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

Walter Priesnitz Filho

UMA FERRAMENTA PARA AUXILIAR O DESENVOLVIMENTO DE MODELOS DE CARGA DE SERVIDORES *WEB*

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

Paulo José de Freitas Filho

FLORIANÓPOLIS, AGOSTO DE 2003

UMA FERRAMENTA PARA AUXILIAR O DESENVOLVIMENTO DE MODELOS DE CARGA DE SERVIDORES *WEB*

Walter Priesnitz Filho

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação com área de Concentração em Sistemas de Computação e aprovada em sua forma final pelo programa de Pós-Graduação em Ciência da Computação.

_____

Prof. Fernando A. O. Gauthier, Dr.

Banca examinadora:

_____

Prof. Paulo José de Freitas Filho, Dr. Orientador

_____

Profa. Elizabeth Sueli Specialski, Dra.

_____

Prof. Luiz Fernando Jacinto Maia, Dr.

“Ser gênio é ter pouco mais do que a capacidade de perceber a realidade de um maneira pouco habitual”

William James

A minha família;

E a minha noiva Mariane Camargo.

Agradeço a Deus pelas oportunidades que tive.

A meus Pais pelo apoio incondicional.

As minhas irmãs Clarissa e Camila pela confiança,

A minha noiva pelo amor, compreensão e paciência,

A meu orientador e amigo Paulo Freitas, o tio Freitas, pelo apoio, paciência, compreensão, e bom humor.

Agradeço ao Rivalino por compartilhar de sua experiência.

Aos meus grandes amigos Dione, Eduardo, Fernando Barreto, Prass, Rafael, Renato, Rodrigo que me ajudaram nesse trabalho.

E também ao pessoal da administração da Rede INF Adamo, Juliano, e ao Professor Mariani pela paciência e presteza com que me ajudaram na obtenção de informações.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

SCRIPT

RESUMO

Este trabalho tem por finalidade mostrar uma ferramenta desenvolvida com o intuito de melhorar o processo de caracterização de cargas de trabalho de servidores *web*, facilitando o trabalho dos administradores desses sistemas. A ferramenta desenvolvida é composta de um *script* de coleta de dados que atua em conjunto com o *script apachectl*, no modo *fullstatus*, realizando coletas periódicas de informações do sistema monitorado e as grava em um arquivo de *log*. E de um sistema que analisa esses dados (*logs*) e efetua a conversão dos mesmos em informações e as armazena em uma base de dados (*Interbase*®). Os resultados obtidos desse trabalho foram satisfatórios uma vez que, com o uso dessa ferramenta, houve um aumento da significância dos dados obtidos para a elaboração dos modelos de cargas do sistema, economia de tempo no processo de avaliação de desempenho, e planejamento de capacidade do referido servidor.

Palavras-chave: Avaliação de Desempenho, Monitoração, Servidores *Web*

ABSTRACT

This work has for purpose to show a tool developed with the intention of facilitating the administrators' of web servers Apache® in what it's performance. The developed tool acts together with the *script* apachectl in the way fullstatus that accomplishes periodic collections of the system monitored for subsequent conversion of the data obtained in information stored in a database. That database was elaborated on the database manager system (DBMS) Interbase®. The obtained results of that work were satisfactory once, with the use of that tool, there was an earnings of time in the process of performance evaluation of the referred server and an important aid in the process of capacity planning of those servers.

Keywords: Performance Evaluation, Monitor Systems, Web Servers

1 INTRODUÇÃO

1.1 Considerações Iniciais

Desde o surgimento dos computadores pessoais é cada vez maior o número de pessoas com acesso às novas tecnologias. Isso implica em uma demanda crescente de serviços oferecidos através de sistemas computacionais.

Através das redes de computadores foi possível vencer barreiras geográficas e atingir os mais distantes pontos do planeta. Não demorou muito para que empresários percebessem a importância dos recursos computacionais para a expansão de suas áreas de atuação.

Os negócios apresentam um processo contínuo de crescimento na *WWW* (*World Wide Web*), e torna-se cada vez mais importante para as empresas à quantificação das medidas de performance observada pelos consumidores. (OLSHEFSKI, NIEH & AGRAWAL, 2002).

Com a *WWW* a gama de serviços disponíveis ao usuário aumentou ainda mais. Porém esse crescimento não pode, nem deve, ser desordenado. Deve ser bem assistido para que os recursos empregados sejam capazes de suprir as necessidades, mantendo a qualidade dos serviços prestados, dos clientes e dos provedores dos serviços.

Através da avaliação de desempenho e do planejamento de capacidade é possível acompanhar como um determinado serviço está respondendo as expectativas de provedores e clientes.

1.2 Justificativa

O número de serviços oferecidos através de redes de computadores tem crescido de maneira acentuada nos últimos anos. Destaca-se, nesse crescimento, a utilização cada vez maior dos serviços baseados na *web* sejam leilões virtuais, *webmails*, horóscopo, entre outros.

Também se sabe que o tráfego gerado na *web* e em sítios de comércio eletrônico varia muito a cada dia da semana, e semana a semana. (ALMEIDA, ARLITT & ROLIA, 2002), (MENASCÉ et all, 2000), (CROVELLA & BESTAVROS, 1996), apud (KANT & WON, 1999).

Essa demanda crescente por serviços requer suporte adequado. E para que este suporte seja adequado o conhecimento do ambiente deve ser o mais completo possível.

Este trabalho justifica-se pela importância da boa administração de recursos de tecnologia de informação, através da avaliação de desempenho e do planejamento de capacidade, e pela aplicação dos conceitos metodológicos na elaboração de um modelo para estes ambientes.

1.3 Problema

O problema a ser abordado neste trabalho envolve a elaboração de modelos de carga de trabalho (*workloads*) através do tratamento das informações (de estado e discos) obtidas de servidores *Web* Apache®. Essas informações podem ser obtidas através de *scripts*, que geram arquivos (*logs*), de coletas periódicas de informações do servidor.

Esses arquivos não apresentam nenhuma formatação, ou padronização, o que torna o tratamento dessas informações bastante lento.

O acompanhamento, ao longo do tempo, do comportamento de um servidor diante de determinadas situações torna-se, praticamente, impossível dado à lentidão do processo de pesquisas nesses arquivos.

Esse fato prejudica sobremaneira a confiabilidade e a representativade de modelos de carga elaborados para avaliar o desempenho destes servidores em intervalos de tempo maiores como dias, por exemplo.

1.4 Objetivos

1.4.1 Objetivo Geral

Este estudo tem por finalidade o desenvolvimento de uma ferramenta que auxilie o analista no processo de avaliação de desempenho de servidores *web* especialmente nas etapas que envolvem a caracterização de sua carga de trabalho e no desenvolvimento dos posteriores modelos de carga.

1.4.2 Objetivos Específicos

Para que se obtenha êxito neste trabalho serão dados os seguintes passos:

Elaboração de um *script* de coleta e armazenamento com os dados do servidor em intervalos de tempo predeterminados através do agendador de tarefas do sistema (*cron*);

Desenvolver a ferramenta que faça a análise dos dados extraídos do servidor;

Desenvolvimento da base de dados que irá armazenar as informações de performance do servidor. Através da qual será possível executar pesquisas sobre informações de performance de épocas anteriores e, também, calcular estatísticas de utilização dos recursos do servidor;

Desenvolvimento do módulo de coleta e análise dos dados obtidos dos arquivos (*logs*). Esse módulo irá percorrer os arquivos com os dados brutos e coletar as informações desejadas;

Validação do módulo de obtenção de dados dos arquivos de *log*. A validação se faz necessária para que as informações sejam confiáveis e o processo de análise consistente;

Desenvolvimento do módulo de exportação de dados. Com esse recurso será mais fácil uma integração entre a ferramenta desenvolvida com outras ferramentas como de simulação, por exemplo, no futuro;

1.5 Estrutura do trabalho

No primeiro capítulo (Introdução) é apresentado o tema, que motivou a elaboração deste trabalho, a justificativa e a importância da realização do mesmo. Assim como o objetivo geral, de melhorar as condições de trabalho de quem faz a avaliação de desempenho de servidores *web* Apache®, e os objetivos específicos necessários para a conclusão do trabalho.

No capítulo dois (Revisão de Literatura) são expostas as teorias envolvidas no processo de execução desse trabalho. Visando, desta forma, deixar o leitor mais bem preparado para compreender este estudo.

São expostos os conceitos de avaliação de desempenho e monitoração de sistemas computacionais, que embasaram as características disponibilizadas pela ferramenta.

O planejamento de capacidade, que será auxiliado com a utilização dessa ferramenta.

O capítulo três apresenta a adequação dos conceitos com os aspectos práticos da elaboração do trabalho.

Nele são expostas as teorias de caracterização de cargas de trabalho que auxiliaram a identificar a carga principal sob o sistema em estudo. São descritas as ferramentas utilizadas para a coleta de dados do ambiente (*apachectl*) assim como o módulo do servidor responsável por essa coleta (*modstatus.so*).

No capítulo quatro a ferramenta desenvolvida é apresentada. São explicadas as etapas percorridas para a obtenção dos dados a serem utilizados.

Também e mostrada a linguagem de programação na qual a ferramenta foi desenvolvida e seu funcionamento, justificando a escolha. Assim como o sistema gerenciador de banco de dados (SGBD) utilizado. O funcionamento e a operação são

mostrados para que os leitores possam compreender melhor o funcionamento e possam, também, utilizar a ferramenta.

O capítulo cinco aborda as conclusões tomadas após a conclusão do desenvolvimento desse trabalho. Nele são expostos os pontos de contribuição do trabalho, assim como as dificuldades encontradas e as sugestões para a continuação deste trabalho.

É colocado no Anexo um trecho dos arquivos coletados no ambiente de teste para que os leitores percebam as dificuldades encontradas no processo de coleta de informações dos arquivos de *log*.

Ao final do trabalho são apresentadas as referências bibliográficas dos materiais utilizados na elaboração deste trabalho.

2 REVISÃO DA LITERATURA

Neste capítulo serão explorados os conceitos e a metodologia sobre planejamento de capacidade. Busca-se, desta forma, situar o leitor no contexto atual desse tema.

Inicialmente é abordado o planejamento de capacidade, começando pela conceituação de planejamento de capacidade, objetivos do planejamento, descrição da metodologia, erros mais comuns, e as dificuldades encontradas no processo.

Após os conceitos referentes às redes de computadores também são expostos para que a percepção do leitor sobre o assunto seja facilitada.

Pretende-se, ao final deste capítulo, que o leitor consiga compreender de forma mais significativa à importância do que está sendo proposto pelo presente trabalho.

2.1 Planejamento de Capacidade

A pouco tempo a administração de sistemas resumia-se em corrigir os problemas à medida que os mesmos iam aparecendo, ou seja, era adotado em sistema de gerenciamento reativo. Onde se fazia necessário uma análise em arquivos de *logs* para que fosse encontrada a causa do problema que havia ocorrido no sistema.

Com o passar do tempo tornou-se necessário prever os problemas que iriam ocorrer no sistema em função do aumento da carga de trabalho, implantação de novos serviços, e assim impedir a perda acarretada pelos mesmos.

Surgiu, então, o planejamento de capacidade como alternativa a essa situação. Através dele era possível ter um gerenciamento pró-ativo do sistema.

Segundo GUTJAHR (1999), o gerenciamento de performance consiste no processo de planejar e gerenciar dia-a-dia as cargas de trabalho ou aplicações para encontrar os requisitos de repostas e *throughputs* de maneira efetiva e econômica.

MENASCÉ & ALMEIDA (1998) conceituam o planejamento de capacidade como o processo de prever quando as entradas de um sistema irão saturar o mesmo, e de

determinar a maneira mais economicamente viável de impedir esta saturação o máximo possível. A previsão deve considerar a evolução das cargas de trabalho, assim como as novas aplicações, as já existentes, e os níveis de serviços desejados.

Ainda MENASCÉ & ALMEIDA (1994), afirmam que previsão é uma palavra chave no que se refere a planejamento de capacidade. Pois é impossível fazer o planejamento de capacidade sem que seja possível fazer uma previsão da performance do sistema.

