga os clientes via SNMP em intervalos configuráveis. Todo processo é feito por um processo filho o qual retorna os dados ao processo pai via *socket* do UNIX. Uma variável é

[5] *trap* é um mecanismo de aviso de problema do agente para o gerente.

utilizada para especificar o número de interrogações simultâneas permitidas. Um contador guarda o número de interrogações filhas junto com o conjunto de seus `id´s`.

Os pedidos de serviço da GUI são feitos pelo satoold que aceita conexões pela porta 4017 (0xFB1 em hexadecimal). O protocolo usado é similar com o SMTP (*Simple Mail Transport Protocol*). Os comandos são: HELO (diz hello ao *daemon*), HELP (escreve uma mensagem curta de ajuda), LIST (lista todas máquinas clientes no banco de dados) e GET (pega os dados de uma máquina específica).

Os dados dos clientes são mantidos num banco de dados `ndbm` pelo nome completo da máquina no domínio. Uma cópia do banco de dados é mantida em disco por segurança.

Arquivo de configuração do satoold possui parâmetros que podem ser configurados pelo arquivo `/usr/local/etc/satoold.conf`. O intervalo de interrogação é definido pela entrada *interval default*, o valor inicial é 300 segundos. Valores por máquinas individuais são definidos pela entrada *interval* <*hostname*>. Máquinas pré-carregadas são especificadas pela entrada *preload* <*hostname*>.

A GUI é baseada em X para visualizar o conteúdo do banco de dados satool. Foi escrita em Tcl/Tk, utiliza arquivo de configuração para configurar visão hierárquica das máquinas monitorizadas, especificar limites para valores de dados e para tipos de objetos (*widgets*). É indicada para rodar em estações de trabalho de administradores ou estações de trabalho dedicadas, fica trabalhando em segundo plano até que um alarme é ultrapassado notificando o administrador. Tcl/Tk foi utilizada por ser portável, extensível, fácil de usar e é poderosa na comunicação utilizando o comando *send* que permite que diferentes processos Tcl/Tk comuniquem-se uns com os outros.

Satool representa grupos de máquinas hierarquicamente, como o conceito de `netgroup` da Sun Microsystems Inc. Grupos são definidos pelo arquivo de configuração `satool-display.conf` usando a sintaxe do arquivo `/etc/aliases`:

```
groupname: member, member, ...
```

Limites de alarmes para cada variável monitorizada também são especificados por máquinas. Macros, membros individuais ou por grupo são aceitos. Mais duas entradas são adicionadas a este arquivo: domínio e servidor de designação.

Os objetos para visualizar os dados do banco de dados, conforme o tipo de dado, são na forma de histogramas, termômetros e no formato texto. A freqüência com que os valores do *widget* são mostrados pode ser alterada. O valor inicial é 5 minutos. Histogramas são usados para mostrar médias, termômetros para dados unidimensionais (percentual/escalar) e texto mostra o nome da máquina sendo monitorizada e o item de dado pedido.

O administrador pode alterar seu ambiente satool (GUI) escolhendo a máquina, tipo de *widget*, limites de alarmes e ações e layout da tela, gravando esta configuração no arquivo `.satoolrc` no seu diretório pessoal.

Além dos objetos de dados satool possui 4 tipos de janelas: *main window*, *viewer window*, *help* e *message window*. A *main window* contém a raiz da hierarquia da máquina. A *viewer window* mostra os membros do grupo e fornece acesso ao usuário para monitorizar os dados. *Help* está disponível pela *Help Menu*. Alertas quando um valor é ultrapassado, objetos que mudam de aparência (cor, vídeo reverso, novo ícone, etc) são mostrados pela *message window*, que também mostra as mensagens de erro.

Extensões:

O satool tem facilidade em adicionar novas variáveis para serem monitorizadas através de mudanças no agente, servidor e GUI. Adicionando-se algumas linhas nos arquivos de configuração destes componentes.

Arquiteturas suportadas:

Agente SNMP (na versão utilizada pelo satool) trabalha somente sobre 4.3 BSD. Os elementos do agente rodam sobre ULTRIX 4.2/4.3 e são portáveis para outras versões de UNIX. A GUI é compatível com qualquer sistema que rode Tcl/Tk. O *daemon* satoold roda em qualquer versão de UNIX que suporte *sockets* Berkeley e `ndbm`.

3.1.3. SAFO

O SAFO [27] (Sistema Agregador de Ferramentas de Operação de rede) é uma ferramenta para monitorização da rede que agrega alguns utilitários (10 ao total) em forma de linha de comando UNIX numa interface gráfica.

O levantamento dos utilitários usados no SAFO foi feito via pesquisa em listas de discussão na Internet. Os utilitários utilizados são mostrados na tabela 3.2:

Comando	**Descrição**
`ping`	Verifica se uma máquina, rede ou interface está ativa, é considerado uma ferramenta básica e imprescindível para o gerência.
`traceroute`	Fornece a rota por pacote
`nslookup`	Programa para consulta de nome de máquina e endereço IP na Internet.
`netstat`	Mostra o estado da rede.
`traffic`	Mostra o tráfego Ethernet graficamente, fornecendo diferentes visões (janelas) do tráfego.
`etherfind`	Examina os pacotes que passam pela interface de rede e armazena a descrição do tráfego num arquivo texto.
`ifconfig`	Faz a configuração dos parâmetros da interface de rede.
`rup`	Fornece o estado das máquinas locais através de um broadcast.
`tcpdump`	Interpreta e escreve num log os cabeçalhos de vários protocolos como ethernet, IP, ICMP, UDP, TCP, NFS entre outros.
`ps`	Mostra o estado dos processos correntes da máquina.

Tabela 3.2 - Comandos UNIX utilizados pelo SAFO

Todos estes utilitários possuem vários parâmetros de configuração e execução.

O SAFO é modular e escrito em C, utiliza para a implementação da interface gráfica o pacote GUIDE (*Graphical User Interface Design*) e o GXV. Fornece uma interface amigável e totalmente direcionada a janelas com botões. Os resultados dos utilitários são redirecionados para janelas.

Fornece um *help on-line* e funções através de botões de:

- Inclusão/exclusão de utilitários (sem necessidade de recompilações, a menos que necessitem de parâmetros de execução);
- Saída (*quit*);
- Redirecionamento de saída para arquivo texto ou *tty*;
- Timer: para cada utilitário pode-se incluir os dados na `crontab` para execução em segundo plano e escalonamento;
- Assistant: auxílio no entendimento quando o resultado de execução do utilitário não é o esperado (com erro).

Na execução pode-se utilizar valores *default* ou não, sendo totalmente configurados através de parâmetros.

3.1.4. Olho Vivo

O objetivo deste protótipo denominado Olho Vivo [28] é fazer uma gerência pró-ativa utilizando monitores remotos e sistema especialista. Ele é composto de monitor remoto RMON (*Remote Monitoring*) MIB e um conjunto de regras de produção, consideradas o bojo do sistema especialista.

Componentes:

1) RMON MIB: foi escolhida por possuir a definição de objetos através dos quais é possivél a realização de uma análise de tendências, fornecendo subsídios para prever a ocorrência de problemas numa rede.

Os grupos que compõem a RMON MIB são listados na tabela 3.3:

Grupo	**Descrição**
Statistics	Contém estatísticas medidas pelo monitor para cada interface monitorizada pelo dispositivo.
History	Registra amostras estatísticas periodicamente e as armazena para posterior recuperação.
Host	Contém estatísticas associadas a cada máquina da rede.
Capture	Permite que pacotes sejam capturados através de filtros e que o sistema crie múltiplos *buffers* de captura.
Event	Controla geração e notificação de eventos.
HostTopN	Utilizado na preparação de relatórios por máquinas ordenadas por estatísticas.
Filter	Permite captura de pacotes com filtros arbitrários.
Alarm	Coleta amostras estatísticas de variáveis do monitor periodicamente e compara com os limites predeterminados.
Matrix	Armazena estatísticas de tráfego e número de erros entre pares de máquinas.

Tabela 3.3 - Grupos RMON

A RMON MIB tem evoluído para RMON II, permitindo o gerência da rede local desde o nível físico até o nível de aplicação.

2) Sistema Especialista: Segundo Artola [28], sistemas especialistas são programas que atuam como consultores inteligentes, permitindo que o conhecimento e a experiência de um ou vários especialistas sejam extraídos e armazenados num computador. O conhecimento é um dos elementos principais de um sistema especialista. As regras de produção descrevem este conhecimento em forma de regras do tipo situação-ação ou se-então. Em gerência de redes, os sistemas especialistas são comumente orientados a diagnósticos e utilizam regras de produção na representação do conhecimento, um método de inferência de encadeamento para frente e um padrão de comparação para localizar o conhecimento relevante.

O sistema OLHO VIVO:

É um sistema que funciona como um vigilante da rede, observando os indicadores de degradação do desempenho da rede através dos conceitos de agentes remotos (RMON MIB) e sistemas especialistas e buscando soluções para os problemas encontrados.

É composto pelo agente RMON, módulo inteligente, um gerente e uma interface. O agente RMON utilizado foi o btng (*Beholder The Next Generation*), ele implementa a RMON MIB completa e possui alguns objetos que o grupo DNPAP, criador deste *software*, adicionou. O `beholder` é um conjunto de coletores, onde cada um é responsável por um grupo de objetos da RMON MIB. Foi feita uma monitorização com o `beholder` para a geração de uma *baseline*.

O módulo inteligente é o principal componente do Olho Vivo, analisando os resultados coletados pelo monitor (`beholder`) e sugerindo soluções ao administrador sobre os problemas de desempenho da rede.

Foram implementadas 9 regras, cada uma referindo-se a problemas específicos da rede como:

regra 1: referente ao número de pacotes descartados pelo monitor.

regra 2: referente ao nível de *broadcast*, devendo este nível ser menor que 8%.

regra 3: referente ao nível de *multicast*.

regra 4: erros de alinhamento ou CRC, a taxa de erros deste tipo deve ser inferior a 2% do tráfego total.

regra 5, 6 e 7: avaliam contadores de pacotes com erro de comprimento.

regra 8: taxa de colisões.

regra 9: taxa de utilização, não deve ultrapassar 40%.

O gerente utilizado foi o software SNM (SunNet Manager da Sun MicroSystems) e a interface com o administrador é o correio eletrônico. Quando a regra é verdadeira pela primeira vez envia um mail indicando o problema e o nome do arquivo de *log* (*baseline*).

3.1.5. HLMMIB

O HLMMIB [29] (*Host Load Metrics MIB*) é uma extensão da MIB, específica para a manutenção de carga do sistema para a administração de um sistema distribuído. HLMMIB realiza medidas de uso de CPU, memória por exemplo, para ter-se uma visão global de uso dos recursos do sistema. Pode ser usado para decisões de configuração ou para um mecanismo de balanceamento de carga.

Uma biblioteca também foi definida baseada na HLMMIB, chamada HLMLIB.

Agente SNMP HLMMIB:

Foi realizada uma extensão de um agente SNMP fazendo uma espécie de subagente, estendendo a MIB definida especificamente para a monitorização de carga das máquinas e dando suporte a uma aplicação gerente de balanceamento de carga.

Este subagente registra-se junto ao agente e informa que é responsável por uma parte da MIB (HLMMIB). Quando os pedidos dos gerentes relacionados a algumas destas variáveis da HLMMIB são recebidos pelo agente, ele repassa ao subagente os pedidos. O subagente executa a ação e devolve ao agente o resultado que será repassado ao gerente.

A HLMMIB define grupos de variáveis relacionados com a CPU, memória física, memória virtual, interfaces de rede e unidades de armazenamento. Cada grupo deve conter características de arquitetura, parâmetros de configuração e estatísticas de utilização.

Para obter as informações de carga acessou-se as estatísticas que o UNIX mantém sobre o comportamento e utilização dos recursos de *hardware* como CPU, discos, memória,

etc., pode-se usar estes dados como medida de ociosidade da máquina servindo como base para um balanceamento de carga.

As informações para a implementação do agente SNMP HLMMIB são obtidas acessando estas estatísticas mantidas pelo UNIX diretamente do núcleo do sistema, algumas delas são: número de tarefas na fila de processos prontos, quantidade de memória livre, taxa de *page faults*, taxa de chamadas ao sistema, taxa de trocas de contexto, tempo CPU livre durante intervalo determinado, taxa de acesso ao disco. Foi utilizado na implementação da extensão do agente SNMP o pacote CMU-SNMP (Carnegie Mellon).

Biblioteca HLMLIB:

A biblioteca HLMLIB (*Host Load Metrics Library*) foi escrita em C++ e modela os dispositivos remotos gerenciados (memória, CPU, disco, ...) e é independente de protocolo. Ela funciona como uma camada intermediária entre as aplicações gerentes e os protocolos de gerência oferecendo uma interface de mais alto nível. Uma aplicação gerente usa esta biblioteca através de sua interface para implementação de aplicações chamada HLMAPI.

Aplicação:

Foi implementada uma aplicação para utilização desta biblioteca utilizando PVM (*Parallel Virtual Machine*) e visando o balanceamento de carga usando os dispositivos ociosos da rede.

3.1.6. Buzzerd

BUZZERD [30] é um sistema de monitorização e notificação de sistemas de rede projetado para detectar problemas e fornecer informações de diagnóstico de máquinas na rede com o objetivo de minimizar o custo de *downtime*. Este nome foi escolhido pois um *pager* digital pode vibrar (som buzzz...) quando recebe uma mensagem. O sistema buzzerd é dividido em monitorização de sistema e notificação de problema, onde a monitorização é feita pelo subsistema denominado `sysmond` e a notificação é realizada através do *daemon* notificador `buzzerd`.

`Sysmond` é executado em cada máquina da rede e periodicamente executa programas para verificar características específicas do sistema (monitores). Quando `sysmond` detecta um problema, retransmite a informação para o *daemon* `buzzerd`. Este

processo transmite a mensagem ao administrador por correio eletrônico, *pager* digital, *pager* alfanumérico ou registro em *log*. A informação é registrada e caso volte a ocorrer o administrador é novamente notificado. As notificações ao administrador podem ser configuradas dependendo do tipo de severidade do problema, hora do dia que ocorreu, tendo para isto um arquivo de configuração para cada administrador.

O sistema possui alguns utilitários como:

- `ibuzz`: programa interativo que faz o controle dos *daemons* `sysmond` por máquina. Realiza as tarefas de mudança de configuração (parâmetros), adição de monitores, atualizações dos *daemons* e instalação.

- `ackpage`: programa que permite acesso as informações de diagnóstico e dados dos problemas notificados pelo `buzzerd` ao administrador, sendo eliminados da lista depois de resolvidos.

- `checkind`: programa que garante que todos os processos `sysmond` estejam rodando.

- `ezpage`: utilitário que permite que usuários enviem mensagens (*pager*) para uma pessoa ou problema específico, fazendo breve descrição do problema.

Subsistema sysmond:

O subsistema de monitorização (sysmond) foi projetado com duas funcionalidades: um conjunto de monitores para detectar e analisar diferentes problemas como espaço em disco, conexões de rede, carga média do sistema, entre outros e um *daemon* que executa os monitores em intervalos regulares e transmite, quando necessário, informações de diagnóstico ao *daemon* centralizado `buzzerd`.

O conjunto de monitores e o *daemon* do sysmond usam um arquivo de configuração para manutenção, configuração e instalação do sysmond nas máquinas da rede chamado `sysmond.cf`. O `sysmond.cf` é um arquivo texto com 3 campos: o programa que usa o parâmetro, o nome do parâmetro e o(s) valor(es) do parâmetro. A freqüência de execução e quais monitores o *daemon* `sysmond` deve utilizar, parâmetros globais como onde os *daemons* `buzzerd` e `checkind` estão, são informações que estão também no `sysmond.cf`.

Uma vez que os monitores do sysmond detectam e analisam o problema, esta informação deve ser passada ao buzzerd desta forma: nome da máquina que ocorreu o problema, qual monitor detectou o problema, severidade do problema (como estimada pelo monitor), diagnósticos coletados pelo monitor. Através de um código de problema, cada máquina, monitor e severidade possui um código único. Este código é associado a informação pelo buzzerd.

Para escrever monitores para serem integrados no sysmond, o monitor deve retornar um nível de severidade, alguma informação de diagnóstico como saída e via `syslog` e deve utilizar o `sysmond.cf` para qualquer parâmetro de configuração.

O subsistema sysmond possui uma pequena biblioteca com facilidades compatíveis com estes requisitos que contém procedimentos para armazenar informação de diagnóstico, calcular níveis de severidade, extrair parâmetros do `sysmond.cf` e extrair variáveis SNMP.

O sysmond foi escrito em C e implementado como um *daemon* que periodicamente divide o processo em processos filhos para manipular a execução de monitores e possíveis contatos com buzzerd. O sysmond executor é utilizado para fazer esta divisão/manipulação de processos, utilizando também um *timeout*. O conjunto de monitores e a biblioteca sysmond foram escritas em Perl e C. O sysmond também monitora o `syslogd`.

Subsistema buzzerd:

Utiliza dois subsistemas primários de comunicação via *sockets* TCP/IP e um protocolo buzzerd. O *daemon* notificador (`buzzerd`) envia mensagem ao administrador, mantém um banco de dados dos incidentes e retransmite a mensagem quando o problema reaparece ou é desconhecido. Suporta várias conexões de rede ao mesmo tempo e não permite o envio de mensagens desnecessárias. Utiliza códigos de problemas para converter os problemas reais para serem enviados via *pager* digital. O código é definido como uma série de 3 dígitos separados por * ou -, que identificam a máquina com problemas, o monitor que descobriu o problema e a severidade da situação.

O banco de dados de problemas é composto por informações como: único `page-id`, data e hora do problema, código do problema, nome da pessoa notificada, número de vezes que foi notificada, tempo para reiniciar a notificação e outras informações adicionais do problema.

Através do arquivo de configuração `buzzerd.cf` é determinada para quem será enviada a notificação e qual método será usado (*log*, correio eletrônico, *pager* digital, *pager* alfanumérico). O uso de código de problema único (máquina*monitor*severidade) funciona como um filtro para a checagem no banco de dados evitando a ocorrência de retransmissões desnecessárias de mensagens, quando possível. O buzzerd utiliza modem para enviar mensagens aos *pager*.

O protocolo buzzerd:

ADD <código do problema> : relata um problema

DELETE <page> <usuário> : resolve um problema

INFO <page> <data> : fornece informações adicionais sobre o problema

LIST : mostra todos os problemas correntes

ONCALL : lista a pessoa que receberá a notificação

HELP : mostra uma ajuda básica do protocolo buzzerd

QUIT : fecha a conexão

buzzerd foi escrito em C e implementado como um *daemon* o qual divide o processo em processos filhos para manipular conexões. Cada processo filho comunica-se com o processo pai usando pares de *sockets* (para simplificar a ação do processo filho na adição do banco de dados). O banco de dados de mensagens aos *pager* é mantido como uma lista na memória com armazenamento em arquivos em caso de falhas. O arquivo de configuração é um arquivo texto.

3.1.7. I-DREAM

Sistema I-DREAM (Intranet-baseD REsource and Application Monitoring) [31] baseado em *intranet* para a monitorização de redes heterogêneas de computadores. A interface do I-DREAM é baseada em páginas *web* que são geradas dinamicamente através de informações armazenadas em banco de dados. O sistema é modificado e estendido quando o banco de dados é atualizado permitindo que vários eventos sejam automaticamente detectados.

Características:

- Facilmente modificado para refletir a evolução da rede monitorizada.
- Usado de qualquer máquina na *internet* para monitorizar uma rede local com segurança.
- Permite várias visões da rede para diferentes classes de usuários.

Modelo conceitual:

Conceitualmente o I-DREAM é composto de 5 subsistemas: controle, definição, visualização, análise e diagnóstico da informação (DiaAnalysis) e atuação e coleta da informação (ActGather). Os subsistemas são organizados em dois níveis: controle, definição e visualização compõem o nível superior e DiaAnalysis e ActGather compõem o nível inferior. Existem duas categorias de usuários (que somente acessam o I-DREAM pelo nível superior): usuários de rede e administradores de sistemas. Os usuários de rede acessam somente informações disponíveis e não podem modificar dados. Os administradores tem acesso privilegiado (mecanismo de `login/password`) e podem alterar definições e funções de controle do sistema.

Arquitetura:

A arquitetura tem 5 componentes: interface usuário/sistema (USI), DBMS (Data Base Management System), gerente, ToolBox e sociedade de agentes. A arquitetura modular foi derivada do modelo conceitual, desta forma:

- Módulo interface usuário/sistema é derivado dos subsistemas controle, definição e visualização.
- Módulo gerente é derivado do subsistema controle.
- A funcionalidade de banco de dados dos subsistemas definição e ActGather é obtida pelo DBMS.
- As funcionalidades de solução, identificação e diagnóstico dos problemas do subsistema DiaAnalysis é obtida pelo agente de raciocínio da sociedade de agentes.

- A funcionalidade de coleta de informação do subsistema ActGather é obtida pelo gerente, ToolBox e agente de monitorização da sociedade de agentes.

- A funcionalidade de corrigir problemas é obtida pelo agente de execução da sociedade dos agentes.

A sociedade dos agentes é composta de 3 tipos de agentes:

- Agentes de monitorização: são agentes que monitorizam a rede para coletar informação sobre seu comportamento atual e armazenam esta informação no banco de dados.

- Agentes de raciocínio: são agentes inteligentes que encontram os problemas comparando o comportamento atual da rede com a informação do comportamento esperado da rede.

- Agentes de execução: são agentes que realizam as ações demandadas pelos agentes de raciocínio para fazer a rede funcionar como o esperado.

O módulo gerente utiliza as ferramentas de monitorização do ToolBox para monitorizar os recursos da rede. Estas ferramentas são utilizadas pelos agentes de monitorização para coletar informações do comportamento real da rede. O administrador pode iniciar ou parar a monitorização através do módulo gerente. Os agentes interagem diretamente com o módulo de banco de dados para armazenar a informação monitorizada. O ToolBox armazena as ferramentas de monitorização utilizadas pelo I-DREAM.

O gerente acessa a lista de recursos de rede que devem ser monitorizados pelo DBMS e é responsável pela ativação do agente de execução quando o agente de raciocínio identifica que uma ação deve ser acionada para corrigir o problema. O gerente atua como um *daemon*, que permanece a espera para acionar ou parar agentes de monitorização ou execução ou verificar se os agentes estão no ar.

A interface usuário/sistema permite controle sobre o comportamento do sistema ativando, ajustando e desativando ferramentas de monitorização e agentes. É também um *front-end* do DBMS. Ela é a interface de acesso dos usuários e administradores, permitindo através de mecanismos de autorização que somente administradores interajam com o DBMS e que usuários somente possam visualizar a informação.

Modelo navegacional:

Permite navegação vertical e horizontal através dos recursos monitorizados.

O I-DREAM foi desenvolvido utilizando as linguagens Perl, Java, HTML e CGI. Pode ser executado nas plataformas IBM/AIX, Sun/Solaris e Sun/SunOS.

3.1.8. SAGRES

O SAGRES [32] é um sistema baseado em conhecimento projetado para a área de gerência de sistemas[6] . O SAGRES possibilita a coleta, a análise, o tratamento e a geração de ações relacionadas ao comportamento da rede, como conseqüência de regras previamente cadastradas em uma base de conhecimento, para tanto foram utilizados os conceitos de SGRBCs (Sistemas de Gerenciamento de Redes Baseado em Conhecimento). Esse sistema segue o modelo de gerência de redes adotado na *internet*, o SNMP v2 (Simple Network Management Protocol versão 2).

O SAGRES é formado por 9 componentes. A descrição de cada um de seus componentes é relacionada a seguir:

- Usuário: Pode ser usuário ou administrador do sistema. Necessita de uma conta e senha para acessar as funcionalidades do sistema. O administrador deve fazer a manutenção de todas as contas do sistema.

- MIB: Repositório de dados contendo todas as informações dos objetos gerenciados da rede.

- Coletor de Dados: Realiza a coleta dos dados da rede e os armazena numa base de dados para posterior visualização em forma textual ou gráfica. Este módulo coleta dados da MIB utilizando comandos GET.

- Tratamento dos Dados: É alimentado pelo módulo Coletor de dados e os dados são tratados antes de serem disponibilizados ao administrador ou usuário em forma de informação.

[6] Gerência de sistemas é a aglutinação das tarefas de administração de sistemas e gerenciamento de redes heterogêneas [32].

- Visualização de Dados: É alimentado pelo módulo de Tratamento de dados.
- Cadastro de Regras: É responsável pela interação entre administrador e Base de Conhecimento (BC), é através deste módulo que as inclusões, alterações e remoções de ações e regras na BC são realizadas.
- BC (Base de Conhecimento) : É o repositório das ações e regras do sistema.
- Analisador de Regras: Utiliza os dados coletados, ações e regras cadastradas para detectar e corrigir problemas no comportamento da rede.
- Ação na MIB: Faz as alterações nos parâmetros da rede mediante o processo de inferência sobre as regras do módulo Analisador de Regras.

