Universidade de São Paulo
Instituto de Física de São Carlos

Servidores de Arquivos de Alto Desempenho

Carlos Norberto Fischer

Orientadora: Profa. Dra. Regina Helena Carlucci Santana

Tese apresentada ao Instituto de Física de São Carlos para a obtenção do título de doutor em Ciências: Física Aplicada - opção Física Computacional.

São Carlos - São Paulo

1996

MEMBROS DA COMISSÃO JULGADORA DA TESE DE DOUTORADO DE **CARLOS NORBERTO FISCHER** APRESENTADA AO INSTITUTO DE FÍSICA DE SÃO CARLOS, UNIVERSIDADE DE SÃO PAULO, EM 15/03/1996.

COMISSÃO JULGADORA:

Regina HC Santana

Prof. Dra. Regina Helena Carlucci Santana

Eduardo Marques

Prof. Dr. Eduardo Marques

Jan Frans William Slaets

Prof. Dr. Jan Frans William Slaets

Valt Obac Roda

Prof. Dr. Valentin Obac Roda

Solon Venancio de Carvalho

Prof. Dr. Solon Venâncio de Carvalho

Agradecimentos

Agradeço a Profa. Dra. Regina H. C. Santana pela colaboração e orientação durante o desenvolvimento deste trabalho.

Agradeço a colegas docentes e funcionários do Departamento de Estatística, Matemática Aplicada e Computacional - Rio Claro pela colaboração recebida.

Ao Departamento de Computação e Estatística do ICMSC - São Carlos pelo suporte para o desenvolvimento de parte deste trabalho.

A CAPES/PICD e FUNDUNESP pelo apoio financeiro.

Especial agradecimento a minha esposa Olga pelo apoio, carinho e paciência, e a meu filho Arthur.

E a Deus pela força maior.

RESUMO

Nesta tese são analisados projetos alternativos para servidores de arquivos baseados em rede local e as suas influências sobre o desempenho desses sistemas. Para a realização dessas análises, modelos de simulação foram utilizados como ferramentas auxiliares.

O servidor de arquivos do sistema distribuído TRICE foi utilizado como ambiente de referência para o desenvolvimento do sistema de simulação usado nas análises. O modelo construído representa as características de interesse daquele servidor de arquivos. Dados colhidos junto ao sistema TRICE foram utilizados nos processos de construção e de validação do sistema de simulação. Os programas de simulação utilizados nas análises são implementados em GPSS/H, uma linguagem de programação com ferramentas apropriadas para simulações de sistemas.

Várias técnicas que podem ser aplicadas na construção de servidores de arquivos, visando a obtenção de sistemas de alto desempenho, são analisadas utilizando o sistema de simulação desenvolvido. Resultados das simulações demonstraram os grandes benefícios que podem ser obtidos a nível de desempenho se as técnicas analisadas forem consideradas em projetos de servidores de arquivos. Dentre as técnicas consideradas, destaca-se a utilização de múltiplos discos, onde os acessos a dados são realizados apenas a áreas parciais dos dispositivos. A utilização desta técnica permite a obtenção de um servidor de arquivos de alto desempenho e de grande confiabilidade. Um protótipo foi construído para demonstrar a exequibilidade de implementação dessa técnica.

ABSTRACT

Alternative designs of LAN-based file servers are evaluated and the influence of these designs on the file server performance is investigated in this thesis. Simulation models were used as auxiliary tools in the analysis.

The TRICE file server was utilized as a reference environment for development of the simulation system used in the analysis. The model represents the main characteristics of a file server. Data acquired from the TRICE system were used to construct and validate the simulation system. The simulation programs utilized in the analysis were implemented in GPSS/H, a programming language that presents appropriate tools for systems simulation.

Several techniques that can be applied on the construction of file servers to obtain high performance are analysed using the simulation system developed. Simulation results demonstrate the benefit obtained on the performance when the techniques considered are used in file server designs. A technique that deserves special attention is the utilization of multiple disks, where the access to data are realized in partial areas of the devices. The utilization of this technique allows obtaining a high performance file server with high reliability. A prototype was built to demonstrate that the technique proposed can be implemented.

Conteúdo

Capítulo 1	Introdução.....	1
Capítulo 2	Revisão Bibliográfica: Servidores de Arquivos Para Redes Locais e Modelos de Simulação.....	6
2.1	Servidores de Arquivos Baseados em Rede Local.....	7
2.1.1	Sistemas Computacionais Distribuídos.....	7
2.1.2	Servidores de Arquivos Para Redes Locais.....	9
2.1.2.1	Conceitos Básicos Importantes.....	10
2.1.2.2	Servidores de Arquivos.....	11
2.1.2.3	O Servidor de Arquivos Ideal.....	12
2.1.3	Projeto de Servidores de Arquivos.....	13
2.1.3.1	Desempenho.....	13
2.1.3.2	Segurança.....	19
2.1.3.3	Confiabilidade.....	20
2.1.4	Revisão de Alguns Servidores de Arquivos.....	21
2.2	Modelamento Para Simulação.....	23
2.2.1	Técnicas de Aferição.....	23
2.2.2	Técnicas de Modelagem.....	26
2.2.3	Simulação de Sistemas Computacionais.....	28
2.2.3.1	Modelos de Sistemas.....	28
2.2.3.2	O Processo de Modelagem.....	28
2.3	Considerações Finais.....	32
Capítulo 3	Estudo de Caso: TRICE.....	35
3.1	O Sistema Distribuído TRICE.....	36

3.2 O Servidor de Arquivos do TRICE.....	36
3.2.1 As Características Gerais.....	37
3.2.2 A Estrutura do Sistema de Arquivos.....	39
3.3 Caracterização do Tempo de Resposta.....	40
3.3.1 As Características Internas do Servidor de Arquivos.....	40
3.3.2 A Caracterização da Demanda de Serviço.....	41
3.4 Os Dados Coletados Junto ao TRICE.....	48
3.5 Considerações Finais.....	54
 Capítulo 4 O Simulador do Servidor de Arquivos.....	 57
4.1 O Servidor de Arquivos de Referência.....	58
4.2 O Sistema de Simulação.....	59
4.2.1 O Modelo do Servidor de Arquivos.....	59
4.2.2 O Programa de Simulação.....	60
4.3 Validação do Modelo de Simulação.....	63
4.4 Adequação do Sistema de Simulação.....	70
4.4.1 Adequação da Carga de Trabalho.....	71
4.4.2 Adequação dos Recursos.....	75
4.5 Considerações Finais.....	84
 Capítulo 5 Técnicas Para a Obtenção de Servidores de Arquivos de Alto	
Desempenho.....	86
5.1 O Modelo e os Resultados das Simulações.....	87
5.2 Técnicas Para a Obtenção de Alto Desempenho.....	88
5.2.1 Redução no Número de Acessos a Disco.....	89
5.2.2 Aumento no Número de Discos.....	95
5.2.3 Comparação Entre as Técnicas Abordadas.....	101
5.3 A Estrutura Proposta e os Níveis de Carga de Trabalho.....	104

5.4 Considerações Finais.....	107
Capítulo 6 Exequibilidade de Implementação de Um Servidor de Arquivos.....	109
6.1 A Configuração do Servidor de Arquivos.....	110
6.2 Exequibilidade de Implementação.....	111
6.2.1 Exemplos de Recursos Disponíveis.....	111
6.2.2 Um Exemplo de "Hardware".....	115
6.2.3 Um Exemplo de Algoritmo.....	127
6.3 Considerações Finais.....	132
Capítulo 7 Considerações Finais, Conclusões e Proposições.....	133
7.1 Considerações Finais.....	133
7.2 Conclusões.....	138
7.3 Proposições.....	141
Referências Bibliográficas e Bibliografia Geral.....	143
Anexo A Distribuições Utilizadas no Sistema de Simulação.....	153
Anexo B Diagramas da Interface.....	159
Anexo C Listagens dos Programas.....	163

Índice de Figuras

Figura 3.1	Estrutura Básica do Servidor de Arquivos do TRICE.....	37
Figura 3.2	Tempo Médio de Resposta do Servidor de Arquivos.....	42
Figura 3.3	Tempo Entre Solicitações de Uma Estação de Trabalho.....	43
Figura 3.4	Caracterização dos Tempos Entre Solicitações.....	44
Figura 3.5	Blocos de Solicitações de Uma Estação de Trabalho.....	45
Figura 3.6	Distribuição dos Tempos Entre Solicitações.....	49
Figura 3.7	Distribuição dos Tipos de Solicitações.....	51
Figura 3.8	Distribuição do Número de Blocos dos Diretórios.....	54
Figura 4.1	Diagrama do Modelo do Servidor de Arquivos.....	60
Figura 4.2	Comparações Entre os Tempos de Resposta - Sem "Cache".....	68
Figura 4.3	Comparações Entre os Tempos de Resposta - Com "Cache".....	69
Figura 4.4	Tempos de Resposta do Servidor de Arquivos.....	74
Figura 4.5	Tempos Gastos na Fila do Disco.....	75
Figura 4.6	Tempos Gastos nas Filas do Processador e do Disco.....	76
Figura 4.7	Taxas de Utilização para o Processador e o Disco.....	77
Figura 4.8	Tempos de Resposta do Servidor de Arquivos.....	78
Figura 4.9	Tempos Gastos nas Filas do Processador e do Disco.....	80
Figura 4.10	Tempos de Resposta Usando Processadores Diferentes.....	81
Figura 4.11	Tempos de Resposta do Servidor de Arquivos.....	83
Figura 5.1	Taxas de Utilização para o Processador e o Disco.....	88
Figura 5.2	Tempos de Resposta para Diferentes Taxas de Acerto do "Cache".....	90
Figura 5.3	Tempos de Resposta para Diferentes "Caches".....	92
Figura 5.4	Tempos de Resposta para "Caches" com Escrita Atrasada.....	94
Figura 5.5	Modelo de Simulação com Vários Discos.....	96
Figura 5.6	Tempos de Resposta para Diferentes Números de Discos.....	97

Figura 5.7	Regiões de Acesso dos Discos.....	99
Figura 5.8	Tempos de Resposta para Diferentes Números de Discos.....	100
Figura 5.9	Comparação Entre as Técnicas Abordadas.....	104
Figura 5.10	Taxas de Utilização para os Sistemas de Processamento e de Disco.....	105
Figura 6.1	Diagrama de Blocos do Interfaceamento T800 - Disco IDE.....	118
Figura 6.2	Diagrama de Blocos da Interface Projetada.....	119
Figura 6.3	Tempos de Resposta para Diferentes Processadores.....	127

Índice de Tabelas

Tabela 3.1	Composição dos Tipos de Solicitações.....	47
Tabela 3.2	Composição dos Tempos das Solicitações.....	53
Tabela 4.1	Variância Interna dos Resultados das Simulações.....	67
Tabela 4.2	Comparações Entre os Tempos de Resposta - Sem "Cache"	68
Tabela 4.3	Comparações Entre os Tempos de Resposta - Com "Cache"	68
Tabela 4.4	Comparações Entre os Números de Acessos a Disco.....	69
Tabela 4.5	Relações Entre Velocidades de Processadores.....	73
Tabela 4.6	Tempos de Resposta e Gastos nas Filas do Processador e do Disco.....	79
Tabela 4.7	Tempos de Resposta do Servidor de Arquivos.....	82
Tabela 5.1	Comparação Entre as Técnicas Abordadas.....	103

CAPÍTULO UM

INTRODUÇÃO

Sistemas computacionais distribuídos estão sendo utilizados em muitos tipos de aplicações, desde a provisão de computação de propósito geral até aplicações comerciais, ou mesmo para simples troca de dados ou textos. O tamanho de um sistema distribuído varia desde uma rede local, com alguns computadores, até uma rede mundial (por exemplo, a Internet), que pode agregar milhões de máquinas.

Um *sistema computacional distribuído* [CO94, MU93, TA95] pode ser definido como uma coleção de unidades de processamento autônomas interligadas por uma rede de comunicação e gerenciadas por um "software" de integração. Todas as transferências de informações entre os elementos que compõem um sistema distribuído (computadores, estações de trabalho, impressoras, etc.) são realizadas através da sua rede de interligação, comumente chamada de *rede de computadores* [CO94, TA95].

Redes locais de computadores oferecem altas taxas de transferências, com poucos erros de transmissão, a um custo relativamente baixo. Esse tipo de rede providencia a comunicação entre os elementos de um sistema distribuído separados por curtas distâncias. Algumas características interessantes que um sistema distribuído baseado em rede local pode oferecer são [CO94, TA95]: compartilhamento de recursos por um número elevado de usuários, disponibilidade, flexibilidade na substituição de recursos ou na introdução de novos, tolerância a falhas e transparência. Dentre essas características, uma se destaca: o compartilhamento de recursos e de informações. Essa característica pode trazer reduções significativas nos custos de implantação

de um sistema distribuído de computadores baseado em rede local, sem prejudicar o bom desempenho do sistema. Quando um recurso é compartilhado por um grande número de usuários, podem ser adotados recursos de melhor qualidade sem com isso interferir fortemente no custo do sistema. O compartilhamento de informações oferece uma série de características fundamentais de um sistema distribuído: mobilidade do usuário, troca de dados, etc.

Para providenciar o compartilhamento de recursos e de informações, em um sistema distribuído de computadores baseado em rede local são implementados *servidores* que oferecem *serviços especializados*. Exemplos de servidores encontrados em redes locais são os servidores de arquivos, de impressão, de sistema ("boot") e de correio eletrônico.

Em especial, a utilização de *servidores de arquivos* [MI82, SV84, TA95] é uma forma de providenciar o compartilhamento de arquivos e bases de dados a todos os usuários da rede de computadores. Isso introduz maior disponibilidade de informações no sistema e aumenta a mobilidade dos usuários (que podem utilizar qualquer máquina da rede) e a segurança das informações contra falhas de dispositivos ou recursos (obtida através da replicação de arquivos).

Entretanto, cada servidor de arquivos pode estar servindo a vários clientes e assim se tornar um recurso bastante exigido dentro de um sistema distribuído. Se isso acontece, o tempo que cada cliente fica esperando pelas respostas de suas solicitações, enviadas a tal servidor, pode se tornar longo; isso tem influência direta sobre o desempenho total do sistema. Servidores de arquivos devem ser projetados para, entre outras qualidades, responder rapidamente aos seus clientes.

A obtenção de servidores de arquivos com baixos tempos de resposta tem se mostrado uma área bastante atrativa para a realização de pesquisas. O trabalho apresentado nesta tese foi desenvolvido com o objetivo de analisar algumas técnicas que podem ser aplicadas na construção servidores de arquivos baseados em rede local, visando a obtenção de sistemas de alto desempenho.

No entanto, para auxiliar nessas análises de desempenho de servidores de arquivos, necessitava-se de uma ferramenta adequada. Duas formas comumente usadas em processos de avaliação de desempenho de sistemas computacionais são [HA88b, MA87, SO92]: *técnicas de aferição* (por exemplo, coleta de dados e construção de protótipos) e *técnicas de modelamento* (analítico ou para simulação).

A técnica utilizada neste trabalho para a realização das análises de desempenho de implementações de servidores de arquivos foi *modelamento para simulação*, pois essa técnica é uma ferramenta bastante interessante e adequada para ser usada em processos de avaliação de sistemas computacionais. Simulação oferece alta flexibilidade, permitindo a verificação de alternativas para o projeto de um sistema de forma rápida e a um custo relativamente baixo. Além disso, resultados bastante precisos podem ser obtidos através de simulação, se o modelo for confiável.

Através de simulações foram analisadas diferentes abordagens para servidores de arquivos. Esses servidores são compostos basicamente por um sistema de processamento e por um sistema de armazenamento (disco). Utilizando simulações pôde-se observar a influência de cada um desses sistemas no desempenho do servidor.

Após demonstrar que o sistema de disco é o elemento que requer maior atenção no projeto de um servidor de arquivos, esta tese apresenta e analisa uma série de possibilidades para diminuir o tempo relativo a acesso a esse recurso. Toda a análise é feita através de resultados obtidos em simulações.

É importante ressaltar que o programa de simulação utilizado baseia-se em dados reais colhidos em um sistema distribuído e foi devidamente validado.

Finalmente, a proposta considerada mais adequada, considerando-se desempenho, custo e confiabilidade, foi caracterizada como viável através da construção de um protótipo.

O trabalho desenvolvido é apresentado nesta tese em sete capítulos. O primeiro capítulo é uma introdução, onde são feitas considerações quanto a área de interesse do trabalho e as motivações para o seu desenvolvimento.

O capítulo dois apresenta uma revisão sobre dois tópicos importantes dentro do contexto do trabalho descrito nesta tese. Inicialmente, é feita uma revisão sobre servidores de arquivos baseados em redes locais de computadores; conceitos e aspectos de interesse são revistos. A outra parte daquele capítulo está relacionada a modelos de simulação de sistemas computacionais. Os passos envolvidos num processo de modelamento de um sistema são descritos.

No capítulo três são apresentadas as características de interesse do ambiente utilizado como referência para o desenvolvimento do sistema de simulação: o servidor de arquivos do sistema distribuído TRICE. Detalhes importantes desse servidor são discutidos. Na última seção desse capítulo são apresentadas informações, colhidas junto ao sistema TRICE, que complementam a caracterização desse ambiente.

O modelo e o programa de simulação, utilizados nesta tese para a avaliação de desempenho de servidores de arquivos, são apresentados no capítulo quatro. Os critérios para a construção do sistema de simulação, bem como as técnicas utilizadas para a sua validação, são mostrados nesse capítulo.

No capítulo cinco são apresentadas discussões sobre as análises de algumas técnicas que podem ser aplicadas na implementação de servidores de arquivos visando a obtenção de sistemas com alto desempenho. O modelo e o programa de simulação do servidor de arquivos de referência foram utilizados como ferramentas na realização dessas análises.

O capítulo seis tem por objetivo demonstrar a viabilidade de implementação de um servidor de arquivos de alto desempenho utilizando uma técnica proposta nesta tese. Exemplos de recursos que podem ser utilizados em tal implementação, bem como formas de desenvolvê-la, são mostrados.

Finalmente, as considerações mais importantes sobre este trabalho, as contribuições e as sugestões para trabalhos futuros são apresentadas no capítulo sete.

CAPÍTULO DOIS

REVISÃO BIBLIOGRÁFICA: SERVIDORES DE ARQUIVOS PARA REDES LOCAIS E MODELOS DE SIMULAÇÃO

Este capítulo apresenta uma revisão bibliográfica sobre servidores de arquivos baseados em redes locais de computadores e sobre modelos de simulação, tópicos relevantes ao trabalho desenvolvido nesta tese.

Na primeira parte desta revisão, são feitas considerações quanto a conceitos e aspectos importantes dentro do contexto relacionado a servidores de arquivos. Inicialmente, são apresentadas definições e características relevantes sobre sistemas distribuídos de computadores. Na sequência, destaque especial é dado a servidores de arquivos baseados em rede local. Comentários sobre alguns conceitos básicos e requisitos necessários relativos a servidores de arquivos e sistemas de arquivamento são feitos. Também são apresentados projetos de alguns servidores de arquivos descritos na literatura.

Na segunda parte da revisão discute-se a utilização de modelos de simulação. Considerações sobre algumas técnicas direcionadas a avaliação de sistemas computacionais são feitas inicialmente.

2.1 Servidores de Arquivos Baseados em Rede Local

Neste item são revistos alguns conceitos importantes relacionados a Servidores de Arquivos Baseados em Rede Local de Computadores. Conceitos e aspectos básicos considerados relevantes ao tema deste trabalho são descritos e analisados.

2.1.1 Sistemas Computacionais Distribuídos

Um sistema computacional distribuído [CO94, MU93, TA95] pode ser definido como um sistema composto por um "hardware" baseado em múltiplas unidades de processamento (computadores ou processadores) autônomas interligadas através de uma rede e por um "software" de integração, explorando o compartilhamento implícito de recursos (discos, impressoras, etc.) e formando um ambiente computacional integrado e tolerante a falhas.

As características essenciais para um sistema distribuído são [CO94, TA95]:

- . Compartilhamento de Recursos: é uma característica fundamental de sistemas distribuídos; periféricos (como discos e impressoras) e "softwares" (como arquivos, aplicativos e base de dados) podem ser compartilhados pelos usuários do sistema; isso proporciona uma das grandes vantagens de sistemas distribuídos sobre sistemas tradicionais: a relação custo/desempenho favorável;
- . Disponibilidade: falhas de "hardware" não provocam maiores problemas para os usuários do sistema; quando uma estação de trabalho falha, o usuário pode se deslocar para outra estação; se um recurso falhar, outros recursos conectados a rede podem ser usados;
- . Flexibilidade: a substituição ou a introdução de novos recursos podem ser realizadas sem a necessidade de alterações no sistema e nos programas aplicativos, não afetando o funcionamento normal do sistema;
- . Tolerância a Falhas: através da replicação de recursos, um sistema distribuído pode se tornar tolerante a falhas; por exemplo, arquivos podem ser replicados em vários locais aumentando a

confiabilidade do sistema de arquivos, além de melhorar a disponibilidade dos dados e também a velocidade de obtenção dos mesmos pelos usuários;

- **Transparência:** o ambiente deve ser visto como uma coleção de elementos integrados em um sistema único, ocultando a natureza distribuída dos recursos utilizados na realização das tarefas dos seus usuários.