Este processo, o planejamento de capacidade, torna-se indispensável à medida que as empresas necessitam de um gerenciamento de recursos cada vez mais eficiente. Buscando manter os níveis de serviço com taxas de ocupação otimizadas para uma boa relação custo/benefício. Visando, desta forma, manter o cliente satisfeito com os serviços, e com a qualidade do atendimento.

O planejamento de capacidade, segundo MARTINEZ (2001), tem por objetivo principal prever o impacto das atividades dos negócios na estrutura tecnológica, e dar um retorno para os responsáveis pelo negócio em termos de custos financeiros.

2.1.1 Importância do Planejamento de Capacidade

Segundo MENASCÉ & ALMEIDA (1998), o Planejamento de Capacidade é importante nos seguintes aspectos:

2.1.1.1 Insatisfação dos Usuários:

A falta do planejamento de capacidade do Serviço de Informação implicará em uma situação que certamente violará os níveis de serviços desejados. A simples solução de um problema pode ser mais longa do que pode ser tolerado. Onde os usuários enfrentarão um sistema degradado. E, dependendo do tipo de *workload,* esta degradação pode variar desde um pequeno incômodo até sérios problemas organizacionais.

2.1.1.2 Imagem Externa da Companhia:

Em muitas situações, a performance do serviço de informação facilita à

companhia sua percepção por seus usuários. A falta de planejamento pode levar a situações de degradação de performance que serão sentidas diretamente pelos clientes.

2.1.1.3 Decrescimento da Produtividade:

A baixa performance do sistema pode implicar em baixa produtividade. Pode-se considerar o exemplo de um sistema CAD rodando em um mainframe. Gerando efeitos negativos no trabalho como um todo e em metas não alcançadas, prazos,...

2.1.1.4 Contenções Orçamentárias:

Na maioria dos casos, quando a performance atinge seu ponto de degradação, a solução envolve expansão de *hardware* ou a reescrita do *software*, com implicações financeiras que dificilmente irão adequar-se ao orçamento previamente aprovado e aos limites orçamentários.

2.1.1.5 Risco de Perdas Financeiras:

Um sistema degradado pode levar certas companhias a tomar decisões e determinadas ações de modo inapropriado, podendo acarretar em perdas financeiras.

2.1.1.6 Controle de Ambiente de Serviços de Informação:

Em um ambiente degradado as pessoas tendem a buscar soluções para seus problemas. Muitas vezes estas soluções implicam em aquisição de novas máquinas, novos *softwares*. O que pode gerar perdas para a empresa, pois podem existir vários aplicativos e vários tipos diferentes de equipamentos a serem mantidos gerando um custo maior do que *hardware* e *software* padronizado.

2.1.2 Erros no Planejamento de Capacidade

Durante o processo de planejamento de capacidade de um sistema é comum que ocorram erros, seja por inexperiência do responsável ou até mesmo por falta de conhecimento específico sobre o assunto.

Dentre os erros mais comuns no planejamento de capacidade temos:

- O responsável pelo planejamento de capacidade não levar em consideração a previsão de evolução das cargas de trabalho.

- O responsável pelo planejamento não considerar os níveis de serviços desejados.

Sumarizando, o planejamento de capacidade não pode ser feito sem que sejam considerados os níveis de serviço, a evolução das cargas de trabalho, e os modelos de previsão de performance do sistema.

2.1.3 Elaboração dos Modelos de Performance

Previsão de performance é o processo de estimar as medidas de performance de um sistema computacional a partir de um determinado conjunto de parâmetros. Medidas de performance típicas incluem tempo de resposta, *throughput*, utilização do recurso, e tamanho da fila do recurso. (MENASCÉ & ALMEIDA, 1998)

Os parâmetros são divididos em categorias, como segue:

Parâmetros do Sistema: Características do sistema que afetem a performance. Como exemplos pode-se citar os protocolos da rede, a número máximo de conexões suportadas por um servidor *web*, e o número máximo de conexões suportadas por um sistema gerenciador de bancos de dados.

Parâmetros de Recursos: Características intrínsecas dos recursos que afetam a performance. São exemplos de parâmetros de recursos os tempos de pesquisa de um disco, taxas de transferência e de latência, largura de banda, e velocidade da CPU.

Parâmetros de Workload: São derivados da caracterização das cargas de trabalho e são divididos em:

Parâmetros de Intensidade das cargas de trabalho: Provêm as medidas da carga do sistema indicadas pelo número de unidades de trabalho contidas em cada recurso. Exemplos: número de visitas a um servidor *web proxy*, numero de requisições por segundo submetidas a um servidor de arquivos,...

Parâmetros de Demanda de Serviços das cargas de trabalho: Especificam o montante total do tempo de serviço requerido por cada componente básico de cada recurso. Exemplos: tempo de CPU das transações em um servidor de bancos de dados, tempo total de *I/O* em um servidor *web proxy* para requisições de imagens e vídeos utilizados em disciplinas que utilizem educação à distância.

A previsão de performance requer o uso de modelos. Dois tipos de modelos podem ser utilizados: modelos de simulação e modelos analíticos. Ambos devem considerar a concorrência ao acesso e as filas de cada recurso do sistema (CPUs, discos, roteadores, etc). As filas também devem ser consideradas em recursos de *software* (*threads*, portas de protocolos).

2.1.4 Previsão de Performance

Para prever a performance de um sistema, precisa-se estar apto a resolver o modelo de performance que representa este sistema. Os modelos de simulação são feitos através de aplicativos que imitam o comportamento do sistema como transações através dos recursos que compõem este sistema.

Há várias alternativas para elaboração de modelos. Quanto mais elementos forem representados com mais detalhes, maior será a precisão do modelo. A obtenção de dados também irá aumentar.

2.1.5 Validação do Modelo de Performance

Um modelo de performance é dito válido se as métricas de performance (tempos de resposta, utilização de recursos, e *throughputs*) calculadas pelo modelo forem iguais, ou muito próximas, das medidas do sistema real.

Durante a caracterização das cargas de trabalho, as medidas são tomadas para as demandas de serviços, intensidade das cargas de trabalho, e para medidas de performance como tempo de resposta, *throughput*, e utilização dos dispositivos. As mesmas medidas são computadas por médias do modelo de performance.

Se os valores obtidos pelo modelo não forem válidos, o modelo deve ser calibrado.

2.1.6 Análise Custo/Performance

Uma vez que o modelo de performance esteja feito e resolvido e o modelo de custos desenvolvido, as análises podem ser feitas. Os modelos de performance e de custos podem ser usados para avaliar vários cenários e configurações. Para cada cenário, pode-se fazer a previsão de performance de cada componente básico das cargas de trabalho global e os custos associados a cada cenário.

A comparação de vários cenários resultará na elaboração de um plano de configuração, um plano de investimentos, e um plano de pessoal.

O plano de configuração irá especificar quais *upgrades* deverão ser feitos nas plataformas de *hardware* e *software* existentes, quais mudanças deverão ocorrer na topologia da rede e qual arquitetura de sistema deverá ser empreendida.

O plano de investimentos especifica o a que tempo que cada *upgrade* deve ser feito. O plano pessoal determina que mudanças no tamanho e na estrutura da equipe de suporte para acompanhar as mudanças do sistema.

2.1.7 Visão Geral da Metodologia de Planejamento de Capacidade

A metodologia do Planejamento de Capacidade baseia-se em três modelos: o modelo de *workload*, o modelo de performance, e o modelo de custos.

O modelo de *workload* trata das demandas dos recursos e das características da intensidade das cargas de trabalho para cada componente das cargas de trabalho global em um intervalo de tempo representativo.

O modelo de performance é usado para prever a performance do sistema como uma função da descrição do sistema e dos parâmetros de *workload*. As saídas do modelo de performance incluem tempos de resposta, *throughputs*, taxa de utilização dos

recursos do sistema, e o tamanho das filas. Estas métricas são comparadas com os acordos de níveis de serviços para determinar se a capacidade do sistema está adequada.

O modelo de custos contabiliza os custos com software, hardware, telecomunicações e suporte.

2.1.8 Compreensão do Sistema e de suas Necessidades

A fase inicial da metodologia consiste em aprender qual tipo de hardware (clientes e servidores), software (sistemas operacionais, *middleware*, e aplicações), conectividade da rede, e protocolos estão presentes no ambiente. Também envolve a identificação dos períodos de pico, estruturas de gerenciamento, e acordos de níveis de serviços. Esta informação é obtida de várias formas: encontros de grupos de usuários, auditorias, questionários, arquivos do *help desk*, documentos de planejamento, entrevistas, e outras técnicas de obtenção de informações.(MENASCÉ & ALMEIDA, 1998).

2.1.9 Caracterização das Cargas de Trabalho

Cargas de trabalho são tipos únicos de trabalhos sobre o sistema de informação. Usualmente um aplicativo, ou um grupo de usuários executando um trabalho similar. (GUTJAHR, 1999).

A caracterização das cargas de trabalho consiste na análise dos *workloads* em unidades mensuráveis de trabalho (transações, *jobs* em *batch*) para que seja possível entender a utilização corrente dos recursos do sistema e projeções futuras.

Para MENASCÉ & ALMEIDA (1998) a caracterização das cargas de trabalho consiste em descrever precisamente a carga global em função dos componentes principais do sistema. Cada componente das cargas de trabalho é decomposto em componentes básicos. Os componentes básicos são então caracterizados pela intensidade das cargas de trabalho e pela demanda de serviço em cada recurso.

2.1.10 Modelos de Previsão de Performance

Um aspecto importante do gerenciamento de capacidade envolve a previsão de até

quando o sistema irá prover métricas de performance que atinjam os níveis de serviços desejáveis ou aceitáveis.

2.1.11 Desenvolvimento do Modelo de Custos

A metodologia do planejamento de capacidade requer a identificação das maiores fontes de custo assim como a determinação de como estes custos irão variar com o tamanho e a arquitetura do sistema. Os custos são caracterizados em iniciais e operacionais.

Os custos iniciais são aqueles decorrentes da instalação do sistema, enquanto que os custos operacionais são os gastos anuais decorrentes da manutenção e atualização *hardware* e *software* do sistema, para prevenir que o sistema fique obsoleto, com a performance degradada, e com vulnerabilidades na segurança. Aplicam-se ao *hardware*, *software* e infraestrutura.

Custos operacionais estão relacionados à manutenção e atualização de *hardware* e *software*, custos pessoais, treinamento, serviços de telecomunicações, e consultorias.

2.2 Avaliação de Desempenho

Os sistemas cliente/servidor são sistemas de missão crítica no dia a dia das empresas. Medir a performance de ambientes baseados em redes de computadores é o processo chave na obtenção da satisfação de clientes e na obtenção do sucesso da empresa. (MENASCÉ & ALMEIDA, 1998).

A representatividade de um modelo de performance depende da qualidade dos dados que são usados como entradas. Antes de iniciar o processo de coleta de dados deve-se observar quais os dados de performance devem ser considerados e quais a ferramentas disponíveis para monitorar os tempos de resposta e a utilização de recursos.

Os dados coletados de um sistema são utilizados, basicamente, para detecção de problemas operacionais, ajustes de funcionamento e para o planejamento de capacidade. Para descobrir o que está acontecendo com o sistema os administradores recorrem aos

monitores.

Essas ferramentas coletam dados do sistema continuamente e exibem os valores que podem indicar problemas em potencial como utilização de CPU, atividades de paginação, utilização de memória, etc. Esses dados auxiliam na visualização da situação global do sistema e verificar se algum segmento do sistema está funcionando de forma anômala.

Os ajustes no funcionamento do sistema podem ser feitos a partir da análise ao longo do tempo dos dados obtidos via monitoração para que situações já ocorridas não venham a prejudicar o sistema novamente.

O planejamento de capacidade faz uso dos dados coletados para a elaboração de modelos de performance que irão prever o funcionamento do sistema com novas configurações, o prever futuros problemas decorrentes do crescimento natural do mesmo.

Segundo MENASCÉ & ALMEIDA (1998), em qualquer processo de medição de performance é importante entender o que está sendo medido e o quão precisos e confiáveis são os números que estão sendo obtidos. Portanto, é imprescindível conhecer as ferramentas de medição, suas qualidades e suas limitações. As técnicas utilizadas para a coleta de dados podem ser divididas em duas categorias: baseados em eventos e baseados em amostragem.