A aplicação SAGRES é implementada parte em ambiente UNIX e parte em PC. A linguagem Delphi foi utilizada para o ambiente PC e a linguagem Tcl/Tk (Tcl versão 7.6 e Tk versão 4.2) e a ferramenta Scotty (versão 2.1.5) foram utilizadas para o ambiente UNIX.

3.2. Pacotes comerciais

Foram pesquisados através da WWW (World Wide Web) 11 pacotes proprietários de empresas que trabalham na área de gerência de rede que são: Transcend, Optivity, Solstice, CiscoWorks, OpenView, Spectrum, Tivoli Enterprise, SymbEl, Big Brother, Unicenter TNG e SiteScope.

3.2.1. Transcend da 3COM

O Transcend pode ser assim dividido: Transcend Enterprise Manager Version 4.2 para UNIX na gerência de dispositivos fim-a-fim, Transcend Traffix Manager para UNIX na gerência RMON/RMON2 e Transcend LANSentry Manager.

Características comuns aos produtos Transcend:

- Agentes de gerência inteligentes SmartAgent: O *software* SmartAgent está embutido em cada dispositivo da 3COM. Automaticamente checa e monitoriza os dispositivos e retorna ao console de gerência um relatório quando uma exceção ocorre. Este agente adiciona inteligência as MIBs padrão, assim

dispositivos podem ser gerenciados remota e pró-ativamente. Através da característica Autocalibrate configura limites de alarmes automaticamente, baseado no nível de tráfego atual. Passando somente dados selecionados para o console, diminuindo o tráfego em toda a rede. Combinando SmartAgent com RMON, Transcend continuamente diagnostica e cria registros (*log*) de desempenho da rede.

- Inteligência fim-a-fim: o *software* TranscendWare está embutido em produtos de acesso 3COM que coletam dados de gerência, otimiza desempenho de rede e aplica políticas configuradas pelo administrador. Pode-se ainda gerenciar: virtual LAN (VLAN), emulação de LAN (LANE) e ATM.

1) Transcend Enterprise Manager Version 4.2 para UNIX :

Transcend Enterprise Manager para UNIX [33] é um sistema de gerência integrado de rede, consolidado para configuração, monitorização e solução de problemas de concentradores, *switches* corporativos, *bridges*/roteadores para qualquer plataforma.

Características:

- Gerencia plataformas como: Solaris SunNet Manager, HP OpenView e NetView para AIX. [34]

- Constrói com a utilização do SmartAgent tarefas de gerência para dispositivos 3COM baseadas em agentes no padrão SNMP. O *software* SmartAgent protege a rede com mecanismo de acesso multinível e multiusuário que previne acessos não autorizados.

- Realiza tarefas para consolidar atividades de rede tais como administração de dispositivos, gerência RMON para diversos dispositivos e distribuição remota de *software* e configuração através da rede.

- Configura dispositivos para monitorizá-los através do uso de tecnologia RMON, configurando limites para volume de tráfego, erros e congestionamento de acordo com o dispositivo, segmento de rede ou grupo virtual.

- Possui orientação por mapas de cada dispositivo que detalha todas as portas do dispositivo e o estado em tempo real.

- Possui acompanhamento de alarmes para registro e organização em *log* de eventos para análise e visualização.

- Faz controle dos dispositivos remotamente em tempo real ou escalonado, *download* remoto de agentes atualizados usando TFTP (*Trivial File Transfer Protocol*), uso de acesso por `Telnet` para gerência de parâmetros de sistemas 3COM e também de dispositivos de outras marcas.

- Faz visualização gráfica de dados RMON e obtenção de estatísticas de segmento para isolar problemas ou estabelecer ajustes de *benchmarks*.

- Utiliza também a aplicação LANSentry que será descrita a seguir.

2) Transcend LANsentry Manager:

Transcend LANsentry Manager [35] é um conjunto de ferramentas RMON para monitorização e resolução de problemas que possui uma visão gráfica da LAN e fornece informações da atividade da rede. Disponível para Windows e UNIX, pode-se monitorizar tráfego, analisar pacotes e diagnosticar problemas que afetam o desempenho da rede. Para analisar e entender a rede pode-se visualizar os dados em tempo real ou visualizá-los na forma de gráficos, *dials* e tabelas, além de informações obtidas por cliques em componentes gráficos. Pode monitorizar redes *Token Ring*, *Ethernet*, FDDI ou *Fast Ethernet* ao mesmo tempo.

Características:

- Resolução e detecção de falhas: filtragem e captura de pacotes num formato cliente/servidor. Com a interface gráfica faz-se a configuração dos filtros na captura de pacotes.

- RMON completo: LANSentry suporta todos os 10 grupos RMON. Que são: Statistics (RMON Group 1), History (RMON Group 2), Alarm e Event (RMON Groups 3 & 9), Hosts e Host TopN (RMON Groups 4 & 5), Matrix (RMON Group 6), Filter e Capture (RMON Groups 7 & 8), Token Ring (RMON Group 10).

- Decodificação de pacotes: decodifica o pacotes capturados das sete camadas da rede e mostra o conteúdo dos mesmos através de uma janela em 3 níveis de

detalhe: sumário de informações, pacotes decodificados e conteúdo do pacote em hexadecimal.

- Tradução de endereço: através do SmartAgent e RMON2 no *probe*, pode-se construir uma tabela de endereços MAC (*Media Access Control*) para endereços de rede de qualquer pacote e traduzí-los para nomes e endereços IP. Auxilia na eliminação de IPs duplicados com detecção e notificação automática. Gera relatórios históricos e periódicos usando as informações do banco de dados obtidas pelo componente chamado Collector.

- Geração de tráfego: oferece, somente para plataformas UNIX, a funcionalidade de teste de estresse de tráfego em segmentos de rede *Token Ring* e *Ethernet* a partir de uma estação central. Podendo-se com esta geração, capturar e analisar pacotes. LANSentry para Windows executa em modo *standalone* ou integrado com HP OpenView para Windows. E para UNIX também executa em modo *standalone* ou integrado a plataformas de gerência como SunNet Manager, HP OpenView, e IBM NetView.

3) Transcend Traffix Manager para UNIX:

RMON [36] é uma MIB definida pela *Internet Engineering Task Force* (IETF) e IETF RFC 1757 (Ethernet) e RFC 1513 (*Token Ring*) fornece estatísticas da camada MAC bem como filtros e captura de pacotes por análise de protocolos. RMON2 (*Remote Monitoring 2*) é um extensão da MIB definida pela IETF RFC 2021 e RFC 2074 que adiciona estatísticas de rede e aplicações de camada da pilha de protocolos. RMON2 permite realizar medidas de tráfego acima da camada 3, inclusive. Transcend Traffix Manager coleta e relaciona dados de múltiplos *probes* RMON/RMON2 para mostrar uma visão global do tráfego da rede para um gerência de desempenho, análise de tendências e resolução de problemas. Com sua interface mostra o padrão de comunicação em detalhes das aplicações cliente/servidor que estão usando a rede.

Características:

- Coleta de informação: usando dados de dispositivos RMON e RMON2, bem como agentes RMON e RMON2 embutidos nos dispositivos de rede 3COM, pode-se gerenciar o fluxo do tráfego e as informações são organizadas por conexão.[37]

- Agrupamento lógico da rede: pode-se agrupar os objetos de rede no mapa de acordo com o usuário e suas necessidades.

- Banco de dados relacional: permite gerenciar uma grande quantidade de dados, sendo coletados automática e continuamente para todos os protocolos e aplicações e acessados a qualquer momento. Suporta também interface SQL (*Standard Query Language*).

- Resolução de dados através da maximização de espaço em disco: a resolução de dados pode ser personalizada para armazenar e retornar os dados coletados, alocando espaço igualmente entre escolhas de resolução de dados (de hora em hora, de 6 em 6 horas, diariamente e semanalmente). Checa o espaço em disco e mostra a taxa de dados coletados, usados e a taxa estimada de espaço em disco disponível para cada resolução.

- Escalabilidade para crescimento da rede através de agregação e filtragem dos dados: dependendo do tamanho da rede, a taxa de dados coletada pode ser grande e consumir espaço demasiado no banco de dados. O Traffix Manager possui duas opções para escalonar estes dados: agregação e filtragem de dados. Agregação de dados consolida um intervalo de endereços de rede sobre um simples dispositivo. Filtragem de dados limita a taxa de dados coletados pelo *probe* em um valor predeterminado.

- Relatórios: pode-se analisar a rede em vários níveis através de relatórios em forma de gráficos, desde um dispositivo até vários segmentos de acordo com a necessidade. A saída dos mesmos pode ser direcionada para HTML ou *postscript*. Os tipos de relatórios disponíveis são: atividade de conexão, atividade de dispositivo, atividade de grupo, atividade de segmento, conexões TopN, dispositivos TopN, grupos TopN e segmentos TopN.

- Gráficos flexíveis: gráficos podem ser plotados em qualquer nível da rede fornecendo uma visão histórica da mesma.

3.2.2. Optivity da Bay Networks

A família Optivity [38] é composta de um conjunto de aplicações, logicamente conectadas que fornecem características de gerência para todo o ambiente de rede. Baseadas em padrões da indústria como SNMP, RMON e RMON2 e com a característica de topologia multicamada.

Para a plataforma UNIX e para grandes redes, o Optivity Enterprise inclui: Optivity LAN, Optivity Internetwork, Optivity Analysis e Optivity Planning. Individualmente cada componente realiza uma aplicação específica: Optivity LAN para monitorização, configuração e solução de problemas de *hubs*, *switches* e redes locais; Optivity Internetwork para roteadores, WANs e *internetworks*; Optivity Analysis e Optivity Planning para análise de tendências e projeto de rede.

A habilidade para monitorizar e registrar fluxos de tráfego, executar análise de tendências, antecipar "gargalos" e simular topologias de redes requerem um gerência no nível de sistema e são alternativas antes de realizarem-se mudanças físicas na rede. O sistema de gerência de rede Optivity está focalizado no nível de sistema.

O Optivity através da inteligência distribuída na forma de agentes de *software* desempenha 3 funções críticas na gerência:

1) Os agentes monitorizam uma parte específica da rede, capturando, processando e condensando os dados na fonte para reduzir o tráfego entre agentes/servidores.

2) Os agentes comunicam-se entre si para construir um completo banco de dados da topologia multicamadas da rede, descobrindo não somente relacionamentos físicos entre os dispositivos mas também conexões lógicas e virtuais através da rede. Relatando os inter-relacionamentos complexos entre dispositivos, subredes e LANs virtuais (VLANs).

3) Os agentes continuamente monitorizam o tráfego baseado em padrões existentes como RMON e RMON2, acumulando e traçando o desempenho e estatísticas de erro fornecendo visibilidade da rede.

Características:

- Apresenta uma interface baseada em gráficos, fornecendo janelas com gráficos da rede para possibilitar a monitorização e controle do tráfego cliente/servidor

por locação. Através da GUI pode-se habilitar e desabilitar partes da rede, alterar políticas para detectar e relatar falhas, observar o fluxo de pacotes através da empresa. E também oferece um controle completo sobre *hubs*, roteadores e *switches*. Estas tarefas evitam o deslocamento de pessoal técnico localmente para solucionar os problemas.

- Visões físicas e lógicas, isolamento de falhas e resolução de problemas, gerência de configuração global, traçado de rotas e monitorização de aplicações são algumas outras características desta ferramenta.

- A topologia multicamada localiza todas as entidades na rede e caracteriza completamente suas relações de conectividade através da visão lógica, física e virtual da rede. Pode-se mudar instantaneamente de visão e ter informações de IP da subrede e detalhes das máquinas da rede como conexões, número de placa e portas da interface.

- Optivity é integrado com várias plataformas existentes no mercado. Para redes maiores, Optivity Enterprise trabalha com HP Open View Network Node Manager, Tivoli NetView 6000 e SunConnect Solstice Domain Manager.

- Inteligência distribuída: coleta e reduz dados onde eles são gerados e envia somente informações críticas, é a chamada inteligência distribuída que diminui a carga do processamento em recursos centralizados.

Aplicações comuns:

- NETarchitect: para a flexibilidade na configuração, Optivity NETarchitect realiza complexas questões de gerência de configuração no nível de sistema. Consiste de duas ferramentas básicas orientadas a sistemas: o File Manager e o Configurator Editor.

- Enterprise Health Advisor: faz a gerência pró-ativa. Identifica e isola problemas de rede e pode-se imediatamente tomar uma ação corretiva. Fornece o estado de todos os roteadores em uma simples visão, oferece uma variedade de ferramentas de diagnóstico para encontrar e resolver problemas de roteamento.

- Built-in Network Tap (NTAP) e Packet Capture (PCAP): provêem uma característica de captura remota de pacotes que torna a estação UNIX uma ferramenta para análise de protocolos remoto.

- Rede Virtual: permite a configuração da rede e alocação de recursos, trazendo o conceito de rede virtual. A aplicação Network Atlas, por instância, utiliza a autotopologia para apresentar uma visão global da rede, permitindo desempenhar monitorização e solução de problemas. Mostrando o relacionamento entre visões de rede lógica e física, Network Atlas visualiza e resolve problemas na camada escolhida.

- Centro de comando e gerência orientado a objeto: através da característica de topologia multicamada, Optivity descobre todos os dispositivos da rede, bem como seus relacionamentos físicos e lógicos. Estes dados são mostrados através do centro de comando orientado a objetos oferecendo um sumário das locações da rede. Selecionando uma locação que revela uma série de pastas representando todas WANs, LANs e subredes, bem como *hubs*, *switches*, *switches* ATM, roteadores e *probes* RMON. Com a interface orientada a objeto não há necessidade de conhecer a MIB para obter detalhes de falha, desempenho e dados de configuração.

- SuperAgent: a tecnologia do SuperAgent atua na consolidação dos dados em um console de gerência que permite empregar gerentes de domínio no nível médio através da rede. Estes gerentes de nível médio coordenam as atividades dos agentes dos dispositivos de nível baixo, consolidando as tarefas altamente repetitivas tais como analogia de falha, resolução de topologia, monitorização de roteadores e determinação do estado dos dispositivos.

- Agentes Advanced Analyzer: coletam dados de falhas, desempenho e configuração no nível de porta, habilitando funcionalidades como interface gráfica Expanded View e análise de desempenho no nível de dispositivo. O agente Advanced Analyzer também fornece detalhes da configuração da rede através da característica de mapeamento dinâmico - Autotopologia. E também suporta 4 grupos RMON permitindo análise de desempenho e formação de *baseline* pela empresa. Para um gerência adicional o agente Advanced Analyzer fornece total funcionalidade RMON e RMON2 que dá uma visibilidade total na rede.

- Advanced Analyzer: faz a resolução de problemas remotos da rede. Dispositivos remotos equipados com este agente podem monitorizar e solucionar problemas de uma rede a partir de uma GUI. O agente Advanced Analyzer fornece ampla visibilidade na atividade da rede, permitindo encontrar, diagnosticar e resolver problemas rapidamente.

- Optivity Internetwork: as aplicações PathMan, RouterMan e SiteManager do Optivity Internetwork eliminam a adivinhação, auxiliando na identificação e resolução de problemas rapidamente. Usando o PathMan para traçar o caminho dos dados entre o usuário e o servidor para identificar roteadores com problemas. RouterMan alerta imediatamente que o dispositivo está com falhas na interface. Site Manager permite reconfigurar o roteador para desviar o tráfego até a interface ser consertada. Toda esta operação é realizada pelo console de gerência.

- Optivity Analysis: como as redes atualmente são muito distribuídas e necessita-se controlá-las através de uma locação central, Optivity Analysis possui ferramentas RMON que fornecem capacidades de monitorização e resolução de problemas. Permitindo examinar o tráfego ou fazer um *zoom* em pontos específicos de qualquer parte da rede. Ferramentas RMON2 fornecem um nível de visibilidade de como o tráfego está movimentando-se pela rede. TrafficMan oferece características RMON2 provendo uma visão da camada de rede a camada de aplicação, detalhando quais dispositivos estão comunicando-se. Analisando os padrões de tráfego através da rede, TrafficMan monitoriza a atividade cliente/servidor de qualquer lugar da rede, incluindo locações remotas, eliminando a necessidade de viagens e de técnicos ao local com problemas.

- DecodeMan oferece protocolo completo para as sete camadas, decodifica solução de problemas para cliente/servidor, permitindo examinar todo tráfego ou focar um problema específico. Quando informações mais detalhadas são necessárias, a aplicação NodalView oferece uma ferramenta para coletar falhas baseadas em RMON no nível de estação e estatísticas de desempenho tais como utilização, erros ou distribuição de protocolos. OmniView permite observar e comparar tráfego de rede e dados de diagnósticos por múltiplos *hubs*, anéis, *slots* e portas. As ferramentas RMON do Optivity Analysis fornecem uma visibilidade para monitorizar e analisar a rede em tempo real.

- Optivity Planning: as aplicações de Optivity Planning - NetReporter e DesignMan - coletam e analisam dados *baseline* da rede, para planejar uma evolução da rede. NetReporter com a inteligência distribuída através da rede coleta dados de desempenho detalhados, fornecendo uma idéia para planejar um crescimento futuro. Coletas flexíveis e opções de saída, incluindo acesso a WWW, permitem produzir relatórios revelando tendências para planos de futuras expansões da rede.

- DesignMan utiliza dados históricos para auxiliar na identificação de configurações ótimas da rede. Juntamente com o agente Advanced Analyzer, a aplicação ajuda a identificar os padrões de tráfego de rede relatando informações sobre *top talkers*, caminhos de alta utilização e comunicação remota X local. Usando esta informação e uma ferramenta de simulação integrada, pode-se experimentar várias configurações para encontrar a melhor distribuição possível de roteadores, *hubs*, usuários e outros recursos, antes de realizar qualquer mudança.

3.2.3. Solstice da Sun Microsystems

Solstice [39] é uma família de *software* que oferece soluções para *mail/messaging*, TMN (*Telecommunications Management Network*), segurança, integração de computadores pessoais, conectividade e gerência de redes. Na área de gerência de redes UNIX está dividida em dois produtos: Solstice Site Manager e Solstice Domain Manager. Todas as ferramentas do SunNet Manager estão incorporadas no Solstice Site Manager e no Solstice Domain Manager.

Solstice Site Manager: inclui o SunNet Manager 3.3 e o Soltice Cooperative Console que permite gerenciar dados (topologia, eventos e *traps*) para ser enviado ao Solstice Domain Manager.

Solstice Domain Manager: ferramenta para grandes locações ou gerência de múltiplas locações. Inclui também a versão completa do Solstice Cooperative Console e Layout Tool avançado. E também inclui as mesmas ferramentas do Solstice Site Manager.

Configurações do Solstice Site Manager e Solstice Domain Manager:

Solstice Domain Manager é usado em três configurações gerais:

a) Como uma plataforma *standalone* para gerência de grandes locações.

b) Como um gerente central onde sistemas Solstice Site Manager estão conectados como distribuidores para um ou mais sistemas Solstice Domain Manager.

c) Como múltiplos sistemas Solstice Domain Manager interconectados como plataformas de gerência cooperativas que enviam e recebem informações entre elas.

Características do Solstice Site Manager e Solstice Domain Manager:

Agentes proxy: a distribuição da carga de gerência é feita através do uso de agentes proxy. Estes processos (agentes proxy) são melhor caracterizados como gerentes de nível médio, pois eles atuam como mini-gerentes em dispositivos gerenciados.

A funcionalidade do proxy é:

a) Engenharia de protocolo: o proxy pega os pedidos de serviços de gerência interna e traduz para pacotes de protocolos de gerência de rede correspondente, pedidos pelo dispositivo gerenciado. Por exemplo: SNMP, CMIP (*Common Management Information Protocol*).

b) Gerência de *Polling/Thresholding*: o console de gerência delega muitos pedidos para os processos proxy. Os proxy tem a habilidade de tornar um pedido de gerência simples (por exemplo: coletar BER a cada 60 segundos) e gerar um dado apropriado requerido pelo dispositivo gerenciado. Isto reduz o tráfego de pedidos no console de gerência centralizado. O proxy também faz verificação de limite localmente (por exemplo: avise quando BER exceder de X). Somente quando a condição limite é alcançada o proxy envia um pacote de resposta ao console central, com isto o tráfego entre o console e o proxy fica somente com pedidos simples e respostas de alarmes.

Agentes de estatísticas de CPU: possuem agentes de estatísticas de CPU gerenciando plataformas Solaris. Estes agentes são baseados em agentes RPC, remotos ou locais, que coletam estatísticas de plataformas que rodam Solaris 2.X com um ou mais processadores. O administrador pode monitorizar estatísticas de carga de CPU como percentual de tempo em modo usuário, estatísticas de sistema como número de chamadas de sistemas e estatísticas de memória virtual como taxa de paginação que está ocorrendo. Monitorando desta forma o desempenho de servidores Solaris.

Distribuição de Console: é a chave para distribuir a administração entre várias locações usando para isto os agentes proxy que permitem a gerência remota. O agente proxy minimiza a taxa de pedidos de informação que circula pela WAN, alertando somente quando problemas ocorrem.

A distribuição de console é baseada na tecnologia de Solstice Cooperative Console que fornece ao usuário do Solstice Domain Manager um modelo de dados distribuído para gerenciar cooperativamente médias e grandes redes.

Solstice Cooperative Console envia a rede as atualizações que ocorrem em uma plataforma para outra plataforma. A segunda plataforma verá todas as mudanças nos mapas de topologia e também eventos ou *traps* que ocorrem na rede. Eventos ocorrem quando condições predefinidas são encontradas como a utilização de CPU de certos computadores excederam a um nível percentual específico. *Traps* ocorrem quando eventos não definidos acontecem como uma conexão da rede local que cai, por exemplo.

A distribuição de console é segura através de uma lista de controle de acesso (dos consoles que podem receber informações) filtrando a taxa e tipos de informações entre as plataformas.

Gerência de eventos e dados: basicamente existem 2 diferenças entre os tipos de pedidos que os agentes proxy podem enviar:

a) Pedidos de dados: uma aplicação de gerência pede a um agente que informe o conteúdo de um grupo ou tabela em um certo período do intervalo. Por exemplo: poll do agente hostperf a cada 60 segundos, recuperar a informação e devolvê-la ao usuário.

b) Pedidos de eventos: checagem do agente hostperf a cada 60 segundos e se um certo atributo encontra algum critério, como a utilização de CPU é maior do que 90%, então enviar um relatório. Um relatório de evento somente será gerado quando um critério é encontrado.

Ações baseadas em eventos: Solstice Domain Manager fornece um gerência de eventos para monitorizar a saúde da rede e seus dispositivos, enquanto provê ferramentas para construir os pedidos de eventos. Com a característica de ações baseadas em eventos, quando eventos predefinidos ocorrem, um pedido de evento subsequente pode ser enviado. O administrador pode escolher concatenar múltiplos pedidos predefinidos para resolver problemas rapidamente.

Esta característica não limita-se a verificar se um dispositivo está fora ou não. Pode-se configurar pedidos de eventos para monitorizar a saúde de cada elemento. E também pode-se parar o pedido de evento, permitindo que tais pedidos de eventos sejam automaticamente parados na recepção de um evento.

Autogerenciamento aberto: Solstice Domain Manager permite o administrador consistentemente gerenciar diferentes tipos de dispositivos pela organização, independente da localização física dos mesmos.

Outras características de gerência:

- Pedidos escalonados: Solstice Domain Manager permite que dados e eventos sejam escalonados. O administrador não tem como acionar ou parar no momento um pedido que foi iniciado ou parado.

- Filtragem de *traps*: Solstice Domain Manager permite que *traps* SNMP sejam configurados em diferentes prioridades (baixa, média e alta).

- Envio de *traps* SNMP para outras plataformas de gerência: *traps* SNMP recebidos pelo Solstice Site Manager e Solstice Domain Manager podem ser configurados para serem enviados para outras plataformas Solstice e não-Solstice.

- Estado pendente: Solstice Domain Manager fornece o estado pendente quando o ícone do dispositivo está sombreado e eventos subsequentes e *traps* são ignorados. O estado pendente pode ser iniciado manualmente via menu ou automaticamente via pedido de evento. No último caso, se o evento ocorre o dispositivo será colocado em estado pendente. Uma vez que o dispositivo é consertado, o administrador pode mudar o ícone ao estado normal via menu. Neste ponto, os eventos e *traps* do dispositivo serão recebidos novamente se ocorrerem.

Ferramentas:

- Console de gerência: é a aplicação central de gerência de ambos. É o lugar onde as tarefas de gerência são iniciadas e para onde a informação é retornada. O console apresenta uma interface gráfica que pode ser ajustada para um domínio em particular. Permitindo ao usuário configurar preferências globais.

- O console relata dados e eventos, incluindo mecanismos auditivos, visuais e programáveis: reporta dados habilitando os usuários a direcionar agentes para enviar relatórios de dados de gerência periodicamente, reporta eventos habilitando os usuários a direcionar agentes para relatar somente quando condições são alcançadas, isto é, quando eventos ocorrem.

- Console somente para leitura: permite que um operador gerencie os dispositivos tais como enviar pedidos e fazer *get/set* em dispositivos da MIB. Entretanto, não é possível adicionar, mover, remover elementos se estiver em modo somente para leitura. O operador também não pode criar pedidos predefinidos.