Vários tipos de servidores podem compor uma rede local de computadores, providenciando diferentes modalidades de serviços; alguns exemplos de serviços são [MI82, SV84, TA95]: de nome, correio, hora, sistema ("boot"), impressão e de arquivo.

Todas as comunicações entre os elementos que compõem um sistema distribuído são realizadas através da rede de interligação do sistema, conhecida como rede de computadores. Isso torna essa rede um elemento crítico em sistemas distribuídos, pois, a partir da ocorrência de falhas na rede, todo o sistema pode ser comprometido. As técnicas utilizadas na construção da rede de comunicação de um sistema distribuído devem providenciar que as transferências de dados sejam realizadas de formas rápida e segura.

Sistemas distribuídos podem ser implementados sobre duas grandes classes de redes de computadores [CO94]:

- **Rede de Grande Área (WAN):** esta rede interliga componentes de um sistema distribuído separados por longas distâncias (os computadores podem estar em diferentes cidades, países ou mesmo continentes). O resultado disso é que as transmissões de dados ocorrem em velocidades relativamente baixas. Este tipo de rede interliga computadores, chamados *hospedeiros*, isolados ou que estão conectados a redes locais.
- **Rede de Área Local (LAN):** este tipo de rede é projetado para providenciar a comunicação entre elementos separados por distâncias relativamente curtas dentro de um sistema distribuído. Mensagens são transmitidas em velocidades relativamente altas dentro da rede, que pode

utilizar cabos coaxiais ou de fibras ópticas para a interligação direta entre os componentes do sistema.

Tendo em vista o interesse do trabalho, a sequência desta revisão está direcionada para sistema distribuídos baseados em redes locais de computadores (LAN).

Para o funcionamento eficaz das redes de computadores são necessários mecanismos especiais (protocolos de comunicação) para o controle das transmissões de dados [AL94, CO94, MA93, MI82] e de uma interface de rede para cada elemento do sistema.

Os tipos de topologias comumente usados na estruturação de redes locais de computadores são [CO94]:

- . Topologia de Barramento ("Bus") [AL94, GR84, HW85]: a via de comunicação é um barramento comum, onde os componentes do sistema são conectados;
- . Topologia em Anel ("Ring") [AL94, GR84, HW85]: a via de interligação dos componentes da rede é estruturada na forma de um anel fechado.

Alguns exemplos de implementações de sistemas distribuídos são: Xerox Distributed System [CO94, MI82, SV84], Cambridge Distributed Computing System [MI82, SV84, TA95], Locus [LE90, PO85], TRICE [SA90a, SA90b], V-System [TA85, TA95], Amoeba [CO94, MU93, TA95], Mach [AC86, CO94], Sprite [LE90, NE88], Chorus [CO94, RO88] e Andrew [CO94, HO88, LE90].

2.1.2 Servidores de Arquivos Para Redes Locais

Um sistema distribuído baseado em rede local de computadores apresenta uma série de vantagens em relação a sistemas computacionais tradicionais. Dentre essas vantagens está uma das características mais interessantes daquela classe de sistemas: o compartilhamento de recursos,

tanto a nível de periféricos como de "softwares" [CO94, LE90, SV84, TA95]. Essa característica pode trazer reduções significativas no custo total do sistema.

2.1.2.1 Conceitos Básicos Importantes

Existem, inicialmente, três conceitos básicos (serviço, servidor e cliente) que podem ser definidos pela frase: "um *servidor* é um subsistema que providencia um tipo particular de *serviço* para os *clientes* de um sistema" [SV84]. Por exemplo, o "servidor de arquivos" é um servidor que providencia aos usuários o "serviço de gerenciar o armazenamento de seus arquivos".

Mitchell, em [MI82], define o *serviço* como "uma entidade de software rodando sobre uma ou mais máquinas" e *servidor* como "o software de serviço rodando sobre uma única máquina". Define, ainda, um *cliente* de um serviço como "um software que pode ou não estar operando em nome de um usuário humano". O cliente implementa uma interface adequada para o usuário se comunicar com um servidor [MI82, SV84]. A *interface do cliente* é um conjunto de operações que permite a clientes de um servidor de arquivos, por exemplo, ter acesso e controle sobre o serviço de arquivo.

Outro conceito importante é o de *sistema de arquivo* ou *sistema de arquivamento* que pode ser implementado no servidor de arquivos, no cliente ou parcialmente em cada um desses. A função básica desse sistema é gerenciar diretórios e arquivos de clientes de modo rápido e confiável. A questão de se saber onde e como implementar o sistema de arquivamento é um ponto chave do projeto de um servidor de arquivos; essa questão é conhecida como *o problema da decomposição* [BI80, FR85]. Esse problema pode ter duas soluções extremas possíveis na implementação de servidores de arquivos [BI80]:

- interface de baixo nível: o servidor somente controla o dispositivo físico de armazenamento; ou seja, o sistema de arquivamento é implementado no cliente;
- interface de alto nível: além da função citada para a interface de baixo nível, o servidor controla arquivos, diretórios, etc.; ou seja, o sistema de arquivamento é implementado no servidor.

A utilização do primeiro tipo de modelo facilita a construção de um servidor de arquivos relativamente simples. Para o segundo caso, os clientes podem ser simples e todos têm as vantagens de um sofisticado sistema de arquivamento. Birrel e Needham, em [BI80], apresentam uma solução intermediária: *o servidor de arquivos universal*. Este seria um dispositivo de armazenamento remoto capaz de suportar clientes com sistemas de arquivamento distintos usando o mesmo servidor de arquivos.

2.1.2.2 Servidores de Arquivos

A utilização de *Servidores de Arquivos* é uma forma de compartilhar arquivos e bases de dados [LE90, SV84, TA95], introduzindo maior disponibilidade de informações no sistema e aumentando a mobilidade do usuário (por exemplo, no caso de falha de uma estação de trabalho) e a segurança com relação as informações (gerenciando cópias e recuperação automaticamente). Um servidor de arquivos é normalmente um sistema computacional que roda um "software" especializado na execução de operações de armazenamento e recuperação de dados, de forma rápida, segura e compartilhada.

As principais razões que estimulam a implementação de servidores de arquivos em sistemas distribuídos baseados em rede local são [MI82, MU93, SV84, TA95]:

- . economia: o custo de dispositivos de armazenamento secundário pode representar uma parcela significativa do valor total de um sistema computacional. Uma vez que o atual preço por Mbyte de armazenamento decresce com o aumento da capacidade do dispositivo secundário, a centralização de recursos torna-se uma opção bastante atrativa, possibilitando até mesmo o uso de estações de trabalho sem disco;
- . compartilhamento de informações: num modelo em que cada computador tem seu próprio disco, os usuários podem ter suas próprias versões de um arquivo ou programa, tornando difícil manter versões consistentes desse arquivo ou programa no sistema. Também nesse caso, não é possível garantir que uma informação estará disponível quando outro usuário desejar ter acesso

a mesma. Através do compartilhamento de informações, pode-se providenciar o gerenciamento dos dados para acesso seguro a todos os usuários do sistema;

- expansibilidade: os aumentos nas velocidades de acesso a dados e de armazenamento do sistema podem ser conseguidos através da adição de servidores de arquivos e de dispositivos de disco.

Além da sua função primária, um servidor de arquivos pode, também, servir de base para a implementação de outros tipos de servidores necessários em redes locais de computadores, como por exemplo os servidores de impressão, de nome e de correio eletrônico [TA95].

2.1.2.3 O Servidor de Arquivos Ideal

Tanenbaum, em [TA95], cita algumas questões básicas no projeto de servidores de arquivos para redes de computadores; entre essas estão: proteção, desempenho e confiabilidade.

Baseado nessas questões, também observadas em outras literaturas [LE90, MI82, MU93, SV84], e associado a uma das razões que estimulam a implementação de servidores de arquivos (o compartilhamento de informações), pode-se citar algumas características que são desejáveis para a implementação de um *servidor de arquivos ideal* [LE90, MU85]:

- rapidez: o usuário deve observar um desempenho pelo menos igual ao de um disco local;
- segurança: excelente controle de acesso contra usuários não autorizados;
- confiabilidade: imunidade contra quebras ("crashes") do cliente, do servidor de arquivos ou dos dispositivos de armazenamento;
- flexibilidade: as informações podem ser compartilhadas por diferentes usuários ao mesmo tempo.

Implementar um servidor de arquivos ideal não é uma tarefa fácil, tendo em vista que algumas dessas características podem ser conflitantes (por exemplo, rapidez e confiabilidade).

2.1.3 Projeto de Servidores de Arquivos

Na seção anterior foram descritas características desejáveis que um servidor de arquivos baseado em rede local deve apresentar: rapidez, segurança, confiabilidade e flexibilidade [LE90, MU93]. Neste item são feitas considerações sobre essas características, bem como observações com relação a outros aspectos que devem ser relevados no projeto de tal tipo de servidor.

2.1.3.1 Desempenho

Desempenho é um ponto extremamente importante a ser considerado no projeto de servidores de arquivos. O desempenho de um servidor de arquivos é especialmente dependente das características internas do mesmo, como por exemplo os sistemas de processamento e de disco, o esquema de alocação de dados em disco e as técnicas de "cache" de disco implementadas. Entretanto, outros fatores podem afetar tal desempenho; os mais relevantes são: a velocidade de comunicação da rede e o número de clientes do servidor de arquivos.

A influência dessas características e alguns mecanismos para melhorar o desempenho e confiabilidade de servidores de arquivos são analisados a seguir.

O Sistema de Processamento

Vários autores [LA86, ME85, SA85a] apresentam estudos sobre a utilização dos dispositivos internos de servidores de arquivos baseados em rede local. A conclusão nesses trabalhos é que, dependendo da carga de trabalho imposta pelos clientes na rede, o sistema de processamento de um servidor de arquivos pode rapidamente se tornar o componente mais sobrecarregado do servidor, passando a ser o fator limitante do desempenho desse servidor.

Baseado nisso, pode-se concluir que, para o projeto de um servidor de arquivos, deve haver grande preocupação com relação a minimização dos tempos gastos relativos ao sistema de processamento; uma forma de se obter isso é através da utilização de um processador que apresente alto desempenho.

O Sistema de Disco e o Esquema de Alocação de Dados

As características do dispositivo de armazenamento utilizado no servidor de arquivos e os esquemas usados para alocação de espaço em disco têm grande influência sobre o tempo total de acesso a arquivos.

A utilização de discos com altas taxas de transferência e baixos tempos de acessos a dados traz grandes benefícios ao desempenho de servidores de arquivos [FR85]. Também, a utilização de características adequadas podem promover aumento de tal desempenho, como, por exemplo, o armazenamento temporário ("bufferização") de blocos parciais do disco no servidor de arquivos e a busca antecipada de blocos de dados no disco [PO85], além do aumento no tamanho do bloco de transferências com o disco (mais informação pode ser transferida em um único acesso) [FR85].

Outra forma de melhorar o desempenho do sistema de armazenamento é construir servidores de arquivos com múltiplos discos. A replicação de discos, conhecida como *vetor redundante de discos* (RAID) [GA94, HA92, MU91, PA88b], pode proporcionar aumento na taxa de transferência de dados ou aumento na taxa de atendimento de solicitações.

Para ambientes computacionais onde cada solicitação requer a transferência de um arquivo com grande quantidade de dados, cada arquivo pode ser dividido em vários blocos de dados; o primeiro bloco é escrito no disco A, o segundo bloco no disco B, e assim sucessivamente, até o acesso ao N-ésimo disco compondo o RAID. As transferências (tanto para leitura como para escrita) sobre os N discos considerados poderiam ser realizadas em paralelo, diminuindo os tempos de transferências de cada arquivo em relação aos tempos obtidos para um dispositivo de disco padrão, o que aumentaria o desempenho do sistema de armazenamento.

Para sistemas computacionais onde um grande número de pequenas solicitações de entrada/saída são geradas, se o RAID considerado permite que seus discos atuem de forma independente, cada disco poderia atender a uma solicitação diferente; isso significa que um RAID de N discos poderia processar N solicitações simultaneamente, diminuindo o tempo médio de atendimento de cada solicitação.

Dependendo da implementação, se os arquivos forem replicados, o RAID aumenta a confiabilidade do sistema com relação a falhas de dispositivo.

Quanto a alocação de espaço em disco, a utilização de algoritmos apropriados, que providenciem melhor alocação dos dados, deve minimizar os tempos de busca para escrita/leitura do arquivo [FR85].

Técnicas de "Cache" de Disco

Um modo de melhorar o desempenho de servidores de arquivos é através da utilização de técnicas de "cache" de disco [JA89, KA94, NE88, TE87a]. Nesses esquemas, blocos de arquivos ou mesmo arquivos inteiros do disco do servidor são armazenados temporariamente na memória principal do servidor. O uso de "caches" reduz o número de acessos a disco. Como consequência, ocorre uma minimização do tempo total relativo ao sistema de disco (deixam de ocorrer muitos posicionamentos das cabeças de leitura/escrita e muitas transferências de dados entre o disco e a memória principal do servidor).

Em sistemas distribuídos, quando essa forma de implementação do "cache" (apenas na memória principal do servidor de arquivos dos clientes) é considerada, tal modelo de implementação é conhecido como "*cache*" *centralizado* [TE87a]. Num outro modelo de implementação, conhecido como "*cache*" *distribuído* [LE90, SV84, TE87a], os "caches" são implementados em outras máquinas: na memória principal ou em disco local do cliente e/ou nas memórias principal e secundária de outros servidores.

A vantagem do modelo centralizado é a relativa simplicidade para implementar e manter o "cache" (gerenciamento de apenas duas cópias). O desempenho do sistema melhora uma vez que acessos a disco são evitados. A desvantagem desse modelo é o tamanho possível para o "cache", uma vez que esse é limitado pela quantidade disponível de memória principal do servidor. Além disso, esse tipo de "cache" evita apenas acesso a disco, mas não minimiza acessos a rede e ao servidor

O "cache" distribuído pode evitar acessos ao servidor de arquivos e a rede (cópias nos clientes) ou minimiza a distância entre os dados e o usuário (cópias em outros servidores), diminuindo os tempos gastos em comunicação através da rede, com aumento de desempenho, confiabilidade e disponibilidade do sistema. A desvantagem do modelo distribuído é que o gerenciamento das cópias distribuídas é mais complexo (esquema ótimo de alocação de dados e consistência das possíveis cópias de um arquivo).

Quanto a consistência de várias cópias de um arquivo [DA85, LE90, NE88, TE87a], esse problema é menos sério quando o "cache" está no servidor de arquivos, uma vez que o "cache" precisa ser consistente somente com o disco. Mas, se o "cache" está no cliente, o problema de consistência se torna mais complexo, uma vez que cópias de um arquivo podem existir em muitos lugares. Esse problema pode ser tratado através das seguintes técnicas [DA85, HA88a]:

- . os arquivos no "cache" ou as cópias duplicadas são imutáveis: nessa técnica as cópias de um arquivo somente podem ser lidas;
- . atualizações são coordenadas por uma das cópias: somente uma cópia específica pode ser alterada por vez;
- . técnica de votação: nesse esquema, são realizadas comparações entre dados de vários servidores de arquivos para validar as cópias de arquivos.

Ainda quanto a consistência de arquivos, sistemas que possibilitam a replicação de dados não podem permitir modificações independentes conflitantes para diferentes cópias do mesmo arquivo. Algumas técnicas para evitar este problema são [SA85a, TE87a]:

- . consistência do "cache" iniciada pelo cliente: o servidor de arquivos é consultado para validação do "cache" quando um arquivo é aberto;
- . consistência do "cache" iniciada pelo servidor: na modificação de um arquivo, o servidor envia uma mensagem aos clientes para descartarem o arquivo do "cache".

A primeira técnica pode fazer com que muitas das chamadas recebidas pelo servidor sejam para confirmar a validade do "cache" [SA85a, TE87a]. Várias literaturas apresentam mecanismos para gerenciar replicação, atualização, controle de consistência e recuperação de arquivos em ambientes distribuídos [BE84, DA85, HA88a, HO81, SA85a, TE87a, WE86]

Quanto ao projeto e operação de "caches", alguns pontos a serem considerados para sistemas distribuídos são [KA94, LE90, NE88, OU85, PR90, SM82]:

- . procedimento de escrita: quando os dados do "cache" são modificados, o armazenamento secundário precisa ser atualizado. Isso pode ser feito, basicamente, de duas formas:
 - . escrita imediata: atualiza o conteúdo do disco toda vez que uma escrita é solicitada;
 - . escrita atrasada: atualiza o conteúdo do disco em condições predeterminadas (por exemplo, tempo decorrido e disco ocioso);
- . tamanho do "cache": o tamanho do "cache" depende das condições de custo, desempenho e local de sua implementação;
- . tamanho do bloco: o tamanho do bloco do "cache" (a unidade de armazenamento) em sistemas distribuídos pode ser o arquivo inteiro ou blocos de disco;
- . algoritmo de substituição: modelos de substituição de um bloco (ou um arquivo) para, por exemplo, quando o "cache" está cheio, são:
 - . pela utilização: o bloco que não é acessado a mais tempo é retirado (LRU);
 - . por tempo: o bloco que estiver a mais tempo no "cache" é retirado (FIFO);
- . algoritmo de seleção para o "cache": os dados a serem armazenados no "cache" podem ser selecionados de duas formas:
 - . busca de demanda: o dado é selecionado quando acesso ao mesmo é requisitado;

- busca antecipada: o dado é selecionado antes de ser requisitado.

A Velocidade de Comunicação

As taxas de transferências de informações entre clientes e servidores de arquivos são bastante afetadas pelos tempos adicionais exigidos nos processos de comunicação do sistema. Entre os fatores que influenciam diretamente esses tempos estão os tipos de protocolos utilizados, a quantidade de mensagens trocadas para a realização de uma tarefa e o tamanho dos pacotes de transferência pela rede [SV84, TA85].

Em geral, um cliente usa um tipo de mecanismo de *chamada remota de procedimento* (RPC) [BI85, CO94, SV84, TA95] para interagir com os servidores de arquivos remotos. O protocolo básico de comunicação consiste de dois passos [LE90, SV84, TA95]: *envia_pedido*, *recebe_resposta*. O cliente faz o pedido de serviço com o nome e parâmetros da operação desejada e o servidor envia os resultados como resposta para o cliente. Um terceiro passo pode ser adicionado: um *reconhecimento* do cliente quando recebe a resposta [SV84, TA95].

A interface do servidor de arquivos e os protocolos de comunicação devem ser cuidadosamente projetados a fim de minimizar os custos de geração de mensagens e também o número de mensagens trocadas [SV84]. Svobodova, em [SV84], descreve diferentes implementações de mecanismos de comunicação.

O uso de "caches" ou de armazenamento secundário na estação cliente pode minimizar os tempos gastos com comunicação, pois o acesso a arquivos situados nesses "caches" evita transferências pela rede. Esse procedimento pode melhorar o desempenho do sistema como um todo, pois minimiza o congestionamento da rede.

O Sistema de Arquivos Distribuído

O aumento no número de clientes de um servidor de arquivos pode causar uma saturação no serviço, ou seja, pode haver uma degradação no desempenho do servidor. Uma

solução imediata é replicar o número de servidores de arquivos [CO94, DA85, HA88a] provocando um aumento de disponibilidade e de desempenho de todo o sistema.

Se esses múltiplos servidores estiverem trabalhando de forma cooperativa, gerenciados por um único sistema de arquivamento, o conjunto atua como um único sistema lógico, proporcionando melhor desempenho que um servidor de arquivos único. Isso pode trazer grandes benefícios ao sistema, como por exemplo: aumento de capacidade de armazenamento e processamento (mais que um servidor para armazenar dados e processar as transferências), melhor confiabilidade e robustez do sistema (quebra de um servidor não afeta todo o sistema), aumento de disponibilidade dos dados (arquivos podem ser replicados em vários servidores), manutenção mais fácil (manutenção de um servidor pode ser feita sem afetar o resto do sistema) e expansibilidade (mais fácil inclusão de novos servidores de arquivos).

Um conjunto de servidores de arquivos trabalhando dessa forma recebe o nome de *Sistema de Arquivos Distribuído* [MI82, LE90, SV84].

Entretanto, a replicação de servidores de arquivos pode provocar alguns problemas. O primeiro é decidir a alocação ótima de arquivos para cada cliente. De acordo com Woodside e Tripathi, em [WO86], a solução é equalizar, tanto quanto possível, o uso dos servidores para alocação. Outro problema é quanto a transparência de alocação, ou seja, o usuário não precisa saber qual servidor de arquivos está sendo usado. A solução é o sistema de arquivos gerenciar a alocação de arquivos dos clientes.

O uso de múltiplos servidores de arquivos permite a criação de várias cópias de um mesmo arquivo. Essa característica introduz o problema de manter várias cópias de um arquivo em diferentes servidores de forma consistente [DA85, HA88a]. As soluções para isso podem ser similares às descritas anteriormente para consistência entre "caches" em um sistema distribuído.