2.2.1 Monitores

Monitores são ferramentas utilizadas para observar os níveis de atividade de um sistema (uma rede, um servidor, etc.). A principal função de um monitor é coletar dados sobre o funcionamento do sistema. Os monitores não devem interferir no funcionamento do sistema sob observação. Ou seja, o monitor não deve degradar a performance do sistema monitorado. (MENASCÉ & ALMEIDA, 1998).

Os monitores normalmente permitem observar a performance de sistemas, coletam estatísticas, analisam os dados e apresentam os resultados. Alguns identificam

problemas e sugerem soluções. Podem ser usados para avaliar o desempenho de sistemas já existentes. (JAIN, 1991).

Os monitores pode ser classificados quanto ao modo de operação (eventos e amostragem) e ao tipo:

2.2.2 Baseados em Eventos

Uma ferramenta que efetua a coleta de dados de performance baseada em eventos vai coletar as informações sempre que houver uma alteração no estado do sistema. Como, por exemplo, a chegada de uma nova requisição *HTTP*.

Se a taxa de eventos a serem monitorados for muito alta sobrecarga no sistema torna-se alta prejudicando a avaliação do sistema monitorado.

2.2.3 Baseados em Amostragem

A coleta de dados classificada como baseada em amostragem vai ocorrer em intervalos determinados de tempo. A ferramenta toma o tempo inicial da monitoração e coleta os dados a partir daquele momento.

A sobrecarga causada por uma ferramenta dessas depende de dois fatores: do número de variáveis medidas e do intervalo de tempo entre cada coleta.

2.2.4 Monitores de *Software*

Monitores de Software (MS) são conjuntos de instruções em um sistema para coletar informações de estado e eventos que ocorrem nesse sistema. Essas rotinas coletam dados de performance sobre a execução de um, ou mais, programas e sobre os componentes de *hardware*.

Esses monitores podem armazenar qualquer informação que esteja disponível ao sistema operacional. Essa característica aliada a flexibilidade de aumentar, ou reduzir, a quantidade de dados de performance a ser analisada tornam os monitores de *software* uma ferramenta poderosa na análise de sistemas computacionais. (MENASCÉ &

ALMEIDA, 1998).

Contudo ao rodar as rotinas de coleta de dados do ambiente monitorado são consumidos recursos do sistema observado, o que pode interferir significativamente no funcionamento do ambiente.

As vantagens de um MS são sua facilidade de instalação e de uso. E suas desvantagens são a dependência do sistema operacional e a sobrecarga do sistema que pode ser introduzida sobre o sistema observado.

Os dados obtidos via MS pode ser agrupados em duas categorias conforme o escopo da informação:

2.2.4.1 Informações em Nível de Sistema

Exibe informações sobre a utilização dos recursos do sistema de modo geral. Como utilização CPU, memória, etc.

2.2.4.2 Informações em Nível de Aplicativos

Exibem informações relacionadas à execução do aplicativo. Tempo de execução, consumo de CPU (pelo aplicativo), etc. Os sistemas de contabilidade constituem a maior fonte de informações sobre a execução de aplicativos.

Existem algumas classes de ferramentas que podem ser consideradas monitores de *software*. São elas:

Sistemas de Contabilidade: Esses sistemas coletam informações sobre a utilização dos sistemas pelos usuários. Essas ferramentas, geralmente, fazem parte dos sistemas operacionais (SOs) multi-usuários.

Esses sistemas permitem determinar qual usuário executou qual aplicativo e medir a utilização de CPU por esses aplicativos e quanto, em recursos do sistema, foi utilizado pelo usuário.

Analisadores de Programas: São ferramentas desenvolvidas para coletar

informações sobre a execução de programas específicos. Também podem ser utilizados como ferramentas de otimização mostrando as partes do aplicativo que consomem mais recursos do sistema.

Logs: Os arquivos de *log* guardam informações sobre as requisições recebidas por um sistema, as repostas dadas e a origem dessas requisições. No caso dos servidores *web* os arquivos com os *logs* são, geralmente, em formato texto contendo informações sobre as atividades do servidor.

Existem várias ferramentas de monitoração (gratuitas ou não) disponíveis na rede, para analisar *logs* de servidores Web. Algumas das métricas usadas para formalizar e sintetizar os dados coletados são: número de acessos ao sítio por dia, número de paginas acessadas por dia, tempo médio de visita, tempo médio decorrido entre visitas, etc.

Exemplo: A tabela abaixo (Tabela 2-1) apresenta um log de acesso a um sítio *Web* coletado durante um curto espaço de tempo. A partir deste, deseja-se coletar medidas de desempenho, tais como taxa média de chegadas e tamanho médio dos documentos requeridos.

Tabela 2-1 - Log de acesso a o Web site

Host name	[dd/mm/yy:hh:mm:ss tz]	Request	status	bytes
perf.xyz.com - -	[24/Jan/19xx:13:41:41 –0400]	“Get i.html HTTP/1.0”	200	3185
perf.xyz.com - -	[24/Jan/19xx:13:41:41 –0400]	“Get 1.gif HTTP/1.0”	200	1210
h0.south.com - -	[24/Jan/19xx:13:43:13 –0400]	“Get i.html HTTP/1.0”	200	3185
h0.south.com - -	[24/Jan/19xx:13:43:14 –0400]	“Get 2.gif HTTP/1.0”	200	2555
h0.south.com - -	[24/Jan/19xx:13:43:15 –0400]	“Get 3.gif HTTP/1.0”	200	36403
h0.south.com - -	[24/Jan/19xx:13:43:17 –0400]	“Get 4.gif HTTP/1.0”	200	441
cs.uni.edu - -	[24/Jan/19xx:13:46:45 –0400]	“Get i.html HTTP/1.0”	200	3185
cs.uni.edu - -	[24/Jan/19xx:13:46:45 –0400]	“Get 2.gif HTTP/1.0”	200	2555
cs.uni.edu - -	[24/Jan/19xx:13:46:47 –0400]	“Get 3.gif HTTP/1.0”	200	36403
cs.uni.edu - -	[24/Jan/19xx:13:46:50 –0400]	“Get 4.gif HTTP/1.0”	200	98995
Sys1.world.com - -	[24/Jan/19xx:13:48:29 –0400]	“HEAD index.html”	400	-

Fonte: JAIN 1998.

Na primeira linha, por exemplo, o *log* indica que a requisição originou-se de perf.xyz.com, as 13:41:41 (–0400: zona leste de tempo) no dia 24/Jan/19xx. O documento requisitado foi uma página índice. O código 200 indica que a resposta do servidor foi um sucesso. O código 400, por sua vez, indica requisição mal sucedida devido a erro no cliente. O tamanho do arquivo transferido foi de 3185 bytes.

Como se observa, apesar destes *logs* ajudarem na caracterização da carga de trabalho, eles não apresentam informações diretas sobre parâmetros necessários aos modelos de desempenho. Por exemplo, não existe informação para o tempo necessário a transferência de um documento. Apesar de tudo, algumas informações podem ser deduzidas.

2.2.5 Monitores de Hardware

Monitores de Hardware (MH) são ferramentas de medição que coletam dados de um sistema a partir da detecção de sinais predefinidos. O MH analisa o estado do

sistema monitorado via dispositivos eletrônicos conectados ao circuito e armazena essas medições. (MENASCÉ & ALMEIDA, 1998).

Existem algumas vantagens na utilização de MH:

Baixa sobrecarga do sistema: Por serem dispositivos externos ao sistema monitorado, não consomem recursos do mesmo;

Portabilidade: É possível mudar um MH de um sistema para outro, desde que o monitor não seja dependente do SO do sistema monitorado.

Porém, existe a desvantagem na utilização de MH no que se refere ao consumo de recursos do sistema por *softwares*.

Segundo JAIN (1998), Monitores de Hardware (MH) consistem em equipamentos separados que são conectados ao sistema monitorado. Geralmente possuem baixa sobrecarga do sistema, não consumindo recursos do sistema monitorado. Os MH atuais são programáveis, possuem seu próprio processador, memória e dispositivos de I/O. Não são recomendáveis para a obtenção de dados específicos, relacionados com as aplicações.

2.2.6 MS x MH

Dado um problema de monitoração, como saber qual tipo de monitor é o mais adequado? Algumas considerações devem ser feitas para poder escolher adequadamente o tipo de monitor. São elas:

2.2.6.1 Domínio

Os MS podem ser usados para monitorar níveis altos de informação: Uso de recursos, acessos a bancos de dados, atividade do sistema operacional, utilização da rede, tempo de execução de procedimentos, etc.

Os MH podem ser usados para capturar informações de baixo nível, como freqüência de instruções executadas, sinais elétricos nos barramentos e outros eventos

de hardware.

2.2.6.2 Taxa de Entrada

Os MS consomem tempo de CPU para serem executados, e não podem monitorar eventos muito freqüentemente. É limitado pelo tempo de execução do procedimento de observação.

Os MH podem monitorar eventos muito rapidamente, limitado apenas pela velocidade do monitor.

2.2.6.3 Total de Informação Armazenada

Se a quantidade de informação armazenada por um MS for elevada, o desempenho é afetado. Pode ser usada compressão de dados, com overhead adicional.

Os MH possuem sua própria área de armazenamento. Não interfere no sistema monitorado.

2.2.6.4 Concorrência

Os MS são programas, e naturalmente seqüenciais. Se houver apenas um processador, então apenas um evento é observado num dado instante.

Os MH podem possuir vários conectores e monitorar vários eventos simultaneamente.

2.2.6.5 Portabilidade

Os MS são desenvolvidos numa linguagem de programação e executam apenas numa arquitetura específica.

Os MH podem ser conectados a vários sistemas diferentes. São geralmente portáveis.

2.2.6.6 Monitoração de Eventos Anormais

Os MS normalmente não funcionam quando ocorre uma falha.

Os MH não são influenciados e funcionam mesmo com falha no sistema monitorado

2.3 Redes de Computadores

Segundo SOARES (1995), uma rede de computadores é formada por um conjunto de módulos processadores capazes de trocar informações e compartilhar recursos, interligados por um sistema de comunicação.

Nos seus primeiros vinte anos de existência os sistemas computacionais eram acondicionados, geralmente, em uma grande sala. Empresas de médio porte e universidades contavam com um ou dois computadores. Enquanto as grandes instituições contavam com algumas dezenas.

A fusão dos computadores e das comunicações teve uma forte influência na maneira como os sistemas computacionais eram organizados. O modelo de um computador atendendo a todas as necessidades da organização foi substituído pelas redes de computadores, onde os trabalhos são realizados por vários computadores interconectados. (TANNENBAUM, 1997)

Dentre os princípios de um projeto de redes está o de estruturação da rede em camadas hierárquicas, onde cada camada é construída utilizando-se das funções e dos serviços oferecidos pelas camadas inferiores. As camadas devem ser pensadas como um programas ou processos que se comunicam com os processos correspondentes na máquina com a qual o diálogo está sendo feito. (SOARES, 1995).

2.3.1 Modelo de Referência TCP/IP

Segundo SOARES (1995) a arquitetura internet *TCP/IP* teve seu desenvolvimento patrocinado pela *DARPA* (*Defense Advanced Research Projects Agency*). Esta arquitetura é baseada, basicamente, em um serviço de transporte orientado a conexão, tendo o *TCP* (*Transmission Control Protocol*), e um serviço não orientado a conexão,

não confiável, tendo o protocolo *IP* (*Internet Protocol*) como responsável.

2.3.1.1 A Camada Host/Rede

Não existe nada especificado para esta camada no MR-*TCP/IP*, exceto que ela deve conectar-se com a rede utilizando um protocolo que possibilite o envio de pacotes *IP*.

2.3.1.2 A Camada Inter-Redes

A função da camada inter-redes é possibilitar aos *hosts*, independente da rede onde estejam, enviarem pacotes e garantir que estes pacotes sejam transmitidos independentemente do destino. O roteamento e o controle de congestionamentos também são funções da camada inter-redes.

Nesta camada é definido um formato padrão de pacote, e o protocolo que atua nesta camada é o *IP*.