- Descobridor de IP: encontra automaticamente dispositivos SNMP e IP e coloca-os no banco de dados. Existem dois modos de operação: descobridor tenta encontrar dispositivos alcançavéis pelo console, pode-se ajustar o número de *hops* para delimitar a rede a ser descoberta. Monitor compara os elementos armazenados no banco de dados um a um para encontrar novos membros adicionados a rede desde a última execução do descobridor. Descobridor utiliza vários algoritmos de pesquisa para acelerar a busca dos dispositivos e possui vários ícones de dispositivos como: *hubs*, roteadores, computadores pessoais, estações UNIX.

- Ferramenta de Layout: lê a informação do banco de dados de gerência e automaticamente coloca os dispositivos e conexões em um dos 3 estilos de layout: hierárquico (organiza os elementos em subredes hierárquicas), circular (agrupa subredes em círculos) e simétrico (mostra os elementos em um subgrupo numa distribuição simétrica na tela). Mostra também uma janela com a porção da rede que está correntemente sendo vista e permite que o mapa da topologia seja impresso.

- Ferramenta Browser: fornece ao administrador uma gerência de dados para análises gerais e para plotar gráficos. Dados são organizados em relatórios *streams*, os quais contém o resultado de cada dado ou pedido de evento de um dispositivo em particular.

- Ferramenta Set : é usada para recuperar e inicializar atributos SNMP na MIB dos dispositivos gerenciados. Um administrador pode tomar mais informações de um dado atributo clicando no botão de detalhes do atributo. Informações

incluem nomes de atributos, tipos de atributos, informação de acesso e endereço de rede.

- Ferramenta Grapher: permite ao administrador visualizar valores de atributos retornados de pedidos de dados ou dados armazenados enviados do Browser. O Grapher pode intercalar múltiplos atributos para plotar o gráfico.

- Interfaces de aplicação: Solstice Domain Manager possui as ferramentas User e Developer. Três interfaces de programação e aplicação (APIs) que permitem aos usuários a construção de ferramentas complementares a ferramenta User, que são:

  a) Serviços de gerentes API: fornecem serviços de gerência de aplicações usando Solstice Site Manager e Solstice Domain Manager. Estes serviços incluem: registro de relatórios de dados recebidos, eventos e *traps*; inicializar valores de atributos e relacionar relatórios de erros.

  b) Serviços de agentes API: os serviços de agente fornecem habilidade para gerenciar ambientes com múltiplos protocolos via protocolo intermediário. Solstice Site Manager e Solstice Domain Manager também possuem código fonte de agentes exemplo que podem ser usados em agentes padronizados.

  c) Banco de Dados/Mapa de Topologia API: os serviços de mapa de topologia e banco de dados de gerência fornecem mecanismos para modificar o banco de dados e padronizar a tela da topologia. Por exemplo, aplicações de gerência podem carregar e salvar arquivos ASCII para o banco de dados de gerência do Solstice Domain Manager em tempo de execução.

3.2.4. Cisco Works da Cisco

CiscoWorks [40] é um conjunto de aplicações de gerência de rede baseadas em SNMP. Aplicações CiscoWorks são integradas a vários sistemas de gerência de redes como: SunNet Manager, Site Manager, Domain Manager e Enterprise Manager para Solaris; HP OpenView para sistemas HP ou Sun e IBM NetView para AIX, que permitem monitorização do estado de dispositivos, facilidades de manutenção e resolução de problemas de dispositivos Cisco.

A seguir, algumas aplicações presentes no CiscoWorks:

- Gerente de auto-instalação: permite instalar remotamente um roteador novo utilizando um roteador vizinho.

- CiscoView: mostra graficamente uma visão dos dispositivos físicos Cisco e informações como estado dinâmico, estatísticas e de configuração dos *switches*, roteadores, *hubs*, servidores de acesso, concentradores e adaptadores Cisco. Oferece também funções de monitorização e resolução de problemas.

- Gerência de arquivos de configuração: possui uma ferramenta que informa quem fez mudanças e quando, podendo detectar mudanças de configuração não autorizadas na rede.

- Gerência de dispositivo: cria e mantém um banco de dados que armazena um completo inventário da rede: *hardware*, *software*, versões de componentes operacionais, responsáveis por manutenção de dispositivos e locais associados.

- Facilidade de comando global: informações similares compartilhadas como senhas e listas de acesso podem ser configuradas para serem aplicadas automaticamente para um grupo comum de dispositivos.

- Monitor de saúde: fornece informações sobre estado de dispositivos, incluindo *buffers*, carga de CPU, memória disponível e protocolos/interfaces sendo utilizadas.

- Análise *offline* da rede: coleta dados históricos da rede para análise *offline* de tendências de desempenho e padrões de tráfego. Com o SQL do Sybase projetado para variáveis SNMP é possível fazer consultas e gerar gráficos.

- PathTool: permite análise de rota entre dois dispositivos, coletando dados de utilização e erros.

- Security Manager: configura procedimentos de segurança para aplicações e dispositivos de rede selecionados contra acessos não autorizados, protegendo o ambiente CiscoWorks através de pedido de `login`/senha.

- Software Manager: minimiza o custo de atualização, permitindo aos administradores centralizar a distribuição e gerência de *software* de roteador através da rede. É dividido em:

  a) Software Library Manager: fornece um repositório central para todo *software* Cisco.

  b) Software Inventory Manager: permite rapidez e facilidade para localizar roteadores para atualização.

  c) Device Software Manager: utiliza as funcionalidades da memória Flash de roteadores Cisco guiando usuários através de processos de atualização seguros e simples.

Outros *software* da família Cisco na área de gerência de redes:

CiscoWorks for Switched Internetworks (CWSI):

Gerencia redes locais pequenas a grandes. Por ser um conjunto integrado de aplicações, CWSI fornece funções de mapeamento de topologia, gerência de dispositivo em tempo real, análise de tráfego baseado em RMON, configuração de VLAN, gerência de desempenho e conexão ATM e relatórios.

CWSI pode ser instalado de 2 modos: como um pacote de gerência SNMP independente sem plataformas de gerência ou como um elemento de gerência integrado a alguma plataforma como: HP OpenView, SunNet Manager e NetView/AIX. Ambos modos suportam ambientes UNIX ou NT.

CiscoView:

CiscoView é um aplicação de gerência baseada em GUI que fornece o estado dinâmico, estatísticas e informações de configuração de produtos de rede Cisco (*switches*, roteadores, concentradores e adaptadores). CiscoView mostra graficamente uma visão física dos dispositivos Cisco e fornece funções de configuração e monitorização. A GUI mostra continuamente uma visão atualizada dos componentes físicos e pode ser invocada várias vezes na mesma sessão para mostrar simultaneamente múltiplos dispositivos.

Pode ser integrado com vários sistemas baseados em SNMP e ser executado sobre UNIX nas plataformas de gerência (Sun Microsystems, Site Manager, Domain Manager e

Enterprise Manager, Hewlett-Packard OpenView e Tivoli TME 1.0 NetView para AIX) e Windows.

Cisco Resource Manager 1.1:

É um conjunto de soluções de gerência baseada em Internet. Oferece atualização de *software*, pesquisa de mudanças na rede e isolação de condição de erros para roteadores e *switches* Cisco. As aplicações podem ser visualizadas utilizando um *browser* padrão dando acesso a informações de qualquer parte da rede. Por rodar em modo *standalone* não requer uma plataforma de gerência CiscoWorks e pode co-existir no mesmo sistema com HP OpenView, CiscoWorks ou CWSI. É composto pelas aplicações:

a) Inventory Manager: coleta, mostra e atualiza informações de inventário de *software* e *hardware* de roteadores e *switches*.

b) Software Manager: reduz o tempo necessário para instalar uma nova imagem de *software* do dispositivo, automatizando muitos dos passos associados com escalonamento, *download* e monitorização de atualizações de *software*.

c) Availability Manager: permite monitorizar um dispositivo com segurança e tempo de resposta, relatórios de dispositivos *offline* e recargas de dispositivos.

d) Syslog Analyzer: fornece relatórios de mensagens de `syslog` filtradas para isolar erros de roteadores e *switches* Cisco.

Cisco Netsys Service-Level Management Suite:

Consiste dos seguintes módulos:

a) Connectivity Service Manager: visualiza, avalia e fornece uma solução nas questões de conectividade incluindo disponibilidade, segurança e confiança da rede. Monitoriza os dados atuais de configuração da rede para verificar a disponibilidade dos principais serviços, permite estabelecer políticas de nível de serviço para conectividade, confiabilidade e segurança, utiliza a metodologia VISTA (visão, isolação, solução, teste e aplicação) para automatizar o diagnóstico e solução de problemas.

b) Performance Service Manager: utiliza as funcionalidades do Netsys Connectivity Service Manager para definir, monitorizar e otimizar o desempenho no nível de serviço, usa

os recursos da rede mais eficientemente, diagnostica e resolve problemas de desempenho, faz ajustes na rede e planeja mudanças.

c) LAN Service Manager: é um módulo do Netsys Connectivity Service Manager que mostra uma visão integrada do roteador/LAN e rotas de tráfego, verifica a integridade do domínio melhorando a visão da topologia. Quando combinado ao Netsys Performance Service Manager faz simulações do tipo "o que/se" para analisar os efeitos da movimentação de nós de uma LAN virtual para outra, otimizando seu desempenho.

A Cisco oferece outras ferramentas de gerência, como o CiscoWorks Blue Internetwork Performance Monitor para análises de desempenho de protocolos SNA e IP, e também ferramentas para contabilização como o Cisco Enterprise Accounting. Cisco Voice Manager é uma aplicação de gerência de rede baseada em web que fornece uma solução para configurar, monitorizar e diagnosticar rede de voz sobre IP (VoIP). O Cisco AccessPath Manager é um sistema de gerência de acesso baseado em web para operar grandes, complexas e distribuídas *dial pools*. E o CiscoWorks Blue Internetwork Performance Monitor (IPM) permite operadores de rede diagnosticar a fonte de "gargalos" de desempenho e gerenciar pró-ativamente com ferramentas que isolam problemas, diagnosticam atrasos e analisam tendências da rede.

3.2.5. OpenView da Hewlett Packard

O conjunto de produtos HP OpenView [41]: HP OpenView Professional Suite, HP OpenView Network Node Manager para Windows NT e o HP OpenView Network Node Manager para UNIX compartilham um objetivo comum que é fornecer um conjunto de serviços essenciais que forneçam as bases para um gerência de ambientes com múltiplos fabricantes, distribuído e de rede. Fornecendo serviços de:

- Interface de usuário: a interface do usuário é o mapa, por ele tem-se uma visão hierárquica da rede, representando suas subredes e elementos básicos, sendo estes elementos representados por ícones que mudam de cor conforme o estado do elemento na rede.

- Gerência de evento: notificam as aplicações que eventos específicos ocorreram e que alguma ação deve ser tomada.

- Descobridor: o descobrimento da rede é a checagem dos dispositivos da mesma e posterior representação dos elementos descobertos no mapa por ícones.

- Banco de dados de gerência: é o repositório central onde os dados sobre a rede são armazenados, podendo ser um banco de dados proprietário ou não. Estes dados serão utilizados para relatórios, análise e plotagem de gráficos.

- Infraestrutura de comunicações: para a descoberta dos dispositivos é necessário uma infraestrutura de comunicação por duas razões: existência de múltiplos protocolos e facilidade de conexão entre aplicações e objetos gerenciados, por isto o uso do SNMP.

- Serviços integrados: as API (*Application Programming Interfaces*) fazem parte dos serviços de integração do HP OpenView e servem para integrar as aplicações entre si.

- Gerência de nó: as soluções de gerência de rede devem ser capazes de descobrir e dimensionar em mapas gráficos os dispositivos de rede (roteadores, *bridges*, *switches*, servidores, *desktops*, impressoras) e nós. Estes elementos são geralmente numerosos. Os alarmes devem acionar o administrador assim que algum evento relevante acontece com algum dispositivo ou nó.

Soluções de gerência de rede HP OpenView:

HP OpenView é uma família de *software* de gerência de sistemas e rede baseado em padrões de mercado, projetado para administradores atuarem pró-ativamente e para monitorização de ambientes heterogêneos.

A família de *software* HP Open View é composta dos produtos: HP OpenView Professional Suite, HP OpenView Network Node Manager for Windows NT e HP Network Node Manager for UNIX. Como o objetivo deste trabalho é direcionado para a plataforma UNIX, somente o produto HP Network Node Manager for UNIX será abordado, tendo os demais apenas uma sucinta descrição.

1) HP OpenView Professional Suite fornece gerência de *desktop*, sistema e rede para ambientes Windows95 e Windows NT. Permite controle e gerência pró-ativa de redes locais de computadores pessoais heterogêneas. É baseado nos padrões SNMP e DMI (*Desktop Management Interface*).

2) Network Node Manager (NNM) para Windows NT é uma solução de gerência de rede que fornece visão em profundidade das conexões de rede em um formato gráfico. E oferece ao administrador capacidades para avaliar desempenho da rede, prever quedas e antecipar crescimento da rede.

3) HP OpenView Network Node Manager para UNIX

Fornece máxima visibilidade da rede, indicado para pequenos grupos de trabalho até empresas inteiras.

Características:

- Network Node Manager(NNM) é uma aplicação de gerência gráfica SNMP que fornece gerência de desempenho, configuração e falha para redes distribuídas TCP/IP.
- Projetado para manipular grande variedade de aplicações e equipamentos de rede.
- Aplicações complementares permitem a extensão do ambiente.
- Descobrimento automático e layout da rede incluindo 2 níveis de dispositivos.
- Arquitetura escalável e distribuída dispersa estações de coleta para domínios específicos. Estações de coleta UNIX e NT relatam descobrimento de dispositivos e mudam uma ou mais estações de gerência NNM para UNIX através da WAN. Reduzindo o tráfego e otimizando recursos computacionais.
- HP OpenView Windows Graphical User Interface (GUI) permite visualizar a rede com clareza e em detalhes.
- Subsistema de eventos personalizados que indicam problemas potenciais pela empresa.
- Coleta de dados em tempo real com limites definidos pelo usuário para disparar eventos, possibilitando melhor monitorização e gerência de funcionalidades e desempenho da rede.
- HP OpenView Web, interface web somente para leitura para o NNM acessar os estados e eventos da rede.

3.2.6. Spectrum da Cabletron

Os produtos de gerência da família Spectrum [42] para a plataforma UNIX se dividem em SPECTRUM Enterprise Manager e SPECTRUM Portable Management Applications para UNIX.

1) SPECTRUM Enterprise Manager [43] é uma plataforma de gerência orientada a objetos, com tecnologia de inteligência artificial. Possui dois componentes o SpectroGRAPH e o SpectroSERVER. O SpectroGRAPH é uma GUI que fornece acesso cliente ao SpectroSERVER. O SpectroSERVER é um servidor de gerência de rede onde reside a inteligência artificial do SPECTRUM.

Aplicações do SpectroSERVER:

- Autodescobrimento: automatiza a criação e manutenção do modelo de *software* usado para gerenciar a rede. Opera com intervalos de endereços IP e outros, explora a rede e cria modelos individuais de dispositivos e outras entidades de rede encontradas. Estes modelos são armazenados no banco de dados no SpectroSERVER com informações sobre seus relacionamentos e interconexões.

- Interface de linha de comando (CLI): fornece acesso em linha de comando ao banco de dados do SpectroSERVER. É particularmente usada em locações remotas quando uma estação de trabalho com funcionalidades gráficas (SpectroGRAPH) não está disponível. Pode-se interagir com o SPECTRUM usando comandos da CLI ou escrevendo *scripts* CLI. Permite que usuários leiam e escrevam no banco de dados e utilizem qualquer uma das funções oferecidas pelo SpectroGRAPH. *Scripts* CLI podem ser executados automaticamente usando o programador de tarefas do painel de controle ou o `cron` do UNIX. Os *scripts* devem ser invocados pela linha de comando UNIX, *shell scripts* ou através do menu do sistema.

- Modelo adaptativo dinâmico (DAM): examina os dispositivos e descobre quais as MIBs suportadas, DAM então une estas MIBs aos modelos catalogados das MIBs comumente suportadas e dinamicamente monta o modelo do dispositivo.

- Gerente de dados distribuídos: é um repositório central virtual das informações da rede, o qual permite aos usuários acesso aos dados através de vários SpectroSERVERs.

- Ligações ativas: mostram a condição da conexão entre dispositivos significativos via uma linha colorida. Clicando em uma destas ligações tem-se uma visão da ligação, mostrando os relacionamentos desta ligação em particular com todos os objetos ao redor, os objetos e seus estados.

- Ferramentas MIB: uma coleção de aplicações que permitem acesso e gerência de qualquer dispositivo compatível com SNMP através da MIB. O núcleo das ferramentas MIB é o banco de dados suportado por dispositivos na rede, os quais as aplicações usam quando comunicam-se com os dispositivos de rede.

- Ferramenta de alocação de endereço MAC (MALT): usada para localizar dispositivos na rede quando o endereço físico ou o MAC é desconhecido. MALT interroga *hubs* inteligentes do banco de dados SpectroSERVER para determinar o *hub* e porta onde o endereço está conectado. Desde que exista várias rotas para o mesmo dispositivo, MALT apresenta uma lista de *hubs* e portas, organizado primeiro com a porta mais provável.

- *PathView*: descobre e mostra a rota entre dois dispositivos na rede. PathView fornece uma representação visual dos modelos de pacotes de dados pela rede, e pode ser usado para diagnosticar problemas de rede. Como possui os dispositivos destino e fonte válidos, a aplicação mostra graficamente a rota da fonte ao destino, mostrando cada dispositivo pelo caminho.

- SpectroWATCH: fornece mecanismo para adicionar limites e *log* de dados históricos sem ter que fazer qualquer programação ou modificação de gráficos. Permite selecionar elementos a serem monitorizados em mais alto nível em detalhes e fornece ferramentas que mostram e analisam os dados coletados. Notificações de limites ultrapassados são enviados através das funcionalidades padrão de alarme e evento. Isto inclui habilidade para receber notificações via telefone (SpectroPHONE), *pager*, interface gráfica (SpectroGRAPH) ou através de uma interface *script* externa via correio eletrônico.

- Controle de painel: é uma interface *point-and-click* para realizar várias funções administrativas, incluindo programação de tarefas e *backup online*. O painel de controle elimina a necessidade de programas usando comandos UNIX.

- Exportação de dados (SDE): fornece exportação imediata ou escalonada do banco de dados para formatos de dados no padrão de mercado (ASCII, SAS, Oracle, Ingres e Sybase).

- Gerador de relatório (SRG): consiste de 3 componentes para formatar, gerar e mostrar/imprimir relatórios gráficos e tabelas. Existem 6 tipos de relatórios, todos personalizáveis: estatístico, relacional, inventário, evento, alarme e uptime/downtime.

- Relatórios baseado em web: permitem obter relatórios vitais do estado da rede.

- Escalonador: habilita a execução automática de *scripts* e trabalhos de aplicações. Tarefas podem ser programadas para executar uma vez, de hora em hora, diariamente, semanalmente ou mensalmente. Estas tarefas podem ser programadas durante horas ou em um período base através de uma interface de usuário.

- *UserEditor*: permite aumentar a segurança da rede e simplificar a gerência, administrando usuários individualemte ou como grupos (UserGroups).

- *Backup Online*: grava automaticamente o banco de dados em intervalos definidos pelo usuário sem interromper o SpectroSERVER e pode restaurar o banco de dados também.

2) SPECTRUM Portable Management Applications

O SPECTRUM Portable Management Applications [44] (SPMAs) é uma família de módulos e aplicações de gerência de elementos que permite gerenciar o *hardware* Cabletron com uma grande variedade de plataformas de gerência de elemento e sistemas operacionais. Para ambientes onde não é necessário uma plataforma de gerência de elemento, o SPMA tem uma versão *standalone*.

Características do SPMA:

- Integração com plataformas de gerencimento de elemento existentes: SPMA pode ser integrado com HP OpenView Unix, NetView AIX, SunNet Manager e Soltice Enterprise Manager. Dispositivos são representados por ícones específicos Cabletron usados pelas aplicações SPMA para gerenciar e monitorizar os equipamentos Cabletron. Todas as características de

mapeamento, relatórios, recebimento de *traps*, eventos e notificação de alarme permanecem como funções das plataformas proprietárias existentes.

- *Standalone launcher*: pode-se construir um banco de dados de dispositivos da rede manualmente ou via opção *discovery* com o *standalone launcher*. Os dispositivos são representados pelo tipo do dispositivo num formato de lista que provê o estado atual. O *standalone launcher* é incluído em cada elemento de gerência do SPMA. Fornece também segurança aos mapas de topologia.

- Gerência de dispositivo: dependendo da configuração do dispositivo, pode-se monitorizar o dispositivo no *chassis*, *board*, repetidor ou a nível de porta.

- Aplicação MIBTree: é uma MIB que fornece acesso a muitos padrões IETF e MIBs específicas mostradas no formato de árvore. A funcionalidade Editor possibilita a importação de qualquer MIB compatível com ASN.1 em um banco de dados existente. Cada entrada do banco de dados contém o nome do objeto, identificador OID, sintaxe, modo de acesso, estado e descrição.

- RMON: SPMA RMON suporta os 10 grupos RMON e possui as seguintes características: visão gráfica de dados estatísticos, funções de exportação e impressão de dados estatísticos, captura e filtros de pacotes para análise de protocolo. Através do uso de ações MIB, pode-se ter a habilidade de configurar eventos SNMP quando um evento RMON ocorre, podendo ser usada com opções de *trap* e *log* SNMP disponíveis com o padrão RMON.

- Utilitário para encontrar endereço MAC: habilita pesquisa de um endereço particular MAC em um dispositivo ou de múltiplos. O resultado é uma lista de dispositivos com o endereço incluindo informações de endereço IP, módulo, porta e tipo de porta.

- Monitorização de estatísticas por gráficos e medidores: SPMA permite que usuários visualizem estatísticas em gráficos formato de barra ou pizza com dados absolutos, cumulativos ou delta. A opção de medidor mostra estatísticas em intervalos baixo, médio e alto.

- Aplicação LAN Emulation Client (LEC): é uma aplicação que permite manutenção básica e configuração de LEC para dispositivos de acesso ATM Cabletron.

- Aplicações de nome de comunidade, tabela de *trap* e *download* TFTP: fornecem funcionalidades remotas de configuração de endereços de destino *trap* e séries de segurança. A aplicação de TFTP Download é usada para atualizar o pacote no dispositivo.

- Suporte MIB e *trap*: quando um *trap* de um dispositivo Cabletron é recebido, a SPMA pode interpretar a informação e mostrar uma mensagem significativa. Os utilitários de SNMP Browser e OID estão disponíveis.

- Ano 2000: todos os componentes de SPMA estão de acordo com o ano 2000.

3.2.7. Tivoli Enterprise da Tivoli

Tivoli Enterprise [45] fornece soluções de administração de sistemas para pequenas e grandes organizações centralizando a gerência dos recursos computacionais. A base da administração de sistemas empresariais Tivoli é o Tivoli Management Framework, com um banco de dados orientado a objetos, interface gráfica e serviços básicos utilizados por outros produtos.

A plataforma de gerência Tivoli é baseada na arquitetura ORB (*Object Request Broker*) e utiliza completamente a especificação de grupo de gerência de objetos CORBA 1.1.[46] Em cada máquina onde o ambiente de gerência Tivoli está instalado roda um processo ORB chamado `oserv`. O `oserv` é um processo de longa duração que atua como um agente na máquina. Recebe todos os pedidos locais através da GUI e interface de linha de comando (CLI), comunica-se com outros `oserv` de outras máquinas e inicia qualquer processo/método necessário na máquina local.[47]

O Tivoli permite gerenciar diversas arquiteturas e variados sistemas operacionais UNIX e de PC como DOS, OS/2, Windows, Windows 95 e Windows NT. E possui facilidades básicas como programação de tarefas, notícias/mensagens de eventos, níveis de usuários/administradores dividindo as ações por permissões de uso (super,senior,admin,user), interface gráfica, CLI. Utiliza um banco de dados relacional para armazenar os dados e possui ferramentas para *backup* destes dados.

Possui uma linha de produtos em áreas como segurança, armazenamento, controle remoto, administração de *mainframe*, entre outros, que são instalados sobre esta base principal, Tivoli Management Framework.[48] Dentre eles:

- Tivoli Inventory: mostra e registra *software* e *hardware* instalados em sistemas remotos.

- Tivoli Software Distribution: automatiza a distribuição, instalação e atualização pela rede de pacotes de *software*.

- Tivoli Distributed Monitoring: monitoriza recursos e serviços do sistema, detecta pró-ativamente e corrige automaticamente problemas em potencial enquanto escalona e envia as situações (problemas na forma de eventos divididos por alarmes - warning, critical, severe) para o Tivoli Enterprise Console.

- Tivoli Enterprise Console: coleta informações de eventos de recursos variados, como o Tivoli Distributed Monitoring. Relaciona as informações por níveis de severidade para determinar a causa dos problemas e ações podem ser tomadas para a solução destes problemas.