2.1.3.2 Segurança

Uma preocupação que se deve ter no projeto de servidores de arquivos para sistemas distribuídos é o *controle de acesso*: quem está autorizado a ler e a escrever em cada

arquivo. Essa questão pode ser resolvida usando-se *listas de controle de acesso, capacidades e codificação* ("encryption") [BI80, CO94, MI82, SV84, TA95].

Com listas de controle de acesso, a cada arquivo é associada uma lista de usuários autorizados a usá-lo. Junto com algum identificador de arquivo (FID), o usuário precisa apresentar alguma prova de sua identidade, a qual lhe dará direitos de acesso ao arquivo solicitado.

Em sistemas baseados em capacidades, o cliente só precisa apresentar uma senha especial para cada acesso a arquivos provando sua autorização para isso. Em muitos sistemas o FID é usado como capacidade.

Quando codificação é usada, cada arquivo tem uma chave única e o arquivo é armazenado e transferido em uma forma codificada usando aquela chave. Neste caso, o servidor de arquivos não valida a capacidade; o arquivo é enviado ao cliente, que só tem acesso ao seu conteúdo se tiver a chave de codificação.

2.1.3.3 Confiabilidade

Inicialmente, para se garantir a confiabilidade de um servidor de arquivos deve-se projetá-lo cuidadosamente, usando disco de boa qualidade, e ocasionalmente fazer cópias ("backups") de arquivos [TA85]. Entretanto, mesmo com esses cuidados, outros fatores podem causar modificações ou destruição dos dados, como por exemplo, falhas do servidor de arquivos ou do cliente.

Uma forma de evitar perda de dados é utilizar *transações atômicas* [CO94, HO81, MI82, SV84, TA95]. Uma operação é dita atômica se ao seu final os dados estão consistentemente em um novo estado (a operação termina com sucesso) ou no estado antigo (nenhuma atualização foi executada). Da mesma forma, uma transação é atômica se todas as suas operações são executadas ou, no caso de falha, nenhuma é executada.

A execução de uma transação atômica pode ser dividida em três etapas [SV84]:

- . "schedule": o processo de concorrência entre solicitações é definido;
- . operações: as solicitações são executadas;
- . comprometimento: as modificações são feitas permanentes.

Em geral, mais de uma transação pode tentar o acesso a um arquivo de forma concorrente. Isso traz a necessidade de se ter alguma forma de controle de concorrência. Alguns mecanismos que um servidor de arquivos pode usar para controle de concorrência são [BE84, CO94, MI82, SV84, TA85]:

- . mecanismo de bloqueamento ("lock"): permite acesso exclusivo a um conjunto de dados (a unidade de bloqueamento);
- . "timestamp": um número único, relacionado ao tempo de execução, associado a cada transação;
- . controle otimístico: considera que não haverá acessos conflitantes.

2.1.4 Revisão de Alguns Servidores de Arquivos

Existe um grande número de servidores de arquivos implementados com as mais distintas características. Na sequência, são brevemente revisados alguns desses servidores de arquivos:

- . Servidor de Arquivos Alpine [BR85, SV84]: desenvolvido na Xerox Palo Alto Research Center (PARC), EUA, é dirigido para aplicações de base de dados, suportando gerenciamento integrado de diferentes formas de informação; implementado em linguagem Cedar, está em operação desde 1983;

- Servidor de Arquivos Amoeba [CO94, MU86, TA95]: desenvolvido na Universidade de Vrije, Amsterdam, Holanda, é um servidor que usa um mecanismo de controle de concorrência otimístico e um modelo mínimo de RPC; seu projeto e desenvolvimento foi iniciado em 1981;
- Sistema de Arquivos Andrew [CO94, HO88, LE90]: implementado na Universidade de Carnegie-Mellon, EUA, é direcionado para providenciar um mecanismo eficiente de compartilhamento de informações aos usuários; desenvolvimento iniciado em 1983;
- Servidor de Arquivos de Cambridge - CFS [MI82, SV84, TA95]: desenvolvido na Universidade de Cambridge, Inglaterra, visava implementar um servidor de arquivos universal [BI80], podendo ser usado por computadores com diferentes sistemas operacionais; escrito em BCPL, está em operação desde de 1980;
- Sistema de Arquivos Chorus [CO94, RO88]: desenvolvido no Instituto Nacional de Pesquisas em Informática e Automática (INRIA), França, usa mensagens de comprimento variáveis (limitadas a 64 Kbytes); implementado em C++, está em uso desde 1988;
- Coda [CO94]: implementado a partir de 1989 na Universidade de Carnegie-Mellon, EUA; o projeto visa envolver seu uso em sistemas distribuídos de larga escala em redes locais (LAN) e de grande área (WAN);
- Sistema de Arquivos Locus [LE90, PO85]: desenvolvido na Universidade da Califórnia, EUA, o seu projeto enfatiza questões relativas a tolerância a falhas, com sofisticadas características voltadas ao gerenciamento automático de dados replicados e atualizações atômicas de arquivos; está em uso desde de 1980;
- Sistema de Arquivo Sprite [LE90, NE88, OU88]: projetado na Universidade da Califórnia, EUA; a estrutura em árvore do sistema de arquivos engloba todos os arquivos e dispositivos da rede, tornando-a transparente e igualmente acessível a todos os clientes; desenvolvido a partir de 1982;

- Sistema de Arquivos em Rede Sun - Sun NFS [CO94, LE90]: sua implementação é parte do sistema operacional SunOS; este sistema de arquivos é usado para acesso transparente a arquivos remotos em ambientes UNIX; em funcionamento desde 1981;
- Servidor de Arquivos Trice [SA90b]: desenvolvido no Departamento de Eletrônica e Ciências da Computação da Universidade de Southampton, Inglaterra, é implementado em BCPL e apresenta uma interface do cliente de nível alto.

2.2 Modelamento para Simulação

Muitas vezes, torna-se necessário fazer análises quanto ao desempenho de sistemas computacionais em geral e de servidores de arquivos, em particular. Para essas situações, deve-se ter a disposição ferramentas apropriadas que permitam a realização dessas análises de formas segura e correta. Tais ferramentas podem se tornar bastante úteis nas seguintes situações [HA88b, KO78, MA87]:

- análise de sistemas existentes: objetivando maximizar eficiência e utilização de recurso, minimizar custos e tempo de resposta, etc;
- análise de sistemas em desenvolvimento: visando prever o comportamento de um novo sistema;
- seleção entre sistemas existentes: determinação do sistema mais adequado.

A seguir são descritas algumas técnicas que podem ser utilizadas em processos de análise de desempenho de sistemas computacionais.

2.2.1 Técnicas de Aferição

As ferramentas utilizadas na avaliação de sistemas computacionais podem ser construídas a partir de experimentação prática de um sistema existente [KO78, MA80]. Nesse caso, o sistema é instrumentado (por "hardware" ou por "software") para a coleta das informações

necessárias para a avaliação do seu desempenho, com posterior análise desses dados. Os esquemas de obtenção de informações junto ao próprio sistema recebem o nome de *Técnicas de Aferição*.

Algumas técnicas de aferição que podem ser utilizadas em processos de avaliação de sistemas computacionais são descritas a seguir.

Coleta de Dados

Coleta de Dados [KO78] é uma das técnicas que fornece os resultados mais precisos na avaliação de desempenho de sistemas computacionais. Essa técnica pode ser utilizada quando o objetivo é a obtenção de dados visando a análise direta de um sistema ou quando a intenção é apenas coletar informações precisas visando outros fins. A coleta de dados deve ser aplicada sobre sistemas existentes, operando em condições normais.

A obtenção das informações necessárias para a avaliação do sistema computacional pode ser realizada através da utilização de "hardware" ou "software" direcionados para aquele fim [KO78]. Para a primeira forma, o sistema é instrumentado por meio de monitores de "hardware" (sensores eletrônicos, dispositivos lógicos para monitoramento de sinais, etc.). Para o outro caso, a coleta dos dados pode ser feita através de monitores de "software", que são conjuntos de rotinas executadas junto ao processamento normal do sistema.

A coleta de dados é também muito útil para a obtenção de estimativas para os parâmetros de entrada requisitados por modelos analíticos e de simulação [KO78]. Também, dados coletados junto a um sistema real podem ser usados para a validação de tais tipos de modelos [SA84]. Essa validação é realizada através da comparação entre os resultados obtidos com a resolução do modelo considerado e os dados colhidos junto ao sistema real.

Um dos inconvenientes dessa técnica é que cuidados especiais devem ser tomados para que os mecanismos usados na coleta dos dados não interfiram no comportamento do sistema em análise, pois os tempos gastos nas medições podem estar sendo incluídos nas medidas de desempenho sendo tomadas [KO78]. Uma vez que a coleta de dados necessita que o sistema em

análise exista fisicamente, essa técnica não é adequada para situações onde alternativas de projeto devem ser consideradas.

Uso de Índices de Desempenho

Índices de desempenho ("benchmarks") geralmente são usados quando se deseja fazer comparações entre vários sistemas computacionais. Um "benchmark" é um programa computacional que produz uma carga de trabalho que deve ser executada sobre todas as máquinas consideradas [TA92, WE90]. Os tempos de execução dessa carga nas máquinas são confrontados, fornecendo um bom índice para os desempenhos dos sistemas.

Um bom "benchmark" deve apresentar as seguintes características [WE90]:

- . deve ser escrito em linguagem de alto nível, para rodar em qualquer máquina;
- . deve ser representativo de algum estilo de programação (numérico, comercial, etc.);
- . possa ser avaliado facilmente;
- . possuir larga distribuição.

Um inconveniente dessa técnica é que os "benchmarks" disponíveis podem não ser representativos para uma aplicação desejada. Mas, a principal desvantagem dos "benchmarks" é que estes não possibilitam analisar o comportamento do sistema quando diferentes alternativas para o mesmo devem ser consideradas.

Alguns exemplos de "benchmarks" são [WE90]: Whetstone, Dhrystone, Linpack, EDN e SPEC.

Construção de Protótipos

Esta técnica consiste em criar um protótipo do sistema a ser analisado. O protótipo [HA88b, ST87] é uma versão funcional simplificada de um sistema, onde faltam características

não essenciais que estariam presentes apenas no sistema real. Essa é uma forma de se obter boa precisão para os dados a serem considerados nas análises.

A construção de protótipos é bastante útil quando se deseja a obtenção de informações de um sistema ainda não existente [ST87]. Com isso, informações sobre o comportamento futuro do sistema podem ser obtidas previamente, possibilitando a detecção de erros de projetos em um tempo menor, o que reduz os custos envolvidos.

As grandes desvantagens de protótipos são relativas ao custo envolvido na sua implantação (tornado-os ferramentas caras) e ao longo tempo gasto em suas construções. Isso torna essa técnica pouco adequada para a análise de diferentes alternativas para um sistema [ST87].

2.2.2 Técnicas de Modelagem

As técnicas de aferição, descritas na seção anterior, em geral fornecem resultados bastante precisos quando empregadas na avaliação de desempenho de sistemas computacionais. Entretanto, essas técnicas apresentam alguns inconvenientes, tais como exigência de disponibilidade do sistema ou a implementação de um protótipo do mesmo; além disso, não são adequadas para a análise de projetos alternativos para um sistema. Assim, outros tipos de ferramentas, direcionadas para a avaliação de desempenho de sistemas computacionais, devem ser considerados.

Uma forma alternativa de obter ferramentas direcionadas a avaliação de sistemas computacionais é através de abstração do sistema considerado [KO78, MA87, SO92]. Quando a abstração de um sistema é utilizada, a análise é realizada a partir da criação de um modelo teórico representativo desse sistema. Nesse processo de análise, o modelo, após a sua criação, deve ser resolvido. Os esquemas de abstração de sistemas recebem o nome de *Técnicas de Modelamento*.

A seguir, são descritas as técnicas de modelamento mais utilizadas na avaliação de sistemas computacionais.

Modelamento Analítico

A técnica de modelamento analítico [MA87, SO92] possibilita a obtenção de relações funcionais entre os parâmetros do sistema em análise e os critérios de desempenho escolhidos. Essas relações são descritas em termos de equações que podem ser resolvidas analiticamente.

Uma característica bastante interessante do modelamento analítico é o fato dessa técnica poder ser usada tanto para sistemas existentes quanto para sistemas a serem implementados. Além disso, resultados com boa precisão podem ser obtidos com essa técnica, a custos relativamente baixos [MA87, SH88].

Entretanto, a medida que aumenta a complexidade do sistema considerado (ou o número de detalhes e características a serem relevados), cresce também a complexidade do modelo em uso, dificultando a resolução analítica do mesmo [MA87]. Outro inconveniente de modelos analíticos surge quando se deseja avaliar possíveis modificações no sistema em análise. Para refletir essas modificações no modelo associado, deve-se refazer parte ou mesmo todo o modelo e ainda ter que solucioná-lo [MA87, SO92]. Também, a validação do modelo (garantia de que o modelo é uma representação consistente do sistema real) pode ser tornar uma tarefa bastante difícil [SA84, SO92].

Simulação

Simulação [MA87, SA84, SH88, SO92] é a técnica que consiste em modelar um sistema e então transformar o modelo obtido em um programa computacional que represente o sistema real com um bom nível de fidelidade. Os resultados obtidos com o programa são prognósticos do comportamento possível de um sistema se este for submetido a condições variadas [MA80].

Essa técnica é bastante flexível, podendo ser aplicada para sistemas existentes ou não. Em especial, sua aplicação é bastante recomendada em situações onde alternativas para a implementação ou a adequação de um sistema devem ser investigadas [MA87, SO92]. Da mesma

forma como acontece com a técnica de modelamento analítico, simulação possibilita a obtenção de resultados com boa precisão a baixo custo [MA87, SH88].

Para algumas aplicações, dependendo do nível de complexidade do modelo de simulação construído, grande tempo de processamento do programa associado pode ser necessário. Mas, também nesta técnica, o maior inconveniente apresentado está relacionado com a validação do modelo de simulação obtido e do seu programa associado [SA84, SO92].

2.2.3 Simulação de Sistemas Computacionais

O que se observa a partir das descrições feitas anteriormente é que a Técnica de Simulação pode apresentar bons níveis de precisão para os resultados com custos de avaliação muito baixos [MA87, SH88]. Além disso, como descrito anteriormente, a aplicação dessa técnica é bastante recomendada em situações onde alternativas para a adequação de um sistema existente ou para a implementação de uma arquitetura alternativa devem ser investigadas [MA87, SO92] (esse é o caso do trabalho apresentado nesta tese). Isso ocorre porque simulação oferece grande flexibilidade, permitindo que alterações no modelo sejam feitas com relativa facilidade (comparativamente com as outras técnicas descritas).

2.2.3.1 Modelos de Sistemas

Os modelos de sistemas podem ser classificados de acordo com o modo no qual o modelo representa as mudanças de estado no sistema em consideração. Essas mudanças podem ocorrer de forma contínua ou de forma discreta. Desse modo, os modelos podem ser classificados em [MA80, MA87, SO92]:

- **Modelos para Sistemas Contínuos:** representam sistemas nos quais as mudanças de estado ocorrem de forma contínua no tempo; como exemplo, os modelos para a análise de dissipação de calor e de aplicações biomédicas;

- . **Modelos para Sistemas Discretos:** representam sistemas em que as mudanças de estado ocorrem em pontos discretos do tempo (em intervalos fixos de tempo); como exemplo, sistemas computacionais, que em muitas situações executam atividade em intervalos regulares.

2.2.3.2 O Processo de Modelagem

O processo de modelagem de um sistema consiste em abstrair as características do sistema em um modelo [KO78, MA87]. O processo de modelagem de um sistema pode ser dividido em três etapas básicas [MA80, MA87, SH88]: desenvolvimento, testes e análises de resultados. Essas etapas são descritas a seguir.

A Etapa de Desenvolvimento

Dentro dessa etapa, vários passos precisam ser seguidos [MA87, SO92]. Após o entendimento do sistema considerado, os elementos e os aspectos relevantes do sistema, a serem incluídos no modelo, devem ser identificados e caracterizados. As metas pretendidas e a quantidade de detalhes devem ser definidas clara e objetivamente, para então iniciar-se a construção do modelo. Durante o desenvolvimento do modelo, são gerados os requisitos para os dados de entrada e internos do mesmo; esses dados servirão como parâmetros para o modelo. Os dados de entrada utilizados em um modelo podem ser obtidos de várias formas, como por exemplo junto a sistemas existentes ou gerados usando números aleatórios. A partir da criação do modelo, deve-se escolher o método de análise a ser utilizado: ferramentas analíticas ou simulação.

A sequência desta revisão está direcionada para a Técnica de Simulação. Assim, o próximo passo consiste em transformar o modelo obtido em um programa computacional.

A criação do programa de simulação depende do sistema sendo modelado e da disponibilidade de ferramentas. Basicamente, esses programas podem ser implementados utilizando:

- . **linguagens de simulação:** uma linguagem direcionada a simulação possui todas as estruturas necessárias para a criação de um ambiente de simulação; nesse caso, uma série de ferramentas

específicas se encontram disponíveis, permitindo a simulação de um grande número de tipos de sistemas. Exemplos de linguagens de simulação são GPSS [BA89a], SIMULA [SA85b] e SLAM II [SO92].

- linguagens convencionais: nesse caso, qualquer linguagem de programação de propósito geral pode ser utilizada [SA90], como por exemplo C, Pascal e Occam [BU88, JO88]. Aqui, o programador é responsável pela criação de todo o ambiente necessário para a simulação. A vantagem desse esquema é que o modelador pode escolher a linguagem que melhor lhe convém [MA87]. Além disso, o programa de simulação criado pode ser executado em qualquer computador que tenha compilador para a linguagem escolhida. Uma desvantagem é que o programador tem que desenvolver todas as estruturas exigidas para um ambiente de simulação;
- pacotes de propósito específico: esses pacotes são conjuntos de programas criados para aplicações específicas; esses pacotes são muito úteis em simulações para as quais foram projetados, fornecendo ferramentas bastante adequadas. Entretanto, pouca flexibilidade a mudanças é fornecida, uma vez que são projetados para tipos específicos de sistemas. Exemplos de pacotes de propósito específico são: PET [SO92] (usado na modelagem de redes de computadores) e IDSS [SO92] (aplicados na modelagem de sistemas aeroespaciais);
- extensões funcionais de linguagens de uso geral: são pacotes com procedimentos especiais para simulação utilizados junto a linguagens de programação convencionais; dessa forma, tem-se os aspectos gerais de uma linguagem convencional de alto nível junto a ferramentas específicas de simulação. Entretanto, produzem códigos maiores e menos "legíveis" para os programas que as linguagens de simulação. Por exemplo, SMPL [MA87] é uma extensão da linguagem C.

A Etapa de Testes

Após a criação do programa de simulação, esse deve ser conferido para verificar a possível ocorrência de erros tanto de programação como de lógica [MA87]. Em seguida, deve ser verificado se o programa é uma implementação válida do modelo usado. Para modelos simples, a verificação pode ser feita por inspeção. Entretanto, no caso de modelos maiores, uma verificação

mais detalhada deve ser realizada; uma forma de se fazer isso é através da comparação com modelos analíticos [MA87, SH88].

O modelo de simulação (com seu programa associado) deve ser uma representação válida do sistema considerado. Isso significa que aquele deve reproduzir o comportamento do sistema com fidelidade, a fim de atingir os objetivos propostos. Assim, o modelo deve passar por um processo que recebe o nome de *validação*. Sargent, em [SA84], apresenta um série de discussões sobre técnicas de verificação e validação de modelos.

Basicamente, existem três formas de validar um modelo de simulação [SA84, SO92]:

- . comparar os resultados obtidos nas simulações com medidas colhidas junto ao sistema considerado (quando o sistema existe);
- . comparar os resultados das simulações com dados de outros modelos já validados, por exemplo modelos analíticos;
- . obter nas simulações resultados conhecidos ou previsíveis (por exemplo, quando considerando um projeto novo para o sistema modelado).

A partir da validação do modelo, o programa de simulação pode ser utilizado para a análise do sistema e para prever o possível comportamento do sistema quando considerando modificações na sua estrutura.

As Análises de Resultados

Para complementar processos de modelagem de sistemas deve-se verificar a precisão dos resultados obtidos nas simulações. A precisão dos resultados é extremamente importante para que o modelo possa refletir com fidelidade o comportamento do sistema considerado [SA84].

Com relação a precisão dos resultados das simulações, os aspectos descritos a seguir devem ser considerados.

Utilização de Números Aleatórios

Em geral ferramentas computacionais usadas para simulação de sistemas utilizam variáveis aleatórias e assim a estimativa do valor da característica observada depende da sequência dos números aleatórios utilizada durante a execução da simulação [BA89, SA84]. Uma nova execução da simulação (replicação) sobre o mesmo modelo usando outro conjunto de números aleatórios produzirá uma estimativa diferente para a característica considerada. A variância entre as estimativas obtidas a partir de replicações independentes pode ser usada para avaliar a precisão do modelo, uma vez que uma grande variabilidade coloca os resultados sob suspeita, implicando que o modelo não é apropriado para a aplicação ou que o número de replicações foi insuficiente [MA87].