2.3.1.3 A Camada de Transporte

Esta camada tem como função fazer com que as entidades pares (peer entity) dos hosts de origem e destino mantenham uma conversação. Os dois protocolos que atuam nesta camada são o *TCP* e o *UDP* (*User Datagram Protocol*), onde o *TCP* oferece serviços orientados a conexão, e o *UDP* oferece serviços não orientados a conexão.

2.3.1.4 A Camada de Aplicação

Nesta camada estão presentes os protocolos de alto nível. Dentre eles estão:

A *World Wide Web* (*WWW*) é um repositório de informações em larga escala que os usuários podem procurar usando um programa aplicativo interativo chamado navegador (*browser*). A maioria dos navegadores possui uma interface de apontar e clicar. As informações exibidas incluem texto, imagens, sons,... Além disso, algumas das informações exibidas são destacadas para indicar que um item é selecionável, indicando um novo documento relacionado. (COMER, 2001).

O *FTP* (*File Transfer Protocol*) está entre os protocolos de aplicativo mais antigos e ainda em uso na *internet*. É um dos aplicativos mais pesadamente usados. No início da história da *internet*, datagramas carregando arquivos eram responsáveis por aproximadamente um terço de todo o tráfego da rede. (COMER, 2001)

Ele permite a transferência de arquivos e inclui um mecanismo que permite que os arquivos tenham propriedades e restrições de acesso. Por esconder os detalhes dos sistemas de computadores individuais, o *FTP* acomoda a heterogeneidade (pode ser usado para transferir uma cópia de um arquivo entre um par arbitrário de computadores). (SOARES, 1995)

O *DNS* (*Domain Name System*) é um esquema de atribuição de nomes hierárquico, baseado em domínios. É bastante utilizado para mapear nomes do *hosts* e destinos de mensagens de correio eletrônico em endereços *IP*.

Para que um nome seja mapeado em um endereço *IP*, um aplicativo chama um procedimento de biblioteca denominado resolvedor (*resolver*) e passa seu nome para ele como parâmetro. O resolvedor envia um pacote *UDP* para um servidor *DNS* local que procura o nome e retorna o *IP* para o resolvedor. Que, por sua vez, retorna o endereço *IP* para o aplicativo que fez a solicitação. Tendo conhecimento do endereço este aplicativo pode estabelecer uma conexão *TCP* com o destino.

Para a transferência de uma mensagem de correio eletrônico, *e-mail*, uma conexão *TCP* com um servidor *SMTP* (*Simple Mail Transfer Protocol*) é feita. Uma vez estabelecida essa conexão o protocolo é seguido permitindo ao usuário se identificar, especificar um receptor e enviar a mensagem. Embora o processo pareça simples questões como a entrega confiável e a existência do destinatário devem ser observadas. (COMER, 2001).

O *POP* (*Post Office Protocol*) é um protocolo bastante simples utilizado na obtenção das mensagens contidas em uma caixa postal. O protocolo dispõe de comandos para que o usuário estabeleça *logins* e *logouts*, além de comandos para obter e eliminar mensagens. O objetivo do *POP* é que o usuário possa obter suas mensagens

em uma caixa postal remota e armazena-las em sua máquina local para leitura.

2.4 Servidor *Web*

A principal função de um *Web Server* é transformar uma *URL* (*Universal Resource Locator*) em um nome de arquivo e então mandar esse arquivo, ou o resultado da execução desse arquivo, de volta a quem o requisitou. (LAUREN & LAUREN, 1999).

O servidor Apache é um servidor HTTP, baseado no servidor *web* que foi desenvolvido pela NCSA (*National Center for Supercomputing Applications*), poderoso, flexível com suporte a HTTP/1.1 (*RFC2616*). É altamente configurável e extensível através de módulos externos, pode ser customizado através da escrita de novos módulos utilizando o módulo Apache API, disponibiliza o código fonte, roda em Windows 95-8/NT/2000/XP, NetWare 5.x e superiores, OS/2 e na maioria das versões do UNIX.

O servidor é mantido por um grupo de programadores que trabalha gratuitamente em melhorias e correções de falhas encontradas no servidor. (APACHE, 2003).

O Apache é um programa (aplicativo) que roda em sistemas operacionais (SOs) multitarefa. Normalmente o aplicativo é executado em *background*. Cada instância inicializada do aplicativo atua sobre um determinado sítio, que nada mais é do que um diretório contendo os arquivos de exibição e configuração do sítio.

Além desses arquivos existem ainda quatro subdiretórios:

conf: contém os arquivos de configuração do sítio dos quais o mais importante é o *httpd.conf* que contém as diretivas de funcionamento do servidor;

htdocs: contém os arquivos *html, php, jsp, asp,* que serão disponibilizados aos clientes. Qualquer subdiretório de *htdocs* será acessível a qualquer pessoa na Internet;

log: contém os *logs*, tanto de acesso quanto os de erros;

cgi-bin: nesse subdiretório estão contidos os *scripts,* ou aplicativos, que podem ser executados pelos clientes.

Quando não está processando uma requisição o Apache não faz nada além de escutar uma determinada porta em um determinado endereço IP (*Internet Protocol*). Quando chega uma requisição em uma porta válida é feita uma análise dos cabeçalhos enviados. Então são aplicadas as regras encontradas no arquivo de configuração (*httpd.conf*) e tomadas as medidas apropriadas.

2.4.1 Como o Apache Utiliza o TCP/IP

Um servidor pode ser analisado, olhando-se de fora, como uma caixa onde existe um computador, o(s) *software*(s) e a conexão com o mundo externo. O meio pelo qual pode-se obter essa conexão é conhecido por interface, e essa interface é conhecida no mundo pelo seu endereço IP.

Um servidor pode prover vários serviços em uma mesma interface como: *WWW*, POP, SMTP, DNS, etc. E decide como atender as requisições porque o endereço IP que consta em cada requisição é acompanhado de um número de dois *bytes* que representam a porta do serviço desejado.

O administrador de um sítio, o *webmaster*, pode decidir qual serviço vai "escutar" em qual porta. Porém deve-se observar que qualquer alteração deve ser passada aos clientes para que os mesmos possam configurar seus softwares clientes e continuar utilizando o serviço. A porta na qual o Apache "escuta" é a porta 80, utilizada como padrão para os serviços *WWW*.

Com a utilização de endereços IP virtuais é possível configurar mais de um domínio, como *www.walter.filho.nom.br*, em um mesmo servidor. Para isso são alteradas as configurações no *httpd.conf* na seção *virtual domains*.

2.4.2 Requisições dos Clientes

Quando o cliente envia uma requisição ao servidor utilizando o *Browser*, este analisa a requisição e verifica a presença do "*http*" e deduz que irá

utilizar esse protocolo. Então verifica qual o domínio que irá buscar no servidor *DNS* para a obtenção do endereço IP.

De posse desses dados o cliente cria uma conexão TCP na porta 80 desse endereço e envia a seguinte mensagem: "*GET index.html HTTP/1.0 <CR><LF><CR><LF>*"

As *tags* <CR> (*Carriage Return*) e <LF> (*Line Feeds*) são as responsáveis pela separação do *header* HTML de seu corpo. Se a requisição for do tipo "*POST*" então ela deverá ser seguida de dados. Com isso o servidor envia a resposta ao cliente e fecha a conexão.

3 METODOLOGIA

Este capítulo descreve como a metodologia será aplicada para que os objetivos do trabalho sejam alcançados. Também são enfocados os tópicos específicos da metodologia de planejamento de capacidade que se enquadram com a proposta deste trabalho.

3.1 Análise das cargas de trabalho Global

Quando a intensidade das cargas de trabalho é alta, muitos conjuntos de medidas das cargas de trabalho podem ser obtidos. Ao trabalhar com estes conjuntos raramente será obtido êxito, especialmente se os resultados das cargas de trabalho forem utilizados para previsão de performance através de modelos analíticos. Cada conjunto de valores deve ser substituído por uma representação mais compacta, uma para cada componente básico. Esta representação é chamada de modelo das cargas de trabalho, o produto final do processo de caracterização das cargas de trabalho. (MENASCÉ & ALMEIDA, 1998).

3.2 Previsão de *Workload*

A previsão das cargas de trabalho é o processo de prever como a carga do sistema vai variar no futuro. Através desse processo podem-se responder questões do tipo: “Como irá variar o número de clientes que visitam o sítio nos próximos seis meses?”, “Como irá variar no tempo o número de compradores de livros de computação?”. Responder essas questões envolve a avaliação das direções das cargas de trabalho, se dados históricos estão disponíveis, e/ou analisando os negócios ou planos estratégicos da organização e então mapear estes planos de negócios em mudanças no nos processos dos negócios. (MENASCÉ & ALMEIDA, 2000).

Durante a previsão de cargas de trabalho, os componentes básicos das cargas de trabalho são associados aos processos do negócio então as mudanças na intensidade das cargas de trabalho desses componentes podem ser derivadas a partir dos processos do negócio e dos planos estratégicos.

3.3 Apresentação dos Dados

Segundo MENASCÉ & ALMEIDA (1998), em situações ideais, os monitores de performance e sistemas de contabilização são usados para determinar os valores dos parâmetros para cada componente básico.

Em muitos casos, é possível detectar um número limitado de aplicativos que são responsáveis por uma porção considerável da utilização dos recursos. Estas medições devem ser feitas separadamente nos clientes e nos servidores através de rotinas representando o comportamento padrão dos usuários no uso de cada aplicativo.

Tabela 3-1 Volume de dados por Serviço

Serviços	Volume de Dados (MB)
HTTP	701,6
OUTROS	375
FTP	213,7
Mail	197,2
SSH	43,5
DNS	13,9
NETBIOS	1,2
NFS	0,189
DHCP/BOOTP	0,137

Fonte: Coleta realizada pelo *software* Ntop no ambiente de rede do INE.

3.3.1 Validação do Modelo de Workload

Ao elaborar um modelo, são feitas abstrações da realidade sendo modelada visando simplicidade, maior facilidade para obtenção de dados e uso, e eficiência computacional do processo modelado.

No processo de validação do modelo de *workload* se os resultados obtidos têm de 10-30% de margem de erro, o modelo é considerado válido. Por outro lado, o modelo

deve ser refinado para que seja obtida maior precisão para representar o *workload* atual.

3.4 Definição do Foco

Depois de feito o levantamento das cargas de trabalho do sistema foi definido que o foco do trabalho seriam as transações realizadas pelo servidor Apache, responsável pelo maior volume de dados transmitidos pela rede.

Essa primeira medição foi feita com a utilização da ferramenta *ntop* (*Network Top)* que é uma ferramenta que mostra a utilização de uma rede de computador, similar ao comando *top* do Unix.

O ntop é baseado na biblioteca de captura de dados *libpcap* e foi escrito de modo a poder rodar em, praticamente, todas as plataformas UNIX e Win32. Essa ferramenta foi utilizada por separar por protocolos o volume de dados que trafegam na rede. (DERI, 2002).

3.5 modstatus.so

O módulo de estado do Apache (*modstatus*.so) permite ao administrador do servidor verificar como o servidor está trabalhando. Uma página HTML é gerada e contendo as estatísticas do servidor de forma simples. Se necessário essa página pode ser atualizada automaticamente, desde que o *browser* suporte o a atualização automática (*refresh*).

Os dados de estado obtidos são:

O número de processos filhos atendendo as requisições;

O número de processos filhos disponíveis;

O estado de cada processo filho, o número de requisições que o processo filho atendeu e o número de *bytes* transferidos por esse processo;

O número total de acessos e o volume de dados servidos;

A hora em que o servidor foi iniciado, ou reiniciado, e a hora que ele começou a rodar;

Médias do número de requisições atendidas por segundo, do número de bytes transferidos por segundo, e a média de bytes por requisição;

A porcentagem de ocupação de CPU de cada processo filho e do Apache;

E as requisições sendo processadas no momento.

(MODSTATUS, 2003)

3.6 Apachectl

O *apachectl* é um *front-end* para o servidor APACHE. Ele foi desenvolvido com o intuito de auxiliar o administrador no controle do servidor.