- Tivoli NetView: permite aos usuários descobrir redes TCP/IP, mostra as topologias de rede, relaciona e gerencia os eventos e *traps* SNMP, monitoriza a saúde da rede e coleta dados de desempenho.

- Tivoli Netview Performance Monitor: fornece informações necessárias associada a rede do cliente para a gerência de desempenho, rede e determinação de problemas.

- Tivoli Security Management: fornece uma solução aberta para gerência de segurança da empresa baseado em regras.

- Tivoli User Administration: realiza gerência de servidores, grupos e usuários.

- Tivoli Service Desk: maximiza a eficiência e eficácia dos serviços corporativos e operações de suporte na gerência de problemas, planos e de inventário.

- Tivoli Remote Control: fornece controle remoto em tempo real de sistemas, computadores pessoais, servidores e aplicações distribuídas sobre redes locais e WAN´s.

Tivoli possui várias outras ferramentas específicas para aplicações proprietárias IBM e para parceiras.

3.2.8. Symbel:

SymbEL (SE) [49] é uma linguagem interpretada que fornece ferramentas e utilitários para medir e ajustar o desempenho. É composta por *scripts*, e inclui também uma biblioteca gráfica baseada em MOTIF e pacotes de regras. Não é um produto Sun e funciona somente em sistemas Solaris 2. É um produto aberto podendo sofrer manutenções, modificações e/ou melhorias de terceiros. É composto por três pacotes: RICHPse (interpretador SymbEl), RICHPsex (pacote de extensões X e SE) e ANCrules (ferramentas e regras).

Por ter um código em *scripts*, nenhuma chamada de rotina manipula tempo e todas as chamadas relacionadas a tempo não assumem o ano com 2 dígitos, sendo assim é compatível com o ano 2000.

A razão básica de monitorizar desempenho [50] de um sistema é definir o estado da máquina para verificar se está funcinando bem ou se está lenta. Divide-se o sistema em componentes principais como por exemplo discos, rede, memória e verifica-se o estado separadamente.

SE é um conjunto de regras de desempenho aplicadas a estes componentes principais do sistema onde cada estado é relacionado com uma cor, conforme tabela 3.4.

Cor/Estado	**Descrição**
branco	completamente ocioso, intocável
azul	ocioso enquanto outras instâncias do recurso estão sobrecarregadas
verde	estado normal de operação, sem problemas
ambâr	condição de aviso
vermelho	sobrecarga ou problema detectado
preto	problema crítico que pode causar falhas em processos ou no sistema inteiro

Tabela 3.4 - Tabela de cores/estado do Symbel

Cada regra foi inicialmente definida em termos da saída padrão dos comandos do sistema. Por exemplo vmstat30.r significa: executa o comando `vmstat` com intervalos de 30 segundos e verifica a coluna rotulada com um r.[51] Estas regras que compõem o *software* SymbEl (SE) serão listadas a seguir:

1) Regra de disco:

A regra para discos é complexa, porque existem muitos discos no sistema. Escolheu-se programar uma regra que acusa sobre desbalanceamento no uso de discos. Dois casos especiais foram escolhidos e foi reconhecida a presença de uma unidade de disquete e uma unidade de CD-ROM como dispositivos mais lentos. Discos que tenham mais do que 20% de ocupação e estejam com o tempo de serviço com mais de 30-50ms são preocupantes.

2) Regra de rede:

É similar a regra de disco. Muitas instâncias de rede são combinadas e um desbalanceamento é relatado. Um caso especial é que a regra somente trata com *ethernet*, colisões são usadas como métrica. Não foi encontrada uma métrica satisfatória que indique sobrecarga para rede *token ring* ou FDDI.

3) Regra NFS RPC cliente:

Esta regra é baseada na observação que *timeouts* na chamada de procedimento remoto NFS pode ser associada com respostas duplicadas após a chamada ser retransmitida. Isto indica que a rede está funcionando, mas que servidores estão respondendo lentamente.

4) Regra espaço de *swap*:

Esta regra verifica se existe espaço de *swap* suficiente. É uma das poucas regras que programa o estado preto. Quando não existe espaço de *swap* suficiente, aplicações geralmente falham.

5) Regra demanda de RAM:

O sistema de memória virtual indica que necessita de mais memória quando procura por páginas ociosas para uma recuperação para outros usos. Máquinas com pouca memória tendem a mostrar a sobrecarga de páginas em disco, mas não uma taxa de verificação grande o suficiente. Uma aproximação para páginas ociosas residentes é utilizada para a verificação da demanda de RAM.

6) Regra memória do `kernel`:

O `kernel` dinamicamente aloca memória da memória livre quando necessita criar novas estruturas de dados. Se a alocação falha, processos podem falhar na inicialização. Isto mostra como tentativas de *login* ou conexões de rede falham inesperadamente. Existem duas causas possíveis: o `kernel` atingiu o limite do endereço de espaço; a lista livre não contém qualquer página para alocar. Entretanto a lista livre pode ter recuperado algo como uma pequena taxa de memória livre. Isto pode ser uma sinal de que um problema pode estar passando despercebido.

7) Regra de poder de CPU:

O número de CPUs em uso no sistema deve ser conhecido, nas regras isto é referenciado como ncpus. O tamanho da fila de execução é dividido pelo número de CPUs, baseado na aproximação de que toda CPU executa um trabalho fora da fila de execução em cada pedaço de tempo.

8) Regra conteção de `kernel`:

O `kernel` usa exclusão mútua para sincronizar operações e mantém múltiplas CPUs para acessar concorrentemente regiões de código crítico e dados.

9) Regra diretório de *cache*:

Existe um componente de diretório de *cache* global chamado *directory name lookup cache* ou DNLC. Uma perda de *cache* significa que a entrada do diretório deve ser lida de um disco e verificada para encontrar um arquivo correto. Cada entrada de DNLC contém um nome e uma referência para o *cache* de `inode`, descrita abaixo.

10) Regra *cache* de `inode`:

Um `inode` no sistema de arquivos UNIX é uma entidade que armazena os atributos de um arquivo. Tamanho, tempo de modificação e localização dos blocos de dados por um arquivo são as informações armazenados no `inode`. O nome do arquivo é armazenado separadamente em um diretório, e a entrada no *directory name cache* são conectadas as entradas no *cache* do `inode`. Qualquer arquivo aberto trava a entrada no *cache* do `inode`. Arquivos inativos são mantidos na memória com um `inode` inativo, que pode ter páginas de dados relacionadas até que a página seja recuperada. Se um `inode` inativo é reusado, páginas de dados serão retornadas para a lista livre. Uma perda de *cache* de `inode` e uma

- Programas de *pager*: o cliente (bb) envia linhas de informação para o servidor designado (bbd) e roda um script (bb-page.sh) que utiliza o Kermit via modem para enviar ao *pager* designado a mensagem, este servidor é denominado de BBPAGER.

- Programas de comunicação entre máquinas: um cliente envia informação aos servidores de BBDISPLAY e BBPAGER usando a porta 1984.

O Display Monitor é uma página *web* composta por uma tabela de máquinas e funções monitorizadas cujos os elementos são códigos de cores que informam o estado atual do sistema. A tabela mostra os estados verde (ok), amarelo (aviso), vermelho (crítico), púrpura (sem contato) e *clear* (não testado) para refletir as condições da rede. A ordem de severidade é: verde, amarelo, azul e vermelha. Para obter-se maiores informações sobre cada elemento é necessário clicar sobre o mesmo.

Através do arquivo etc/bb-hosts cada máquina é acessada em intervalos de 5 minutos. O uso de disco e de CPU e os processos de `sendmail` e `http` são monitorizados para todos os sistemas. Outros processos também podem ser monitorizados e todos os parâmetros são configuráveis.

O sistema monitoriza o arquivo de mensagens, procurando por condições NOTICE e WARNING. Se houver condição NOTICE envia imediatamente uma mensagem ao *pager* do administrador e para a condição WARNING aciona um código amarelo.

O Big Brother somente identifica o problema e notifica o administrador, ele não monitoriza explicitamente componentes de *hardware*, não é um monitor de desempenho de rede, servidores, banco de dados ou qualquer aplicação individual, porém fornece informação sobre carga de CPU e informações sobre tempo de resposta como conexões telnet.

Requerimentos:

- Compilador C para versões UNIX.

- Kermit e modem para envio de mensagens via *pager*.

3.2.10. Unicenter TNG da Computer Associates

Unicenter TNG [55] é um *software* da Computer Associates (CA) para gerência de redes heterogêneas, sistemas, *desktops*, banco de dados, aplicações, *internet* e *intranet*.

Possui as seguintes características:

- Descobrimento automático dos dispositivos, relacionamentos, conexões e topologia.
- Arquitetura agente/gerente propiciando redução no tráfego da rede e fornecendo escalabilidade e eficiência.
- Utilização de gerenciamento preditivo com a tecnologia Neuagent (agente neural), utilizando a funcionalidade de auto-aprendizado (causa/efeito/resolução).
- Uso de gerentes que filtram, analisam e correlacionam informações dos agentes e pró-ativamente tomam ações baseadas em regras predefinidas.
- Unispace: facilidade que visualiza os componentes de *software* da empresa.
- Mapa em 2-D ou *browser* fornecendo informações e estado da rede e sistemas.
- Tecnologia *wireless*.
- Gerência de protocolos TCP/IP, IPX, SNA, DECnet a partir de um console central.
- Gerência de *desktops* e servidores para: distribuição de *software*, inventário, proteção contra vírus, *backup*, controle remoto e configuração de *software*.
- Gerência de armazenamento com compressão de dados para otimizar utilização de espaço em disco, integridade de mídia, encriptação e proteção contra vírus.
- Escalonamento de tarefas.
- Suporte a outras plataformas.

3.2.11. SiteScope da Freshwater

O *software* SiteScope [56] e o serviço SiteSeer monitorizam o desempenho e a disponibilidade de aplicações de comércio eletrônico (*e-commerce*), locações *web* e servidores *internet* para plataformas Sun e NT continuamente (24 horas/dia).

O serviço SiteSeer realiza o monitorização no nível de serviço de locações remotas por 24 horas, notifica imediatamente erros via correio eletrônico, *pager* ou *trap* SNMP, realiza testes de transações de comércio eletrônico, verifica senhas de páginas protegidas, faz suporte de URL's seguras.

Capítulo 4

A Ferramenta SIMON

Existem várias ferramentas tanto comerciais quanto acadêmicas destinadas a auxiliar as tarefas de gerência e administração, como foi abordado no Capítulo 3. O objetivo da ferramenta proposta neste trabalho é o de monitorizar os sistemas em rede, coletando dados específicos por área de desempenho e mostrando-os de forma clara e concisa, fornecendo os subsídios necessários ao administrador para realizar os ajustes nos sistemas.

Os dados serão divididos por áreas de desempenho, a saber, memória, CPU, sistemas de disco, rede e atividade de I/O como a do NFS; ficando a cargo do administrador a ação de correlacionar os dados coletados por áreas de desempenho mostrados por esta ferramenta.

Não é possível cobrir todos os problemas e oferecer todas as possíveis soluções, contudo pode-se agilizar este trabalho fornecendo estes dados de forma a facilitar a ação do administrador, visualizando-os através de gráficos e tabelas, otimizando as ações do administrador.

Outro problema em ambientes de rede UNIX é a diversidade de sistemas e versões. Neste protótipo optou-se em apenas monitorizar-se o sistema Solaris 2.X, pois é o UNIX mais encontrado em empresas e universidades e porque é utilizado na Fundação CPqD (onde esta ferramenta foi implementada), como mencionado na Capítulo 2. Entretanto as possibilidades de monitorização de outros sistemas/versões UNIX não estão fechadas por esta ferramenta, pois ela permite monitorizar outras plataformas UNIX.

Esta ferramenta foi denominada de SIMON (SIstema de MONitorização) e é dividida em 3 módulos distintos porém interligados (figura 4.1). São eles:

- Módulo de coleta: formado por coletores (*shell scripts*) que rodam em cada máquina monitorizada;
- Módulo de transferência: transfere os coletores da máquina central para as locais e transfere os dados coletados das máquinas locais para a máquina central;
- Módulo de apresentação: interface gráfica para visualização dos dados na forma de tabelas e gráficos.

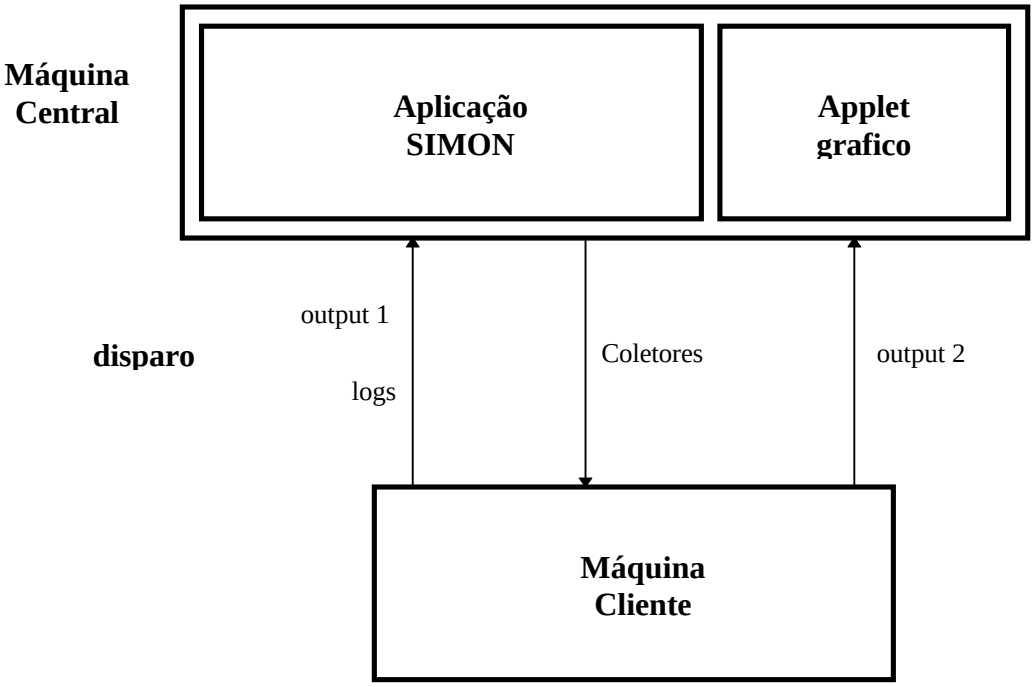

Figura 4.1 Esquema da ferramenta SIMON

output 1 <-- dados para tabelas
output 2 <-- dados para gráficos

O módulo de apresentação do SIMON mostrará sempre dados de forma instantânea: ou um dado instantâneo real ou a média coletada/calculada mais recentemente.que estará disponível em forma de arquivo ASCII logo após a execução dos coletores e subsequente transferência destes arquivos para o diretório `/home/user01/SIMON/` da máquina central. Os coletores são acionados pela execução do arquivo `disparo` na aplicação Java. Somente os dados mais recentes ficam armazenados nos arquivos. Estas informações são transparentes ao administrador no uso da ferramenta.

4.1. Módulo de coleta

Diversos são os fatores que devemos monitorizar na utilização de um disco e dos demais componentes do sistema e diversas são as formas de fazê-lo. Pode-se utilizar diversas ferramentas para tal finalidade; optou-se por usar comandos de linha do UNIX que coletam dados específicos de utilização de disco, por exemplo, como quanto tempo um

determinado processo de um usuário dispende de um disco, ou ainda a quantidade de leituras/escritas num disco.

Optou-se por utilizar comandos de linha do UNIX devido à:

- Simplicidade na aquisição dos dados;
- Sobrecarga mínima para o sistema, não interferindo de um modo prejudicial na monitorização, sendo a freqüência de amostragem estabelecida pelo administrador;
- Facilidade na troca e verificação das saídas dos comandos;
- "Não reinventar a roda", desnecessário buscar no *kernel* dados *in natura*;
- Facilidade em estender para outros sistemas.

Este módulo de coleta de dados é formado por coletores divididos em 5 áreas (memória, CPU, disco, rede, I/O). Estes coletores são *scripts* em *korn shell*. Cada coletor é formado por um comando de linha principal e alguns comandos de linha auxiliares para obter-se os dados desejados.

Todos os coletores possuem procedimentos comuns que foram agrupados num arquivo chamado `funcoes`, também em *korn shell script*. Este arquivo é chamado no código de cada coletor e possui também definições de variáveis globais, como é listado a seguir:

Arquivo `funcoes`:

```
# Inicializações globais dos comandos utilizados nos coletores:
#

DIRDF=/usr/bin/df
...
DIRUP=/usr/bin/uptime

# Path absoluto
#

HOME=/home/user01/SIMON
DIR=/tmp

# Funcoes globais usadas nos coletores:
#

#-- Verificação hostname, sistema e nome coletor --#
Verifica_maq_versao()
{
MAQUINA=`uname -n`;
SYS=`uname`-`uname -r`;
```

```
prog=`basename $0|cut -d'.' -f1`
}

#-- Quantidade de memória real --#
Verifica_mem_real()
{
if [ $SYS == "SunOS-5.5" ] || [ $SYS == "SunOS-5.5.1" ]
then
   tm=`$DIRDMESG | $DIRGREP "mem = " | $DIRGREP -v "avail mem = " |\
   $DIRAWK '{print $3}' | line`
   echo total memory = $tm >> output.$prog
fi
}

#-- Verificação pacotes perdidos e media (ping) --#
RoundTrip()
{
    loss=`tail -2 aux | $DIRGREP loss | $DIRAWK '{printf $7}'`
    media=`tail -2 aux | $DIRGREP min | $DIRAWK '{printf $5}'`
}

#-- Saida --#
# Colocação dos dados coletados no arquivo padrão output.$prog.
# $prog é o nome do coletor.

Output()
{
 if [ ! -f "output.$prog" ]
 then
    echo $MAQUINA > output.$prog
    echo $SYS >> output.$prog
    cat eixos.$prog >> output.$prog
    cat aux.$prog >> output.$prog
 else
     $DIRM output.$prog
     echo $MAQUINA > output.$prog
     echo $SYS >> output.$prog
     cat eixos.$prog >> output.$prog
     cat aux.$prog >> output.$prog
 fi
}

#-- Log --#
# Registro em arquivos ASCII (logs) das execuções dos coletores.

Log()
{
 data=`date`
 if [ ! -r "log.$prog" ]
 then
    echo "Data: " $data > log.$prog
    cat output.$prog >> log.$prog
 else
    echo "Data: " $data >> log.$prog
    cat output.$prog >> log.$prog
 fi
}

#-- Remoção dos arquivos auxiliares --#
Remocao_aux()
{
 if [ -f aux ]
 then
    $DIRM aux
 fi
 if [ -f aux1 ]
 then
```

```
    $DIRM aux1
 fi
 if [ -f aux2 ]
 then
    $DIRM aux2
 fi
 if [ -f aux3 ]
 then
    $DIRM aux3
 fi
 if [ -f aux.$prog ]
 then
    $DIRM aux.$prog
 fi
 if [ -f eixos.$prog ]
 then
    $DIRM eixos.$prog
 fi
}
```

Os coletores seguem um padrão, diferindo apenas nos comandos e modos de operação para a captura dos dados de desempenho. Um esqueleto dos coletores é listado abaixo:

```
#!/bin/ksh
#
# nome_do_coletor.sh

# chamada do arquivo com funções e variáveis globais:
# . <arquivo_com_funcoes>

. $DIR/funcoes

Verifica_maq_versao # função pertencente ao arquivo $DIR/funcoes

# Significado dos valores posicionais da saída do comando <comando>
# <nomevar1>=$1
# <nomevar2>=$2

case $SYS in

  SunOS-5.5) <lista de comandos> > aux.$prog;;

  SunOS-5.5.1) <lista de comandos> > aux.$prog;;

  SunOS-5.6) <lista de comandos> > aux.$prog;;

esac

#-- Saída --#
Output # chamada de função que grava os dados coletados em
            # arquivos output.<nome_do_coletor>.

#-- Log --#
Log          # chamada de função que grava dados coletados cumulativos da
             # execução dos coletores em arquivos log.<nome_do_coletor>.

#-- Remoção dos arquivos auxiliares --#
Remocao_aux  # função que remove todos arquivos auxiliares utilizados pelo
             # coletor.
```

A ferramenta permite facilmente a inclusão de novos coletores, para tanto é necessário que o arquivo de saída `output.nome_do_coletor`, arquivo ASCII gerado pela execução do coletor, contenha os dados na forma em que a interface gráfica compreenda, desta maneira:

```
<hostname>
<sistema_operacional-versão>
<legenda_eixo_x_do_gráfico>
<legenda_eixo_y_do_gráfico>
nome_da_variável1|valor_da_variável1
...
nome_da_variáveln|valor_da_variáveln
```

Caso a inclusão do novo coletor, não necessite que a saída seja em forma de gráfico, mas em forma de tabela, os dados não necessitam estar desta forma no arquivo `output.nome_do_coletor`.

A inclusão do nome do coletor deve constar nos módulos de transferência e apresentação, que serão mostrados posteriormente nas seções 4.2 e 4.3, respectivamente.

Modificações nos coletores existentes para adicionar novos sistemas e versões de UNIX é ainda mais simples e fácil, bastando acrescentar uma nova cláusula na declaração `case` do coletor desejado, assim:

```
novo-sistema-versão) lista de comandos >> aux.$prog;;
```

Deve-se ainda acrescentar no *script* `funcoes` o caminho correto da lista de comandos do novo sistema-versão adicionado, em:

```
# Inicializações globais dos comandos utilizados nos coletores:
DIRNOVO1=/xxx/yyy/comando1
...
DIRNOVOn=/xxx/yyy/comandon
```

4.1.1. Memória

Memória é um dos primeiros setores onde ocorrem problemas de desempenho. O desempenho pode degradar quando o sistema de memória (RAM e discos) não mantém a demanda de páginas, problemas de falta de área de *swap*, processos utilizando grandes porcentagens de memória, entre outros. Três coletores que monitorizam memória foram implementados.

Como cada coletor possui um comando de linha UNIX principal e cada comando é composto por diferentes opções, a descrição detalhada de cada comando e respectivas opções é realizada no Apêndice A.

Coletor psefom.sh

O coletor `psefom.sh` utiliza o comando `ps` que mostra informações sobre processos ativos [57]. Ele verifica os 15 processos que estão ocupando a maior porcentagem de memória da máquina local. O formato do comando `ps` utilizado foi:

```
ps -efo pmem,user,uid,pid,nice,stime,args
```

A saída deste comando irá fornecer uma lista contendo informações por processo como porcentagem de memória utilizada, nome e `id` do usuário, número do processo, prioridade de execução, tempo em execução e o comando completo em execução.

Exemplo: `output.psefom`

```
aldebaran
SunOS-5.5.1
%MEM   USER  UID   PID NI    STIME COMMAND
 1.6 user01 3700  1254 20   Feb_22 emacs -T em@aldebaran@user01 -cr gold prog1
 1.6 user02 2629  1246 20   Feb_22 emacs -T em@aldebaran@user02 -cr gold prog2
 1.7   root    0  5808 20   Feb_17 xterm -ut -T CONSOLE -n CONSOLE -fn 7x13bold
 1.7   root    0  5874 20   Feb_17 xterm -ut -iconic -fn 8x13bold -g 100x35 -bg
 1.7   root    0  5908 20   Feb_17 xterm -ut -T user01@pato -n user01@pato -fn
 1.7   root    0  6129 20   Feb_17 xterm -ut -T user02@iota -n user02@iota -fn
 1.8   root    0  5819 20   Feb_17 xterm -ut -T user03 -n user03 -iconic -fn
 1.8   root    0  5881 20   Feb_17 xterm -ut -iconic -fn 8x13bold -g 100x35 -bg
 1.8   root    0  6111 20   Feb_17 xterm -ut -iconic -fn 8x13bold -g 100x35 -bg
 2.1 user04 2428  5802 20   Feb_17 X :0
 2.3 user05 3699 14297 20 08:42:02 emacs /home/user03/8.4/sis.mkf
 2.5 user04 2428 18105 20   Feb_17 /net/local/netscape-4.05/netscape
 3.5 user01 3268 17828 20 13:25:24 emacs -T em@aldebaran@user01 -cr gold dr299.d
11.2 user07 3268 16150 20 11:53:55 dr.exe
36.8 user06 3268 17631 20 13:05:49 m.exe
```

Coletor swaps.sh

O coletor `swaps.sh` utiliza o comando `swap` que fornece um método para adicionar, remover e monitorizar as áreas de *swap* do sistema [21].

O formato utilizado para o coletor `swaps.sh` foi `swap -s`. Esta opção mostra a área de *swap* dividida da forma:

`allocated:` Taxa total de espaço de *swap* (em blocos de 1024-bytes) correntemente alocada.

`reserved:` Taxa total de espaço de *swap* (em blocos de 1024-bytes) não correntemente alocada, mas reservada para um uso futuro por processos em execução.

`used:` Taxa total de espaço de *swap* (em blocos de 1024-bytes) que é alocada ou reservada.