Efeitos Transientes

No começo da execução de uma simulação podem ocorrer condições iniciais não representativas, como por exemplo a ausência de transações no sistema e recursos desocupados [BA89a]. Entretanto, ao passo que transações são inseridas, os recursos vão se tornando mais ocupados, até se obter as condições conhecidas como "estado permanente de um sistema". Isso significa que as condições iniciais das simulações podem influenciar na precisão dos resultados obtidos [BA89a]. Para reduzir os efeitos dessas condições iniciais sobre os resultados das simulações realizadas, pode-se utilizar a técnica chamada de *exclusão* ("warmup"), na qual a execução de uma simulação é dividida em duas partes: uma fase de inicialização, onde as estatísticas obtidas são descartadas, e uma fase de consideração dos dados, onde as estatísticas coletadas são usadas como estimativas para os valores das características de interesse.

Deve-se observar que, muitas vezes entidades ainda não servidas nas simulações podem estar sendo incluídas nas estatísticas colhidas; isso afeta a confiabilidade dos resultados obtidos. Cuidados devem ser tomados para que enganos, como do tipo citado, não sejam

cometidos. Portanto, as condições finais das simulações devem ser identificadas, para que seus possíveis efeitos inadequados sejam eliminados.

2.3 Considerações Finais

Uma revisão sobre os aspectos mais relevantes ao trabalho apresentado nesta tese foram descritos neste capítulo.

Na primeira parte da revisão, foi descrito que uma das características mais importantes de sistemas distribuídos de computadores é o compartilhamento de recursos ("hardware" e "software") pelos usuários do sistema; algumas das literaturas citadas são especialmente voltadas a esse contexto. Por exemplo, [CO94] apresenta uma extensa descrição sobre sistemas distribuídos, descrevendo sobre muitos conceitos pertinentes ao assunto (definições, vantagens, redes de computadores, sistemas de arquivamento, servidores, etc.); traz também uma discussão sobre vários sistemas distribuídos existentes. Em [TA85], uma série de tópicos especialmente relacionados a sistemas operacionais distribuídos são revistos; conceitos, definições e outros pontos são apresentados. Também são apresentadas discussões sobre alguns sistemas operacionais distribuídos implementados.

Dentre as características interessantes de um sistema distribuído de computadores, uma se destaca: o compartilhamento de recursos; isso pode proporcionar reduções razoáveis nos custos de implantação de sistemas computacionais distribuídos. Vários servidores podem estar disponíveis em um sistema distribuído; um servidor indispensável é o servidor de arquivos. Sendo assim, esse servidor tem merecido atenção especial de muitos pesquisadores.

Várias literaturas descrevem sobre sistemas de arquivamento (tanto a nível de "software" quanto de "hardware") em redes locais de computadores. Por exemplo, [LE90], [MI82] e [SV84] discutem as características de servidores de arquivos. Conceitos importantes (como por exemplo, desempenho, confiabilidade, consistência de informações e transações atômicas), além da descrição de projetos e implementações de servidores de arquivos, são revistos nesses trabalhos. Outros textos descrevem projetos e implementações de sistemas de

arquivamento que visam obter sistemas com alto desempenho e grande confiabilidade; técnicas como *multi-estruturação do sistema de arquivos* [MU91, CH91], *escritas sequenciais em disco* [HA92, OU89, RO91] e *vetor redundante de discos (RAID)* [GA94, HA92, MU91, PA88b] são exemplos de implementações.

No final dessa parte da revisão são apresentados vários exemplos de servidores de arquivos baseados em rede local de computadores (ou sistemas de arquivos associados).

A segunda parte da revisão é relativa a simulação de sistemas computacionais. Vários aspectos relativos a modelamento para simulação de sistemas, considerados relevantes ao trabalho apresentado nesta tese, são revistos.

A simulação foi usada neste trabalho como uma ferramenta auxiliar dentro do processo de análises de projetos alternativos para servidores de arquivos, visando a obtenção de sistemas com alto desempenho (objetivo deste trabalho). Essa técnica oferece grande flexibilidade a baixo custo, podendo ser utilizada na avaliação de sistemas existentes ou não. As alternativas de projetos para um sistema podem ser avaliadas a partir de modificações relativamente simples no programa de simulação associado. Tudo isso torna simulação uma técnica bastante adequada para atingir objetivos como o proposto para o trabalho desenvolvido.

CAPÍTULO TRÊS

ESTUDO DE CASO: TRICE

No capítulo anterior discutiu-se as características gerais de sistemas distribuídos e de servidores de arquivos. Neste capítulo é detalhado o servidor de arquivos do sistema distribuído TRICE. A apresentação desse servidor tem dois objetivos: discutir em detalhes um servidor de arquivos real e utilizar esse ambiente para o desenvolvimento de um sistema de simulação.

No item 3.1 são apresentados os aspectos principais do sistema TRICE, relevantes para a construção do sistema de simulação.

As características do servidor de arquivos modelado, a sua composição (descrevendo a função de cada componente) e a estrutura do sistema de arquivos são descritas no item 3.2.

No item 3.3 discute-se os fatores que influenciam no tempo de resposta do servidor de arquivos, tais como: características internas (processador, disco, "cache", etc) e demanda de serviço imposta sobre o servidor (número de clientes, tipos de solicitações, etc). No final, são apresentadas as distribuições estatísticas coletadas junto ao servidor de arquivos do TRICE.

3.1 O Sistema Distribuído TRICE

O ambiente utilizado para o estudo de caso é o sistema TRICE, desenvolvido no Departamento de Eletrônica e Ciência da Computação da Universidade de Southampton, Inglaterra.

O TRICE (Transportable Integrated Computing Environment) [SA90a, SA90b] é um sistema distribuído baseado em rede local, fracamente acoplado, e organizado de acordo com o modelo cliente-servidor. A estrutura do sistema TRICE consiste de várias redes locais, em forma de anel ("Cambridge Ring"), conectadas a uma rede mestre que segue uma topologia estrela. Cada rede local é composta por estações de trabalho e estações servidoras; por ser um ambiente integrado, diferentes tipos de estações podem fazer parte do TRICE. O "software" do sistema foi escrito em BCPL [RI80], uma linguagem direcionada para a implementação de sistemas.

Uma vez que o ambiente TRICE segue o modelo cliente-servidor [MI82, MU91, SV84, TA95], as estações dos usuários se comportam como clientes que solicitam serviços específicos aos servidores do sistema. O processo de comunicação entre clientes e servidores é feito através de chamadas remotas de procedimentos [BI84, SV84, TA95], implementado em protocolos que, basicamente, controlam a transferência de minipacotes (para trocas de informações) e de pacotes de dados. Na maioria dos casos, um mecanismo de dois passos é usado; assim, para cada solicitação que chega no servidor, uma resposta ou uma mensagem de reconhecimento é retornada para a respectiva estação de trabalho.

3.2 O Servidor de Arquivos do TRICE

O servidor de arquivos do TRICE é responsável pela organização e gerenciamento do sistema de arquivos do ambiente, providenciando o armazenamento e a recuperação de informações. Esse servidor é implementado sobre uma estação de trabalho dedicada.

3.2.1 As Características Gerais

Cada estação servidora de arquivos da versão considerada do TRICE é composta por um processador de entrada/saída (FEP), um processador principal, um sistema de discos, uma interface para acesso direto a memória (DMA) e memória RAM. A figura 3.1 mostra o diagrama descrevendo a estrutura básica do servidor de arquivos.

Figura 3.1 - Estrutura Básica do Servidor de Arquivos do TRICE.

O FEP é responsável pela transferência de informações entre a rede e o servidor. Todos os pacotes recebidos da rede são colocados pela interface DMA (que é controlada pelo FEP) numa memória compartilhada com o processador principal. As informações necessárias para a localização dos dados nessa memória são escritas pelo FEP em outra memória compartilhada, um conjunto de RAMs porto-dual. O FEP envia um sinal de interrupção para o processador principal, para que este execute a tarefa solicitada. Quando dados devem ser enviados para a rede (após a execução de uma tarefa pelo processador principal), os dados são escritos na memória compartilhada e na memória porto-dual são escritas as informações necessárias para a localização

dos dados. Em seguida, um sinal de interrupção é enviado para o FEP, para que este gerencie a transferência dos dados para a rede. Dados coletados junto ao sistema TRICE mostraram que mais de 99% dos pacotes transmitidos ao servidor de arquivos foram recebidos com sucesso. Isso demonstra que o FEP executa suas funções de forma bastante eficaz e que este não representa um fator limitante do desempenho do servidor de arquivos.

O processador principal controla todas as tarefas dentro do servidor de arquivos. Esse processador responde às interrupções geradas pelo FEP e pelo sistema de disco, e tem a responsabilidade de gerenciar diretórios, providenciar informações relativas a arquivos e diretórios, fazer procura de dados no "cache" e solicitar acessos ao sistema de discos. Além disso, atende interrupções geradas pelo subsistema de relógio, para controlar o tempo dentro do servidor.

O sistema de discos é composto por dois discos rígidos utilizando interface padrão SCSI. Um dos discos é utilizado para operações normais de acesso e o outro é usado como dispositivo para manter cópias de arquivos (a geração do disco-cópia é realizada todas as madrugadas); esse esquema proporciona maior segurança aos dados armazenados. O sistema de discos responde a solicitações feitas pelo processador principal e ao final de cada operação, esse sistema gera um sinal de interrupção para o processador principal.

Cada um dos dois recursos anteriores (o processador principal e o sistema de discos) possui uma estrutura que usa o mecanismo de fila (primeiro a entrar, primeiro a sair). Quando o recurso a ser utilizado não está disponível, a solicitação espera na fila pela liberação do mesmo.

A memória principal do servidor de arquivos é usada para manter o "software" que controla o servidor e implementar um sistema de "cache". O "software" do servidor é escrito em BCPL, sendo que o subprograma que controla o processador é estruturado como uma coleção de co-rotinas [AN83, KI91, SN92] que garante a estrutura para a execução concorrente de processos neste servidor. O TRICE é um sistema baseado em sessões; isso significa que, antes de um usuário iniciar seu trabalho, ele deve estabelecer uma sessão de trabalho criando um caminho entre a sua estação e um servidor de arquivos. Para cada caminho criado, uma co-rotina é inicializada.

Parte da memória principal é utilizada para manter o "cache" de disco implementado no servidor de arquivos. Esse "cache", destinado somente a operações de leitura de dados, aumenta o desempenho do servidor, pois reduz o número de acessos ao disco, diminuindo o tempo médio de resposta do sistema. Três esquemas de "cache" foram implementados nesse servidor: para arquivos do sistema (armazena os mais usados), para diretórios (diretórios do sistema e de usuários, até que sejam modificados) e para trilha do disco (armazena trilhas lidas contendo blocos de arquivos ligados e de diretórios). Quanto a escrita, esse "cache" segue o mecanismo de *escrita imediata*, isto é, quando informações armazenadas no "cache" são modificadas, essas são escritas imediatamente no disco e os dados do "cache" são invalidados. A taxa total de acerto obtida no sistema TRICE com esses esquemas de "cache" é de aproximadamente 70%.

3.2.2 A Estrutura do Sistema de Arquivos

O sistema de arquivos do TRICE é estruturado como uma árvore de subdiretórios e de arquivos, sendo o diretório raiz dessa árvore composto por arquivos de comandos do servidor de arquivos, diretórios dos usuários e de subdiretórios do sistema. Esse sistema de arquivos permite trabalhar com arquivos ligados e contíguos.

Os arquivos ligados são compostos por blocos de dados interligados que podem ser armazenados em qualquer parte do disco, sem a preocupação com a sequencialidade física dos mesmos. Cada bloco tem o tamanho de 512 bytes, sendo os 8 primeiros reservados para informações relativas ao número de bytes válidos dentro do bloco e às ligações desse bloco com o seu antecessor e com o seu sucessor; o campo com os 504 bytes restantes são utilizados para o armazenamento dos dados. Para a leitura ou a escrita de arquivos ligados, as estações de trabalho devem emitir uma solicitação de acesso ao servidor de arquivos para a transferência de cada bloco de dados

Os diretórios, também formados por blocos ligados, são organizados de forma similar aos arquivos ligados, exceto que as referências a arquivos e subdiretórios são armazenadas

como entradas nos campos de dados de cada bloco, trazendo informações relativas a esses arquivos e subdiretórios (tais como nome e endereço do primeiro bloco).

Os arquivos contíguos são igualmente estruturados em blocos. Esse tipo de arquivo necessita que seus blocos sejam armazenados de forma consecutiva no disco; isso dispensa a utilização de campos relativos a informações de ligação com outros blocos (possibilitando o armazenamento de até 512 bytes de dados por bloco) e permite a realização de leitura e de escrita de arquivos em uma operação única e rápida. Entretanto, surgem restrições quanto a alocação do arquivo em disco, devido a necessidade de manter os blocos armazenados de forma consecutiva.

O sistema de arquivos é praticamente todo implementado no servidor de arquivos (apenas uma pequena parte foi implementada nas máquinas clientes).

3.3 Caracterização do Tempo de Resposta

O acesso a arquivos no TRICE é semelhante ao da maioria dos servidores que utilizam o esquema cliente-servidor. Cada estação envia uma solicitação e aguarda pela resposta (na forma de mensagem ou de dados) do servidor. Após processar a solicitação, o servidor envia uma resposta para a respectiva estação cliente.

O tempo gasto para responder a cada solicitação gerada por um cliente depende de dois fatores: as características do servidor de arquivos (tipos de processadores e de disco utilizados, estratégias de "cache" de disco implementadas, disposição dos dados em disco, etc.) e da quantidade de serviço exigida desse servidor (número de usuários conectados ao servidor, tempo entre solicitações de cada usuário e tipos dessas solicitações). Esses fatores serão discutidos nas próximas seções.

3.3.1 As Características Internas do Servidor de Arquivos

As características internas do servidor de arquivos apresentam grande influência sobre o desempenho do mesmo. A capacidade de processamento de um servidor de arquivos está diretamente relacionada ao desempenho dos seus recursos: o processador de entrada/saída, o

processador principal e o sistema de discos (as características principais desses recursos, bem como as funções desempenhadas por estes, foram apresentadas na seção 3.2.1). Entretanto, como descrito naquela seção, o processador de entrada/saída normalmente não representa um fator limitante ao desempenho do servidor de arquivos. Assim, a capacidade de processamento desse servidor fica condicionada ao desempenho do processador e do sistema de discos.

As estratégias de "cache" utilizadas no TRICE apresentam grande impacto sobre o tempo de resposta do servidor, pois o "cache" implementado proporciona uma taxa de acerto total relativamente alta, o que faz com que muitos acessos a disco sejam evitados, reduzindo bastante o tempo médio de permanência das solicitações no servidor.

3.3.2 A Caracterização da Demanda de Serviço

Outro fator que tem grande influência sobre o tempo de resposta do servidor de arquivos é a quantidade de serviço imposta a este servidor pelas estações da rede.

O tempo que o servidor gasta para responder a um pedido pode variar conforme as condições do ambiente da rede. Por exemplo, dados obtidos junto ao sistema TRICE mostram que o tempo médio que um servidor de arquivos gasta para responder a cada solicitação cresce quando o número de usuários utilizando esse servidor (e consequentemente a demanda de serviço) aumenta. A figura 3.2 mostra os tempos médios de resposta daquele servidor de arquivos, em milissegundos (ms), quando diferentes números de usuários estão conectados a esse servidor.

Figura 3.2 - Tempo Médio de Resposta do Servidor de Arquivos.

Para a determinação da quantidade de serviço exigida do servidor de arquivos pelas estações de trabalho é necessário caracterizar a carga de trabalho imposta ao servidor por estação isoladamente. Essa carga de trabalho individual pode ser caracterizada pelo intervalo de tempo entre solicitações enviadas ao servidor e pelo tipo da operação solicitada pela estação.

Tempo Entre Solicitações

A composição básica do tempo entre as requisições solicitadas ao servidor de arquivos por um usuário é representada na figura 3.3.

Figura 3.3 - Tempo Entre Solicitações de Uma Estação de Trabalho.

Após cada solicitação, a estação de trabalho aguarda pela resposta do servidor de arquivos, sendo que uma nova solicitação só é feita (por aquela estação) após a chegada da resposta da solicitação anterior. Assim, os intervalos entre as solicitações feitas por um usuário dependem das atividades sendo desenvolvidas na estação de trabalho (t_{et} , na figura 3.3), do tempo gasto na comunicação entre a estação e o servidor (t_c) e do tempo de serviço no servidor de arquivos (t_{sa}).

O tempo das atividades dentro da estação de trabalho depende do tipo da aplicação sendo executada, da capacidade de processamento da estação e da ação do usuário. Com relação ao tempo de comunicação entre a estação cliente e o servidor de arquivos e ao tempo de execução no servidor, esses dependem de vários fatores, sendo bastante influenciados pela carga de trabalho imposta pelo sistema global.

Salienta-se que a carga total do sistema tem influência sobre o tempo entre solicitações feitas por uma estação de trabalho através dos tempos de comunicação e de serviço no servidor de arquivos. Entretanto, através da caracterização da carga de trabalho imposta ao servidor de arquivos por estação de forma isolada (sem que sofra qualquer influência da carga total do sistema) pode-se ativar várias instâncias da carga de uma estação de trabalho e simular o sistema como um todo.

Para obter a carga de trabalho individual de cada estação independente do sistema total, considera-se que a complementação de uma solicitação só ocorre após a chegada da resposta do servidor de arquivos na estação cliente. Com isso, o tempo entre solicitações é determinado a partir da chegada da resposta na estação de trabalho até o envio da próxima solicitação ao servidor (t_{et} na figura 3.4). Assim, o tempo entre solicitações passa a ser considerado dependente apenas do tempo gasto dentro da estação de trabalho.

Figura 3.4 - Caracterização dos Tempos entre Solicitações.

O tempo de atividades dentro da estação de trabalho pode ser longo se a estação estiver processando um grande programa ou estiver esperando uma ação a ser realizada pelo usuário. Mas, o tempo entre solicitações pode ser relativamente curto quando a estação estiver executando operações que necessitam de vários acessos ao servidor de arquivos (por exemplo, a leitura ou a escrita de um arquivo ligado). Neste caso, as solicitações são intercaladas por processamentos na estação cliente de curta duração e assim o tempo de atividades é dependente apenas da capacidade de processamento da estação.

Esse comportamento da estação cliente, onde várias solicitações são feitas dentro de um pequeno período de tempo seguido por um intervalo de tempo relativamente longo sem a ocorrência de solicitações, apresenta uma característica de tráfego de solicitações em blocos ("burst") pela rede local (observada por Shoch e Hupp em [SH80]). A figura 3.5 mostra a característica em que solicitações são feitas em blocos por uma estação de trabalho ao servidor de arquivos, descrevendo o tempo entre esses blocos (t_{bloco}), quando o tempo entre solicitações (t_{et}) é relativamente longo.

Figura 3.5 - Blocos de Solicitações de Uma Estação de Trabalho.

Os Tipos de Solicitações

O servidor de arquivos processa dois tipos básicos de solicitação: operações de controle e operações em dados. As operações de controle correspondem a solicitações que não requerem trocas de dados entre a estação cliente e o servidor; apenas informações para gerenciar arquivos e diretórios são passadas. Exemplo de operações de controle são: abrir e fechar arquivos, eliminar arquivos, verificar privilégio de acesso a arquivo, criar e eliminar diretórios.

As operações com arquivos são utilizadas para a transferência de dados de arquivos; a partir dessas operações, arquivos inteiros, ou parte destes, podem ser transferidos entre as estações de trabalho e o servidor. Dois tipos de arquivos podem ser tratados pelo servidor de arquivos de referência: arquivos ligados e arquivos contíguos. Quando uma estação de trabalho

deseja a transferência de um arquivo ligado, várias solicitações devem ser feitas pela estação ao servidor (cada solicitação correspondendo a transferência de um bloco do arquivo). Para a leitura ou escrita de um arquivo contíguo, uma única solicitação é feita pela estação para a transferência do arquivo inteiro.

As solicitações que chegam ao servidor de arquivos são inseridas na fila do processador principal, que quando disponível inicia a execução da mesma. Se o processador necessita de alguma informação contida no "cache", o acesso ao "cache" é realizado e o processamento da solicitação continua. Se acesso a disco se torna necessário, o processamento da solicitação é interrompido e a solicitação é colocada na fila do disco; quando este estiver livre, o acesso é executado e a solicitação é novamente inserida na fila do processador. Ao terminar o processamento da solicitação, o servidor de arquivos envia uma resposta ao usuário através da rede. O número de ciclos (alternando acessos ao processador e ao disco) depende do tipo da solicitação sendo executada.