O *script* pode operar em dois modos. Ele pode atuar como um simples *front-end* ao servidor para definir qualquer variável de ambiente e iniciar o servidor passando argumentos. E, também, pode atuar simplesmente iniciando, parando, reiniciando o servidor convertendo as instruções recebidas em comandos ao servidor HTTP. (APACHECTL, 2003).

Se o ambiente onde está instalado o apache não for o padrão de instalação, então o *script* deve ser editado para que os caminhos utilizados pelo mesmo sejam corrigidos possibilitando o bom funcionamento do *script*.

Pode ser executado com as seguintes opções:

start

Inicia o servidor. Retorna uma mensagem de erro se o servidor já tiver sido iniciado. É equivalente a *apachectl -k start*.

stop

Para o servidor. É equivalente a *apachectl -k stop*.

restart

Reinicia o servidor. Se o servidor não estiver rodando ele o colocar a rodar. Este comando verifica automaticamente os arquivos de configuração antes de reiniciar para que o servidor não inicie e pare. É equivalente a *apachectl -k restart*.

fullstatus

Exibe um relatório completo do *estado* obtido do módulo *mod_status*. Para isso é necessário que o módulo esteja habilitado no servidor e que tenha um *browser* modo texto, como o *lynx*, disponível no sistema. A URL utilizada para acessar o relatório de *estado* pode ser definida editando a variável *STATUSURL* no *script*.

status

Exibe um breve relatório com os dados do *estado* do servidor. Similar à opção *fullstatus*, exceto por não listar as requisições que estão sendo processadas no momento da observação.

gracefull

Reinicia o servidor. Se o mesmo não estiver rodando é iniciado. Difere do *restart* porque as conexões abertas com o servidor não são abortadas. Os arquivos de *log* do servidor não são fechados imediatamente. Este comando verifica os arquivos de configuração antes de iniciar o *restart* para ter certeza de que não ocorrerão falahas após a inicilização. É equivalente a *apachectl -k graceful*.

configtest

Executa um teste de sintaxe nos arquivos de configuração. Ele verifica os arquivos de configuração e exibe uma mensagem informando que a sintaxe está correta. Senão,

exibe uma mensagem detalhando o erro encontrado. É equivalente a *apachectl -t*.

4 A FERRAMENTA

Neste capítulo serão explicados os procedimentos realizados para a elaboração da ferramenta. Desde o processo de elaboração dos *scripts* para coleta dos dados do servidor utilizado como ambiente de estudos, a linguagem de programação utilizada, a base de dados onde as informações são armazenadas, etc.

4.1 A Coleta dos Dados

Inicialmente foi necessário o desenvolvimento de um *script shell* que realizasse a coleta dos dados de estado do Apache através do *script apachectl* e da utilização de disco no servidor.

O *apachectl* coleta dados do estado do servidor e das requisições que estão sendo tratadas no momento da observação. O *script* (abaixo) foi utilizado em intervalos de um minuto rodando no *cron* da máquina (*Euryale*), que hospeda o serviço.

```
#!/bin/bash
# diretorio onde os arquivos de log serão armazenados
cd ~walterp
# define a data a ser usada nos nomes dos arquivos
hoje=$(date +%d%m%y)
# coleta os dados do apache
apachectl fullstatus >> log_apache_$hoje.log
# coleta os dados de disco
iostat >> log_iostat_$hoje.log
```

Script 1- Script de coleta de dados do Servidor

Através desse *script* os dados de estado, e de utilização de disco, são coletados e armazenados em um arquivo cujo nome será, por exemplo *log_apache_27_02_2003.log*. Nesse arquivo estarão contidas várias amostras de coletas de informações do servidor realizadas no dia 27/02/2003.

O *apachectl* é executado, nesse *script*, no modo *fullstatus* para que os dados das requisições sendo processadas também sejam coletados. É importante notar a relevância

da coleta das informações das requisições, pois elas auxiliam na caracterização da carga de trabalho do servidor. A figura abaixo (Figura 4-1) ilustra o processo de coleta e tratamento das informações.

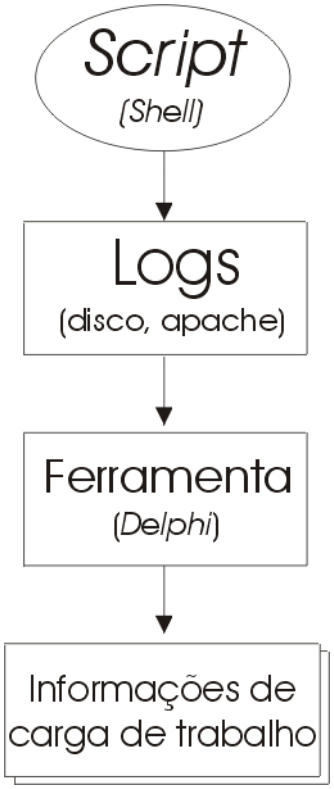

Figura 4-1 - Processo de tratamento de informações

Os dados coletados em cada amostra para análise da ferramenta seguem o formato dos arquivos em ANEXO. A partir desses dados são extraídas as seguintes informações (do Apache):

Data: O dia em que a amostra foi feita;

Hora: A hora em que a amostra foi feita;

Acessos: O total de acessos feitos até o momento da coleta da amostra;

Requisições Correntes: Total de requisições sendo processadas no momento da coleta da amostra;

Servidores Disponíveis: Servidores virtuais que não estão sendo usados pelas requisições;

Tráfego: Tráfego total em MegaBytes desde a inicialização do servidor;

CPU: Porcentagem do total da utilização de CPU pelos servidores;

Requisições/Segundo: Média de requisições feitas por segundo;

KiloBytes/Segundo: Média de KiloBytes transferidos por segundo;

KiloBytes/Requisição: Média de KiloBytes transferidos por requisição.

Dados coletados de cada requisição:

Servidores: Número do processo (servidor) filho;

PID: Identificação do processo no Sistema Operacional;

ACC: Número de acessos à conexão, ao processo filho e ao *slot*;

M: Modo de operação;

CPU: Quantidade de segundos de utilização da CPU;

SS: Quantidade de tempo, em segundos, desde o começo da requisição mais recente;

Requisição: Quantidade de tempo em milésimos de segundos necessário para processar a requisição mais recente;

VHost: *Virtual Host* acessado pela requisição;

Tipo: Tipo de requisição feita (*Get / Post / Head*);

Versão HTTP: Versão do protocolo utilizada;

CONN: KiloBytes transferidos pela conexão;

Child: MegaBytes transferidos pelo processo filho;

Slot: Total de megabytes transferidos pelo *slot*.

E os dados da utilização de disco:

%tm_act: porcentagem de tempo em que o disco (físico) estava ativo;

kbps: total de dados transferidos (leitura/escrita) para o disco em KB/s;

tps: número de transferências por segundo que foram submetidas ao disco (físico);

Kb_read: número total de KB lidos;

Kb_wrtn: número total de KB escritos;

Através dessas informações é possível obter uma visão da situação operacional do servidor sob estudo. E, aplicando-se os conceitos de Teoria de Filas ou Simulação Computacional, sobre esses dados pode-se fazer o planejamento de capacidade de forma mais ágil.

Esse *script* foi executado na máquina *Euryale* entre os dias 26/8/2002 e 14/9/2002 durante as vinte e quatro horas do dia em intervalos de um minuto. Os arquivos de *log* foram armazenados à medida que eram criados conforme o dia do mês a que se referiam.

De posse desses *logs* foi necessário desenvolver um método de migrar esses dados do formato texto para um formato onde pudessem ser feitos agrupamentos e operações. Para isso foi implementado um aplicativo que percorre, cada um desses arquivos de *log* gerados, do início ao fim analisando-os em busca de informações.

4.2 Linguagem de Programação

A linguagem escolhida para o desenvolvimento do aplicativo foi o *Object Pascal* utilizando o ambiente de desenvolvimento *Delphi*® da Borland®.

Segundo LEÃO (1998), o *Delphi* é um ambiente de programação baseado em *Object Pascal*, lançado em 1995. Seu ambiente é baseado em uma biblioteca de componentes chamada VCL (*Visual Component Library*). Esta biblioteca tem como objetivo geral o encapsulamento dos elementos necessários para a programação *Windows*, assim como prover os demais recursos da Orientação a Objetos.

Trata-se de uma ferramenta de desenvolvimento robusta e capaz de disponibilizar aos desenvolvedores recursos que supram suas necessidades, LEÃO(1998).

Destacou-se já na primeira edição do *Delphi* a abordagem baseada em formulários e orientada a objetos, um compilador extremamente rápido, suporte à banco de dados, integração muito forte com a programação em *Windows* e uma tecnologia de componentes, (CANTÚ, 1998).

4.3 O Processamento dos Dados

O arquivo gerado da coleta de dados não apresenta uma estrutura uniforme em observações diferentes (ver **Anexo I**). Devido a este fato foi necessário percorrer os arquivos com as amostras linha-a-linha e caractere-a-caractere para coletar as informações a serem armazenadas na base de dados.

Mesmo não possuindo uma estrutura uniforme é possível identificar os trechos do arquivo que representam os dados a serem coletados através de *tags* e pela contagem de caracteres após a ocorrência de uma determinada informação. O esquema da base de dados é ilustrado na figura abaixo (Figura 4-2).

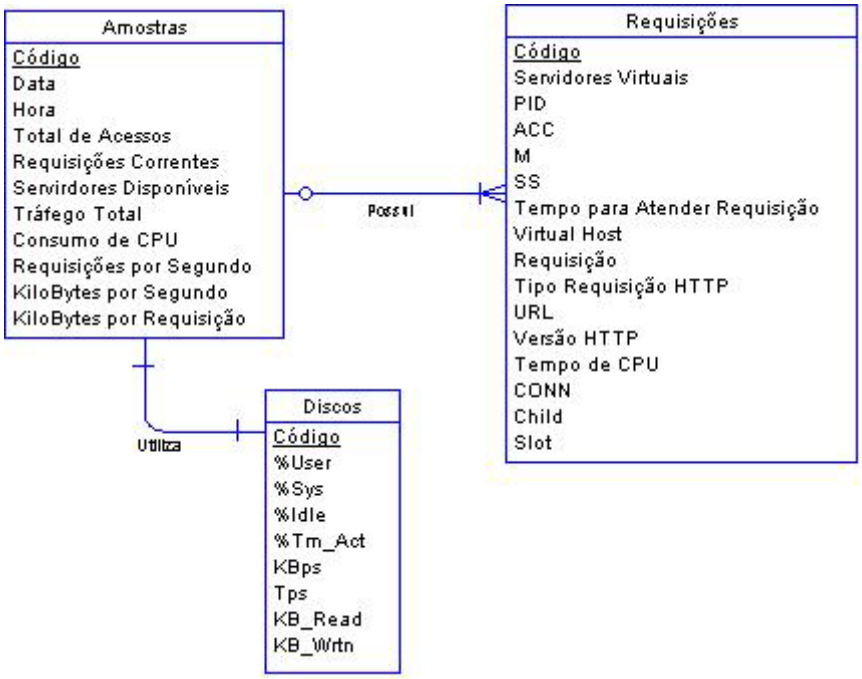

Figura 4-2 - Esquema da Base de Dados

A medida em que as informações vão sendo localizadas no arquivo elas são armazenadas em memória e ao final da coleta do conjunto de dados da amostra, ou da requisição, são gravados no banco de dados.

Optou-se pelo armazenamento das informações obtidas em uma base de dados pelo fato de a mesma oferecer melhores condições de manipulação dos mesmos.

A operação da ferramenta é bastante simples por possuir um ambiente baseado em janelas, o que facilita a familiarização dos usuários com a mesma. Para fazer a análise de um arquivo de *log* é necessário indicar o local onde está o arquivo (Figura 4-3).