`available:` Taxa total de espaço de *swap* (em blocos de 1024-bytes) que está correntemente disponível para futuras reservas ou alocações.

Exemplo: `output.swaps`

```
pato
SunOS-5.5.1
KB
memória
alloc|94744
reser|13316
used|108060
avail|107236
```

Coletor vms.sh

O coletor `vms.sh` utiliza o comando `vmstat` que examina o sistema e relata estatísticas sobre processos, memória virtual, disco, *trap* e atividade de CPU [12].

Este coletor monitoriza a atividade de *swapping* e *paging* da máquina. É calculada a média (KB) dos dados coletados dos campos `si` (*swap-ins*), `so` (*swap-outs*), `pi` (*page-ins*) e `po` (*page-outs*) e armazenada no arquivo de saída `output.vms` na máquina local.

O formato que o coletor `vms.sh` utiliza do comando `vmstat` é `vmstat -S interval count`. Este é um exemplo de coletor onde somente a média coletada contém informação relevante para a ferramenta.

Exemplo: `output.vms`

```
eridano
SunOS-5.5.1
KB
atividade memória virtual
si|0
so|0
pi|75.8
po|22
```

4.1.2. CPU

Quando as aplicações estão funcionando normalmente mas estão iniciando uma saturação de CPU (i.e., um esgotamento de ciclos de CPU disponíveis), as opções para corrigir a situação são limitadas. Podemos otimizar a utilização da CPU, fazendo-se ajustes para aumentar a eficiência do sistema que está no limite. Se a condição de sobrecarga da CPU encaminha-se para continuar indefinidamente, a melhor solução a longo prazo é adicionar mais memória (somente se houver indicação de atividade excessiva de *swap*), adicionar processadores (sistemas multiprocessados), ou trocar o sistema por um mais potente. Foram implementados 3 coletores para a monitorização da CPU.

Coletor psefoc.sh

Coletor responsável em buscar processos que estiverem ocupando a CPU de forma majoritária, degradando o funcionamento da mesma. O comando `ps` também foi utilizado por este coletor. Ele mostra os 15 processos que estão consumindo mais a CPU. Com a sintaxe: `ps -efo pcpu,user,uid,pid,nice,stime,args`. Mostra uma lista com os seguintes dados por processo: porcentagem de CPU utilizada, nome e `id` do usuário, número do processo, prioridade de execução, tempo em execução e o comando completo com argumentos em execução.

Exemplo: `output.psefoc`

```
pato
SunOS-5.5.1
%CPU   USER  UID   PID NI    STIME COMMAND
 0.1   root    0    94 20   Feb_11 /usr/sbin/rpcbind
 0.1   root    0   159 20   Feb_11 /usr/sbin/nscd
 0.1   root    0   220 20   Feb_11 /usr/lib/nfs/nfsd -a 16
 0.1   root    0 11809 20 01:40:41 /usr/lib/autofs/automountd
 0.1 user02 3268 12394 20 09:13:41 emacs -T em@pato@user -cr gold gera.csh
 0.1 user01 2378 12993 20 12:59:58 -csh
 0.2 user01 2378 13125 20 13:01:03 sort +4 -nr
 0.2 user01 2378 13126 20 13:01:03 tail -15
 0.3   root    0     3 SY   Feb_11 fsflush
 0.3   root    0   150 20   Feb_11 nmbd
 0.4   root    0 13123 20 13:01:03 /usr/bin/ps -efo
pcpu,user,uid,pid,nice,stime,args
 0.5 user01 2378 13117 20 13:01:02 /bin/ksh psefoc.sh
 0.5 user03 3531 22847 20   Feb_19 X :0
 1.9 user04 2689 12636 20 10:26:54 /net/local/netscape-4.05/netscape
 1.9 user03 3531  8719 20   Feb_22 /net/local/netscape-4.05/netscape
```

Coletor uptime.sh

O coletor `uptime.sh` utiliza o comando `uptime` que mostra o tempo corrente, o tempo que o sistema está no ar, a média de *jobs* em execução na fila nos últimos 1, 5 e 15 minutos.[11]

Este coletor tem a finalidade de relatar a carga média de utilização de CPU nos últimos 1, 5 e 15 minutos, ou seja, `load average 1`, `load average 2` e `load average 3`.

Exemplo: `output.uptime`

```
iota
SunOS-5.5.1
carga
cargas médias (load average 1,5,15 min)
la1|1.08
la2|0.80
la3|0.64
```

Coletor saru.sh

O coletor `saru.sh` utiliza o comando `sar` que mostra a atividade cumulativa do sistema operacional em n intervalos de t segundos conforme opções.[20]

A função deste coletor é listar o percentual de utilização de CPU por sistema (`sys`), usuários (`us`) e ociosa (`id`). São coletados dados em 5 intervalos de 5 segundos, sendo calculada a média dos dados. A sintaxe do comando `sar` utilizado pelo coletor foi `sar -u interval count`. Este é outro coletor onde a média coletada é mais importante para análise do que valores instantâneos.

Exemplo: `output.saru`

```
iota
SunOS-5.5.1
percentual CPU
tempos de CPU
usr|31
sys|21
idle|46
```

4.1.3. Disco

A vazão (*throughput*) de disco é o fator limitante de desempenho de muitas aplicações. Transferências de dados de e para o disco são muito mais lentas do que

transferências de dados envolvendo CPU ou memória. Conforme explicado no Capítulo 2, seção 2.1.

Muitas vezes uma unidade de disco é exigida demais, constituindo-se em “gargalo” do sistema, ao passo que outras unidades podem estar ociosas. O administrador poderá então decidir pela realocação de informação entre os discos, tentando equilibrar melhor o uso dos dispositivos físicos. Entre tais parâmetros podemos citar tempo de serviço, porcentagem de ocupação dos sistemas de arquivos e porcentagem livre, transferência em KB/s, transações por segundo, duração média de cada transação em milissegundos.

Coletor dfkl.sh

O coletor `dfkl.sh` utiliza o comando `df` que mostra a quantidade de espaço em disco ocupada por sistemas de arquivos montados ou não, diretórios ou recursos montados, a quantidade de espaço usado e disponível e quanto da capacidade total do sistema de arquivo tem sido usada.[14].

Este coletor é responsável por verificar a ocupação em KB dos discos locais do sistema para que se possa otimizar e configurar sua utilização, tanto para usuários e/ou aplicativos da rede. A sintaxe do comando `df` utilizada para este coletor foi `df -kl`.

Exemplo: `output.dfkl`

```
aldebaran
SunOS-5.5.1
capacidade (KB)
partições
/|51
/l/disk0|98
/l/home0|98
/l/home1|86
```

Coletor iosvt.sh

O coletor `iosvt.sh` utiliza o comando `iostat` que iterativamente relata atividade de terminal, disco e I/O e utilização de CPU. A primeira linha da saída fornece informações desde a reinicialização da máquina. [13]

Este coletor verifica o tempo de serviço (ms) disco gasto para um determinado pedido de I/O, o campo `svc_t` mostra esta informação. A sintaxe do comando `iostat` utilizada para este coletor foi `iostat -x interval count`.

Exemplo: `output.iosvt`

```
pato
SunOS-5.5.1
tempo (ms)
discos (scsi disk)
sd1|79.1
sd2|9.2
sd3|59.1
```

4.1.4. Rede

Muitos recursos usados nos subsistemas de rede são alocados e ajustados dinamicamente. O desempenho da rede é afetado somente quando o suprimento de recursos não é capaz de manter sua demanda. A ação de monitorizar a rede fornece dados ao administrador, os quais propiciam subsídios necessários para realizar ajustes no nível de sistema, visando otimizar seus serviços e aplicações.

Coletor netsi.sh

O coletor `netsi.sh` utiliza o comando `netstat` que mostra o conteúdo de várias estruturas de dados relacionadas à rede em vários formatos, dependendo das opções fornecidas. [17]

Este coletor traz informações básicas sobre interfaces de rede, oferecendo informações para o administrador identificar problemas e corrigí-los. Tais informações são listadas por interface de rede sendo: número de erros de entrada (`ierrs`), número de erros de saída (`oerrs`), número de colisões (`collis`) e número de pacotes de saída (`opkts`). A opção do comando `netstat` utilizada para este coletor foi `netstat -i`.

Exemplo: `output.netsi`

```
eridano
SunOS-5.5.1
interface de rede: le0
número de ierrs: 323
número de opkts: 87643097
número de oerrs: 92
número de collis(colisões): 13020578
```

Coletor trafego.sh

O coletor `trafego.sh` faz uso do comando `ping` que utiliza o datagrama ECHO_REQUEST do protocolo ICMP para extrair uma resposta do tipo ECHO_RESPONSE de uma máquina específica. [18]

Este coletor tem por função mostrar se a máquina está no ar e a percentagem de pacotes perdidos, se houverem, e as médias de tempos no envio dos pacotes. O formato utilizado deste comando no coletor foi `ping -s host packetsize count`.

Exemplo: `output.trafego`

```
iota
SunOS-5.5.1
percentagem de pacotes perdidos:  0%
tempos min/avg/max (ms):  0/0/2
```

4.1.5. I/O

A monitorização da atividade de I/O dos serviços dos sistemas em rede e aplicações é uma tarefa importante para ajustar o funcionamento dos mesmos. E permite ao administrador balancear a utilização de memória, CPU, sistemas de discos e sistemas em rede otimizando os recursos, serviços e aplicações do sistema como um todo.

Um exemplo de serviço do sistema que pode utilizar os recursos majoritariamente se não for configurado de modo correto é o NFS. Muitos problemas de desempenho com NFS podem ser atribuídos aos “gargalos” do sistema de arquivos, rede ou sistema de discos. Por exemplo, o desempenho do NFS é severamente degradado pela perda de pacotes na rede. Em particular, *timeouts* de rede podem degradar severamente o desempenho do NFS. Esta condição pode ser identificada usando o comando `nfsstat` e somente a parte cliente do NFS nos interessa nesta monitorização, pois é o lado cliente que detecta a possível falha no servidor.

Coletor nfs.sh

O coletor `nfs.sh` utiliza o comando `nfsstat` que mostra informações estatísticas sobre o NFS e RPC, interfaces do `kernel`. [19]

Este coletor mostra informações do lado cliente NFS e de RPC. As informações de RPC são os campos `calls`, `badcalls`, `badxids`, `timeouts` via UDP e os campos `calls`, `badcalls`, `retrans`, `badxids`, `timeouts` via TCP. A sintaxe utilizada do comando `nfsstat` para obter-se este dados foi `nfsstat -cr`.

RPC pode ser usado via UDP (*connectionless*) ou TCP (*connection oriented*). Em TCP a entrega do pacote é garantida, não necessitando de retransmissões, sendo que o próprio TCP se encarrega delas, se necessário. Os campos *badcalls, badxids* e *timeouts* informam que serviços de alto nível podem ou não estar disponíveis para atender o pacote de chamada. Todos esses eventos são provocados pelo servidor (pois os dados coletados são do cliente). O campo *retrans* indica quando o cliente chegou à conclusão de que a falha deve ser corrigida por ele através de novo envio da mesma chamada (*call)*. O campo *badxids* indica quando o *transaction id* (xid) na resposta do servidor não coincide com o xid do pedido do cliente.

No modo UDP, a entrega não é garantida e portanto fica a cargo do cliente certificar-se de que o pacote foi recebido. Desta forma, contabiliza-se os casos em que foi necessário isto.

Exemplo: `output.nfs`

```
aldebaran
SunOS-5.5.1
qtidade chamadas
chamadas TCP e UDP
co|2786070          <--- calls (oriented)
bco|29              <--- badcalls (oriented)
bxo|29              <--- badxids (oriented)
to|1                <--- timeouts (oriented)
cc|432330           <--- calls (connectionless)
bcc|349             <--- badcalls (connectionless)
rc|878              <--- retrans (connectionless)
bxc|2               <--- badxids (connectionless)
tc|1023             <--- timeouts (connectionless)
```

Coletor balcarga.sh

O coletor balcarga.sh utiliza os comandos `swap -s`, `iostat -D interval count` e `uptime` para ajudar o administrador a balancear a carga da máquina fornecendo dados

como: memória virtual disponível, leituras por segundo, escritas por segundo e percentagem de utilização para cada disco e a carga média de uso de CPU nos últimos 1, 5 e 15 minutos.

Este coletor realiza a união dos dados de 3 agentes (swaps.sh, iosvt.sh e uptime.sh) em forma de tabela pois a interface do prótotipo não permite a junção de 3 gráficos.

Exemplo: `output.balcarga`

```
eridano
SunOS-5.5.1
memory available: 73296k

          sd0           sd1           sd2           sd3
 rps wps util  rps wps util  rps wps util  rps wps util
   0   0  0.7    0   0  0.8    1   0  0.7    1   1  1.7
   0   2  1.8    0   0  0.0   32   5 52.1    0   1  0.7
   0   4  3.7    1   0  1.9   26   7 49.7    1   2  3.3
   0   4  3.4    0   0  0.0   30  18 66.4    3   3  5.1
   0   3  2.8    0   0  0.0   33  17 65.1    4   2  4.5

loadav1: 0.08
loadav2: 0.10
loadav3: 0.08
```

4.2. Módulo de transferência

Este módulo é o responsável por interligar a interface gráfica com os coletores, conforme mostrado na figura 4.1. Ele é implementado através de um programa em C chamado de `disparo`. A linguagem C foi utilizada devido à facilidade em executar, através do comando `system` do UNIX, as ações de execução remota (`rsh`), copia remota (`rcp`[7]) e do aplicativo `appletviewer`.

O programa `disparo`, aceita dois parâmetros de entrada (`máquina_local` e `nome_do_coletor`) e é executado pelo módulo de apresentação. Este programa reside na máquina central.

Esta é a sequência de passos de funcionamento do programa `disparo`:

1) Verifica números de parâmetros de entrada e se forem 2, continua execução.

2) Copia arquivo `funcoes` para o diretório `/tmp` da máquina cliente com o comando `rcp`.

[7] O comando `rcp` faz cópias de arquivos entre máquinas. [23]

3) Copia coletor especificado para o diretório `/tmp` da máquina cliente com o comando `rcp`.

4) Executa na máquina cliente o coletor especificado. Utiliza o comando `rsh -l <usuário> <máquina> <nome_do_coletor>`.

5) Copia o resultado da execução do coletor especificado na máquina cliente (`/tmp/output.nome_do_coletor` e `/tmp/log.nome_do_coletor`) para a máquina central no diretório onde estão os arquivos do SIMON, `/home/user01/SIMON/`, com o comando `rcp`.

6) Chama a aplicação `appletviewer` de acordo com o coletor especificado. Se a saída não for um gráfico, o módulo de apresentação chamará uma função para mostrar os dados em forma tabular.

7) Remove da máquina cliente no diretório `/tmp` o arquivo `funcoes`, o coletor e os arquivos `output.nome_do_coletor` e `log.nome_do_coletor`.

Os dados coletados pelos coletores nas máquinas clientes são armazenados em arquivos em formato ASCII (`output.nome_do_coletor e log.nome_do_coletor`).

Para a utilização do comando `rsh` (`rsh -l <usuário> <máquina> <nome_do_coletor>`) é necessário ter-se na área onde for instalada a ferramenta (`/home/user01` ou `$HOME`) o arquivo `.rhosts`, com o conteúdo:

```
+ nome_do_usuário
```

Para adicionar-se um novo coletor, é necessário apenas, incluir uma cláusula do tipo `if` no programa `disparo`. Esta nova claúsula chamará o aplicativo `appletviewer`, caso a saída seja um gráfico em barras, com o arquivo `output.nome_do_coletor` correto, caso contrário uma tela será mostrada com os dados em forma de tabela. Para ambos casos, o módulo de apresentação também deve ser acrescido, veremos isto na próxima seção 4.3.

Arquivo `disparo.c`:

```
char graf_novo_coletor[] = “appletviewer grafico_novo_coletor.html “;

if (!strcmp(coletor == "novo_coletor"))
{
  if (system(graf_novo_coletor))
  {
    printf("ERRO NA CHAMADA SYSTEM appletviewer grafico_novo_coletor \n");
```

```
    fprintf(fer," ERRO NA CHAMADA SYSTEM appletviewer grafico_novo_coletor");
    exit(1);
  }
}
```

4.3. Módulo de apresentação

O módulo de apresentação que caracteriza a interface gráfica foi implementado em Java. Esta linguagem foi escolhida por ser multiplataforma, característica que permitirá a execução desta ferramenta em diferentes sistemas operacionais, sendo altamente portátil, pois não necessita de recompilação.

Este módulo foi implementado para visualização dos dados coletados no módulo de coleta. A forma de visualização da ferramenta SIMON é por meio de gráficos e tabelas. Este módulo permanece na máquina central, conforme mostrado na figura 4.1.

O módulo de apresentação é dividido em duas partes: a aplicação Java chamada `Simon` e o `applet` Java chamado `grafico`.

A aplicação `Simon` é formada por 6 classes diferentes: `leitarq`, `coletorMEM`, `coletorCPU`, `coletorDISCO`, `coletorREDE` e `coletorIO`. Cada classe é responsável por uma funcionalidade da aplicação.

A classe `leitarq` é responsável pela leitura dos dados dos arquivos `log.*` , `output.*` (para o caso de dados em formato tabular) e arquivos de ajuda (`*.txt`) e visualização destes dados numa interface chamada `TextArea`[8] .

As classes `coletorMEM`, `coletorCPU`, `coletorDISCO`, `coletorREDE` e `coletorIO` são responsáveis por mostrar uma interface contendo duas listas com dois botões (figura 4.2): uma lista contendo os nomes dos coletores de acordo com a área e outra contendo os nomes das máquinas clientes onde os coletores serão executados. Selecionando-se um item de cada lista e clicando no botão chamado disparo, acionará o módulo de transferência, explicado na seção anterior. O outro botão tem a função de fechar esta interface.

[8] TextArea é uma janela que mostra texto especificado com scrollbars, podendo ser editável ou não.

Figura 4.2 - Tela do disparo dos coletores

A aplicação `Simon` é composta por uma interface formada por 4 menus. O primeiro menu *pop-up* chama-se Arquivo (figura 4.3) e contém apenas uma função de saída da ferramenta.

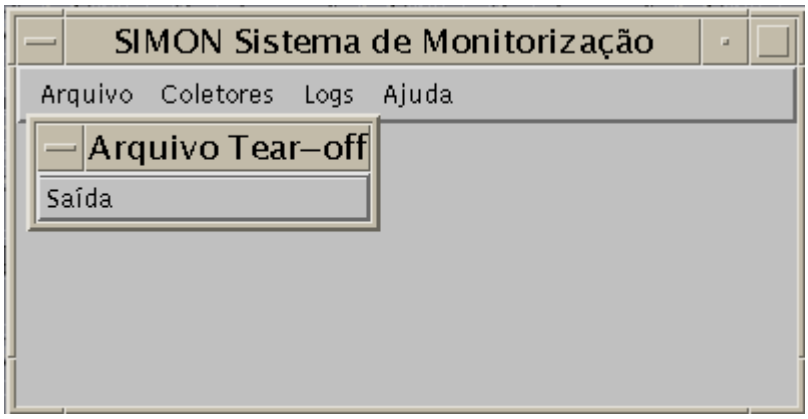

Figura 4.3 - Tela do Menu Arquivo

O segundo menu *pop-up* chama-se Coletores (figura 4.4) e possui 5 funções, cada uma chama as classes `coletorMEM`, `coletorCPU`, `coletorDISCO`, `coletorREDE` e `coletorIO`.

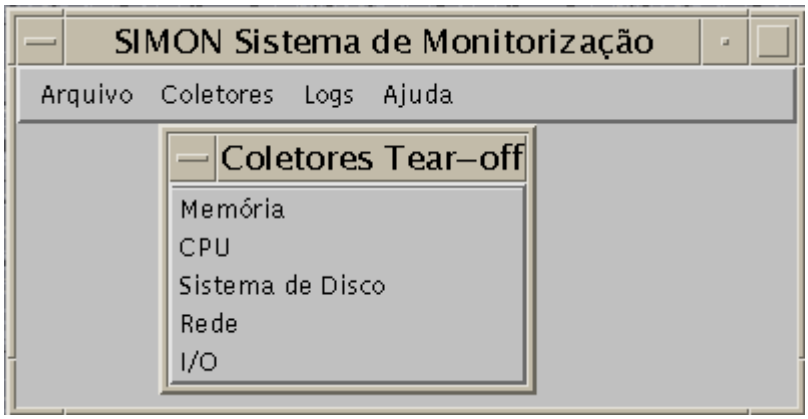

Figura 4.4 - Tela do Menu Coletores

O terceiro menu *pop-up* chama-se Logs (figura 4.5) e possui 12 funções, cada uma chama a classe `leitarq` que faz a leitura e mostra os `log.*` por coletor numa `TextArea`.

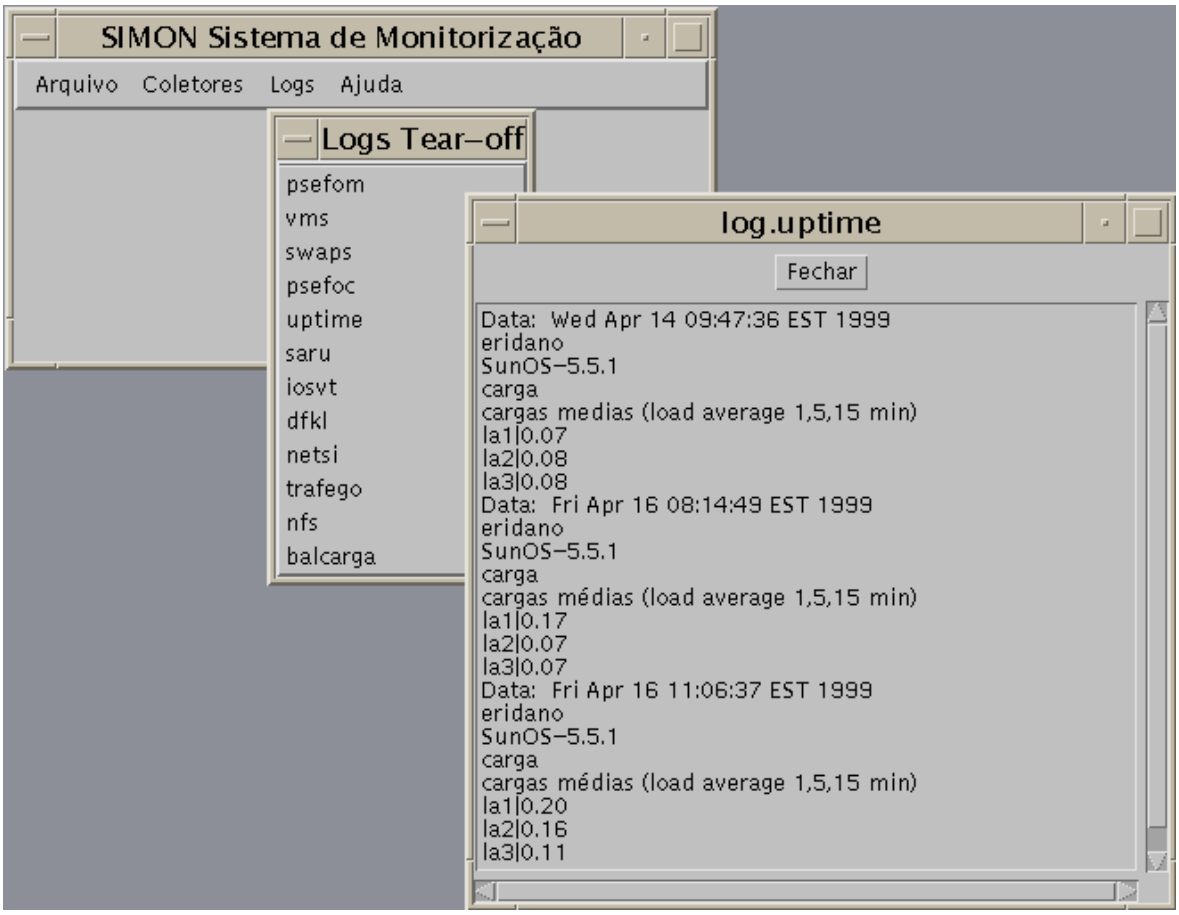

Figura 4.5 - Telas do Menu Logs e log.uptime

É possível com o SIMON o armazenamento dos registros de execução de todos os coletores (os chamados *logs*, figura 4.5). Sendo armazenados com as datas de execução, os logs permitem montar uma base de dados para comparações de desempenho. Por exemplo, utiliza-se esta base com dados históricos para determinar diferenças de desempenho na utilização do sistema ou partes dele. Isolando-se determinadas partes pode-se verificar quais estão mais críticas. A região crítica é isolada e novas medições são feitas para analisar se a região isolada estava realmente afetando o desempenho.

Finalmente o quarto menu *pop-up* chama-se Ajuda (figura 4.6) e possui 3 funções de ajuda. São simples arquivos ASCII contendo textos explicativos sobre a ferramenta Simon, os coletores divididos por área e um texto informativo sobre versão da ferramenta e nome da autora. O menu Ajuda também utiliza a classe `leitarq`.