A execução de cada tipo de solicitação segue o algoritmo básico descrito no parágrafo anterior com características distintas, tais como: tempo de execução no processador, taxa de acerto no "cache" e tipos de acessos ao disco (para ler um bloco de dados de um arquivo, para escrever um arquivo inteiro, etc.). O algoritmo utilizado, juntamente com outros aspectos do sistema (como número de blocos do arquivo ou do diretório, se a informação solicitada está em "cache", etc.), define o número de ciclos de acessos ao processador e ao disco do servidor.

Os tipos de solicitação mais usados pelas estações de trabalho são apresentados na tabela 3.1, que descreve também as formas de acesso ao disco (ou ao "cache") que devem ser realizadas pelo servidor de arquivos.

Tipo de Solicitação	Formas de Acesso
escrita ligada	B
fim de escrita	B <u>B</u> B
leitura ligada	<u>B</u>
escrita contígua	C
leitura contígua	<u>C</u>
fim de leitura	-
criar arquivo	D <u>D</u> P B
eliminar arquivo	<u>D</u> L <u>B</u> B
abrir arquivo	<u>D</u>
fechar arquivo	-
previlégio	<u>D</u> B
renomear	<u>D</u> <u>D</u> B B P B
criar diretório	<u>D</u> P B B
eliminar diretório	P <u>B</u> B B L B
mudar diretório	P D
informações	-
eliminar entrada	L <u>B</u> B
eliminar saída	B L <u>B</u> B
retroceder	-
cota	P
finalizar entrada	-

Tabela 3.1 - Composição dos Tipos de Solicitações

As formas de acesso ao disco, que podem ser necessárias para a execução de solicitações pelo servidor de arquivos, identificadas na tabela 3.1 são:

- . B - acesso único ao disco para a leitura ou a escrita de um bloco de dados;
- . C - acesso(s) ao disco para a leitura ou a escrita de um arquivo contíguo (dependendo do tamanho do arquivo, vários acessos podem ser necessários);
- . L - acesso(s) ao disco para a leitura ou a escrita de um arquivo ligado inteiro;
- . D - acesso(s) ao disco para a leitura de um diretório inteiro;
- . P - procura de uma entrada específica no diretório.

Quando a informação desejada para leitura pode ser encontrada tanto em disco como no "cache" do servidor de arquivos, a abreviação da forma de acesso ao disco está sublinhada. Para cada uma dessas formas, se mais de um acesso a disco for necessário, estes acessos serão intercalados por execuções no processador.

3.4 Os Dados Coletados Junto ao TRICE

Nos itens anteriores foram descritas as características principais do sistema distribuído TRICE e dos seus servidores de arquivos. Nesta seção, são apresentadas informações do sistema TRICE que complementam a caracterização desse ambiente.

As informações apresentadas a seguir foram obtidas a partir de [SA90a], que traz uma série de dados estatísticos coletados junto ao sistema TRICE. Esses dados estatísticos foram colhidos em estações de trabalho distribuídas pela rede local e em estações servidoras de arquivos.

O modelo do servidor de arquivos foi construído a partir da análise desses dados, que foram utilizados na implementação do programa de simulação correspondente (o modelo do servidor e o programa de simulação são descritos no capítulo quatro).

Tempo Entre Solicitações

A demanda total de serviço sobre um servidor de arquivos do sistema depende da carga de trabalho imposta individualmente pelas estações clientes conectadas ao servidor. Essa carga individual é dependente do tempo entre as solicitações de cada estação enviadas ao servidor, sendo que esse tempo é influenciado pela carga total do sistema.

O critério estipulado na seção 3.3.2 para tornar a carga de serviço de cada estação de trabalho independente do sistema global considera que o tempo entre requisições deve ser determinado a partir da chegada da resposta da última solicitação feita pela estação até o envio da próxima requisição ao servidor de arquivos (observar a figura 3.4). Dessa forma, o tempo entre solicitações torna-se dependente apenas do tempo gasto dentro da estação de trabalho; esse tempo é influenciado pelo tipo de aplicação sendo executada, pela capacidade de processamento da estação

e pela ação do usuário (esse critério foi utilizado em [SA90a] para obter os tempos entre solicitações de uma estação de trabalho do sistema TRICE).

A figura 3.6 apresenta a distribuição dos tempos entre solicitações feitas por estação de trabalho; as estatísticas mostradas consideram intervalos de tempo entre requisições de no máximo três segundos.

Figura 3.6 - Distribuição dos Tempos Entre Solicitações.

O que se observa na figura 3.6 é um grande número de ocorrências de valores baixos para os intervalos de tempo entre solicitações e baixa frequência para intervalos maiores. Isso significa que uma grande quantidade de requisições são feitas com curtos períodos de tempo entre estas, caracterizando que as solicitações de cada estação de trabalho são feitas em blocos intercalados por intervalos de tempo relativamente longos sem solicitações.

As distribuições completas dos intervalos de tempo entre solicitações por estação de trabalho, que foram usadas para a implementação do programa de simulação, são apresentadas nas figuras A.1 e A.2 do apêndice A.

Tipos de Solicitações

A carga de trabalho imposta por estação cliente ao servidor de arquivos também é influenciada pelos tipos das operações solicitadas. Dependendo dos tipos das requisições, sequências diferentes de suboperações são executadas pelo processador principal do servidor, com variadas formas de acessos ao sistema de disco. Tudo isso se traduz em diferentes tempos para o servidor responder às solicitações.

O servidor de arquivos pode executar os seguintes tipos de solicitação:

- . operações de controle (sem dados): englobando as requisições das linhas 6 a 21 da tabela 3.1;
- . operações com arquivos de dados:
 - escrever um arquivo ligado: requisições nas linhas 1 e 2 da tabela 3.1;
 - ler um arquivo ligado: requisição da linha 3;
 - escrever um arquivo contíguo: requisição da linha 4;
 - ler um arquivo contíguo: requisição da linha 5.

Nas operações de controle, o servidor de arquivos gerencia arquivos e diretórios. Quando executando requisições com arquivos, o servidor escreve e lê arquivos ligados ou contíguos.

Dados coletados junto ao TRICE mostram que alguns tipos de solicitações são mais requisitados que outros. A figura 3.7 apresenta a distribuição dos tipos de requisições feitas pelas estações clientes ao servidor de arquivos.

Figura 3.7 - Distribuição dos Tipos de Solicitações.

Pela figura 3.7, observa-se que a maioria das solicitações (51%) que são enviadas ao servidor de arquivos requer a leitura de blocos de arquivos ligados. Isso acontece porque, para a transferência de um arquivo ligado, várias requisições são feitas pela estação cliente (uma solicitação para cada bloco) ao servidor e porque o número de requisições para ler arquivos ligados é relativamente alto. Deve-se observar também que são raras as solicitações que requerem a escrita de um arquivo contíguo.

As operações que não requerem transferências de dados representam aproximadamente 31% das solicitações recebidas pelo servidor de arquivos, influenciando significativamente na quantidade de serviço exigida desse servidor. Vários tipos de requisições de controle são executadas pelo servidor de arquivos, cada um necessitando de tempos diferentes de processamento. Para a caracterização dos tipos de solicitações das estações de trabalho torna-se necessário especificar as porcentagens de cada tipo de operação de controle. As dezesseis solicitações de controle mais usadas pelas estações de trabalho do sistema TRICE (encontradas na tabela 3.1) são consideradas nesta tese, destacando-se as operações para abertura e para fechamento de arquivos (cada uma representando em torno de 31% do total de solicitações de controle) e para finalizar leitura de arquivo (28% das solicitações de controle).

As distribuições complementares que mostram as porcentagens de utilização dos tipos de solicitações de controle mais usados pelas estações de trabalho do sistema TRICE são apresentadas na tabela A.1 do apêndice A.

Tempos de Execução das Solicitações

Cada tipo de solicitação requer uma sequência diferente de suboperações que devem ser executadas pelo processador principal do servidor de arquivos. Entre duas suboperações, um (ou mais) acesso ao disco ou ao "cache" é requisitado; se mais de um acesso é necessário (devido ao tamanho do arquivo ou do diretório), os acessos são intercalados por execuções da solicitação no processador.

Os tempos de execução de cada uma das suboperações da sequência considerada e os relativos aos processamentos que ocorrem entre acessos ao disco ou ao "cache" foram medidos junto ao servidor de arquivos do sistema TRICE para os tipos de solicitações mais utilizados e são mostrados na tabela 3.2. O tempo de processamento de cada suboperação da sequência é representado por t_i , o parâmetro t_{bi} representa o tempo de execução no processador entre acessos ao disco ou ao "cache" se dois ou mais acessos a esses recursos são necessários entre suboperações.

Tipo de Solicitação	t1	tb1	t2	tb2	t3	tb3	t4	tb4	t5	tb5	t6	tb6	t7
escrita ligada	2,185	0,000	3,701										
fim de escrita	1,585	0,000	1,046	0,000	1,121	0,000	3,775						
leitura ligada	1,354	0,000	3,761										
escrita contígua	1,424	1,693	3,740										
leitura contígua	1,500	1,693	3,713										
fim de leitura	4,061												
criar arquivo	7,926	3,673	2,870	6,095	22,991	4,409	3,279	0,000	5,927				
eliminar arquivo	5,580	6,095	7,531	1,422	3,895	0,000	1,058	0,000	7,263				
abrir arquivo	3,936	6,095	17,388										
fechar arquivo	7,511												
previlégio	12,210	6,095	7,779	0,000	5,571								
renomear	9,420	6,095	8,743	6,095	12,858	0,000	1,094	0,000	4,038	4,409	1,396	0,000	7,641
criar diretório	8,352	6,095	8,719	4,409	3,717	0,000	4,858	0,000	3,706				
eliminar diretório	11,795	6,095	9,455	0,000	1,092	0,000	2,007	0,000	1,370	1,422	1,300	0,000	4,309
mudar diretório	6,156	6,095	15,540	3,673	10,453								
informações	4,538												
eliminar entrada	2,358	1,422	3,798	0,000	1,069	0,000	4,313						
eliminar saída	1,374	0,000	1,653	1,422	3,781	0,000	1,118	0,000	4,309				
retroceder	4,046												
cota	1,888	6,095	9,383										
fim de entrada	15,777												

Tabela 3.2 - Composição dos Tempos das Solicitações (em milissegundos).

O tempo que o servidor de arquivos leva para responder a uma solicitação é fortemente influenciado pelo número de acessos ao disco (ou ao "cache") necessário para o processamento da solicitação (esse número determina a quantidade de ciclos de acessos ao processador e ao disco). Assim, para complementar a caracterização dos tempos de execução das requisições pelo servidor de arquivos é necessário conhecer as distribuições estatísticas relativas aos tamanhos dos arquivos (ligados e contíguos) e ao número de blocos necessários para armazenar os diretórios que compõem o sistema de arquivos do servidor.

Dados colhidos junto ao servidor de arquivos de referência mostram que a maioria das estruturas usadas para armazenar arquivos e diretórios é composta por poucos blocos de dados

(77% dos arquivos ligados e 73% dos arquivos contíguos são menores que 5 Kbytes, e quase 58% dos diretórios são mantidos em apenas um bloco). A figura 3.8 apresenta dados estatísticos relativos ao número de blocos necessários para manter diretórios, obtidos junto ao servidor de arquivos do TRICE.

Figura 3.8 - Distribuição do Número de Blocos dos Diretórios.

As distribuições estatísticas completas para os tamanhos de arquivos e diretórios são apresentadas nas figuras A.4, A.5 e A.6 do apêndice A.

3.5 Considerações Finais

Um sistema distribuído de computação baseado em rede local, denominado TRICE, utilizado para um estudo de caso e para o desenvolvimento do sistema de simulação (descrito no capítulo quatro) é discutido neste capítulo. Este sistema, organizado seguindo o modelo cliente-servidor, é composto por centenas de estações de trabalho que solicitam serviços a estações

servidoras (dentre estas, servidores de arquivos) distribuídas pela rede. Uma descrição mais detalhada de todo o sistema TRICE pode ser encontrada em [SA90a, SA90b].

O uso desse ambiente como referência se justifica pelo fato do seu servidor de arquivos apresentar características bastante satisfatórias para o desenvolvimento de pesquisas direcionadas a construção de servidores de arquivos com alto desempenho (o objetivo desta tese) e também devido a disponibilidade de farto material [SA90a, SA90b]. Além da descrição das características principais do sistema TRICE, necessárias para o desenvolvimento do modelo criado, em [SA90a] é encontrada uma série de dados estatísticos, obtidos junto às estações clientes e servidoras que compõem aquele sistema, que foram extremamente úteis para a implementação do programa de simulação desenvolvido.

Parte dessas distribuições estatísticas foram colhidas junto às estações de trabalho, através de monitor de "software", como as relativas aos tipos de solicitações enviadas ao servidor de arquivos e as relativas aos tempos entre solicitações por estação cliente. Outras distribuições foram obtidas nos próprios servidores de arquivos, como as de tamanhos de arquivos e diretórios usados no sistema. Os dados relativos ao servidor foram obtidos através de inspeção (tamanho de diretório, etc.) ou através de um monitor de "software" instalado no servidor (tempo de processamento, número de acessos ao disco, etc.) Esses dados foram obtidos junto às estações e aos servidores durante várias semanas de utilização normal do sistema [SA90a].

Outros tipos de dados, colhidos nos servidores de arquivos, foram utilizados nas validações do modelo e do programa de simulação desenvolvidos, como por exemplo, os relativos aos tempos de resposta do servidor e número de acessos a disco.

A partir dos dados obtidos, observou-se que 83% das operações envolvendo acessos a disco estão relacionadas com a leitura de dados. Outra informação importante observada é quanto as técnicas de "cache" de disco implementadas no servidor de arquivos. Os esquemas de "cache" envolvidos proporcionam uma taxa total de acerto em torno de 70%, o que significa uma redução acentuada no número necessário de acessos a disco. Isso aumenta o desempenho geral do servidor,

pois o tempo médio de acesso a disco é da ordem de 30 milissegundos, um tempo bastante superior ao gasto na obtenção de dados em memória "cache".

CAPÍTULO QUATRO

O SIMULADOR DO SERVIDOR DE ARQUIVOS

Este capítulo apresenta o sistema de simulação desenvolvido para a avaliação de desempenho de um servidor de arquivos baseado em rede local. Como referência foi utilizado o servidor de arquivos do sistema distribuído TRICE.

Inicialmente, são apresentadas as características de interesse do servidor de referência, necessárias para a criação do modelo. A seguir, no item 4.2, é descrito o sistema de simulação desenvolvido: o modelo e o programa associado. No item 4.3, o modelo e o programa de simulação são validados utilizando diferentes técnicas.

Para que o sistema de simulação criado pudesse ser usado nas análises de desempenho de servidores de arquivos, esse sistema recebeu algumas adequações. A adequação do sistema de simulação é descrita no item 4.4.

4.1 O Servidor de Arquivos de Referência

Num processo de modelagem, necessita-se de informações suficientes sobre o ambiente de referência, identificando e caracterizando os aspectos de interesse, para que o modelo a ser desenvolvido possa refletir de forma aceitável o comportamento do sistema considerado.

Para a criação do modelo de simulação apresentado neste capítulo foi utilizado o ambiente do sistema distribuído TRICE. A estrutura de referência do sistema TRICE é composta por uma série de servidores (servidores de arquivos, de nome, de impressão, etc), podendo suportar algumas centenas de estações de trabalho de diferentes tipos, muitas delas sem disco, que acessam arquivos em servidores remotos através da rede local. A operação geral do sistema pode ser resumida como um ciclo de solicitações feitas pelas estações de trabalho e de respostas enviadas pelos servidores de arquivos, envolvendo operações de controle e de transferência de dados.

Os servidores de arquivos são estações de trabalho compostas basicamente por um processador principal (para o processamento das solicitações), um processador de entrada/saída - FEP (responsável pela comunicação entre o servidor e a rede) e um sistema de discos rígidos. Um "cache" somente para leitura foi implementado no servidor a fim de reduzir o número de acessos a disco, proporcionando uma taxa de acerto total da ordem de 70%. De uma maneira bastante simplificada, a operação geral de um servidor de arquivos pode ser descrita através dos seguintes passos:

- . cada solicitação é recebida pelo FEP e colocada na fila do processador principal;
- . o processador inicia a execução da solicitação;
- . se um acesso a disco é necessário, a solicitação é inserida na fila desse dispositivo, permanecendo ali até que o disco esteja disponível;
- . após o acesso a disco, a solicitação é novamente colocada na fila do processador.

Quando a execução da solicitação é finalizada, a resposta correspondente é liberada para o FEP, que a envia de volta para a rede. A execução de cada solicitação compreende uma sequência alternada de operações no processador e em disco.

Informações mais detalhadas sobre o sistema TRICE (estrutura e características principais) são apresentadas no capítulo três.

4.2 O Sistema de Simulação

O programa de simulação foi implementado visando caracterizar o comportamento do servidor de arquivos do sistema TRICE. Para a construção do modelo utilizado foram levadas em consideração as características relevantes do sistema real.

4.2.1 O Modelo do Servidor de Arquivos

O modelo criado representa de forma simplificada as funções principais do servidor de arquivos, modelando o processador, o disco e o "cache" de disco, e considera a chegada de solicitações no servidor, o processamento destas (permanecendo em filas quando os recursos necessários estão ocupados) e o envio das respostas para a rede. A execução de cada solicitação pode requerer de um a vários acessos ao processador, e de nenhum a vários acessos a disco (algumas operações de controle necessitam apenas de um acesso ao processador e de nenhum acesso a disco - por exemplo, a operação de fechamento de um arquivo). O diagrama do modelo desenvolvido é apresentado na figura 4.1, demonstrando a alternância na utilização do processador e do disco para o processamento das solicitações.

Figura 4.1 - Diagrama do Modelo do Servidor de Arquivos.

Como observado no capítulo anterior, o processador FEP do servidor de arquivos (cuja função principal é gerenciar as transferências de pacotes de dados entre a rede e o servidor) executa suas funções com extrema eficiência, não provocando atrasos para o processamento das solicitações no servidor. Sendo assim, esse processador não está representado diretamente no modelo criado (no programa de simulação, suas funções são representadas por comandos que providenciam a introdução e a retirada de transações do sistema).

4.2.2 O Programa de Simulação

Dentre as técnicas disponíveis para a implementação do modelo proposto para o servidor de arquivos, simulação foi a técnica escolhida por oferecer flexibilidade quanto a alterações no modelo, facilidade de uso, precisão dos resultados e baixos custos de implantação.

O modelo descrito na figura 4.1 foi utilizado para a implementação do programa de simulação do servidor de arquivos, adotando-se orientação a processos [MA87, SA85b]. Para a implementação desse programa foi utilizado o sistema GPSS/H [BA89a], que fornece um grande número de ferramentas direcionadas para a simulação de sistemas.

No GPSS/H, um sistema é simulado através de entidades que se movem através do mesmo. Essas entidades dinâmicas, chamadas *transações*, se deslocam através de *blocos* padrões,

que representam ações sobre as transações. Outro elemento básico do GPSS/H é o *recurso*, usado pelas transações para executar tarefas específicas. As transações são "atrasadas" quando são "servidas" em um recurso ou quando esperam pela liberação deste, e se movem pelo sistema até serem completadas ou até que a simulação seja interrompida.

O GPSS/H possui um relógio interno que mantém o tempo total simulado; esse relógio controla toda a movimentação de uma transação, desde a introdução da mesma até a sua retirada do sistema.

A estrutura básica do programa de simulação segue o algoritmo descrito a seguir:

começo

 Inicialização das estruturas de dados gerais;

 Enquanto *simulando* fazer

 Chegada de solicitações no servidor de arquivos;

 Determinação do número de acessos a disco para cada solicitação;

 { início de processamento da solicitação }

 Execução da solicitação no processador;

 Se *necessita de acesso a disco*

 Então Se *leitura*

 Então Se *dado em "cache"*

 Então ler "cache"

 Senão ler disco;

 Senão escrever em disco;

 Retornar para execução no processador;

 Senão retirar solicitação do servidor de arquivos;

 { fim de processamento da solicitação }

fim.

No programa simulador, transações representam as solicitações que chegam ao sistema e recursos representam os dispositivos envolvidos (no caso, processador e disco). A chegada de cada solicitação no servidor de arquivos é simulada pela introdução de uma transação no modelo (através do comando *generate* do GPSS/H) e o envio da resposta para a rede é simulada pela retirada da transação do sistema (através do comando *terminate* do GPSS/H).

A entrada de transações no modelo é feita de acordo com a carga de trabalho imposta sobre o servidor; para a caracterização dessa carga, dois outros tipos de entidades do GPSS/H são usadas no programa. O primeiro é o tipo *variável*: uma variável (denominada Nest no programa) é usada para fixar o número de estações de trabalho consideradas na corrente simulação e outra variável (Vest) é usada para designar a velocidade das estações de trabalho ligadas a rede (o valor do parâmetro Vest é determinado em relação as estações que compunham originalmente o sistema TRICE, o MC68008). O outro tipo de entidade utilizada é uma *função* (Tsol), que determina aleatoriamente o intervalo de tempo entre solicitações para cada estação de trabalho do sistema. A função Tsol foi construída a partir das distribuições dos tempos entre solicitações dentro de um pacote e dos tempos entre esses pacotes para cada estação de trabalho; essas distribuições são apresentadas nas figuras A.1 e A.2 do apêndice A.