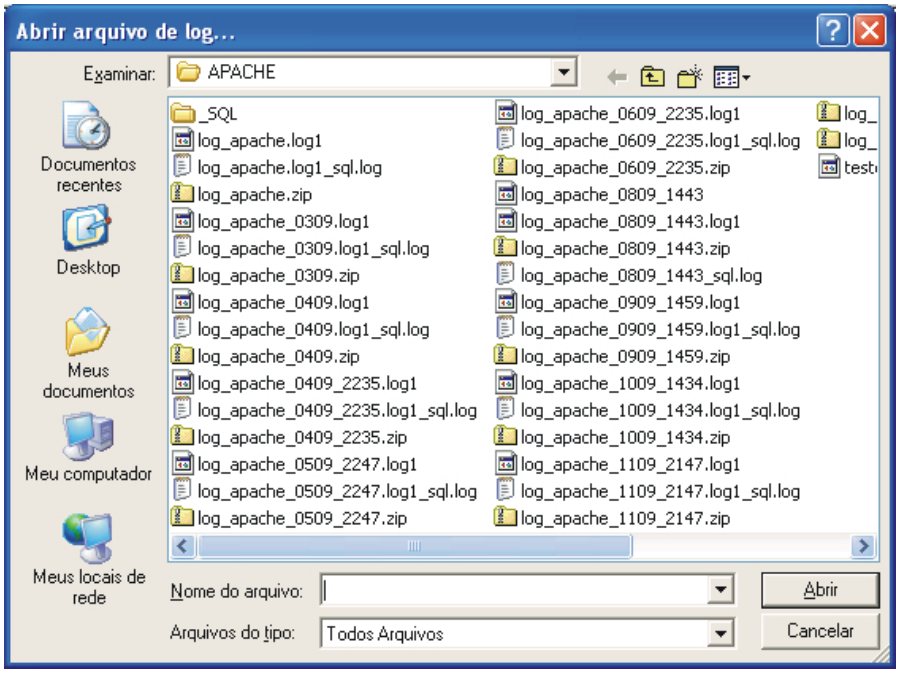

Figura 4-3 - Abrindo um arquivo de log

Depois de informada a localização do mesmo, o processo de análise dos dados é iniciado para convertê-los em informações (conforme descrito anteriormente) a serem armazenadas na base de dados. Essas informações são exibidas na tela principal (Figura 4-4).

Com as informações armazenadas na base de dados é possível a obtenção de estatísticas de utilização desse servidor. O processo de cálculo dessas estatísticas é bastante simples. A base de dados é percorrida do início ao fim, para amostras. As requisições são percorridas da primeira a última de cada amostra.

Para efetuar a análise do desempenho da máquina que provê o serviço são calculados os valores representativos como a taxa de utilização de CPU, o número de conexões ativas, o número de processos ativos que, segundo ANDREOLINI, COLAJANNI & MORSELLI (2002), são fatores importantes que afetam a latência das requisições atendidas por um servidor *web*.

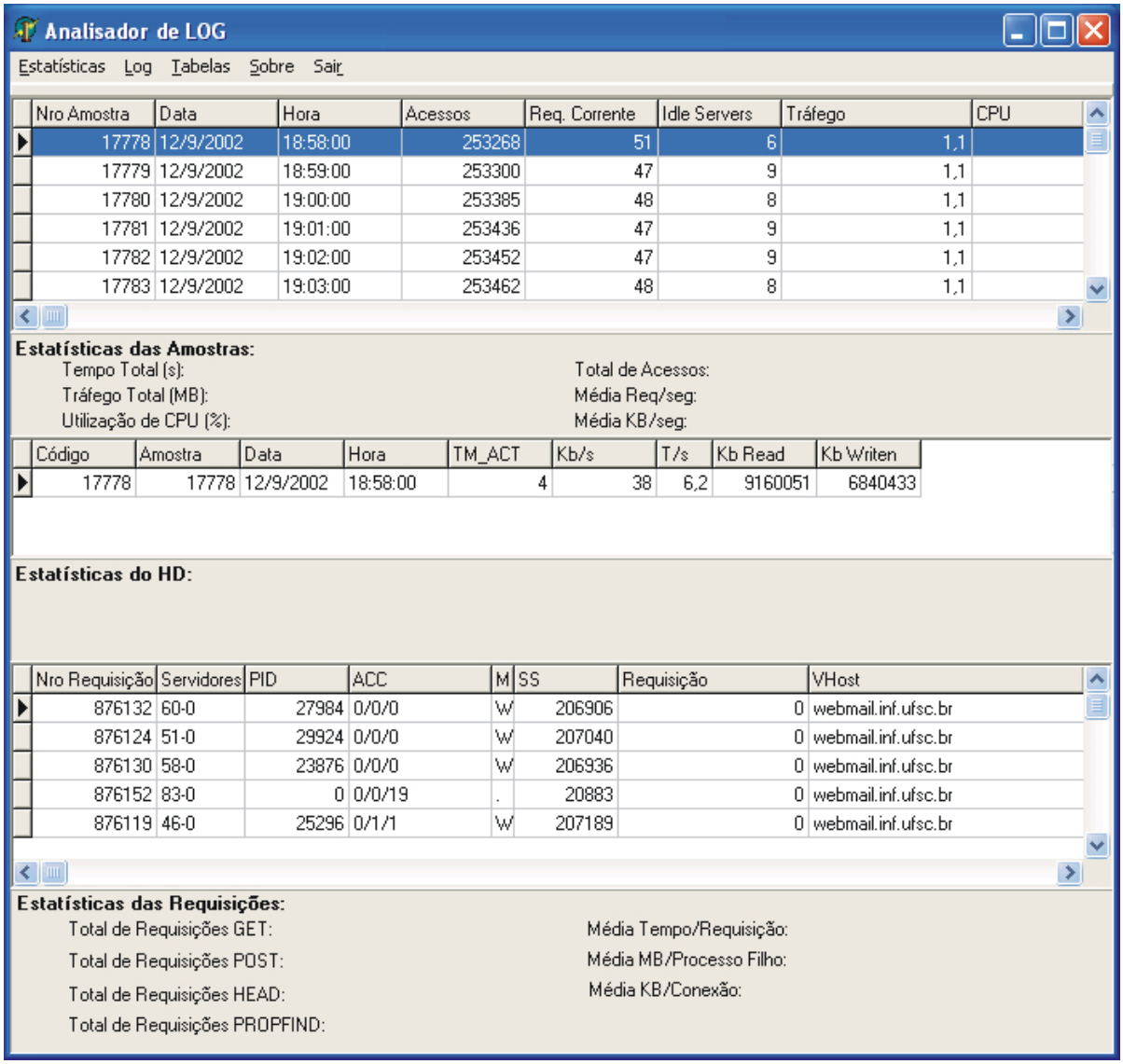

Figura 4-4 – Tela com as estatísticas de requisições e amostras

Através dos recursos oferecidos pelo aplicativo (volume e confiabilidade dos dados) é possível diagnosticar as condições de funcionamento e desempenho do sistema monitorado.

Após a migração dos dados do modo texto para a base de dados é possível fazer a exportação dos mesmos para o formato CSV, que pode ser utilizado no Microsoft Excel para a realização de análises mais detalhadas do comportamento dos mesmos. As figuras abaixo () ilustram o processo de exportação dos dados.

Figura 4-5 - Menu para Exportação dos Dados

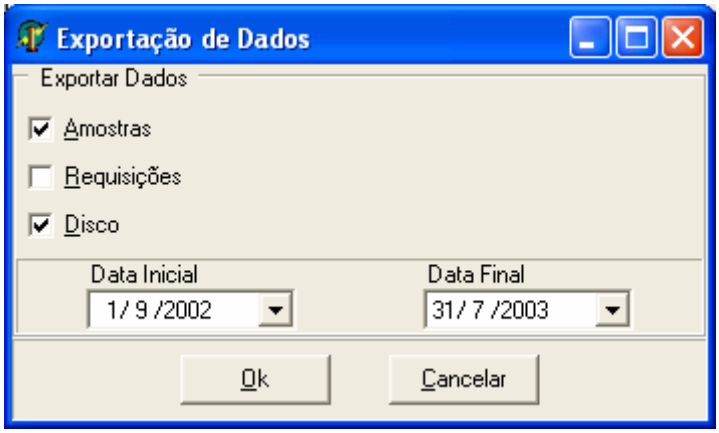

Figura 4-6 - Seleção dos Dados a Serem Exportados

Após e seleção dos dados a serem exportados o aplicativo executa a instrução SQL correspondente nas tabelas selecionadas e percorre as mesmas realizando o processo de exportação no formato CSV.

4.3.1 O Ambiente de Testes da Ferramenta

O Departamento de Informática e Estatística (INE) da Universidade Federal de Santa Catarina – UFSC conta atualmente com nove máquinas disponibilizando os serviços relacionados a redes de computadores. Dentre os serviços oferecidos estão: *SSH*, *HTTP, POP, SMTP, POPS,...*

A estrutura da rede do INE está centralizada na sala da administração da rede. Nesta sala estão os servidores, alguns *switches* e *hubs*, é por essa sala que o departamento tem acesso à rede externa a UFSC. É também nesta sala que são monitorados, configurados e alterados os serviços de TI referente a área de redes do

departamento.

A compreensão do ambiente e suas necessidades consiste em identificar o *hardware*, *software*, conectividade da rede, protocolos presentes no ambiente, períodos de pico, estruturas de gerenciamento, acordos de níveis de serviço. Visando a obtenção dessas informações MENASCÉ & ALMEIDA (1998) citam algumas questões que auxiliam nesta tarefa:

Qual a configuração dos servidores?

Qual a configuração da rede?

Qual a capacidade do *link* de comunicação de dados?

Qual o sistema operacional utilizado?

Qual a descrição dos demais componentes utilizados com *hubs*, roteadores, *firewall*, *proxy*?

4.3.1.1 Equipamentos

O departamento possui 30 linhas telefônicas e 30 modems para receber a ligações para acesso discado aos serviços oferecidos.

Roteadores (1 – Gateway)

Hubs (8 – IBM,3COM)

Switches (2 3COM, 1 IBM) 100Mbps

Links (155 Mbps com o NPD) (30.000Mbps NPD-TELESC)

Tabela 4-1 Servidores do INE

Nome	CPU	RAM	HD(s)	Serviço(s)	SO	Interface
Gateway	PIII-550	64MB	4.5GB	(Firewall camadas 3,4 e 7)	OPENBSD (k2.9)	100Mbps
Admrede3	P –100	64MB	800 MB	MP3, uso da administração.	RED HAT 7.2 (k2.4)	10 Mbps
Admrede2	P – II	64MB	20GB	Uso da administração.	RED HAT 7.2 (k2.4)	100 Mbps
Admrede	P – II	64MB	4.5GB	Uso da administração.	RED HAT 7.2 (k2.4)	100 Mbps
Marte	RISC 6000	128 MB	4.5GB	Portal (PET**), WWW-alunos (c/ PHP)** e COMPILADOR	AIX 4.3.3	10 Mbps
Mercurio	Sparc Axil 240	80MB	1GB e 2GB SCSI	Autentica RAS, DHCP, NIS, AOLPress, NFS	SOLARIS 8	100Mbps
Euryale	RISC 6000	256 MB	29GB SCSI	**WWW**, NFS, DNS, Email, IMAP, POP, **www-user (s/ PHP)**	AIX 4.3.3	100Mbps
Hipolita	RISC 6000	128 MB	10 GB SCSI	mysql, email, IMAP, POP, DNS, POPS, IMAPS, NFS	AIX 4.3.3	10 Mbps
Vênus	Sparc	128 MB	1 GB	SSH (Todos)	Solaris 8	10 Mbps
Carontes		64 MB	90 GB	NIS (Slave), Bkp-User, FTP-Anon	Red Hat Linux 7.2	10 Mbps

Fonte: Levantamento de dados feito junto aos administradores da rede do INE.

De posse desses dados foi possível identificar onde instalar o *script* de monitoração para realizar as coletas de dados. A máquina *Euryale* (RISC 6000, com 256 MB RAM, 29 GB SCSI, rodando o sistema operacional AIX 4.3.3 e com interface de rede de 100Mbps).

4.3.2 Resultados

Após a realização da coleta dos dados do servidor e da utilização do aplicativo sobre os mesmos, foi possível obter uma visão geral sobre as condições de funcionamento do servidor.

Foram coletadas 240,2 MB de dados em formato texto com a utilização do aplicativo desenvolvido, para atuar junto com o script de coleta de dados, esses dados em formato texto foram convertidos em uma base de dados com 220 MB onde é possível fazer inúmeros tipos de levantamentos estatísticos através da utilização da linguagem de pesquisa estruturada (SQL – *Structered Query Language*).