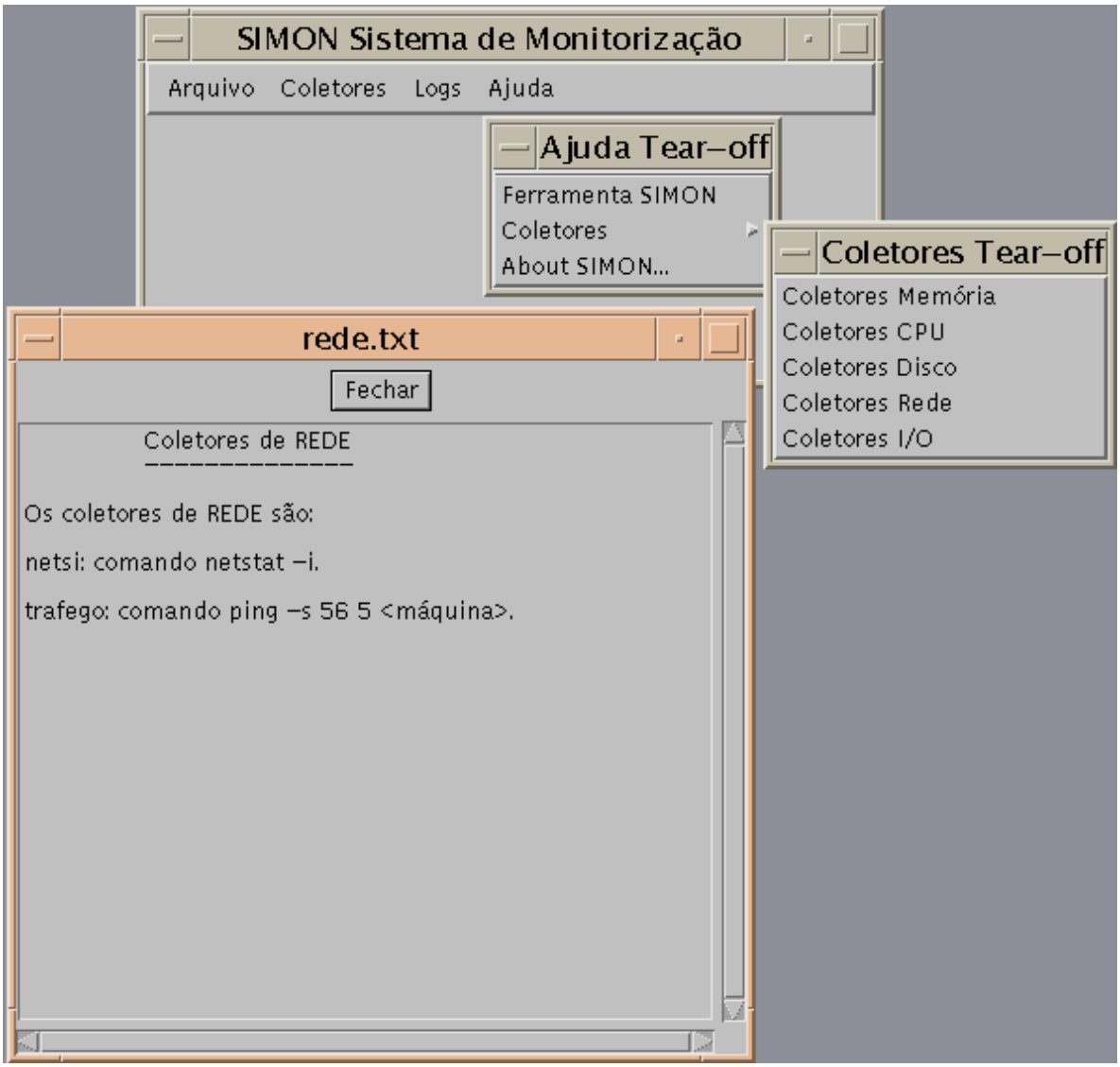

Figura 4.6 - Telas do Menu Ajuda

O `applet grafico` (figura 4.7), que é chamado pelo programa `disparo`, faz a leitura do arquivo `output.nome_do_coletor` e mostra os dados coletados pelos coletores em forma de gráfico de barras.

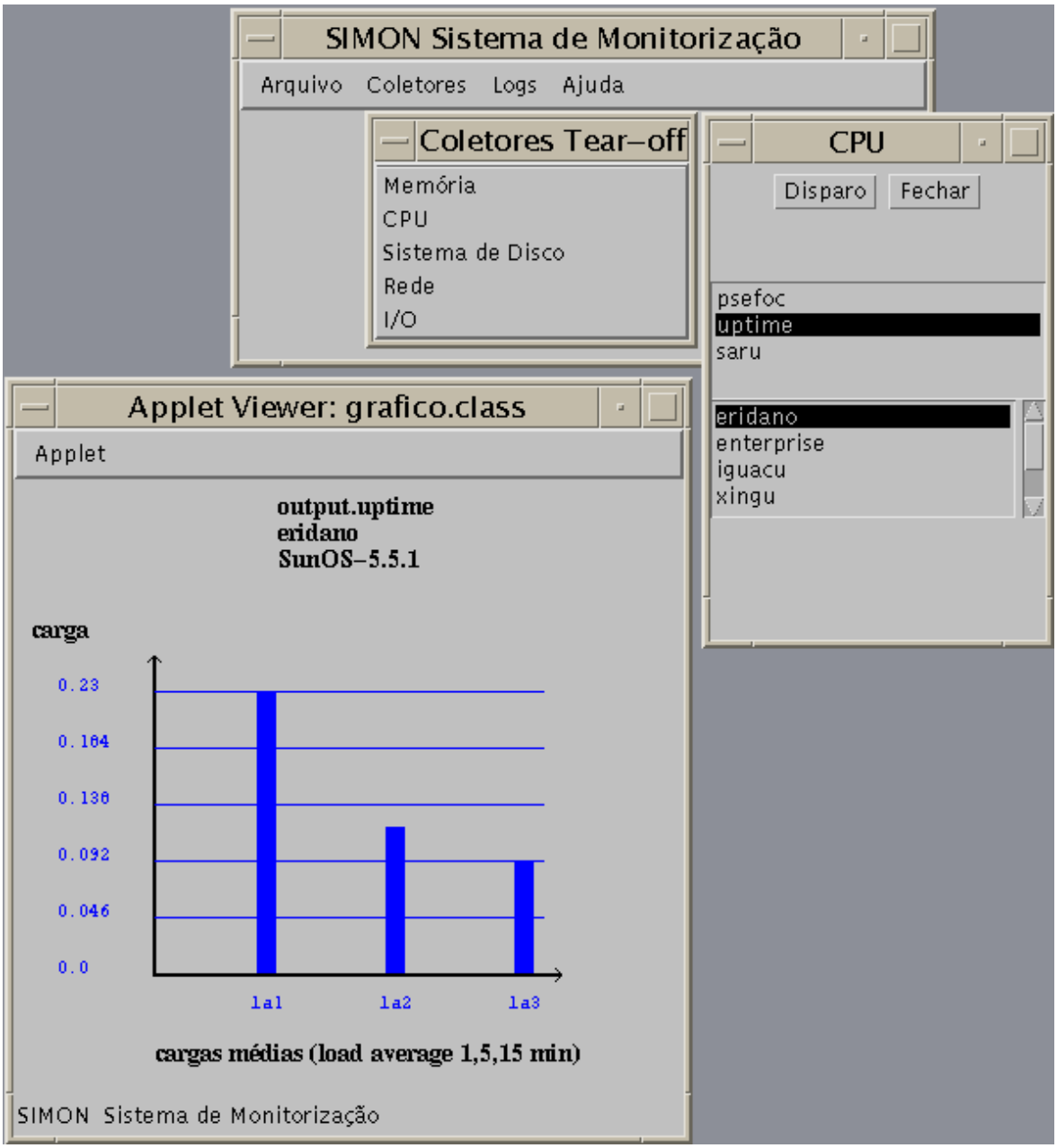

Figura 4.7 - Tela do disparo do coletor uptime

Este `applet` é chamado pelo aplicativo `appletviewer` no módulo de transferência, que por sua vez lê um arquivo .html contendo:

```
<applet
    code="grafico.class"
    width=400
    height=350>
    <param name="arquivout" value="output.<nomedocoletor>">
</applet>
```

Cada coletor com saída na forma de gráfico de barras possui um arquivo .html chamado `grafico_novo_coletor.html`.

Acrescentando coletores à interface Java:

Se a saída do coletor for em forma tabular basta adicionar as linhas abaixo na classe `coletor(MEM/CPU/DISCO/REDE/I/O)` de acordo com a área de desempenho do novo coletor:

```
if ((a.getSelectedItem()) == "novo_agente")
{
  Runtime.getRuntime().exec(exedis);
  File out = new File(path,"output.novo_agente");
  while (!out.exists() && out.length() == 0)
  {
    arq = new leitarq("output.novo_agente");
  }
  arq = new leitarq("output.novo_agente");
  arq.show(true);
}
```

Se a saída do novo coletor for em formato de gráfico de barras, adicionar as linhas:

```
if ((a.getSelectedItem()) == "novo_coletor")
{
    Runtime.getRuntime().exec(exedis);
}
```

Para incluir-se novas máquinas para monitorização, é necessário somente a adição do nome da máquina no arquivo `rel-maq`.

Para a instalação do SIMON uma máquina é considerada central, instalando-se nela o pacote JDK (*Java Development Kit*) 1.1.6 (ou superior). Esta máquina central necessita de espaço físico em disco suficiente para o JDK, em média 30 MB, já que a ferramenta SIMON não consome mais do que 500 KB. É nesta máquina que a interface gráfica do SIMON será executada. As outras máquinas serão as clientes e requerem somente 100 KB de espaço livre no diretório temporário `/tmp`, onde os coletores e seus respectivos resultados ficarão temporariamente armazenados. O SIMON não precisa ter permissões de super-usuário para coletar os dados; ele pode ser instalado e executado em hierarquia pessoal, cuja conta deve conter o arquivo `.rhosts` adequado para que os coletores possam ser executados remotamente e o módulo de transferência possa executá-los remotamente e copiar os dados sem a necessidade de pedido de senha. Além disso, a variável PATH do sistema deve possuir o caminho do diretório onde o JDK está instalado.

4.3.1. Demonstração

O protótipo da ferramenta foi executado para verificação de suas funcionalidades. Para executar a ferramenta na máquina central basta digitar-se `java Simon`. Os resultados são demonstrados a seguir:

A figura 4.8 mostra a tela principal do SIMON que é formada por 4 menus *pop-up*.

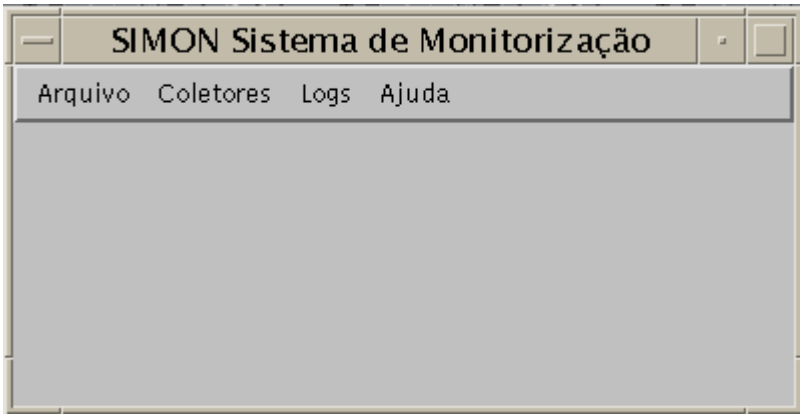

Figura 4.8 - Tela Principal do SIMON

A área de desempenho Memória possui 3 coletores: psefom, vms e swaps, como mostrados na figura 4.9.

Figura 4.9- Tela de disparo dos coletores de Memória

Conforme mostrado na figura 4.10, selecionando-se o coletor psefom (`ps -efo`) na máquina `eridano` os 15 processos que consomem mais memória são visualizados por ordem crescente.

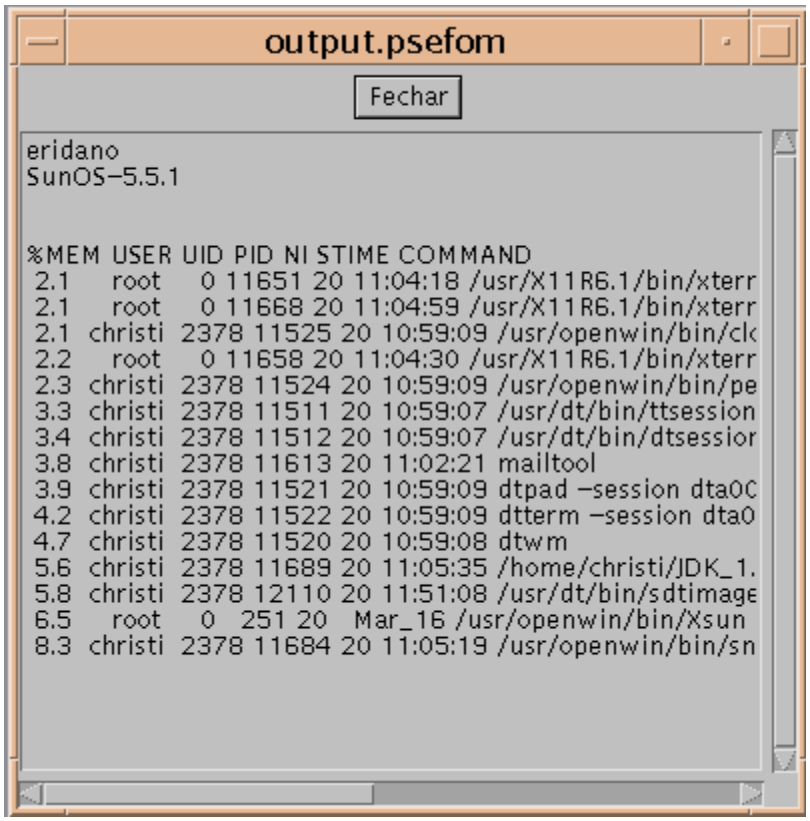

Figura 4.10 - Tela do coletor psefom

A execução do coletor swaps (`swap -s`) na figura 4.11, mostra o gráfico de barras com os campos `alloc` (memória alocada), `reser` (memória reservada), `used` (memória usada) e `avail` (memória disponível) da máquina `eridano`.

Tendo estas informações por várias máquinas o administrador tem uma ampla visão da situação do *swap*, podendo realocar processos naquelas em que o *swap* está crítico, por exemplo.

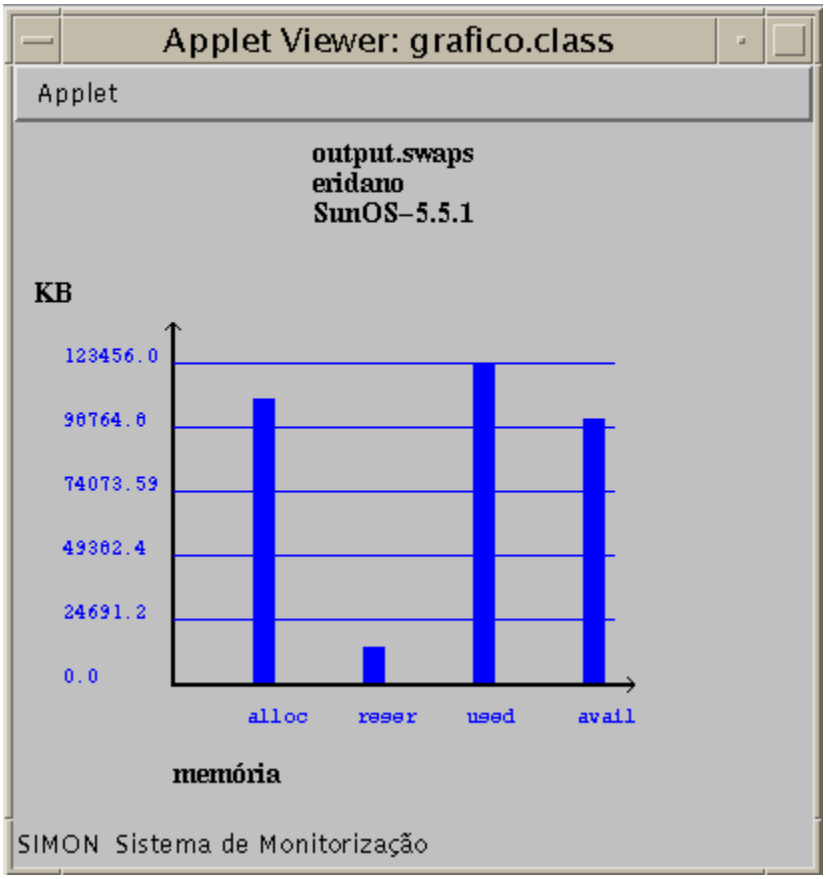

Figura 4.11 - Tela do coletor swaps

O coletor de atividade de `swapping` e `paging`, `vms.sh`, mostra a média dos campos `si` (*swap-in*), `so` (*swap-out*), `pi` (*page-in*) e `po` (*page-out*) na máquina `eridano` na figura 4.12. São feitas 5 medições em intervalos de 5 segundos.

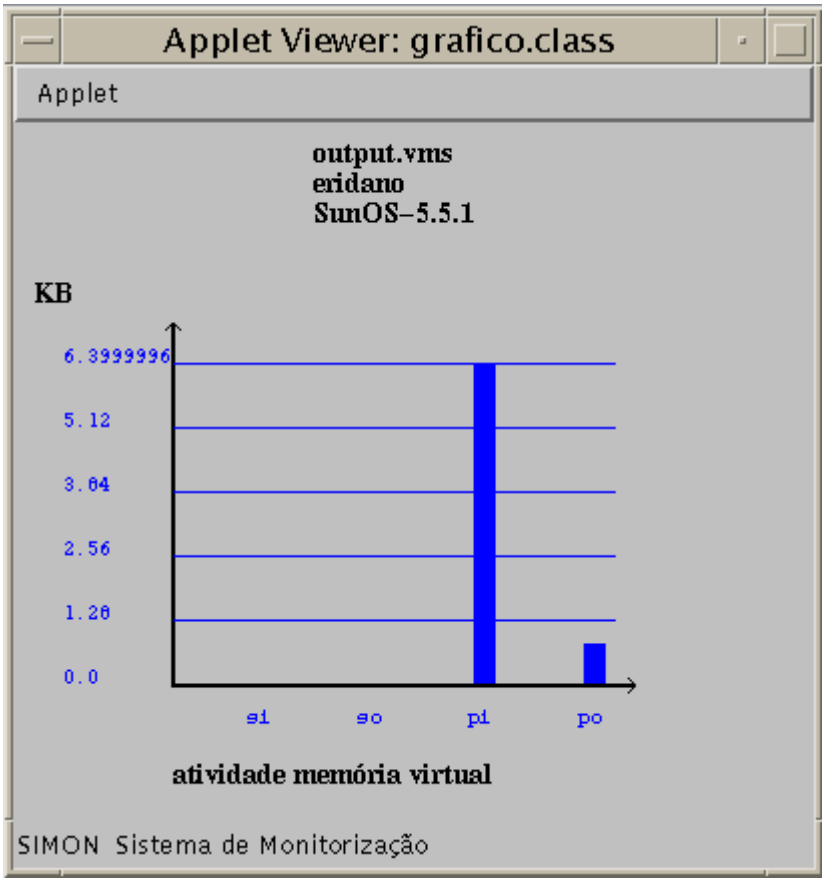

Figura 4.12 - Tela do coletor vms

Acionando-se o item CPU, outra janela composta por 2 botões e 2 listas com *scrollbars* é mostrada (figura 4.13). A primeira lista mostra a relação de coletores de CPU e a segunda a relação das máquinas. O primeito item da cada lista já surge selecionado caso o botão disparo seja executado distraidamente, evitando enviar ao módulo de transferência uma execução “vazia”.

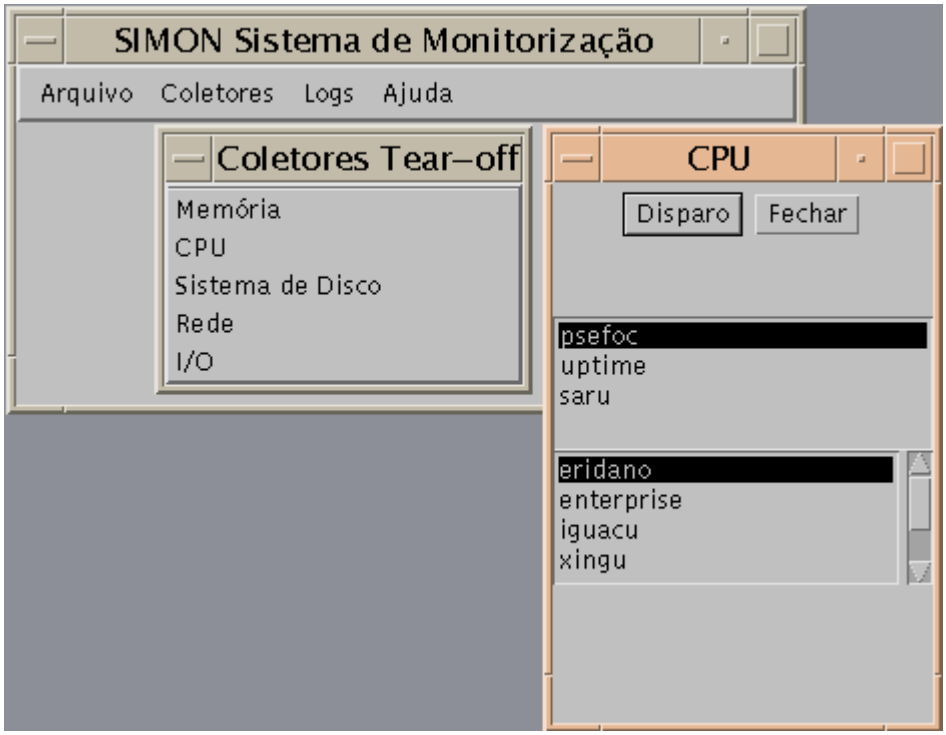

Figura 4.13 - Tela de disparo dos coletores de CPU

Selecionando-se o coletor psefoc (`ps -efo`) e uma máquina, uma janela de texto não editável é mostrada com o resultado da execução deste coletor (figura 4.14). Os 15 processos que consomem mais CPU em ordem crescente de uso são mostrados. Com estes dados o administrador pode verificar quais processos estão consumindo grandes porções da CPU e realizar o mesmo processo para as demais máquinas rapidamente, sem a necessidade de conectar-se ou digitar novamente os comandos; somente são necessários 3 cliques de mouse para obter os resultados para cada máquina

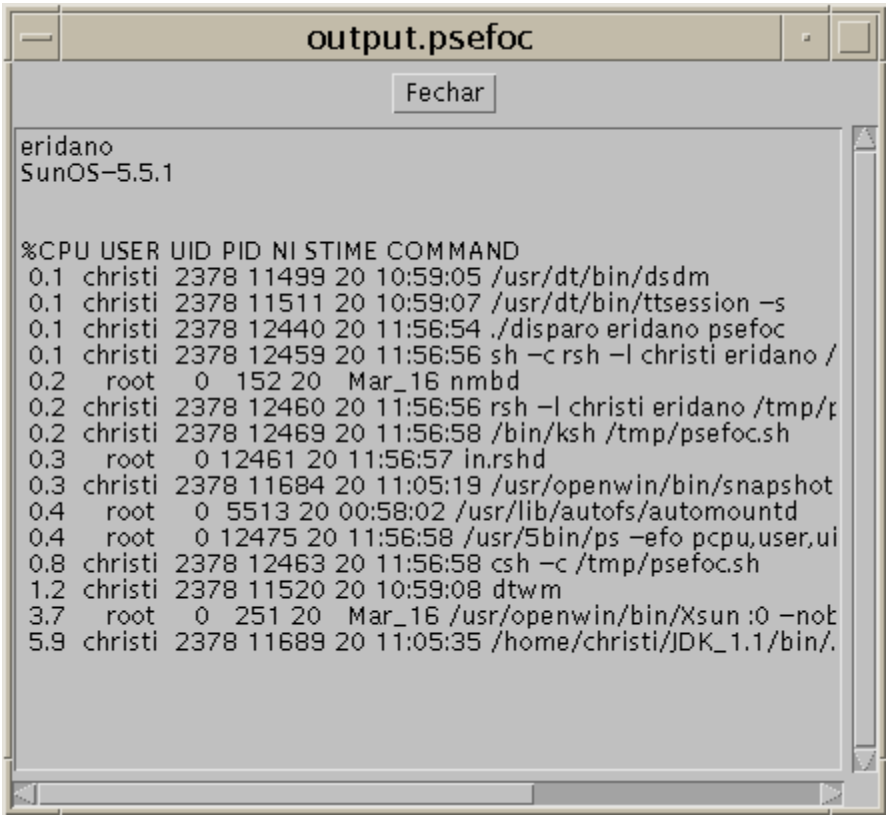

Figura 4.14 - Tela do coletor psefoc

.