Para a determinação do número de acessos ao disco (e, conseqüentemente, ao processador), exigido em cada solicitação, uma nova função é usada (Naces); essa função foi construída a partir das distribuições dos tamanhos de arquivos e de diretórios, apresentadas nas figuras A.4, A.5 e A.6 do apêndice A. O valor produzido pela função Naces corresponde ao número de acessos que devem ser feitos sobre o disco e/ou ao "cache" de disco.

Os tempos de serviço no processador e no disco são obtidos de formas distintas. Para o processador é usada uma função (Tproc) que gera valores aleatórios seguindo a distribuição dos tipos de solicitações (figura 3.7 e tabela A.3 do apêndice A) que podem ser feitos pelas estações ao servidor de arquivos e a partir da tabela dos tempos (tabela 3.2 do capítulo três) exigidos no processador (em relação ao MC68020, o processador original do servidor) para a execução das primitivas que compõem cada solicitação. Uma variável (Vproc), que designa a

velocidade do processador do servidor de arquivos, é também usada no cálculo do tempo de execução nesse recurso (o valor do parâmetro *Vproc* é determinado em relação ao MC68020).

No caso do disco, o tempo de serviço é encontrado de maneira mais simples: é utilizado o tempo médio de acesso a dados (tempo de busca do cilindro mais o tempo de latência) especificado comercialmente para o disco considerado (no caso, o disco original do servidor de arquivos tem especificação para o tempo médio de acesso da ordem de 29 milissegundos) e o tempo para a transferência de um bloco de 512 bytes (para o disco original, esse tempo é de 0,341 ms). Dessa forma, o tempo de execução relativo ao disco é fácil e rapidamente determinado para qualquer disco a ser considerado, sendo o seu valor atribuído a uma variável usada no programa (*Taces*).

No programa simulador, para decidir se o dado desejado está ou não armazenado no "cache" do servidor de arquivos é simplesmente usado o comando do GPSS/H (*transfer*) que transfere de forma probabilística uma transação para um bloco não-sequencial; ou seja, o fluxo normal do programa pode ser alterado a partir da comparação entre um número aleatório, gerado quando a transação se encontra no bloco referente a essa estrutura de transferência, e um operando desse comando (o valor para este operando no programa construído é de 70%, que corresponde a taxa de acerto original para o "cache" de disco).

4.3 Validação do Modelo de Simulação

Em processos de simulação de sistemas é necessário que o modelo construído seja validado para que os resultados obtidos nas simulações sejam considerados confiáveis, refletindo de forma aceitável o comportamento do sistema real. Entretanto, a validação de um modelo (e do programa de simulação associado) é uma tarefa que na maioria das vezes não é realizada com simplicidade. SARGENT, em [SA84], escreve que "normalmente não é prático considerar que um modelo seja absolutamente válido sobre o completo domínio da aplicação destinada; em vez disso, testes e avaliações devem ser conduzidos até que seja obtida confiança suficiente nos resultados, tal que o modelo possa ser considerado válido para a aplicação a qual se destina".

Basicamente, existem três formas de validar um modelo [SA84]:

- comparar os resultados das simulações com dados obtidos a partir de um sistema real (quando o sistema já existe);
- comparar os dados das simulações com dados de outro modelo já validado;
- obter nas simulações resultados conhecidos ou previsíveis (por exemplo, quando considerando um novo projeto para o sistema modelado).

Para a validação do modelo de simulação construído para o servidor de arquivos optou-se por comparar os resultados das simulações com dados obtidos a partir do sistema existente e por produzir nas simulações resultados conhecidos ou previsíveis.

O modelo usado na simulação do servidor de arquivos foi validado levando-se em consideração [SA84]:

- a observação de resultados que eram conhecidos previamente ou que podiam ser calculados manualmente;
- a influência de alterações nos parâmetros do modelo;
- validação interna dos resultados com relação a precisão dos mesmos;
- comparação dos resultados com dados obtidos junto ao sistema TRICE.

As técnicas utilizadas no processo de validação são apresentadas a seguir, destacando-se também resultados observados a partir de suas considerações.

Obtenção de Resultados Previsíveis

Nesta técnica, parâmetros e variáveis internas do programa simulador recebem valores predeterminados a fim de que possam ser gerados resultados conhecidos previamente ou

que podem ser calculados manualmente. A partir da consideração dessa técnica, os seguintes resultados foram obtidos:

- . **filas vazias:** simulações considerando apenas um usuário produziram filas sempre vazias para os recursos do sistema;
- . **tempos de resposta:** predefinindo o número de acessos aos recursos do sistema e fixando os seus tempos de execução, obteve-se valores conhecidos para os tempos de resposta.

Relação Entrada-Saída

Esta técnica é usada para verificar se a lógica do modelo de simulação está correta e para demonstrar que os resultados obtidos nas simulações são razoáveis com relação aos esperados a partir de dados de entrada particulares. Dentre as várias características verificadas destaca-se:

- . **fluxo dentro do programa:** através da consideração de valores específicos para alguns parâmetros (por exemplo, número de acessos a disco e número de usuários) pôde-se observar o comportamento correto do simulador quanto ao fluxo dentro do programa;
- . **tempo entre solicitações:** o aumento do número de usuários reduziu os tempos entre as solicitações inseridas no modelo (aumentando a carga de trabalho imposta sobre o servidor); a consideração de estações de trabalho mais velozes na rede causaram aumento no nível da carga de trabalho;
- . **filas dos recursos:** a elevação no nível da carga de trabalho produziu crescimento no comprimento médio das filas do processador e do disco; as considerações de processador e disco mais rápidos provocaram, além de diminuições nos tempos de execução desses recursos, reduções nos tempos médios de permanência das solicitações nas filas internas; taxas de acerto mais altas para o "cache" de disco reduziram o tamanho da fila do disco;
- . **tempo de resposta:** o tempo médio para o servidor de arquivos responder a uma solicitação aumentou quando o número de usuários cresceu, mas diminuiu quando recursos mais rápidos ou aumentos nas taxas do "cache" foram considerados nas simulações;

- **número de acessos a disco:** o número de acessos a disco diminuiu com o aumento da taxa de acerto do "cache".

Validação Interna

Uma preocupação que se deve ter com o uso de números aleatórios é relativa a precisão dos resultados obtidos nas simulações. Com relação a isso, durante as simulações levou-se em consideração os seguintes aspectos [BA89, SA84]:

1. Em geral, a simulação de sistemas baseia-se na utilização de variáveis randômicas (este é o caso do GPSS/H) e assim o resultado obtido depende da sequência dos números aleatórios utilizada durante a execução da simulação. Para obter estimativas mais precisas, foram calculados os valores médios entre os resultados obtidos em várias replicações usando sequências independentes de números aleatórios;
2. No começo da execução de uma simulação podem ocorrer condições iniciais não representativas, como por exemplo a ausência de transações no sistema e recursos desocupados. Essas condições iniciais podem influenciar na precisão dos resultados obtidos nas simulações. Para reduzir os efeitos dessas condições iniciais sobre os resultados das simulações realizadas, foi utilizada a técnica conhecida como *exclusão* (descrita no capítulo dois).

A tabela 4.1 apresenta os valores mínimos e máximos obtidos a partir dos dados produzidos por várias replicações (dez execuções do programa simulador) para algumas características do sistema simulado considerando diferentes cargas de trabalho, bem como as médias simples desses dados e os respectivos desvios padrões. Os intervalos de confiança de 95% para os resultados das simulações daquelas características e os valores obtidos junto ao TRICE também são mostrados na tabela 4.1.

Característica	Valor Mínimo	Valor Máximo	Valor Médio	Desvio Padrão	Intervalo de Confiança	TRICE
tempo no processador por acesso (1 usuário)	2,769	2,786	2,777	0,006(0,2%)	[2,773 a 2,781]	2,780
tempo de resposta por solicitação (1 usuário)	19,678	20,244	19,964	0,35(1,8%)	[19,714 a 20,214]	20,170
tempo de resposta por solicitação (10 usuár.)	21,949	23,204	22,798	0,32(1,4%)	[22,568 a 23,028]	22,960
número de acessos a disco	21064	22355	21816	672(3,1%)	[21336 a 22296]	21400

Tabela 4.1 - Variância Interna dos Resultados das Simulações.

Observa-se a partir da tabela 4.1 que os valores dos desvios padrões são bastante baixos; esse fato caracteriza as pequenas variâncias entre os resultados das simulações, demonstrando a precisão conseguida para as estimativas geradas.

Comparação de Resultados

Comparações entre dados do sistema real e das simulações mostram até que ponto o modelo está refletindo com precisão aceitável o comportamento do ambiente simulado. Para validar o modelo construído, utilizando esta técnica, foram realizadas comparações entre os resultados produzidos pelas simulações e os dados obtidos junto ao servidor de arquivos do sistema TRICE para várias características de interesse, destacando-se: os tempos médios de resposta do sistema, os tempos médios de execução no processador por solicitação, os tempos médios de permanência das solicitações nas filas dos recursos e o número de acessos a disco.

As figuras 4.2 e 4.3 e as tabelas 4.2 e 4.3 apresentam comparações entre os tempos médios de resposta do servidor de arquivos, obtidos a partir das simulações e do sistema real, para diferentes números de usuários. A figura 4.2 e a tabela 4.2 consideram a inexistência do "cache" de disco e a figura 4.3 e a tabela 4.3 consideram a existência de "cache" com taxa de acerto de 70%.

N. Usuários	1	10	20	30	40
TRICE	50,6	91	264	660	1100
Modelo	50,7	89,7	267	671	1114
Erro (%)	0,2	1,4	1,1	1,7	1,3

Tabela 4.2 - Comparações Entre os Tempos de Resposta - Sem "Cache".

Figura 4.2 - Comparações Entre os Tempos de Resposta - Sem "Cache".

N. Usuários	1	10	20	30	40
TRICE	20,2	23	30	37	55
Modelo	20,0	22,8	28,7	36,1	53,7
Erro (%)	1,0	1,0	4,3	3,0	2,4

Tabela 4.3 - Comparações Entre os Tempos de Resposta - Com "Cache".

Figura 4.3 - Comparações Entre os Tempos de Resposta - Com "Cache".

Outro tipo de dado bastante importante comparado foi relativo a quantidade de acessos a disco. A tabela 4.4 apresenta os números de acessos ao disco do sistema obtidos nas simulações e junto ao servidor de arquivos, considerando a ausência e a presença do "cache" de disco, necessários para a execução de 50 mil solicitações.

	TRICE	Modelo	Erro (%)
Sem "Cache"	71700	73665	2,7
Com "Cache"	21400	21816	1,9

Tabela 4.4 - Comparações Entre os Números de Acessos a Disco.

O que se observou das comparações realizadas foi que os valores médios dos resultados produzidos nas simulações e os dados obtidos junto ao servidor real são bastante próximos, com diferenças sempre inferiores a 5%. Diferenças desse nível são bastante aceitáveis em simulações estocásticas, permitindo considerar válidos o modelo e o programa de simulação construídos e assim possibilitar os seus usos como ferramentas auxiliares na realização de análises dos efeitos das aplicações de algumas técnicas especiais para o desenvolvimento de servidores de arquivos com alto desempenho.

4.4 Adequação do Sistema de Simulação

O sistema de simulação foi criado para a realização de análises de desempenho de servidores de arquivos. Os dados considerados caracterizam o comportamento do sistema distribuído de forma precisa e completa. Precisa, uma vez que foram obtidos através de monitores construídos em um sistema real e em utilização normal. Completa, pois caracteriza todo o funcionamento do servidor e também das estações de trabalho.

No entanto, com o rápido avanço dos componentes de "hardware", algumas das características originais do sistema TRICE se tornaram pouco adequadas. Por exemplo, o disco originalmente usado no servidor de arquivos do TRICE apresentava tempo médio de acesso a dados da ordem de 29 milissegundos (ms) e baixa capacidade de armazenamento. Atualmente no mercado estão disponíveis discos muito mais rápidos, com capacidades de armazenamento várias vezes superiores. Além disso, com relação aos tipos de estação de trabalho que podem compor um sistema distribuído, existem estações muito mais velozes que aquelas que compunham o TRICE. Assim, para que o modelo de simulação desenvolvido pudesse ser usado como base para as análises, tornou-se necessário alterar algumas de suas características, a fim de adequá-lo para o objetivo proposto.

Porém, a adequação do modelo não deve ser feita simplesmente considerando que os mais rápidos processador e disco existentes no mercado seriam adotados na estrutura do servidor

de arquivos de referência, mas sim ser realizada de forma criteriosa, considerando que os benefícios a nível de desempenho desse servidor justificassem os custos de sua construção.

Dessa forma, algumas características do sistema de simulação sofreram alterações visando adequar o modelo; essas alterações, bem como as justificativas para as suas considerações, são apresentadas nas seções a seguir.

4.4.1 Adequação da Carga de Trabalho

O modelo de simulação descrito nos itens anteriores considera um servidor de arquivos que é submetido a cargas de serviço impostas por estações de trabalho baseadas no processador MC68008 da Motorola (as estações que compunham originalmente o sistema TRICE). Entretanto, máquinas mais rápidas podem fazer parte de sistemas distribuídos; esse também é o caso do sistema TRICE que permite a utilização de outros tipos de estação.

Quando estações clientes mais rápidas são utilizadas, uma carga de trabalho mais pesada será impostas pelos usuários aos servidores de arquivos do sistema, influenciando nos tempos de resposta desses servidores. Tendo em vista o exposto, a fim de adequar o modelo de simulação quanto a carga de trabalho, inicialmente foi caracterizada uma nova carga, baseada nas distribuições descritas anteriormente, que poderia ser imposta aos servidores de arquivos quando considerando estações de trabalho com velocidade maior que a das estações originais.

Para a caracterização dessa nova carga, imposta pelas estações clientes, foram analisadas as características básicas da carga de trabalho descrita no capítulo três. Duas características devem ser consideradas: o tempo entre solicitações e o tipo de trabalho solicitado ao servidor. O trabalho solicitado não é influenciado pela velocidade da estação cliente. O tempo entre solicitações sofre uma influência não linear, como será discutido ainda nesta seção.

Observa-se também que uma estação de trabalho fica aguardando pela resposta do servidor de arquivos, sendo que uma nova solicitação só é feita por aquela estação ao servidor após a chegada da resposta da solicitação anterior. Assim, os intervalos entre as solicitações feitas por

um usuário dependem da atividade em desenvolvimento na estação de trabalho, dos tempos gastos nas comunicações entre a estação e o servidor e do tempo de serviço no servidor de arquivos (observar a figura 3.3, no capítulo três). Dessa forma, a carga imposta ao sistema tem influência sobre o tempo entre solicitações feitas por uma estação de trabalho através dos tempos de comunicação e de serviço no servidor de arquivos. No entanto, como a carga foi caracterizada pelo tempo entre a chegada da resposta na estação cliente até o envio da próxima solicitação ao servidor, pode-se considerá-lo dependente apenas do tempo gasto dentro da estação de trabalho (observar a figura 3.4, no capítulo três). Esse tempo de atividades dentro da estação de trabalho depende do tipo da aplicação sendo executada, da capacidade de processamento da estação e da ação do usuário.

Dados coletados junto ao sistema TRICE [SA90a] mostram que os intervalos de tempo entre solicitações menores que 300 milissegundos sofrem influência exclusiva da velocidade do processador da estação cliente e que para valores maiores que 300 ms o processador vai perdendo essa influência de forma gradual até aproximadamente 1 segundo, quando os tempos entre solicitações passam a depender, na maioria dos casos, da ação do usuário. Isto é, o processador da estação de trabalho só influencia na parte de processamento; no entanto, o tempo de ação do usuário permanece constante.

Baseado no exposto, a nova carga de trabalho imposta por cada estação cliente ao servidor de arquivos é caracterizada utilizando as distribuições originais obtidas junto ao sistema TRICE e considerando a relação entre a velocidade de um novo processador e a do processador original da estação de trabalho do TRICE (o MC68008). Essa relação (denominada Vest) é o resultado da divisão da velocidade do novo processador pela velocidade do processador original. Como parâmetro para a determinação dessas velocidades utilizou-se o "benchmark" Dhrystone. Relações de velocidade para alguns processadores mais usuais são observadas na tabela 4.5.

Processador	Dhrystones	Relação
MC68008	$4,4 * 10^3$	1
MC68060	$176 * 10^3$	40
Pentium	$71 * 10^3$	16
PowerPC 604	$170 * 10^3$	39

Tabela 4.5 - Relações Entre Velocidades de Processadores.

Essas relações (Vest) são utilizadas de acordo com os seguintes critérios:

- para intervalos de tempo entre solicitações menores que 300 ms, esses valores são divididos diretamente pela relação (Vest) considerada;
- para intervalos de tempo (IntTempo) entre 300 ms e 1000 ms, os novos valores dos intervalos (NovInt) são obtidos utilizando a seguinte expressão:

$$\text{NovInt} = \text{IntTempo} / (\text{Vest} - ((\text{Vest} - 1) * (\text{IntTempo} - 300) / (1000 - 300)))$$

- os intervalos de tempo maiores que 1 segundo não sofrem alterações.

Para demonstrar qual seria a influência desse novo tipo de carga de trabalho sobre o desempenho do servidor de arquivos considerado, foram realizadas simulações utilizando o referido modelo onde se considerou o uso de estações de trabalho mais rápidas. Para as simulações realizadas na sequência deste trabalho foi considerado um processador cuja relação de velocidade (Vest) é igual a 40. Resultados de simulações comparando o tempo médio de resposta do servidor (obtidos variando-se o número de usuários) para as estações de trabalho original e nova são apresentados na figura 4.4.

Figura 4.4 - Tempos de Resposta do Servidor de Arquivos.

Os resultados obtidos nessas simulações demonstram a influência do novo tipo de carga de trabalho imposto ao servidor, que, por gerar requisições mais rapidamente, provoca a elevação no valor médio do tempo de resposta do sistema. Salienta-se que a elevação observada é resultado do aumento nos valores dos tempos médios de espera nas filas do processador e do disco, uma vez que os tempos médios de execução nesses recursos são constantes para qualquer tipo de carga. Para o caso de um usuário, o tempo médio de resposta (20 ms) não sofre alterações, uma vez que neste caso não existe concorrência pelos recursos do servidor e o tempo gasto em filas é nulo. A figura 4.5 mostra os aumentos nos tempos médios gastos na fila do disco observados nas simulações realizadas.

Figura 4.5 - Tempos Gastos na Fila do Disco.

4.4.2 Adequação dos Recursos

O ambiente de simulação modela um servidor de arquivos composto por um processador MC68020 da Motorola, um disco com tempo médio de acesso a dados (correspondendo a soma do tempo médio de localização da trilha desejada com o tempo médio de latência do disco) de aproximadamente 29 milissegundos (ms) e "cache" de disco com taxa de acerto da ordem de 70%. No entanto, salienta-se que recursos mais rápidos que os citados existem atualmente no mercado. Assim, os parâmetros do modelo referentes às velocidades dos recursos foram alterados para que refletissem melhor os recursos atuais.

Descreve-se a seguir as adequações realizadas no modelo de simulação, bem como os critérios utilizados e os resultados obtidos nas simulações a partir da consideração das mesmas.

A Substituição do Disco

A partir da figura 4.4 pode-se observar que o tempo médio de resposta do sistema é de 20 ms quando considerando um único usuário e que o seu valor cresce quando uma carga mais

alta é considerada nas simulações. Esse crescimento é devido ao aumento dos tempos médios gastos nas filas do processador e do disco quando a concorrência por esses recursos cresce. Na figura 4.5 nota-se o crescimento ocorrido na fila do disco quando a carga varia de 1 até 70 usuários.

Fazendo uma comparação entre as figuras 4.4 e 4.5, conclui-se que a maior parte do tempo gasto nas filas é relativo a fila do disco. Isso pode ser confirmado na figura 4.6, que apresenta os tempos médios gastos nas filas dos recursos do servidor de arquivos.

Figura 4.6 - Tempos Gastos nas Filas do Processador e do Disco

Outro tipo importante de dado coletado nas simulações realizadas é o relativo a taxa de utilização dos recursos (fração do tempo total de execução do sistema na qual o recurso esteve ocupado). A partir dessa taxa é possível verificar qual recurso está sendo mais utilizado no sistema. A figura 4.7 mostra as taxas de utilização (em porcentagem) para o processador e para o disco quando a carga de trabalho varia.

Figura 4.7 - Taxas de Utilização para o Processador e Disco.

O que se observa na figura 4.7 é que a taxa de utilização do disco mostrou-se sempre maior que a do processador (aproximadamente duas vezes), ou seja, o disco permaneceu ocupado um período de tempo maior que o do processador. Deve-se observar também, que para cargas altas o disco passou praticamente todo o tempo de simulação ocupado.

A partir das figuras 4.6 e 4.7, pode-se descrever o disco como o ponto de estrangulamento (gargalo) do servidor de arquivos e, por ser considerado assim, deve ser o primeiro dispositivo a ser considerado no processo de adequação de recursos. Observar que, embora a substituição do processador por outro mais rápido deva provocar reduções nos tempos médios de resposta do servidor de arquivos, os ganhos provavelmente não seriam tão significativos quanto aos obtidos com a substituição do disco.