Nesses 240,2 MB estavam 17787 amostras. Durante as coletas foram realizados 1.367.317 acessos e foram transferidos 8.233,30 MB de dados.

De posse da base de dados com as informações sobre o servidor foi possível fazer uma série de análises sobre o tipo as demandas de carga e de utilização do servidor. A tabela abaixo (Tabela 4-2) ilustra as médias referentes às amostras coletadas.

Tabela 4-2 - Médias Coletadas nas Amostras Realizadas

Dado	Valor
Utilização CPU	8,05
Servidores Disponíveis	7,00
Requisições/Segundo	1,29
KiloBytes/Segundo	642,59
KiloBytes/Requisição	839,69
Requisições Processadas	15,00
Tráfego Diário	256,91
Acessos Diários	203882

Fonte: Análise Realizada pelo Aplicativo

O gráfico abaixo (Gráfico 4-1) ilustra a o volume das requisições recebidas por tipos de arquivos.

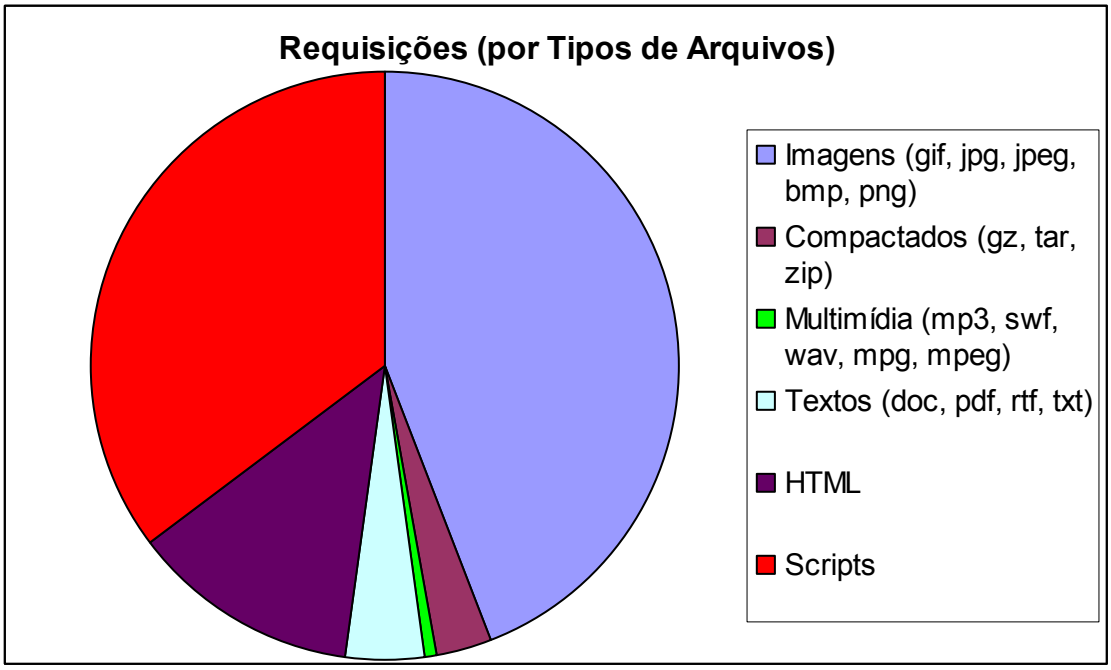

Gráfico 4-1 - Volume de requisições por tipo de arquivo

Também foram obtidas informações quanto à utilização do servidor pelas requisições recebidas no período da monitoração, conforme ilustra a tabela abaixo (Tabela 4-3).

Tabela 4-3 - Taxas Médias de Utilização do Servidor pelas Requisições

Dado	Valor
Tempo das Requisições	13,80
CPU	242,19
CONN	3,19

Fonte: Análise Realizada Pelo Aplicativo

Onde o tempo médio, em milisegundos, para atender a requisição mais recente de cada amostra foi de 13,80; o tempo médio de utilização de CPU foi de 242,19 segundos; a média de kilobytes transferidos por requisição foi de 3,19.

A utilização do disco durante o período de monitoração é representada pelos valores da tabela (Tabela 4-4) abaixo.

Tabela 4-4 - Utilização de Disco

	Tm ACT	KB/s	T/s	KB Lidos	KB Escritos
Média	4,49	39,08	6,46	15224845,38	13650908,38
Desvio Padrão	1,40	14,07	2,06	9451636,69	8334015,10
Variância	1,95	198,08	4,24		

Fonte: Análise Realizada Pelo Aplicativo

De posse dessas informações é possível ter uma visão geral sobre a situação do funcionamento do servidor no que se refere a sua carga de trabalho e suas taxas de utilização.

5 CONCLUSÕES

A avaliação de desempenho e o planejamento de capacidade são de suma importância para o bom gerenciamento de recursos, o que possibilita o provimento de serviços com qualidade.

A Internet é um sistema complexo dado ao grande de número de componentes existentes no processo de medição de desempenho. Este estudo apresentou uma nova forma de trabalhar dados de cargas de trabalho de servidores *web* através de um conjunto *scripts*/aplicativo (ferramenta).

A ferramenta apresentada atua como fonte de informações para um processo contínuo de monitoração ao longo do tempo, possibilitando a avaliação do desempenho e o planejamento de capacidade de forma mais precisa. O que favorece a elaboração de modelos de carga e desempenho mais significativos, contribuindo diretamente na melhoria do processo de planejamento de capacidade.

Como as informações são mantidas em uma base de dados é possível fazer uma análise ao longo do tempo sobre o comportamento do serviço sob diversas condições de tráfego, quantidade de acessos, etc. Uma vez que o analista pode acompanhar o comportamento do serviço em várias condições de funcionamento, assim com em várias épocas (interferências sazonais).

Com a disponibilização do recurso de exportação de dados é possível fazer a análise/utilização dos mesmos por aplicativos de simulação, estatísticos, etc. Que contribuirão, também, com a agilização do processo de caracterização de cargas de trabalho e planejamento de capacidade de servidores *web* Apache®.

Os objetivos propostos pelo trabalho foram alcançados de forma satisfatória. Pois, além da agilização o processo de avaliação de desempenho de servidores web Apache®. Os modelos de cargas podem ser elaborados de forma mais significativas por trabalhar com um volume de dados maior e mais consistentes.

5.1 Dificuldades Encontradas

Durante a realização desse trabalho foram encontradas dificuldades burocráticas e operacionais.

Também foram encontradas dificuldades no processo de obtenção dos dados a serem analisados. As políticas de segurança da Universidade, muito corretas, tornaram o processo moroso.

A falta de ferramentas de monitoração que se adequassem ao que se fez necessário monitorar, também foram um obstáculo.

O processo de extração dos dados a partir dos *logs* gerados pelo *script* de coleta foi difícil devido à inexistência de um padrão destes arquivos. E ao grande volume de dados obtidos no intervalo de tempo e ao curto período utilizado para coletas entre cada amostra.

5.2 Trabalhos Futuros

A partir do trabalho realizado pode-se dar seqüência em pesquisas que acrescentem novas funcionalidades a ferramenta. Pois o mesmo trata de um tema específico que pode ser ampliado de várias formas.

Pode-se continuar o processo de planejamento de capacidade a partir da ferramenta, implementando novas funcionalidades. Ou através de uma especificação melhor das rotinas de obtenção e avaliação dos dados coletados. Complementando a consistência dos resultados já obtidos.

Expandir o escopo do funcionamento da ferramenta para contribuir, também, no gerenciamento proativo de sistemas computacionais como servidores. Detectando e indicando situações críticas, ou possíveis situações de risco para o sistema emitindo alertas e gerando relatórios para os administradores de sistemas.

Criar uma interface entre a ferramenta desenvolvida e uma ferramenta de

simulação, como o Arena®, para que o trabalho dos analistas responsáveis pelo planejamento de capacidade tenham condições de finalizar seus projetos de forma mais consistente e rápida.

Desenvolver a ferramenta em ambiente *web* para que facilite sua utilização em ambientes distribuídos. De forma que o administrador, ou o analista do sistema em estudo possa enviar os dados coletados do sistema e, através dessa interface *web*, obtivesse uma avaliação das características das cargas de trabalho desse sistema. Assim como relatórios de utilização de recursos, etc.

6 ANEXOS

6.1 Anexo I - Arquivo com o *Estado* do Servidor

Apache Server Status for euryale.inf.ufsc.br

Server Version: Apache/1.3.14 (Unix) mod_ssl/2.7.0 OpenSSL/0.9.5a
PHP/4.0.3pl1
Server Built: Oct 13 2000 18:35:32

Current Time: Sunday, 08-Sep-2002 14:43:01 GRNLNDST
Restart Time: Sunday, 08-Sep-2002 08:01:15 GRNLNDST
Parent Server Generation: 3
Server uptime: 6 hours 41 minutes 46 seconds
Total accesses: 18047 - Total Traffic: 207.5 MB
CPU Usage: u681.11 s167.23 cu2.26 cs2.6 - 3.54% CPU load
.749 requests/sec - 8.8 kB/second - 11.8 kB/request
12 requests currently being processed, 5 idle servers

```
WK__KKKWR_RKW_W_..W.....................................................................
................................................................................................
.......................................................................
```

Scoreboard Key:
"_" Waiting for Connection, "S" Starting up, "R" Reading Request,
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,
"L" Logging, "G" Gracefully finishing, "." Open slot with no current
process