A figura 4.15 mostra o resultado da execução do coletor uptime (`uptime`). As cargas médias (la1, la2 e la3) nos últimos 1, 5 e 15 minutos respectivamente são mostradas na forma de gráfico de barras que facilitam o entendimento rápido dos resultados.

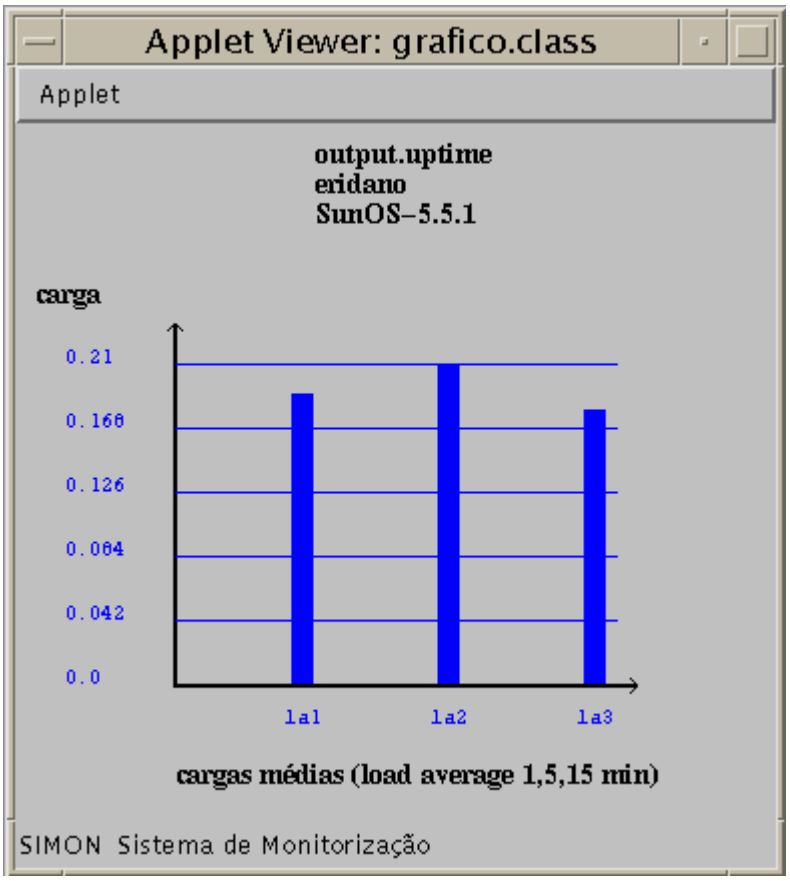

Figura 4.15 - Tela do coletor uptime

A execução do coletor saru (`sar -u`) na máquina `eridano` mostra os percentuais de utilização da CPU em modo usuário (usr), modo sistema (sys) e o tempo que ficou ociosa (idle) esperando I/O, como mostrado na figura 4.16. São feitas 5 medições em intervalos de 5 segundos.

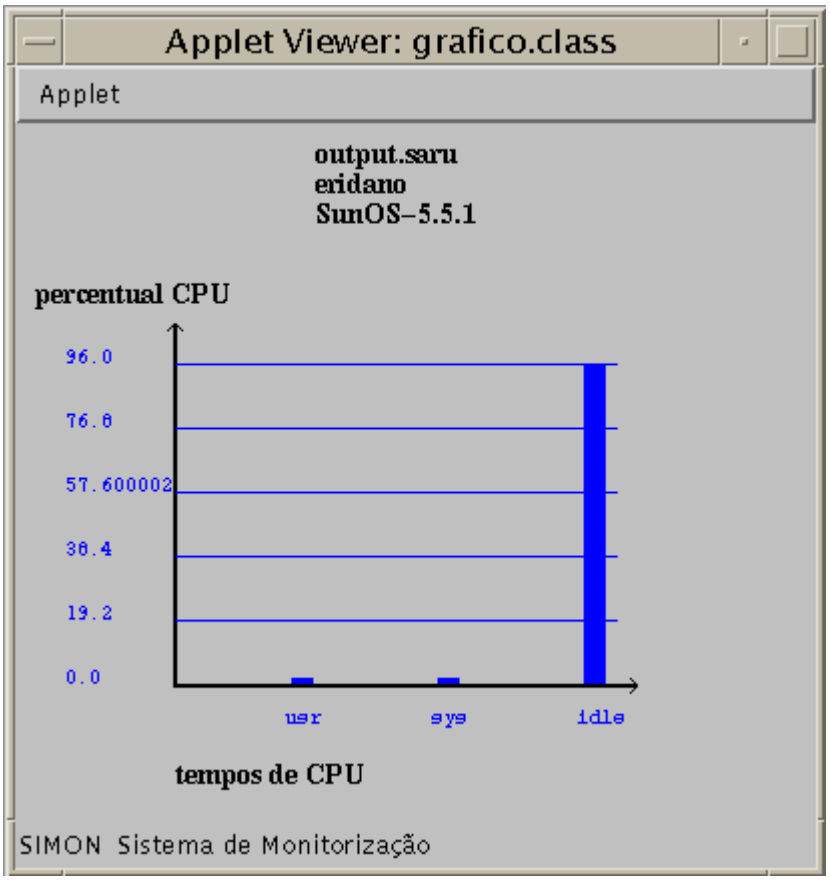

Figura 4.16 - Tela do coletor saru

A área de desempenho Sistema de discos possui 2 coletores: dfkl e iosvt, como mostrado na figura 4.17.

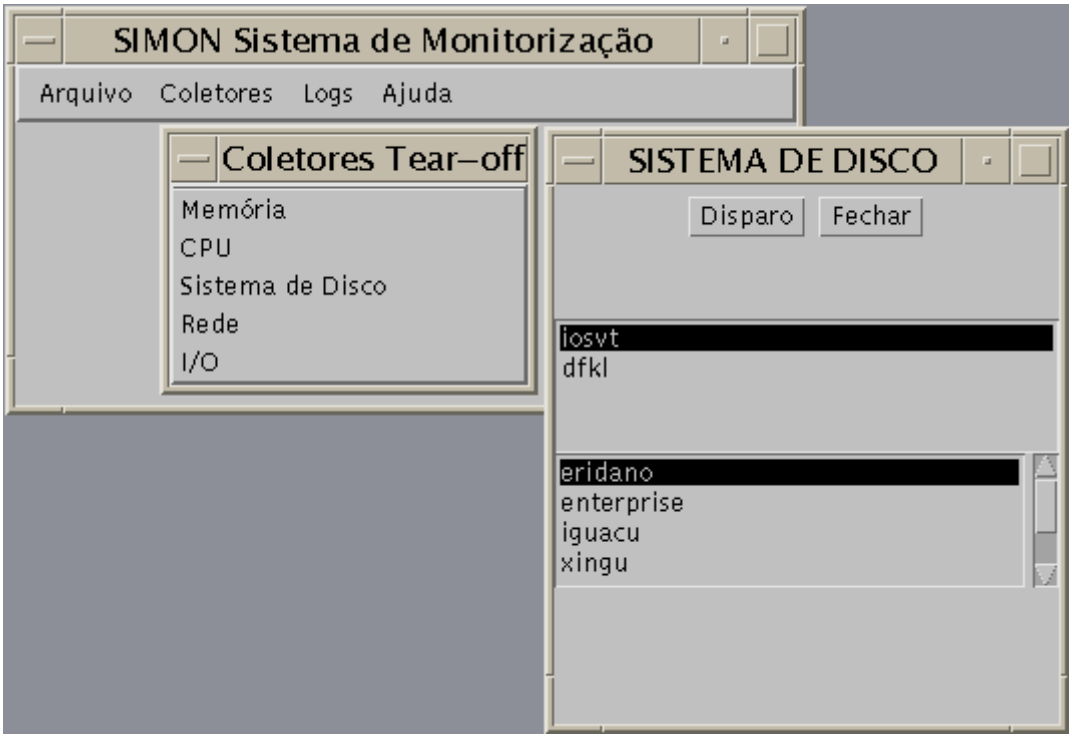

Figura 4.17 - Tela de disparo dos coletores de Sistema de Discos

O gráfico do coletor iosvt (figura 4.18) mostra o resultado da execução do comando `iostat -x 5 1`, onde o valor 5 é o intervalo de medição e o valor 1 é a quantidade de medições. Os tempos de serviço (campo `svc_t`) dos discos locais da máquina `eridano` são mostrados por este coletor. Com base nestes dados pode-se saber quais discos são mais exigidos, e balanceamentos de carga podem ser indicados.

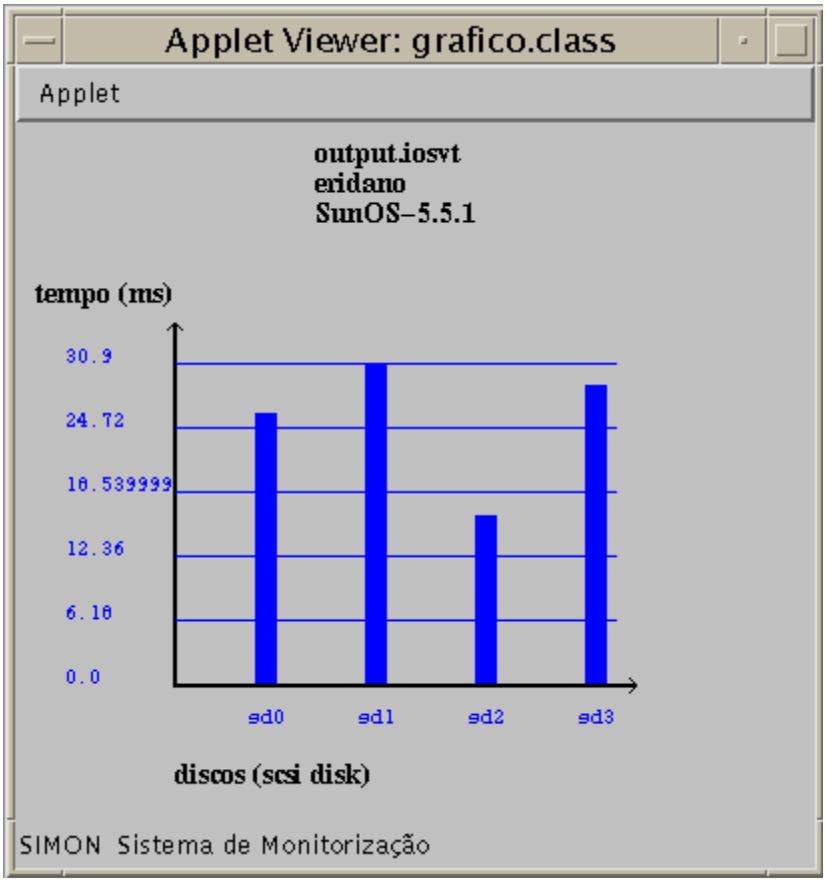

Figura 4.18 - Tela do coletor iosvt

Na figura 4.19, o gráfico do coletor dfkl (`df -kl`) mostra a porcentagem de utilização dos discos locais da máquina `eridano`. Os nomes dos discos locais não são mostrados e sim onde estão montados para facilitar a análise e ação do administrador quando descobre que algum disco está em sua capacidade máxima. O administrador pode optar por realocar os dados para outros discos da máquina ou outras máquinas.

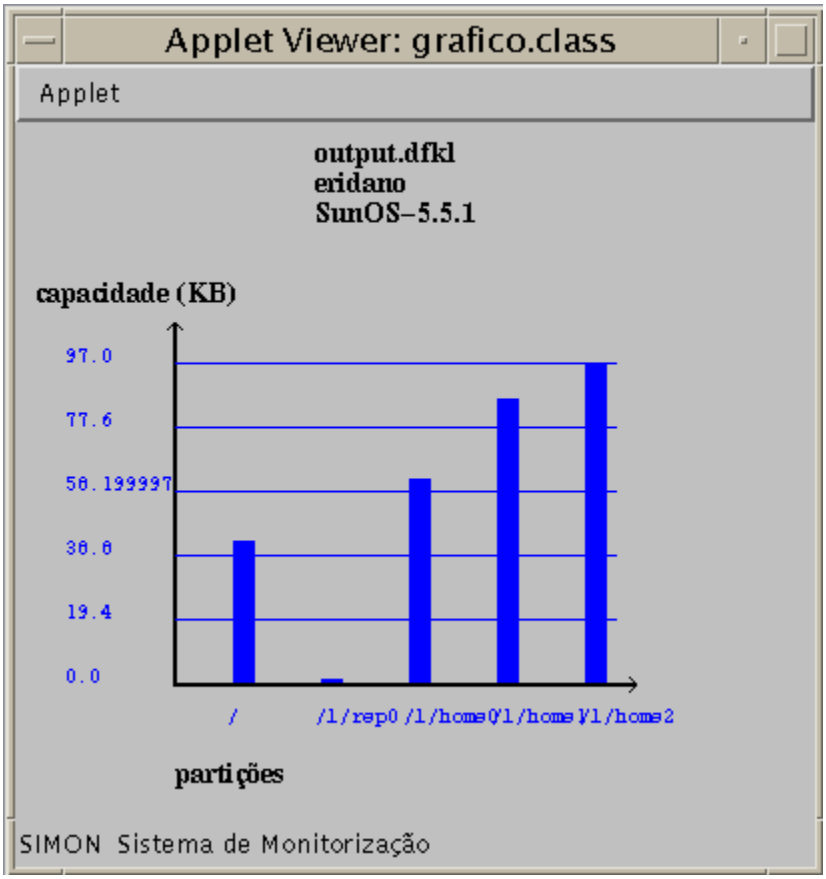

Figura 4.19 - Tela do coletor dfkl

A quarta área de desempenho é Rede. Conforme a figura 4.20 existem 2 coletores: netsi e trafego. Os dados de rede são mostrados em forma de tabelas.

Figura 4.20 - Tela de disparo dos coletores de Rede

O resultado da execução do coletor netsi (`netstat -i`) na máquina `eridano` é mostrado na figura 4.21. Os dados por interface de rede (exceto a interface *loopback*) visualizados são: número de erros de entrada (`ierrs`), número de pacotes de saída (`opkts`), número de erros de saída (`oerrs`) e número de colisões (`collis`).

A análise da interface de rede com sobrecarga, taxa de erros ou colisões alta pode indicar conexões lentas, servidores fora da rede, tráfego pesado de NFS ou problemas físicos na rede (cabeamento ou equipamentos).

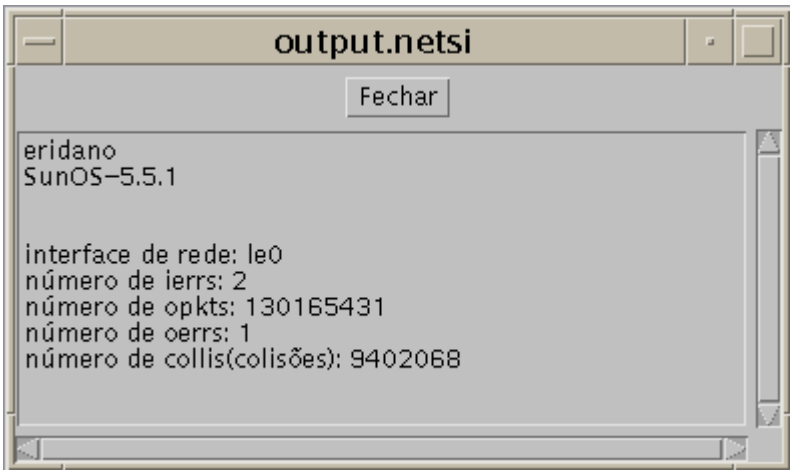

Figura 4.21 - Tela do coletor netsi

O coletor trafego mostrado na figura 4.22 (`ping -s host packetsize interval`) na máquina `eridano` fornece a percentagem de pacotes perdidos no intervalo de coleta e os tempos mínimo, médio e máximo de resposta em ms dos pacotes.

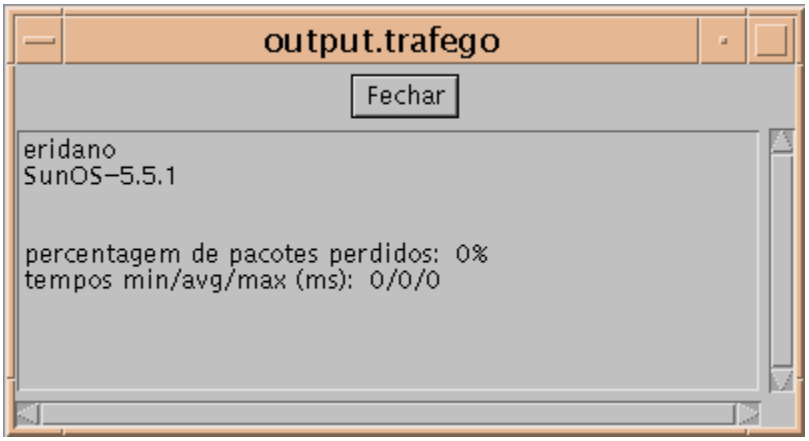

Figura 4.22 - Tela do coletor trafego

A figura 4.23 mostra os dois coletores de I/O denominados nfs e balcarga. Esta área de desempenho tem o objetivo de agrupar os coletores que monitorizam os fatores que afetam o desempenho de I/O. O NFS, por ser um grande usuário da rede e de discos, encaixa-se nesta área. A função deste coletor não é monitorizar toda a atividade do NFS, mas sim fornecer dados básicos de como o NFS está funcionando.

Figura 4.23 - Tela de disparo dos coletores de I/O

O coletor de I/O (figura 4.24) nfs.sh mostra graficamente os dados do lado cliente NFS. Os dados são divididos na saída do comando `nfsstat -cr` em orientado (TCP) e não orientado (UDP). As informações consideradas importantes neste comando são:

co (calls via TCP), bco (badcalls via TCP), bxo (badxids via TCP), to (timeouts via TCP), cc (calls via UDP), bcc (badcalls via UDP), rc (retrans via UDP), bxc (badxids via UDP) e tc (timeouts via UDP).

Segundo [3], se o número de to for igual ou superior a 1% (um porcento) do número de co, pode significar que:

- o servidor pode estar fora da rede ou
- a conexão está lenta ou
- há problemas no *hardware* de rede.

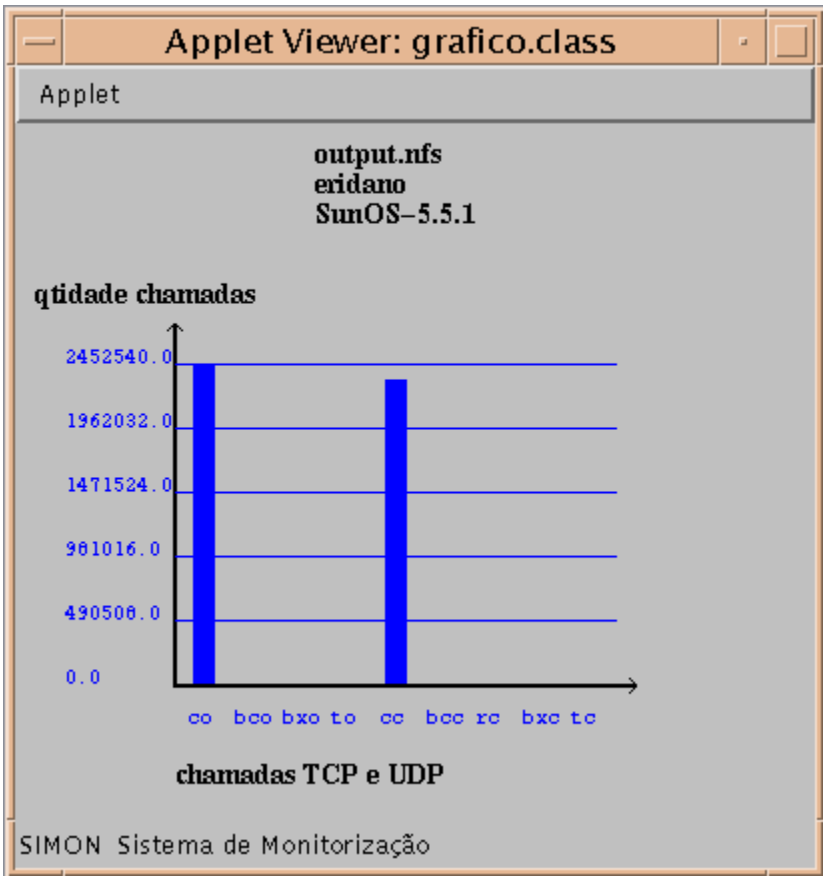

Figura 4.24 - Tela do coletor nfs

O coletor balcarga (figura 4.25) fornece alguns dados como: memória virtual disponível em KB, porcentagem de utilização dos discos locais da máquina e cargas médias dos últimos 1, 5 e 15 minutos. Com estes dados o administrador pode fazer o balanceamento de carga/utilização de cada máquina, distribuindo os serviços do sistema e as tarefas dos usuários por máquinas menos carregadas. É possível a inclusão de outros tipos de informação para uma análise mais apurada da carga nas máquinas.

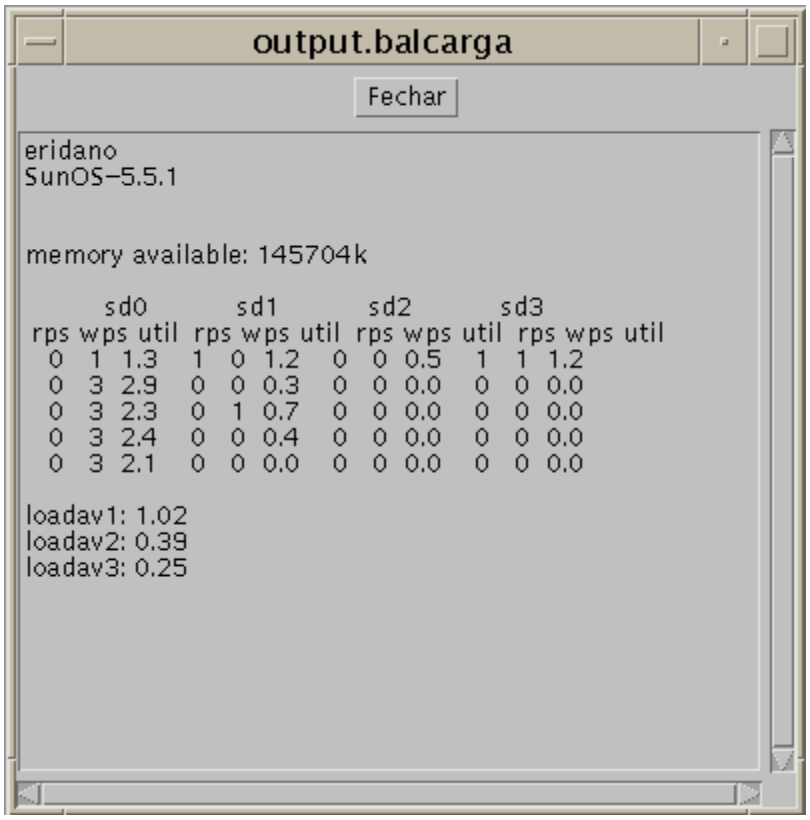

Figura 4.25 - Tela do coletor balcarga

Capítulo 5

Conclusão

O levantamento dos fatores que afetam o desempenho realizado no Capítulo 2 foi importante para focalizar os problemas e formular as heurísticas oferecidas como solução.

Os *software* existentes no mercado e a pesquisa realizada na literatura sobre ferramentas tanto para gerência de redes quanto para administração de sistemas ajudou a perceber a necessidade de uma ferramenta simples que mostrasse os dados de uma forma direta e clara. Gráficos mostram dados desta forma, se bem desenhados e concisos. Contudo, a experiência de uso de algumas delas, junto com as especificações disponíveis, revelou que geralmente elas são demasiado complexas e pesadas, tanto com relação a ciclos de CPU quanto com tráfego gerado na rede. Com certeza, isto advém da tentativa de cobrir uma longa lista de itens benéficos ao administrador.

Das plataformas comerciais pesquisadas, tive a oportunidade de utilizar na minha profissão o Tivoli. É uma ferramenta que encaixa-se na administração de sistemas, formada por um conjunto de *software* com finalidades e funcionalidades diferentes. Ele faz uso de outra ferramenta não pesquisada neste trabalho, chamada NetView, para coletar dados de gerência de rede. O NetView também utiliza gráficos para mostrar os dados coletados de tráfego de rede. Outra ferramenta que o Tivoli utiliza é o Sentry, *software* composto por agentes predeterminados e configuráveis que fazem a monitorização e atuação de desempenho nas máquinas clientes. Os agentes são configurados com níveis de alarmes. Quando o limite é atingido, eventos são enviados ao console de eventos (T/EC) e mostrados de acordo com o grau de severidade previamente configurado. O Tivoli é uma ferramenta completa, contudo exige no mínimo duas máquinas dedicadas, uma com o *framework* (pacotes básicos para que a ferramenta funcione) e outra com o banco de dados relacional.