A fim de verificar a influência da utilização de discos mais rápidos sobre o desempenho do servidor, foram realizadas novas simulações considerando no modelo em uso tempos médios de acesso a dados com valores inferiores ao do disco original (de 29 ms); foram

considerados tempos médios de 20 ms, 15 ms e 12 ms. Esses valores correspondem aos tempos médios de acesso para discos encontrados no mercado [RI95, SE94] (por exemplo, o Hawk2 ST11900N de 15 ms e o Barracuda1 ST11950N de 12 ms, ambos da Seagate). Os tempos médios de resposta obtidos nessas simulações a partir da variação da carga de trabalho são apresentados na figura 4.8 juntamente com os relativos ao disco original.

Figura 4.8 - Tempos de Resposta do Servidor de Arquivos.

Como pode ser observado pelos resultados das simulações, a substituição do disco original por outro com menor tempo médio de acesso pode trazer benefícios bastante expressivos ao desempenho do servidor de arquivos. Observa-se também que as reduções nos tempos médios de resposta foram mais acentuadas quando a carga de trabalho tornou-se mais pesada e que, nessas condições, ganhos de até 86% foram registrados quando considerando o disco de 12 ms. Isso ocorre porque, nas simulações, a substituição do disco original por outro de 12 ms, além de provocar a diminuição do tempo médio de execução relativo a tal dispositivo, promove drástica redução no tempo gasto na sua fila (o tempo gasto na fila do disco quando considerando o

dispositivo de 29 ms, para 70 usuários, foi da ordem de 387 ms, enquanto que para o disco de 12 ms esse tempo foi de aproximadamente 5,6 ms).

Os resultados mostram ainda que os tempos de resposta quando considerando um disco de 15 ms foram apenas pouco superiores ao de 12 ms. Então, devido a custos, a primeira opção de uso seria o disco de 15 ms. Entretanto, baseado em resultados de simulações, verifica-se que o sistema de disco continua sendo o fator limitante do desempenho do servidor de arquivos. Para que o sistema de disco deixe de ser tal gargalo, esse dispositivo deve apresentar maior velocidade. Assim, de acordo com as simulações, chega-se ao disco de 12 ms, o mais rápido atualmente, como a melhor opção para a adequação do sistema de disco.

A Substituição do Processador

Embora ganhos consideráveis tenham sido obtidos nas simulações que consideram a substituição do disco original por outro mais rápido, outros procedimentos, visando reduzir ainda mais os tempos médios de resposta do servidor de arquivos, podem ser investigados. Novas análises utilizando o modelo de simulação foram realizadas, considerando agora o uso de um disco com tempo médio de acesso a dados da ordem de 12 ms.

Das simulações considerando o disco de 12 ms obteve-se informações relativas aos tempos médios gastos nas filas dos recursos; esses tempos são mostrados na tabel 4.6 e na figura 4.9, juntamente com os tempos médios de resposta do sistema, obtidos com a variação da carga de trabalho.

N. Usuários	1	40	70
Tempo Resp.	12,3	21,6	56,6
Fila-Process.	0	7,1	38,7
Fila-Disco	0	2,2	5,6

Tabela 4.6 - Tempos de Resposta e Gastos nas Filas do Processador e do Disco.

Figura 4.9 - Tempos de Resposta e Gastos nas Filas do Processador e do Disco

Observando esses tempos, nota-se que a fila do processador tornou-se maior que a do disco (exceto para um usuário, em que são ambas vazias). Analizando os valores apresentados na tabela 4.6 e na figura 4.9, observa-se que:

- para 40 usuários, o tempo médio para a execução de uma requisição aumentou de 12,3 ms para 21,6 ms, uma diferença de 9,3 ms; desse tempo, 7,1 ms (76%) foi devido ao tempo de espera na fila do processador;
- para 70 usuários, aquele tempo na fila do processador aumenta para 38,7 ms e a porcentagem para 87%.

A partir dessa análise, conclui-se que, com a utilização do disco de 12 ms, o processador se torna o fator limitante do desempenho do sistema. Assim, se melhoramentos forem realizados a nível do processador, ganhos satisfatórios devem ser obtidos no desempenho do

servidor de arquivos. Uma forma de alcançar melhoramentos no processador é através da sua simples substituição por outro mais rápido.

Para analisar o comportamento do sistema quando velocidades superiores são consideradas para o processador, realizou-se simulações utilizando o modelo que adota o uso do disco de 12 ms considerando processadores com velocidades 5 vezes, 10 vezes, 20 vezes e 50 vezes a do processador original do servidor de arquivos, o MC68020 (essas velocidades relativas foram consideradas somente para efeito de testes, não tendo necessariamente relação com processadores existentes no mercado). A seguir, são apresentados resultados relativos ao tempo médio de resposta obtidos nessas simulações, juntamente com os obtidos usando o processador original.

Figura 4.10 - Tempos de Resposta Usando Processadores Diferentes.

A figura 4.10 mostra que benefícios podem ser alcançados com a utilização de processadores mais rápidos que o original. Por exemplo, segundo as simulações, o uso de um processador com fator de 5 para a velocidade pode proporcionar ganhos de até 57% no desempenho do servidor em comparação aos obtidos com o dispositivo original (e disco de 12 ms), quando considerando 70 usuários.

Entretanto, a consideração sucessiva de componentes mais rápidos nem sempre deve trazer ganhos significativos ao desempenho do sistema. Observa-se na figura 4.10 que as simulações considerando o processador com velocidade 5 vezes apresentaram tempos médios de resposta apenas pouco superiores aos obtidos para outros com velocidades maiores. Esse fato ocorreu porque, a partir da consideração de um recurso com fator 5 de velocidade, a taxa de utilização do disco se mostrou bem maior que a do processador (em torno de 4 vezes). Isto é, o disco voltou a ser o gargalo do sistema e a adoção de processadores mais rápidos passaram a apresentar benefícios pouco expressivos ao desempenho do sistema. Tendo em vista o exposto e os custos de processadores muito velozes, o processador com velocidade 5 vezes a do original torna-se a opção mais conveniente no processo de adequação desse tipo de recurso. No capítulo seis será apresentada uma análise sobre a possibilidade de adotar-se um processador 5 vezes mais rápido.

Os Resultados da Adequação dos Recursos

De acordo com as observações anteriores, as considerações de um processador com velocidade 5 vezes a do recurso original e de um disco com tempo médio de acesso a dados da ordem de 12 ms no modelo de simulação utilizado produzem reduções bastante significativas nos tempos médios de resposta do sistema. As reduções resultantes das considerações desses recursos podem ser avaliadas a partir da tabela 4.7 e da figura 4.11, que apresentam os tempos médios de resposta quando no modelo de simulação são considerados os novos recursos e os que originalmente compunham o servidor de arquivos do sistema TRICE.

N. Usuários	1	40	70
Original	20,0	80,6	411,3
Adequado	6,8	11,2	24,2

Tabela 4.7 - Tempos de Resposta do Servidor de Arquivos.

Figura 4.11 - Tempos de Resposta do Servidor de Arquivos.

O que se observa a partir dos resultados das simulações é que a consideração dos recursos mais rápidos podem provocar reduções de até 94% (para 70 usuários) nos valores dos tempos médios de resposta em relação aos valores obtidos considerando os dispositivos originais.

Salienta-se, no entanto, que para a obtenção de novas reduções significativas nos tempos médios de resposta, outras técnicas devem ser consideradas no lugar de procedimentos que simplesmente resultam em substituições de recursos por outros mais rápidos. Para justificar isso, observa-se que o disco mais rápido atualmente (de 12 ms) já está sendo considerado nas simulações e que a utilização de processadores com velocidades superiores a 5 vezes a do dispositivo original não produzem ganhos maiores ao desempenho do sistema que justifiquem os custos de suas possíveis adoções em um servidor de arquivos real. Isto é, com esses valores, o disco mais veloz encontrado atualmente no mercado ainda é o gargalo do sistema. Na impossibilidade de obter discos mais rápidos, deve-se partir para abordagens que evitem acessos a disco ou que tornem esses acessos mais rápidos, mesmo utilizando um dispositivo mais lento.

Dessa forma, o processador de velocidade 5 vezes a do recurso original e o disco com tempo médio de acesso de 12 ms foram os recursos considerados mais indicados para serem relevados no modelo de simulação, a fim de que esse modelo pudesse ser considerado adequado para os propósitos de análises descritos anteriormente.

Após a sua adequação, o modelo de simulação foi utilizado para que pudessem ser analisados os efeitos de aplicações de procedimentos voltados para a obtenção de servidores de arquivos com alto desempenho. Esse modelo é tratado na sequência do trabalho como "modelo base".

4.5 Considerações Finais

Este capítulo apresentou o ambiente de simulação desenvolvido para possibilitar a avaliação de desempenho de servidores de arquivos baseados em redes locais. Para a criação do modelo e do programa de simulação associado utilizou-se o ambiente do servidor de arquivos do sistema distribuído TRICE. O modelo criado representa de forma simplificada as funções de interesse do servidor de arquivos considerado, modelando o processador, o disco e o "cache" de disco, e considera a execução de uma solicitação desde a sua entrada no servidor até a complementação da mesma e consequente envio da resposta para a rede.

Embora o modelo seja simplificado, resultados precisos e confiáveis foram obtidos para as características de interesse. Essa simplificação permitiu a criação de um sistema de simulação sem se ter a preocupação com detalhes do sistema real, o que torna o programa desenvolvido, além de simples, bastante flexível, possibilitando rápidas e fáceis alterações em sua estrutura; além disso, essa simplicidade faz com que seja necessário baixo tempo de execução do programa de simulação, permitindo a realização de um grande número de replicações.

O sistema utilizado para a implementação do programa simulador, o GPSS/H, fornece várias ferramentas direcionadas para a simulação de sistemas; o uso de GPSS/H torna o programa bastante simples, compacto e de fácil entendimento, e permite que alterações sejam realizadas com simplicidade e rapidez no programa. O programa simulador foi desenvolvido

usando um mecanismo orientado a processos e utiliza várias distribuições estatísticas simplificadas a partir de dados coletados junto ao servidor de arquivos real, apresentados no capítulo três.

Várias técnicas, direcionadas para a validação e verificação de modelos de simulação e seus programas associados, foram utilizadas para validar os resultados produzidos pelas simulações executadas. Nesse processo de validação mostrou-se a precisão daqueles resultados. A partir da determinação dos desvios padrões das estimativas para características de interesse do modelo pôde-se observar as pequenas variâncias entre os valores, demonstrando a precisão interna conseguida para os resultados gerados nas várias execuções do programa de simulação. Através de comparações entre os valores médios dos resultados produzidos pelas simulações e os dados obtidos junto ao servidor de arquivos do sistema TRICE, para as características de interesse, observou-se diferenças sempre inferiores a 5%, consideradas baixas, e com isso demonstrando a precisão das estimativas com relação aos dados do sistema real. Aqueles resultados das simulações também permitiram construir intervalos de confiança de 95%, caracterizando a precisão dos mesmos.

Algumas das características do ambiente de referência o tornavam não totalmente adequado para ser usado como base naquelas análises; assim, o passo seguinte foi adequar o modelo através da alteração dessas características. Foram realizadas alterações quanto a carga de trabalho exercida sobre o servidor de arquivos e nos parâmetros relativos as velocidades do processador e do disco. Durante esse processo, baseado nos resultados das simulações realizadas, concluiu-se que a velocidade mais adequada para o processador seria de 5 vezes a do processador original do servidor de arquivos do TRICE, o MC68020, e que o disco mais adequado para ser usado seria o de tempo médio de acesso a dados de 12 ms, o mais rápido no momento.

A partir de suas validações e adequações, o modelo e o programa de simulação foram utilizados como ferramentas na realização de análises dos efeitos das aplicações de técnicas especialmente direcionadas para a implementação de servidores de arquivos com alto desempenho. Essas análises são descritas no capítulo cinco.

CAPÍTULO CINCO

TÉCNICAS PARA A OBTENÇÃO DE SERVIDORES DE ARQUIVOS DE ALTO DESEMPENHO

O modelo de simulação descrito no capítulo anterior foi desenvolvido para possibilitar a análise de abordagens que podem ser aplicadas na implementação de servidores de arquivos visando a obtenção de sistemas com alto desempenho. Este capítulo apresenta algumas dessas abordagens e os resultados obtidos a partir da simulação do modelo. Os resultados são então analisados e conclui-se que abordagens são mais adequadas.

O item 5.1 apresenta observações quanto ao modelo de simulação utilizado e quanto a forma como os resultados das simulações foram gerados.

Algumas técnicas que podem ser utilizadas para a obtenção de servidores de arquivos com alto desempenho são analisadas no item 5.2. Os resultados obtidos nas simulações a partir de suas considerações e as vantagens e desvantagens de cada uma são discutidos. A técnica considerada mais adequada para obter um servidor de arquivos com alto desempenho e alta confiabilidade é salientada.

Finalmente, no item 5.3 são feitas considerações quanto a estrutura sugerida para um servidor de arquivos e aos níveis de carga de trabalho aos quais pode ser exposto.

5.1 O Modelo e os Resultados das Simulações

Neste capítulo são analisadas algumas abordagens que podem ser utilizadas para a construção de servidores de arquivos com alto desempenho. Especial ênfase é dada para a técnica apresentada nesta tese, que considera a utilização de múltiplos discos onde os acessos são realizados apenas em áreas parciais dos mesmos. A forma escolhida para auxiliar na realização dessas análises foi através do uso de um modelo de simulação.

Observa-se que os dados obtidos nas simulações são resultados de simulações estocásticas e devem ser analisados considerando-se os efeitos da utilização de números aleatórios. Uma preocupação devido ao uso de números aleatórios é relativa à precisão das estimativas para os valores de características do sistema em observação, pois essas estimativas dependem dos números aleatórios gerados durante a execução da simulação. Uma nova execução da simulação (replicação) sobre o modelo usando um conjunto diferente de números aleatórios produzirá uma estimativa diferente para as características consideradas. Para obter valores estimados mais precisos, torna-se necessário o cálculo dos valores médios entre os resultados de várias replicações usando diferentes conjuntos de números aleatórios. Esse método reduz a variância dos resultados e permite a construção de um intervalo de confiança predefinido que refletem a precisão obtida. Os valores apresentados nesta tese são as médias de resultados de no mínimo dez execuções das simulações (este valor mínimo é o resultado de testes de simulações procurando produzir um intervalo de confiança de 95%).

Outro ponto que merece atenção é quanto as condições iniciais não representativas que podem ocorrer em simulações (como por exemplo a ausência de transações no sistema e recursos ociosos no início da simulação), em geral apresentando grande influência sobre os resultados obtidos. Para reduzir os efeitos desses transientes iniciais, nas simulações foi usada a técnica de *exclusão*, onde a execução da simulação é dividida em duas partes. Uma fase de inicialização, caracterizada nos exemplos quando a simulação alcança dez por cento do total de solicitações, onde as estatísticas coletadas são descartadas. A segunda parte é uma fase de coleta de

dados, onde as estatísticas coletadas são consideradas como estimativas para os valores das características de interesse. Outras observações relativas a essa técnica são feitas no capítulo dois.

Salienta-se ainda que, neste capítulo, são apresentados os resultados das simulações para apenas três valores de carga de trabalho (aos quais o servidor de arquivos do sistema está exposto): 1 usuário, 40 e 70 usuários. Isso é feito porque esses valores representam níveis de carga de trabalho com características de interesse bem definidas para as análises, dispensando as apresentações de valores intermediários.

5.2 Técnicas Para a Obtenção de Alto Desempenho

No capítulo quatro foram mostrados vários resultados obtidos com o sistema de simulação construído e adequado. Além dos valores para os tempos médios de resposta do sistema quando recursos mais rápidos são considerados no modelo, outro tipo de informação que pode ser obtido das últimas simulações é relativo a taxa de utilização daqueles recursos. A figura 5.1 mostra essas taxas para o processador e o disco quando a carga de trabalho é variada.

Figura 5.1 - Taxas de Utilização para o Processador e Disco.

As taxas de utilização do processador e do disco reforçam a conclusão do capítulo anterior (seção 4.4.2) outros meios, objetivando diminuir os tempos relacionados ao disco, deveriam ser investigados.

Uma vez que a adoção de discos mais rápidos não é viável, duas formas básicas que podem ser usadas para diminuir os tempos relativos ao sistema de disco são: a redução no número de acessos a esse sistema e o aumento na quantidade de discos. A seguir são analisadas técnicas nesse sentido.

5.2.1 Redução no Número de Acessos a Disco

Uma forma de se evitar acesso a disco é manter os dados envolvidos em uma operação de leitura em um "cache" de disco, sendo a solicitação mais rapidamente atendida. Várias técnicas de "cache" são analisadas a seguir.

Redução no Número de Leituras em Disco

O "cache" de disco do servidor de arquivos de referência considera apenas a leitura de dados, evitando em torno de 85% dos acessos a disco relativos a aquele tipo de operação. Isso proporciona uma taxa total de acerto, considerada até agora no modelo de simulação, da ordem de 70%, pois, como descrito no capítulo três, aproximadamente 83% das operações envolvendo acessos a disco estão relacionadas com a leitura de dados.

O que se observa dos valores acima é que a taxa total de acerto do "cache" pode ser elevada, reduzindo o número de acessos a disco. A fim de verificar os benefícios que isso pode trazer ao desempenho do servidor de arquivos, foram realizadas simulações utilizando o modelo base e considerando uma taxa de acerto do "cache" relativo a leituras maior. No modelo de simulação, isso foi conseguido simplesmente alterando o valor da taxa de acerto desse "cache"; na prática, tal aumento pode ser obtido através do melhoramento dos tipos de "caches" já implementados ou utilizando outras formas de "cache" envolvendo leituras [NE88, RI83, SA90a].

Alguns resultados dessas simulações, relativos aos tempos médios de resposta, considerando que a taxa de acerto do "cache" para leituras variou entre 85% (o valor original) e 100%, são apresentados a seguir.

Figura 5.2 - Tempos de Resposta para Diferentes Taxas de Acerto do "Cache".

Nota-se a partir dos resultados da figura 5.2 que reduções razoáveis nos tempos médios de resposta são conseguidas quando a taxa de acerto do "cache" cresce (para a taxa de 100% e 70 usuários, observa-se reduções de 64%), demonstrando os ganhos que podem ser obtidos quando o número de acessos a disco é reduzido.

No entanto, obter "cache" de disco com alta taxa de acerto é uma tarefa que pode implicar grandes custos adicionais. Qualquer que seja a forma utilizada para obter aumentos nas taxas de acerto do "cache" para operações relativas a leitura resulta em crescimento no tamanho desse "cache", devendo envolver custos adicionais de implantação [MU91, NE88]. Além disso, uma vez que escritas são executadas junto a operações de leitura, os mecanismos usados para tratar o problema quanto a consistência entre os dados armazenados em disco e no "cache" provocam sobrecarga ao servidor de arquivos, podendo não trazer o desempenho esperado [NE88].

Escrita no Disco em Tempos Ociosos

Embora a taxa total de acerto do "cache" de disco, conseguida com técnicas que envolvem apenas operações de leitura de dados, possa ser aumentada, essa está limitada ao valor de 83% (porcentagem relativa a leitura de dados do total de operações envolvendo acessos a disco). Para superar este limite, técnicas de "cache" direcionadas a escrita de dados podem ser utilizadas, reduzindo o tempo total envolvendo o sistema de disco.

Uma técnica de "cache" que pode ser usada para reduzir o número de acessos a disco é a que armazena temporariamente na memória principal do sistema os dados referentes às operações de escrita em disco. Os dados que devem ser escritos em disco são inicialmente armazenados nessa memória e, quando o sistema de disco estiver ocioso, tais dados são transferidos para este. Dessa forma, a taxa de acerto desse tipo de "cache" pode ser considerada de 100%, uma vez que, se o princípio dessa técnica for seguido (escritas são realizadas em disco apenas em tempos ociosos), o tempo gasto com tais operações não deve ser considerado no cálculo do tempo médio de resposta do servidor de arquivos.

Baseado no princípio descrito acima, foram realizadas simulações, utilizando o modelo base (em que o "cache" para leituras tem taxa de 85%), considerando a existência de um "cache" para escrita em disco em tempos ociosos (isso foi conseguido pela simples alteração do valor da taxa de acerto para escritas). A figura 5.3 apresenta os tempos médios de resposta obtidos nas simulações com o modelo base considerando apenas o "cache" padrão (somente de leitura) e considerando a existência também do "cache" de escrita em tempos ociosos.

Figura 5.3 - Tempos de Resposta para Diferentes "Caches".

O que se observa dos resultados apresentados na figura 5.3 é uma redução acentuada (de 74% para 70 usuários) nos valores dos tempos médios de resposta. Entretanto, a adoção de um "cache" com essas características apresenta alguns problemas. O principal deles é relativo a consistência das informações, uma vez que os dados armazenados na memória (que contém as últimas atualizações) podem permanecer diferentes dos alocados em disco durante longos períodos de tempo, o que possibilita a perda daqueles dados devido a queda do servidor, falta de energia, etc., uma vez que a sua memória principal é volátil.