Srv	PID	Acc	M	CPU	SS	Req	Conn	Child	Slot	Client	VHost	Request
0-3	13172	0/1146/1146	W	78.75	0	0	0.0	7.59	7.59	200.151.201.82		

```
www.inf.ufsc.br GET /~sauloc/OS.zip HTTP/1.1
1-3 1934 14/1048/1048 K 77.09 0 8 22.3 16.70 16.70 200.154.160.145
www.inf.ufsc.br GET /~euclides/util/mestrado/ipv6/encrypt.gif HTTP/1.1
2-3 12406 0/1112/1112 _ 85.15 18 167 0.0 13.18 13.18 200.176.51.242
www.inf.ufsc.br GET /~rangel/osbunitoes/imagens/guraratuba.jpg
HTTP/1.1
3-3 26678 0/1367/1367 _ 93.27 35 2 0.0 11.67 11.67 200.191.186.45
www.inf.ufsc.br GET /ine5365/mux.html HTTP/1.1
4-3 15774 2/1136/1136 K 80.64 0 10705 33.0 12.62 12.62 200.226.213.86
www.inf.ufsc.br GET /barata/reco.jpg HTTP/1.1
5-3 16280 4/1253/1253 K 88.63 9 2 16.5 14.58 14.58 200.211.144.74
www.inf.ufsc.br GET /localiza.gif HTTP/1.1
6-3 31858 2/1296/1296 K 84.62 14 481 64.6 11.59 11.59 200.176.51.242
www.inf.ufsc.br GET /~rangel/osbunitoes/imagens/jaragua.jpg HTTP/1.1
7-3 17028 0/1346/1346 W 93.49 20 0 0.0 12.46 12.46 200.226.213.86
www.inf.ufsc.br GET /barata/barata1.html HTTP/1.1
8-3 18386 0/990/990 R 68.23 212 17 0.0 13.47 13.47 ? ? ..reading..
9-3 21408 0/1161/1161 _ 81.57 29 17 0.0 13.52 13.52 200.226.173.249
www.inf.ufsc.br GET /barata/etr2.html HTTP/1.1
10-3 20474 0/42/796 R 2.24 1 4 0.0 1.77 11.00 ? ? ..reading..
11-3 22488 7/83/1002 K 5.32 9 2 19.3 3.26 13.93 200.211.144.74
www.inf.ufsc.br GET /secretaria.gif HTTP/1.1
12-3 13638 0/2/968 W 0.07 39 0 0.0 0.02 8.19 127.0.0.1
euryale.inf.ufsc.br GET /server-status HTTP/1.0
13-3 4452 0/0/813 _ 2.91 19 17 0.0 0.00 7.81 209.244.151.85
(unavailable) GET
/cgi-bin/formmail.pl?email=f2%40aol%2Ecom&subject=www%2Einf
14-3 22608 0/25/444 W 2.32 280 0 0.0 0.08 6.30 209.206.212.79
www.inf.ufsc.br GET /~tsp/aha1520a_ug.pdf HTTP/1.1
15-3 28490 0/0/284 _ 0.44 14 2 0.0 0.00 8.65 200.180.90.83
(unavailable) GET /icons/compressed.gif HTTP/1.1
```

```
16-3 - 0/0/351 . 1.88 164 2 0.0 0.00 4.80 200.247.126.132
webmail.inf.ufsc.br GET /graphics/logout.gif HTTP/1.1
17-3 - 0/0/259 . 0.12 177 2 0.0 0.00 4.51 200.180.90.83
www.inf.ufsc.br GET /icons/folder.gif HTTP/1.1
18-3 25766 0/1/324 W 0.08 221 0 0.0 0.00 2.10 200.176.158.5
webmail.inf.ufsc.br GET
/view.php3/cynsfa0065.jpg?mailbox=INBOX&index=1234&bodypart
19-3 - 0/0/221 . 0.22 137 33398 0.0 0.00 2.86 200.180.90.83
www.inf.ufsc.br GET /~brunomm/disciplinas/formais/apostila_formais.zip
HTTP/1.1
20-3 - 0/0/185 . 0.73 198 867 0.0 0.00 3.77 200.196.14.173
(unavailable) GET /select.php3 HTTP/1.1
21-3 - 0/0/112 . 0.07 356 2 0.0 0.00 2.78 200.206.142.43
www.inf.ufsc.br GET /poo/atores/imagens/casa11.gif HTTP/1.0
22-3 - 0/0/108 . 0.06 364 6 0.0 0.00 0.37 200.215.99.237 (unavailable)
GET /cco/cco/imagens/cd.jpg HTTP/1.1
23-3 - 0/0/63 . 0.12 5505 7 0.0 0.00 0.44 200.193.2.80
webmail.inf.ufsc.br GET /graphics/download.gif HTTP/1.1
24-3 - 0/0/43 . 0.09 6735 2 0.0 0.00 0.68 200.196.29.67
webmail.inf.ufsc.br GET /graphics/source.gif HTTP/1.1
25-3 - 0/0/35 . 0.07 6733 2 0.0 0.00 0.08 200.196.29.67
webmail.inf.ufsc.br GET /graphics/addressbook-blue.gif HTTP/1.1
26-3 - 0/0/20 . 0.07 7935 15 0.0 0.00 0.10 200.219.185.101
www.inf.ufsc.br GET /~marcelo/tip218.gif HTTP/1.1
27-3 - 0/0/18 . 0.08 7935 10 0.0 0.00 0.20 200.219.185.101
www.inf.ufsc.br GET /~marcelo/tip219.gif HTTP/1.1
28-3 - 0/0/16 . 1.11 7929 7671 0.0 0.00 0.14 200.180.4.224
webmail.inf.ufsc.br GET /mailbox.php3 HTTP/1.1
29-3 - 0/0/22 . 0.09 7935 5 0.0 0.00 0.03 200.219.185.101
www.inf.ufsc.br GET /~marcelo/tip25.gif HTTP/1.1
30-3 - 0/0/12 . 0.09 7975 11 0.0 0.00 0.04 200.176.25.95
```

```
www.inf.ufsc.br GET /imagens/INE1.gif HTTP/1.1

31-3 - 0/0/5 . 0.06 7972 6 0.0 0.00 0.03 200.242.30.193

www.inf.ufsc.br GET /favicon.ico HTTP/1.1

32-3 - 0/0/5 . 0.05 7976 6 0.0 0.00 0.01 200.226.170.245

www.inf.ufsc.br GET /grafos/livro.html HTTP/1.1

33-3 - 0/0/5 . 0.08 7973 3 0.0 0.00 0.00 200.226.170.245

www.inf.ufsc.br GET /grafos/images/kbrgraph.gif HTTP/1.1

34-3 - 0/0/30 . 1.26 7699 2 0.0 0.00 1.19 200.215.99.6

webmail.inf.ufsc.br GET /graphics/addressbook-red.gif HTTP/1.1

35-3 - 0/0/2 . 0.04 7978 10 0.0 0.00 0.000 200.219.185.101

www.inf.ufsc.br GET /~marcelo/IMG00019.GIF HTTP/1.1

36-3 - 0/0/1 . 0.03 7977 11 0.0 0.00 0.00 200.219.185.101

www.inf.ufsc.br GET /~marcelo/IMG00009.GIF HTTP/1.1

37-3 - 0/0/4 . 0.54 7952 48 0.0 0.00 0.00 151.30.37.33 www.inf.ufsc.br

GET /~veroneze/Maiden/Caricaturas.jpg HTTP/1.1

38-3 - 0/0/2 . 0.09 7969 11 0.0 0.00 0.000 200.226.64.98

www.inf.ufsc.br GET /~renato/download.gif HTTP/1.1

39-3 - 0/0/3 . 0.06 7968 10 0.0 0.00 0.01 200.226.64.98

www.inf.ufsc.br GET /~renato/fundo.gif HTTP/1.1

40-3 - 0/0/14 . 0.10 7951 2 0.0 0.00 0.05 200.241.248.138

www.inf.ufsc.br GET /grafos/images/isomorfo1.gif HTTP/1.1

41-3 - 0/0/10 . 0.13 7950 400 0.0 0.00 0.00 200.177.66.43

www.inf.ufsc.br GET /~awangenh/CG/apostilas/apostilaport.pdf HTTP/1.1

42-3 - 0/0/2 . 0.11 7961 5858 0.0 0.00 0.07 200.204.21.172

www.inf.ufsc.br GET /%7Ebarreto/Tese.htm HTTP/1.1

43-3 - 0/0/8 . 0.11 7958 6 0.0 0.00 0.000 200.226.151.190

www.inf.ufsc.br PROPFIND /EMicro2002/ HTTP/1.1

44-3 - 0/0/1 . 0.52 7963 596 0.0 0.00 0.00 200.215.64.220

webmail.inf.ufsc.br GET

/status.php3?language=pt-BR&message=Pasta%3A+INBOX&status=g

45-3 - 0/0/1 . 0.10 7966 11 0.0 0.00 0.00 200.193.2.80
```

```
webmail.inf.ufsc.br GET /graphics/down.gif HTTP/1.1

46-3 - 0/0/1 . 0.10 7963 32 0.0 0.00 0.01 200.154.218.238

www.inf.ufsc.br GET /barata/barata16.html HTTP/1.0

47-3 - 0/0/1 . 0.12 7960 7 0.0 0.00 0.00 200.215.64.220

webmail.inf.ufsc.br GET /imp.css HTTP/1.0

48-3 - 0/0/1 . 0.08 7963 14 0.0 0.00 0.000 200.204.21.172

www.inf.ufsc.br GET /%7Ebarreto/Tese_files/image001.wmz HTTP/1.1
```

```
 Srv  Child Server number - generation

 PID  OS process ID

 Acc  Number of accesses this connection / this child / this slot

  M   Mode of operation

 CPU  CPU usage, number of seconds

 SS   Seconds since beginning of most recent request

 Req  Milliseconds required to process most recent request

Conn  Kilobytes transferred this connection

Child Megabytes transferred this child

Slot  Total megabytes transferred this slot
```

```
SSL/TLS Session Cache Status:

cache type: DBM, maximum size: unlimited

current sessions: 9, current size: 1368 bytes

average session size: 152 bytes
```

6.2 Anexo II – Arquivo de *log* de estatísticas de disco

```
tty:      tin         tout   avg-cpu:  % user    % sys     % idle    % iowait
          0.1         18.3              33.1     16.9       45.8       4.2
Disks:        % tm_act     Kbps      tps    Kb_read   Kb_wrtn
hdisk3           1.9      32.9       3.3    1807391    177828
hdisk0           8.2      70.2      11.1    2564632   1674984
hdisk1           9.2     184.4      13.7    9864306   1272254
cd0              0.0       0.0       0.0          0         0
```

7 REFERÊNCIAS BIBLIOGRÁFICAS

1 ALMEIDA, V.; ARLITT, M.; ROLIA, J. *Analazing a Web-based System's Performance Measures at Multiple Time Scales*. ACM SIGMETRICS Performance Evaluation Review, New York, Vol. 30, 2, Pag. 3-9, September 2002.

2 ANDREOLINI, M.; COLAJANNI, A.; MORSELLI, R. *Performance Study os Dispatching Algorithms in Multi-tier Web Architetures*. ACM SIGMETRICS Performance Evaluation Review, New York, Vol. 30, 2, Pag. 10 – 20, September 2002.

3 APACHE. *Apache HTTP Server Version 1.3 Documentation*. Documentação *online* do Apache (http://httpd.apache.org/docs/). Disponível em 22/02/2003.

4 APACHECTL. *apachectl - Apache HTTP Server Control Interface - Apache HTTP Server*. Documentação *online* do Apachectl (http://httpd.apache.org/docs-2.0/programs/apachectl.html). Disponível em 22/02/2003.

5 CANTÙ, Marco. **Dominando o Delphi 4: A Bíblia**. São Paulo: Makron Books. 1999.

6 COMER, D. E. **Redes de Computadores**. Porto Alegre, 2001: Bookman.

7 CROVELLA, M.; BESTAVROS, A. *Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes*. Proceedings of the 1996 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, Philadelphia, Vol. 24, 1, Pag. 160-169, May 1996.

8 DERI, L. **What's Ntop**. Documentação *online* do NTop (http://www.ntop.org/overview.html). Disponível em 22/03/2003.

9 GUTJAHR, B. *Tutorial on an Integrated Performance & Capacity Management Process***.** Proceedings of CMG 25th Conference, Reno, Session 1, December 1999.

10 JAIN, R. **The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling.** New York: Jhon Whiley & Sons. Inc. 1991.

11 LAURIE, B.; LAURIE, P. **Apache – The Definitive Guide**. Portland: O'Reilly, 1999. 2nd Edition.

12 LEÃO, M. **Delphi 4 – Curso Completo**. Rio de Janeiro: Axcel Books. 1998.

13 KANT, K; WON, Y. *Server Capacity Planning for Web Traffic Workload*. IEEE Transactions on Knowledge and Data Engineering, September 1999.

14 MARTINEZ, F. *Your First Capacity Planning Process, a survival guide*. Proceedings of CMG 27th Conference, Anahein, December 2001.

15 MENASCÉ, D. A.; ALMEIDA, V. A. **Capacity Planning for Web Performance. Metrics, Models & Methods**. Englewood Cliffs: Prentice Hall, 1998.

16 MENASCÉ, D. A.; ALMEIDA, V. A. F.; DOWDY, L. W. **Capacity Planning and Performance Modeling: from mainframes to client-server systems,** Englewood Cliffs: Prentice Hall, 1994.

17 MENASCÉ, D. A.; ALMEIDA, V. A. F. **Scaling for E-Business,** Englewood Cliffs: Prentice Hall, 1994.

18 MENASCÉ, D. A.; ALMEIDA, V.; FONSECA, R.; RIEDI, R.; RIBEIRO, F.; MEIRA, W. *In Search of Invariants for E-Business Workloads*. Proceedings of the Second ACM Conference on Electronic Commerce, Minneapolis, Pag 56-65 , October 2000.

19 MODSTATUS. *mod_status - Apache HTTP Server* . Documentação *online* do modstatus (http://httpd.apache.org/docs-2.0/mod/mod_status.html). Disponível em 24/02/2003.

20 OLSHEFSKI, D. P.; NIEH, J.; AGRAWAL, D. *Inferring Client Response Time at the Web Server*. Proceedings of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, Marina Del Rey, Vol. 30,1, Pag. 160-171, June 2002.

21 SOARES, L. F. **Redes de Computadores: das LANs, MANs e WANs às Redes ATM**. Rio de Janeiro: Campus, 1995.

22 TANNENBAUM, A. S. **Redes de Computadores**. Rio de Janeiro: Campus, 1997.

23 TSYKIN, M. *On Automated Service Level Reporting*. Proceedings of CMG 25th Conference, Reno, Session 5, December 1999.