O *software* SunNet Manager 2.2 também foi estudado no início da implementação deste trabalho. O objetivo era criar novos agentes de desempenho e utilizar a funcionalidade Grapher deste *software* para mostrar os dados coletados. A idéia inicial era embutir a ferramenta *top*[9] como um novo agente e plotar gráficos com os resultados. Contudo fazer novos agentes é uma tarefa complexa, e o funcionamento da aplicação gerente provou

[9] *top* é uma ferramenta que mostra os processos que consomem mais CPU de modo interativo e em tempo real.

requisitar mais poder de processamento em máquinas dedicadas do que havia ao nosso alcance na época. A experiência piloto baseada em SNMP com aplicações gerentes comerciais foi desapontadora devido a nossos escassos recursos computacionais, o que acabou direcionando a pesquisa para abordagens mais leves e atuais.

Os pacotes comerciais Transcend, Optivity, CiscoWorks, OpenView e Spectrum são plataformas de gerência de redes. Monitorizam basicamente a atividade de tráfego e utilizam implementações SNMP, RMON ou ambas para captura e análise dos dados.

O pacote Solstice traz embutido o SunNet Manager. É uma plataforma de gerência de redes e traz também a vantagem de visualizar os dados coletados de tráfego na rede na forma de gráficos, utilizando a funcionalidade Grapher do SunNet Manager.

Uma ferramenta que não se encaixa exatamente na área comercial e acadêmica, pois foi criada por um especialista da Sun e é de uso livre, é a Symbel. Ela se parece com o SIMON, pois utiliza *scripts* para a monitorização, é dividida por áreas de desempenho e somente trabalha na plataforma Solaris 2.X. O Symbel utiliza um código de cores para analisar o estado do sistema.

O *software* Unicenter TNG e SiteScope fazem uso da *internet* para gerência remota ou para monitorização. Big Brother não utiliza SNMP e sim pequenos *shell scripts* realizando a notificação e a monitorização remota via *internet*, utiliza um código de cores para apresentar os resultados da monitorização, como o SymbEl.

As ferramentas acadêmicas pesquisadas, exceto o SAFO, utilizam o SNMP e/ou RMON para coleta de dados. O sistema Olho Vivo apresenta a característica de inteligência artificial e atua somente no nível de tráfego de rede. A característica de acionamento de alarme quando um evento ocorre é a principal missão da ferramenta Buzzerd pois o objetivo é minimizar o *downtime*. As ferramentas HLMMIB, Satool e Safo utilizam dados coletados de comandos UNIX e o Igor realiza a execução remota de *shell scripts* como o SIMON porém possui tratamento para várias conexões simultâneas.

A característica principal do I-DREAM é a modificação em tempo real da evolução da rede utilizando páginas *web* para a visualização. O SAGRES introduz um novo conceito, gerência de sistemas, onde as ferramentas da gerência de redes e administração de sistemas trabalham unidas. Ambos fazem parte de um projeto de formalização da administração de sistemas heterogêneos, o projeto FLASH da UFPE.

O SIMON é uma ferramenta de administração de sistemas para redes de pequeno porte com uma abordagem voltada para realizar o ajuste fino (*tuning*) dos sistemas. Sua interface gráfica é clara e bastante simples nesta primeira implementação, já que o propósito deste trabalho é provar a viabilidade da abordagem.

É fácil de usar oferecendo uma interface orientada a menus e botões, portável pois tanto os *scripts* quanto a interface gráfica Java não necessitam de recompilações e plota gráficos simples e concisos. Os dados dos coletores são confiáveis e atuais. SIMON é da mesma forma fácil de instalar e configurar.

Embora o SIMON divida a monitorização em 5 áreas de desempenho, a correlação de problemas não é afetada, ao contrário, o objetivo é cruzar os dados de forma a obter uma solução para otimizar o sistema e corrigir os problemas.

Através das informações armazenadas pelos *logs* de execução de cada coletor do SIMON, pode-se, além de averiguar as máquinas clientes com o auxílio dos gráficos e tabelas no momento da monitorização, estudar os dados históricos coletados anteriormente e analisá-los para compreender a situação de desempenho das máquinas num âmbito geral.

Finalizando, é uma ferramenta pequena com menos de 200KB de tamanho de código e é altamente extensível, novos coletores e outras plataformas podem ser facilmente adicionadas ao sistema para monitorização mediante o acréscimo de poucas linhas de código.

Apêndice A

Comando ps

O comando `ps` que mostra informações sobre processos ativos [57].

Formato:

```
ps[ -aAdeflcj ][[ -o format ] ... ][ -t term ][ -u uidlist ][ -U
uidlist ][ -G gidlist ][ -p proclist ][ -g grplist ][-s sidlist ]
```

Opções:

`-a`: lista informações sobre os processos mais frequentemente requisitados.

`-A`: lista informações para todos os processos.

`-d`: lista informações sobre todos os processos exceto sessões *leaders*.

`-e`: lista informações sobre cada processo em execução.

`-f`: gera uma listagem completa.

`-l`: gera uma listagem longa.

`-c`: mostra informações em formato que reflete propriedades do escalonador descritas em `priocntl`.

`-j`: lista processos por `id` e `gid`.

`-o formato`: mostra informações de acordo com formato definido por lista de variáveis fornecidas, as quais são:

```
user ruser group rgroup uid ruid gid rgid pid ppid pgid sid pri
opri pcpu pmem vsz rss osz nice class time etime stime f s c tty
addr wchan fname comm args
```

`-t term`: lista somente dados de processos associados com `term`.

`-u uidlist`: lista informações de dados de processos cujo `id` aparece na `uidlist`.

`-U uidlist`: lista informações de processos cujo `id` aparece na `uidlist`.

`-G gidlist`: lista informações de processos agrupados por `gid`.

`-p proclist`: lista somente dados de processos por `pid`.

`-g grplist`: lista somente dados de processos agrupados por *group leader*. (Um *group leader* é um processo cujo `id` é idêntico ao `gid` do processo).

`-s sidlist`: lista informações de todas sessões cujo `id` aparece na `sidlist`.

Comando swap

O comando `swap` que fornece um método para adicionar, remover e monitorizar as áreas de *swap* do sistema [21].

Formato:

```
swap -a swapname [ swaplow ] [ swaplen ]
 swap -d swapname [ swaplow ]
 swap -l
 swap -s
```

Opções:

`-a swapname`: adiciona uma área específica de *swap*, `swapname` é o nome do arquivo de *swap*.

`-d swapname`: remove uma área específica de *swap*, `swapname` é o nome do arquivo de *swap*.

`-l`: lista o estado de todas as áreas de *swap*.

`-s`: mostra sumário de informações sobre o espaço total de *swap* usado e disponível.

Comando vmstat

O comando `vmstat` que examina o sistema e relata estatísticas sobre processos, memória virtual, disco, *trap* e atividade de CPU [12].

Formato:

```
vmstat [ -cisS ] [ disks ] [ interval [ count ] ]
```

Opções:

`-c`: relata estatísticas de *cache flushing*.

`-i`: mostra o número de interrupções por dispositivo.

`-s`: mostra o número total de eventos do sistema desde a inicialização da máquina.

`-S`: relata atividade de *swapping* e *paging*. Campos: si (*swap-ins*), so (*swap-outs*), pi (*page-ins*), po (*page-outs*).

Comando uptime

O comando `uptime` que mostra o tempo corrente, o tempo que o sistema está no ar, a média de *jobs* em execução na fila nos últimos 1, 5 e 15 minutos.[11]

Formato:

```
uptime
```

Comando sar

O comando `sar` que mostra a atividade cumulativa do sistema operacional em n intervalos de t segundos conforme opções.[20]

Formato:

```
sar [ -aAbcdgkmpqruvwy ][ -o filename] t [ n ]
sar [ -aAbcdgkmpqruvwy ][ -e time ][ -f filename ][ -i sec ][ -s time ]
```

Opções:

`-a`: reporta uso de rotinas de sistema de acesso a arquivo.

`-A`: relata todos dados, é equivalente as opções: `-abcdgkmpqruvwy`.

`-b`: relata atividade de *buffer*.

`-c`: relata chamadas de sistema.

`-d`: relata atividade de cada bloco de dispositivo.

`-g`: relata atividades de paginação.

`-k`: reporta atividades de KMA (*kernel memory allocation*).

`-m`: reporta atividades de semáforos e mensagens.

`-p`: relata atividades de *paging*.

`-q`: reporta tamanho médio da fila enquanto ocupada e porcentagem do tempo ocupado.

`-r`: relata páginas de memória e blocos de disco não usados.

`-u`: reporta utilização de CPU. Campos: `%usr`, `%sys`, `%wio`, `%idle`: porção de tempo em modo usuário, sistema e ocioso com processos esperando I/O e ocioso.

`-v`: relata estado de tabelas de processos, `inode` e arquivos.

`-w`: reporta atividade de sistema de *swapping* e *switching*.

`-y`: relata atividade de dispositivo `TTY`.

`-e time`: seleciona dados por tempo. *Default* é 18:00.

`-f filename`: usa `filename` como estrutura fonte para `sar`. *Default* é o arquivo de dados `/usr/adm/sa/sadd`.

`-i sec`: seleciona dados em intervalos próximos de `sec`.

`-o filename`: grava exemplos em arquivo, `filename`, em formato binário.

`-s time`: seleciona dados por tempo na forma: hh[:mm]. *Default* é 08:00.

Comando df

O comando `df` que mostra a quantidade de espaço em disco ocupada por sistemas de arquivos montados ou não, diretórios ou recursos montados, a quantidade de espaço usado e disponível e quanto da capacidade total do sistema de arquivo tem sido usada.[14]

Formato:

```
/usr/bin/df [-abeFgklnotV ]
```

Opções:

`-a` : mostra todos sistemas de arquivos.

`-b`: número total de KB livres.

`-e`: mostra somente o número de arquivos livres.

`-F tipo_de_sistema_de_arquivo`: especifica o tipo de sistema de arquivo para mostrar.

`-g`: mostra a estrutura `statvfs` completa.

`-k`: mostra os números em KB. A saída consiste de uma linha de informação para cada sistema de arquivo, que inclui nome, espaço total alocado, quantidade de espaço alocado para arquivos, quantidade total de espaço disponível para criação de novos arquivos e percentagem de espaço disponível.

`-l`: mostra somente sistemas de arquivos locais (sistemas de arquivos montados).

`-n`: nome do tipo do sistema de arquivo.

`-o opções_tipo_de_sistema_de_arquivo`: opções para sistemas de arquivos específicos, separados por vírgula.

`-t`: mostra listagem com totais.

`-V`: repete o conjunto completo de sistemas de arquivos especificados na linha de comando, mas não os executa.

Comando iostat

O comando `iostat` que iterativamente relata atividade de terminal, disco e I/O e utilização de CPU. A primeira linha da saída fornece informações desde a reinicialização da máquina. [13]

Formato:

```
/usr/bin/iostat [ -cdDeEIMnpPtx ][ -l n ][ disk ...][ interval[
count ]]
```

Opções:

`-c`: relata percentagem de tempo que o sistema gasta em modo usuário, sistema, esperando por I/O e ocioso.

`-d`: para cada disco relata o número de KB transferidos por segundo, o número de transferências por segundo e a média do tempo de serviço em milissegundos(ms).

`-D`: para cada disco relata leituras por segundo, escritas por segundo e percentagem de utilização de disco.

`-e`: mostra sumário das estatísticas de erro de dispositivo.

`-E`: mostra todas estatísitcas de erro de dispositivo.

`-I`: relata os resultados em cada intervalo.

`-l n`: limita o número de discos na saída com `n`.

`-M`: mostra o *throughput* dos dados em MB/seg ao invés de KB/seg.

`-n`: mostra nomes em formato descritivo.

`-p`: para cada disco mostra estatísticas por partição e por dispositivo.

`-P`: para cada disco mostra somente estatísticas por partição.

`-t`: número de caracteres lidos e escritos no terminal por segundo.

`-x`: para cada disco relata estatísticas estendidas como número de transferências em KB por segundo, com leitura e escrita mostradas em separado, média de comandos esperando na fila, média de comandos sendo processados pelo dispositivo, tempo de serviço e percentagem de tempo que os comandos estão esperando na fila.

Comando netstat

O comando `netstat` que mostra o conteúdo de várias estruturas de dados relacionadas à rede em vários formatos, dependendo das opções fornecidas. [17]

Formato:

```
netstat [ -anv ]
netstat [ -g|-m|-p|-s|-f address_family ][ -n ][ -P protocol ]
netstat {[ -i ] [ -I interface ]} [ interval ]
netstat -r [ -anv ]
netstat -M [ -ns ]
netstat -D [ -I interface ]
```

Opções:

`-a`: mostra o estado de todos os *sockets* e todas as entradas de tabelas de roteamento

`-f address_family`: limita estatísticas para endereço de família inet ou unix.

`-g`: mostra os membros do grupo *multicast* pelas interfaces.

`-i`: mostra o estado das interfaces que são usadas pelo tráfego TCP/IP.

`-m`: mostra estatísticas de *STREAMS*.

`-n`: mostra endereços de rede como números.

`-p`: mostra as tabelas de resolução de endereço (ARP).

`-r`: mostra as tabelas de roteamento.

`-s`: mostra estatísticas por protocolo.

`-v`: mostra informações adicionais para os *sockets* e tabelas de roteamento.

`-I interface`: mostra o estado de um interface em particular.

`-M`: mostra tabelas de roteamento *multicast*.

`-P protocolo`: mostra estatísticas ou estados de todos *sockets* por protocolo.

`-d`: mostra o estado de todas as interfaces que estão sob controle do DHCP (*Dynamic Host Configuration Protocol*).

`-D`: mostra o estado das interfaces configuradas DHCP.

Comando ping

O comando `ping` que utiliza o datagrama ECHO_REQUEST do protocolo ICMP para extrair uma resposta do tipo ECHO_RESPONSE de uma máquina específica. [18]

Formato:

```
      /usr/sbin/ping host [ timeout ]
        /usr/sbin/ping [ -s ][ -dlLnrRv ][ -i nterface ][ - I interval ][
-t ttl ] host [ packetsize ][ count ]
```

Opções:

`-d`: configura a opção SO_DEBUG do socket.

`-l`: perde a rota fonte.

`-L`: retira o loopback dos pacotes multicast.

`-n`: mostra os endereços de rede como números.

`-r`: ignora as tabelas normais de roteamento e envia diretamente para uma máquina ligada na rede

`-R`: registra a rota.

`-v`: repete a saída no vídeo.

`-i interface`: especifica a interface de saída para uso dos pacotes multicast.

`-I interval`: especifica o intervalo entre transmissões sucessivas.

`-t ttl`: especifica o tempo de vida do IP para pacotes unicast e multicast.

Comando nfsstat

O comando `nfsstat` que mostra informações estatísticas sobre o NFS e RPC, interfaces do `kernel`. [19]

Formato:

```
nfsstat [ -cmnrsz ]
```

Opções:

`-c`: mostra informações do lado cliente (NFS e RPC).

`-m`: mostra estatísticas para cada sistema de arquivo NFS montado.

`-n`: mostra informações do lado NFS para ambos lados cliente e servidor.

`-r`: mostra informações do RPC.

`-s`: mostra informações do lado servidor.

`-z`: reinicializa para zero as estatísticas.

Referências Bibliográficas

[1] NEMETH, Evi, SNYDER, Garth, SEEBASS, Scott, HEIN Trent R. **UNIX System Administration Handbook**. Second Edition. Prentice-Hall Inc., 1995. 779p.

[2] COCKCROFT, Adrian. **Sun Performance and Tuning**. SunSoft Press, 1995.

[3] Digital Equipament Corporation. [on line]. Digital Equipament Corporation: 1995 [citado 8 junho 1998]. Disponível em World Wide Web:<URL:http://?.html>.

[4] LOUKIDES, Mike. **System Performance Tuning**. O´Reilly & Associates, Inc., 1990.

[5] Solaris 2.6 System Administrator Collection Vol. 1,Sun Microsystems,Inc. 1997.

[6] Man Page at/batch, Sun Microsystems, Inc. 1995.

[7] Man Page nice, Free Software Foundation, Inc. 199?.

[8] Man Page renice, Sun Microsystems, Inc. 1995.

[9] Man Page cron, Sun Microsystems, Inc. 1994.

[10] Man Page limit/ulimit, Sun Microsystems, Inc. 1995.

[11] Man Page uptime, Sun Microsystems, Inc. 1994.

[12] Man Page vmstat, Sun Microsystems, Inc. 1994.

[13] Man Page iostat, Sun Microsystems, Inc. 1994.

[14] Man Page df, Gnu FileUtils. 1998.

[15] Man Page newfs, Sun Microsystems, Inc. 1994.

[16] Man Page tunefs, Sun Microsystems, Inc. 1994.

[17] Man Page netstat, Sun Microsystems, Inc. 1993.

[18] Man Page ping, Sun Microsystems, Inc. 1995.

[19] Man Page nfsstat, Sun Microsystems, Inc. 1991.

[20] Man Page sar, Sun Microsystems, Inc. 1993.

[21] Man Page swap, Sun Microsystems, Inc. 1994.

[22] Man Page rsh, Sun Microsystems, Inc. 1994.

[23] Man Page rcp, Sun Microsystems, Inc. 1994.

[24] Man Page tmpfs, Sun Microsystems, Inc. 1990.

[25] PIERCE, Clinton. **The Igor System Administration Tool**. Chicago, EUA: Proceedings of the Tenth USENIX System Administration Conference(LISA X), outubro/1996.

[26] MILLER, Todd, STIRLEN, Christopher, NEMETH, Evi. **satool - A System Administrator´s Cockpit, An Implementation**. Monterey, EUA: Proceedings of the LISA VII, novembro/1993. p. 119-129.

[27] MEDINA, Roseclea D.,TAROUCO, Liane M. R. **SAFO-Sistema Agregador de Ferramenta de Operação de Rede**. Fortaleza,CE: Anais II WAIS, maio/1996. p. 17-29.

[28] ARTOLA, Esmilda S., TAROUCO, Liane M. R. **Um Sistema Especialista para Gerência Pró-ativa Remota.** Fortaleza,CE: Anais XIV SBRC, maio/1996. p. 118-139.

[29] UCHÔA, Rodrigo C., RODRIGUEZ, Noemi D. L. R. **Suporte para Monitoramento e Controle de Carga em Sistemas Distribuídos**. Belo Horizonte, MG: Anais XIII SBRC, maio/1995. p. 123-142.

[30] HARDY, Darren R., MORREALE, Herb M. **Buzzerd: Automated Systems Monitoring with Notification in a Network Environment**. Long Beach, EUA: Proceedings of the LISA VI, outubro/1992. p. 203-210.

[31] Silva, Fábio Q. B. da, Franklin, Danielle M., et alli. **I-DREAM: An Intranet-baseD REsource and Application Monitoring System.** [on line]. [citado em 10 abril 1999]. Disponível em World Wide Web: <URL:http://www.di.ufpe.br/~flash/resultados/artigos/idream.htm>.

[32] Filho, Raimir H., Oliveira, Mauro, et alli. **Da Administração de Redes à Gerência de Sistemas Baseada em Conhecimento.** [on line]. [citado em 10 abril 1999]. Disponível em World Wide Web: <URL:http:// www.di.ufpe.br/~flash/resultados/artigos/relatorios/relat-flash0798.ps>.

[33] 3COM Corporation. **Transcend Enterprise Manager Version 4.2 For UNIX**. [on line]. 3COM Corporation: 1998 [citado 18 novembro 1998]. Disponível em World Wide Web:<URL:http://www.3com.com/products/dsheets/400193.html>.

[34] 3COM Corporation. **With Transcend Enterprise Manager for UNIX, you can** ....[on line]. 3COM Corporation: 1998 [citado 18 novembro 1998]. Disponível em World Wide Web:<URL:http://www.3com.com/products/dsheets/400193a.html>.

[35] 3COM Corporation.**Transcend LANSentry Manager**[on line]. 3COM Corporation: 1998 [citado 18 novembro 1998]. Disponível em World Wide Web:<URL:http://www.3com.com/products/dsheets/400306.html>.

[36] 3COM Corporation.**Transcend Traffix Manager Global Traffic Management for UNIX**[on line]. 3COM Corporation: 1998 [citado 18 novembro 1998]. Disponível em World Wide Web:<URL:http://www.3com.com/products/dsheets/400298.html>.

[37] 3COM Corporation.**Transcend Traffix Manager Global Traffic Management for UNIX**[on line]. 3COM Corporation: 1998 [citado 18 novembro 1998]. Disponível em World Wide Web:<URL:http://www.3com.com/products/dsheets/400298a.html>.

[38] Bay Networks.**Optivity**:take control of your network [on line]. Bay Networks: 1998 [citado 6 agosto 1998]. Disponível em World Wide Web: <URL: http://business5.baynetworks.com/>.

[39] Sun Microsystems,Inc.**Solstice Domain Manager and Solstice Site Manager**[on line]. Sun Microsystems: 1998 [citado 23 novembro 1998]. Disponível em World Wide Web: <URL: http://www.sun.com/domainmgr/wp-site_domain_mgr.html>.

[40] Cisco Systems Inc. **Network Management Applications** [on line].Cisco Systems Inc: 1998 [citado 25 novembro 1998].Disponível em World Wide Web: <URL: http ://www.cisco.com/univercd/cc/td/doc/prod_cat/79995.htm>.

[41] Hewlett-Packard Co. **Positioning HP OpenView Network Management Solutions** [on line]. Hewlett-Packard: 1998 [citado 30 novembro 1998].Disponível em World Wide Web: <URL: http://www.hp.com/ovw/whitepaper/wp1.htm>.

[42] Cabletron Systems. **SPECTRUM Family of Management Products** [on line]. Cabletron Systems: 1998 [citado 10 dezembro 1998]. Disponível em World Wide Web: <URL: http://www.cabletron.com/spectrum/>.

[43] Cabletron Systems. **SPECTRUM for Open Systems**[on line]. Cabletron Systems: 1998 [citado 10 dezembro 1998]. Disponível em World Wide Web: <URL: http://www.cabletron.com/spectrum/applications/>.

[44] Cabletron Systems. **SPECTRUM Portable Management Applications - A building-block approach to network management**[on line]. Cabletron Systems: 1998 [citado 10 dezembro 1998]. Disponível em World Wide Web: <URL: http://www.cabletron.com/products/items/SPMA-CORE/>.

[45] Tivoli Systems Inc. **Tivoli Deployment Training Guide**. Austin, EUA: Tivoli Systems Inc., junho/1996.

[46] Tivoli Systems Inc. **Tivoli Management Plataform Training Guide**. Austin, EUA: Tivoli Systems Inc., junho/1996.

[47] Tivoli Systems Inc. **Tivoli Applications Training Guide**. Austin, EUA: Tivoli Systems Inc., junho/1996.

[48] Tivoli Systems Inc. **Tivoli Management Software** [on line]. Tivoli Systems Inc: 1998 [citado 8 dezembro 1998]. Disponível em World Wide Web:<URL: http://tivoli.com/o_products/html/body_products.html>.

[49] Sun Microsystems,Inc.**The SE Performance Toolkit - Release 3.0** [on line]. Sun Microsystems,Inc: 1998 [citado 16 dezembro 1998]. Disponível em World Wide Web:<URL:http://www.sun.com/sun-on-net/performance/se3/>.

[50] Sun Microsystems,Inc. **Performance Toolkit Version 2.5** [on line]. Sun Microsystems,Inc: 1998 [citado 16 dezembro 1998]. Disponível em World Wide Web: <URL: http://www.sun.com/960301/columns/adrian/se2.5.html>.

[51] Sun Microsystems,Inc. **System Performance Monitoring** [on line]. Sun Microsystems,Inc: 1998 [citado 16 dezembro 1998]. Disponível em World Wide Web: <URL: http://www.sun.com/950901/columns/adrian/column1.html>.

[52] Sun Microsystems,Inc. **Advanced Monitoring and Tuning** [on line]. Sun Microsystems,Inc: 1998 [citado 16 dezembro 1998]. Disponível em World Wide Web: <URL: http://www.sun.com/951001/columns/adrian/column2.html>.

[53] MacLawran Group, Inc. **Big Bhother - Monitoring & Notification for Systems and Networks.** [on line]. MacLawran Group, Inc.: 1999 [citado em 10 abril 1999]. Disponível em World Wide Web: <URL: http://www.maclawran.ca/bb-dnld/new-info.html>.

[54] MacLawran Group, Inc. **Big Bhother** [on line]. MacLawran Group, Inc.: 1999 [citado em 10 abril 1999]. Disponível em World Wide Web: <URL: http://maclawran.ca/bb/bb-info.html>.

[55] Computer Associates, Inc. **CA Unicenter TNG** [on line]. Computer Associates, Inc. :1999 [citado em 10 abril 1999]. Disponível em World Wide Web: <URL:http://www.cai.com/products/unicenter/tng-brochure.pdf>.

[56] Freshwater Software, Inc. **SiteScope Software** [on line]. Freshwater Software, Inc: 1999 [citado em 10 abril 1999]. Disponível em World Wide Web: <URL: http://www.freshtech.com/>.

[57] Man Page ps, Sun Microsystems, Inc. 1995.