Uma forma de atenuar o problema de possíveis perdas de informações é através da utilização de um conjunto de memórias não-voláteis para a implementação do "cache" de escrita. No entanto, o "cache" construído assim também apresenta alguns inconvenientes. Torna-se necessária a obtenção de um conjunto de memória diferente do tipo convencional, com um custo mais elevado e necessitando de controle e alimentação separados do restante da memória principal.

Deve-se observar também que a técnica para escrita em tempos ociosos leva em conta que existirão momentos frequentes em que o sistema de disco estará desocupado para a

execução de escritas. No entanto, quando da ocorrência de altas cargas de trabalho, tais momentos podem não existir com a frequência necessária, o que fará com que os dados do "cache" permaneçam inconsistentes com relação aos do disco por períodos de tempo ainda maiores, aumentando as chances de perdas de informações e tornado necessário um "cache" de tamanho muito grande. As altas cargas de serviço devem ser consideradas quando do dimensionamento do tamanho do "cache", pois nessas situações uma maior quantidade de dados deve permanecer armazenada em tal "cache".

Escrita Atrasada em Disco

Para amenizar os problemas descritos com a abordagem discutida na seção anterior (consistência e tamanho do "cache" necessário), pode-se considerar a escrita atrasada em disco. Esta técnica, para implementação de "cache", diminui o número de acessos a disco uma vez que as informações são inicialmente escritas no "cache" e transferidas para o disco algum tempo depois. Essa técnica apresenta algumas vantagens:

- desempenho: uma vez que as escritas são executadas em "cache", estas são completadas muito mais rapidamente que em disco (como acontece com a técnica anterior), não provocando grandes atrasos;
- evita acesso desnecessário ao disco: dados podem ser excluídos antes de serem repassados ao disco, reduzindo bastante o tráfego para o disco;
- minimiza o problema de consistência: a informação no "cache" é enviada ao disco em intervalos constantes e curtos.

Estatísticas obtidas junto ao sistema de arquivos 4.2 BSD UNIX [NE88, OU85] mostram que em torno de 25% de novos dados são excluídos dentro dos 30 segundos seguintes à solicitação de escrita e 50% são excluídos dentro dos próximos 5 minutos. Assim, se esta técnica for utilizada, o número de acessos a disco pode ser reduzido, diminuindo os tempos gastos relativos ao sistema de disco.

Seguindo a idéia apresentada, foram realizadas simulações usando como referência o modelo base e considerando a existência de um "cache" com escrita atrasada em disco. Para essas simulações foram consideradas as taxas de exclusão citadas, obtidas junto ao sistema BSD. A figura 5.4 apresenta os tempos médios de resposta resultantes das simulações com o modelo base e com o modelo que considera a existência também do "cache" para escrita atrasada em disco.

Figura 5.4 - Tempos de Resposta para "Caches" com Escrita Atrasada.

Os resultados acima mostram os benefícios que podem ser alcançados nos valores dos tempos médios de resposta com a adoção da técnica de escrita atrasada: reduções de até 27% são atingidas quando considerando atrasos de 30 segundos e de até 46% para atrasos de 5 minutos (para 70 usuários).

No entanto, o esquema de escrita atrasada apresenta problemas de confiabilidade, desde que dados ainda não escritos em disco serão perdidos no caso de ocorrência de quebra do servidor de arquivos. Assim, quanto menor o atraso na realização da escrita em disco, menores serão as chances de perdas de informações. Com isso, embora os benefícios obtidos com atrasos da

ordem de 5 minutos tenham sido mais expressivos, as chances de perdas de informações com tal esquema aumentam bastante com relação a aquelas quando escritas são atrasadas apenas 30 segundos. Para resolver o problema relativo a consistência dos dados, a mesma solução apresentada para escrita no disco em tempos ociosos (utilizar memórias não-voláteis) pode ser adotada aqui; porém, os inconvenientes salientados anteriormente permanecem válidos também para este caso.

Considerando o problema relativo ao tamanho do "cache" para escrita, tem-se que quanto maior for o tempo de permanência das informações no "cache", maior deverá ser o tamanho do mesmo, a fim de comportar todos os dados ainda não transferidos para o disco (como ocorre no caso da técnica de escrita em tempos ociosos, quando o servidor de arquivos é submetido a altas cargas de trabalho).

5.2.2 Aumento no Número de Discos

Uma forma alternativa de reduzir o tempo total gasto junto ao sistema de disco é através da replicação desse dispositivo. Dessa forma, solicitações que necessitam de acessos a discos podem ser atendidas de forma concorrente, diminuindo os seus tempos de espera em filas e consequentemente os tempos de respostas para as solicitações.

A fim de possibilitar a realização de análises quanto aos efeitos da utilização de vários discos na estrutura de um servidor de arquivos, o modelo base de simulação foi modificado para considerar um número maior de discos. As modificações envolveram apenas aspectos relativos a quantidade desses dispositivos, sendo mantidas as demais características. O diagrama de blocos mostrado na figura 5.5 representa o modelo de simulação utilizado quando considerando mais de um disco. Neste novo modelo, tem-se como recurso um processador e vários discos, sendo que o acesso aos discos são feitos através de um fila única.

Figura 5.5 - Modelo de Simulação com Vários Discos.

As modificações no programa simulador referente ao modelo base foram realizadas através da ativação de várias instâncias do disco e da replicação da sequência de códigos relativos ao disco. Nesse programa foi considerado que os discos envolvidos teriam tempo médio de acesso a dados da ordem de 12 ms, sendo que os valores para os parâmetros relativos ao processador (fator 5 de velocidade) e a taxa de acerto do "cache" (de 70%) foram mantidos.

Uso Convencional dos Discos

Segundo o modelo, as solicitações de acesso a disco são colocadas em uma fila única; dessa forma, não há distinção de acesso entre os discos. Isso significa que os acessos para leitura são realizados sobre qualquer disco e portanto esses dispositivos devem manter sempre as mesmas informações. Para manter os discos iguais, cada solicitação de escrita deve, então, ser realizada sobre todos os discos.

Baseado no modelo descrito, foram realizadas modificações no programa simulador em uso. Utilizando esse programa foram executadas simulações em que foi variado de 2 a 5 o

número de discos envolvidos. Considerou-se nessas simulações que os acessos a dados seriam realizados de forma convencional, ou seja, as cabeças de leitura/escrita poderiam fazer acessos a qualquer local dentro do disco. Quando essa forma de acesso é usada, o tempo médio de localização de dados para os tipos de discos considerados permanece da ordem de 12 ms. A figura 5.6 apresenta os tempos médios de resposta obtidos em simulações considerando o modelo base, com apenas um disco, e o modelo que permite um número maior de discos.

Figura 5.6 - Tempos de Resposta Para Diferentes Números de Discos.

O que se observa a partir dos resultados obtidos para dois ou mais discos é que, com relação as simulações considerando um único disco, pouca melhoria (ou mesmo nenhuma) é obtida quando cargas de trabalho leves são consideradas. Isso ocorre porque as taxas de utilização dos recursos são baixas, fazendo com que os tempos de permanência das solicitações na fila do sistema de discos sejam pequenos e poucas solicitações possam ser executadas de forma concorrente. No entanto, quando altas cargas de trabalho são consideradas, observa-se reduções acentuadas nos tempos médios de resposta do sistema (por exemplo, para 70 usuários e dois discos a redução foi de aproximadamente 60% - de 24,2 ms para 9,8 ms), justamente pela possibilidade de atendimento

concorrente de solicitações que requerem acessos a disco. Isso mostra que os benefícios trazidos com a replicação de disco tornam sua aplicação bastante interessante.

Desses mesmos resultados, nota-se também que a consideração de um número de discos igual ou superior a três leva a reduções nos tempos médios de resposta pouco superiores a aquelas obtidas com dois discos. Assim, uma forma de obter um aumento acentuado no desempenho do sistema, sem envolver maiores custos de implantação, seria através do uso de dois discos.

O esquema de replicação de discos descrito provoca a redução nos tempos médios de resposta do sistema sem introduzir problemas quanto a consistência das informações (como verificado com as técnicas que consideram o uso de "cache"), pois essas são armazenadas diretamente nos discos quando solicitado. Além disso, esse esquema aumenta a segurança dos dados já armazenados com relação a falhas do dispositivo, uma vez que existe(m) cópia(s) dos mesmos em outro(s) disco(s).

Entretanto, tal esquema apresenta um grande inconveniente: para que funcione adequadamente, essa técnica necessita que os discos sempre mantenham as mesmas informações. Ou seja, quando uma solicitação de escrita é feita, essa deve ser replicada e executada sobre todos os dispositivos, o que degrada o desempenho do servidor de arquivos.

Acessos a Áreas Parciais do Disco

A técnica anterior considera que os acessos aos discos em operações normais de serviço podem ser feitos em qualquer local dentro dos mesmos, o que resulta em tempos médios de acesso a dados da ordem dos especificados pelos fabricantes para os dispositivos considerados (no caso em questão, tempo médio de 12 ms).

O tempo médio de acesso a dados especificado comercialmente para um disco é basicamente composto pelo tempo médio de latência e pelo tempo médio de localização do cilindro desejado, sendo este último determinado quando se considera que as cabeças de leitura/escrita

podem fazer acessos em toda a área (a todos os cilindros) do dispositivo. Se os movimentos dessas cabeças forem restritos a uma área parcial do disco, ou seja, se os movimentos forem localizados apenas sobre um conjunto contíguo de cilindros, o tempo médio de busca de cilindros contidos nessa área parcial será reduzido, diminuindo o tempo médio total de acesso ao dado. Como a área parcial designada para um disco pode estar abrangendo apenas parte das informações armazenadas, para que todos os dados possam, então, ser localizados torna-se necessária a utilização de vários discos. Para a obtenção da região parcial, a área total de cada recurso deve ser dividida em um número de partes igual ao de dispositivos utilizados, sendo que o acesso a um determinado dado em operações normais de trabalho só pode ser realizado em um dos discos.

A figura 5.7 representa o esquema descrito considerando o uso de três discos; o aspecto linear de movimento das cabeças de leitura e escrita de cada disco é mostrado. A região sombreada representa a área parcial (os cilindros) em que são feitos acessos em cada disco durante operações normais de trabalho.

Figura 5.7 - Regiões de Acesso dos Discos.

Baseado na técnica que considera que os deslocamentos das cabeças de leitura/escrita dos discos são restritos a regiões parciais de dados (diminuindo os seus tempos médios de localização de cilindros), foram realizadas alterações nas características relativas aos tempos médios de acesso dos discos do programa de simulação adequado para conter mais de um

disco. Para essas alterações levou-se em consideração o seguinte aspecto. Com relação ao disco a ser considerado, o mais veloz do mercado tem especificação para o tempo médio de localização do cilindro desejado da ordem de 8 ms, quando acessos se realizam em toda a área do dispositivo, e tempo médio de latência em torno de 4 ms, resultando em tempo médio de acesso a dados de aproximadamente 12 ms. Assim, para se determinar o tempo médio de localização de cilindros para cada um dos discos envolvidos, considerando que o acesso seria feito a apenas uma parte da área de dados, basta dividir o tempo de localização do cilindro pelo número de dispositivos. Tal resultado, acrescido do tempo médio de latência (4 ms), produz o tempo médio de acesso a dados a ser considerado para os discos envolvidos.

Utilizando esse programa simulador, foram realizadas simulações considerando o uso de dois, três e quatro discos. Resultados relativos aos tempos médios de resposta obtidos são apresentados na figura 5.8.

Figura 5.8 - Tempos de Resposta Para Diferentes Números de Discos

O que se observa dos resultados acima é que, quando comparada ao esquema que considera um único disco, a técnica apresentada provoca reduções mesmo quando um único

usuário é considerado. Essas reduções nos tempos de resposta são razoáveis (no mínimo 25%) para cargas leves e bastante acentuadas para altas cargas. Para a carga de 70 usuários, o uso de dois discos proporciona reduções de até 66% nesses tempos; já o uso de três discos promove ganhos da ordem de 79%, quando comparado ao uso de um disco, e de 39% quando comparado ao de dois discos. Tais resultados mostram os benefícios que podem ser obtidos no desempenho do servidor de arquivos de referência a partir desse esquema de replicação de discos (sem introduzir problemas quanto a consistência das informações).

Da figura 5.8 nota-se também que os ganhos obtidos para quatro discos são pouco superiores aos relativos a três, de onde se conclui que tal acréscimo (de três para quatro discos), com seus custos adicionais, não traria benefícios mais significativos ao desempenho do servidor que justificassem tal acréscimo. Isso acontece porque, já para três discos, as taxas de utilização desses dispositivos são baixas, mesmo para cargas pesadas (14% para 70 usuários), e com isso observa-se filas dos discos sempre pequenas, reduzindo bastante o número de solicitações que são executadas de forma concorrente.

Salienta-se que a técnica em questão não exige que os discos mantenham as mesmas informações (como ocorre com o esquema anterior), eliminando assim a necessidade de replicação das solicitações de escritas, em operações normais de trabalho, que causaria degradação no desempenho do servidor. No entanto, a fim de obter maior segurança dos dados pode-se utilizar os espaços livres dos discos para replicar dados. As degradações no desempenho do servidor de arquivos devido a replicação das solicitações de escritas podem ser bastante atenuadas se cada escrita for executada em apenas um disco e em períodos ociosos replicada nos outros discos, solução que não pode ser aplicada à técnica anterior devido a necessidade de se manter os discos com as mesmas informações.

5.2.3 Comparação Entre as Técnicas Abordadas

Nas seções 5.2.1 e 5.2.2 são descritas algumas técnicas que podem ser usadas para a obtenção de servidores de arquivos com alto desempenho. Nessas seções são mostradas,

basicamente, duas maneiras de diminuir o tempo de execução especificamente relativo ao sistema de disco: a redução no número de acessos a esse sistema e o aumento no número de discos.

A primeira solução é descrita na seção 5.2.1 e mostra que, através do uso de técnicas de "cache" de disco, a redução no número de acessos a disco pode resultar em ganhos acentuados no desempenho de um servidor de arquivos. As formas de implementação de "cache" apresentadas (direcionadas tanto para leitura como para escrita em disco) podem ser implementadas utilizando a memória principal do servidor. No entanto, esses mecanismos podem apresentar grandes inconvenientes. No caso do "cache" relativo a operações de leitura de dados, a obtenção de alta taxa de acerto pode implicar altos custos adicionais (devido ao tamanho necessário para o "cache") e ganho limitado de desempenho (devido ao tratamento para evitar inconsistência entre os dados armazenados em disco e em "cache"). Com relação ao "cache" de escrita, uma vez que a memória principal é volátil, podem surgir sérios problemas relacionados a perdas de informações (devido a quebra do servidor, por exemplo). Embora esse problema possa ser bastante atenuado através do uso de memórias não-voláteis, essa solução acarreta uma série de outros inconvenientes (por exemplo relativos a custo e a gerenciamento e tamanho do "cache"). Salienta-se também que a adoção de técnicas de "cache" para melhorar o desempenho do servidor de arquivos não traz nenhum benefício quanto a segurança dos dados contra falhas de disco.

A seção 5.2.2 apresenta uma forma alternativa para reduzir o tempo total relativo ao disco: a replicação desse tipo de dispositivo. No caso da técnica que considera os acessos a discos de forma convencional, o nível dos benefícios obtidos depende do tipo de carga de trabalho imposto ao servidor de arquivos: pouco ganho é obtido para cargas leves e reduções expressivas nos tempos médios de resposta são registradas quando da imposição de cargas pesadas. Um ponto importante dessa técnica é que os ganhos são obtidos sem introduzir problemas quanto a consistência das informações, como o que acontece nos esquemas que consideram "caches" de disco. Além disso, essa técnica aumenta a segurança dos dados já armazenados nos discos com relação a falhas nos mesmos. A desvantagem desse esquema é principalmente relativa ao custo de sua implantação (utilização de vários discos) e a necessidade de se manter os discos com as

mesmas informações (o que pode provocar degradação no desempenho do servidor de arquivos). A outra técnica apresentada na seção 5.2.2 (acessos apenas a áreas parciais dos discos) promove, além das vantagens observadas para o método anterior, ganhos no mínimo razoáveis para todos os tipos de carga (inclusive considerando um usuário) e pode eliminar a necessidade de se manter os discos com as mesmas informações. Todavia, o inconveniente relativo aos custos para a implantação dessa técnica é o mesmo do caso anterior.

O objetivo dentro deste item 5.2 foi o de analisar algumas técnicas que podem ser aplicadas para a obtenção de servidores de arquivos com alto desempenho. A tabela 5.1 e a figura 5.9 apresentam comparações entre os tempos de resposta obtidos nas simulações para as técnicas discutidas neste item. Nessas comparações, na técnica de "cache" para leitura é considerada uma taxa de 100% e na de escrita atrasada o tempo de atraso usado foi de 5 minutos (observar que foram utilizadas as condições mais favoráveis para essas duas técnicas); no caso de uso de múltiplos discos, ambas as técnicas consideram três dispositivos.

N. Usuários	1	40	70
Padrão	6,8	11,2	24,2
Leitura 100%	4,4	5,7	8,7
Tempo Ocioso	3,7	4,5	6,3
Atraso 5 min.	5,3	7,5	13,1
3 Discos Conv.	6,8	7,2	8,3
3 Discos Parc.	4,4	4,6	5,0

Tabela 5.1 - Comparação Entre as Técnicas Abordadas.

Figura 5.9 - Comparação Entre as Técnicas Abordadas.

A partir das comparações acima e das observações anteriores, relativas às análises, conclui-se que a técnica que utiliza vários discos e leva em conta acessos apenas a áreas parciais desses recursos é a mais adequada para ser aplicada, porque além de proporcionar reduções bastante acentuadas nos tempos médios de resposta do servidor, não introduz problemas quanto a consistência das informações e aumenta a segurança dos dados já armazenados em disco, pois praticamente elimina a possibilidade de perdas desses dados devido a falhas de disco. Segundo as simulações, para os níveis de carga de trabalho considerados, o uso de três discos no servidor deve apresentar os melhores resultados sem provocar maiores custos de implantação. Apesar de acarretar custos adicionais, a utilização dessa técnica possibilita a obtenção de servidores de arquivos com alto desempenho e alta confiabilidade.

5.3 A Estrutura Proposta e os Níveis de Carga de Trabalho

De acordo com os resultados obtidos nas simulações realizadas, um servidor de arquivos, estruturado basicamente com um processador com velocidade 5 vezes a do MC68020, três discos com tempo médio de acesso a dados de 12 ms, sendo os acessos restritos apenas a

regiões parciais contíguas desses discos, e "cache" de disco com taxa de acerto em torno de 70%, deverá apresentar bom desempenho quando submetido aos níveis de carga de trabalho considerados nas simulações. Observa-se que para esse servidor apresentar níveis de desempenho da ordem dos obtidos nas simulações torna-se necessário que o gerenciamento dos discos seja feito de forma concorrente.

No entanto, algumas observações devem ser feitas com relação ao esquema de estruturação de um servidor de arquivos utilizando vários discos e aos níveis de carga de trabalho aos quais poderia ser exposto.

A partir das simulações realizadas com o modelo do servidor de arquivos composto por um processador e três discos (conforme as considerações feitas anteriormente) obteve-se informações quanto as taxas de utilização dos recursos envolvidos. A figura 5.10 apresenta essas taxas (em porcentagem) quando a carga de trabalho varia.

Figura 5.10 - Taxas de Utilização para os Sistemas de Processamento e de Discos.

Observa-se a partir da figura 5.10 que as taxas de utilização dos discos são sempre baixas, mesmo para cargas pesadas (14% para 70 usuários, uma carga de trabalho considerada alta para o tipo de servidor considerado).

O fato de um recurso apresentar taxa de utilização baixa significa que esse passa por longos períodos de ociosidade, sem executar qualquer ação, sendo o comprimento médio da sua fila sempre curto. Então, como os discos apresentaram baixas taxas de utilização, para os níveis de carga de trabalho considerados, pode-se concluir que o número de solicitações que seriam atendidas de forma concorrente pelo sistema de discos é pequeno. Isso inviabiliza a utilização de um número maior de discos na estrutura do servidor de arquivos, procedimento que poderia ser usado na tentativa de provocar diminuição significativa nos tempos totais relativos ao sistema de discos. (Observar a figura 5.8, onde se nota que, segundo as simulações, a utilização de quatro discos em vez de três na estrutura do servidor de arquivos não deve trazer benefícios mais significativos ao desempenho do mesmo que justifiquem os custos adicionais envolvidos.)

Baseado no exposto, as reduções nos tempos em discos só seriam significativas se os tempos de execução nesses recursos fossem diminuídos (por exemplo, utilizando dispositivos mais rápidos) ou evitando o acesso aos mesmos (através do uso de "cache" de disco). Como o disco mais rápido atualmente já está sendo considerado para fazer parte da estrutura do servidor de arquivos, a solução que poderia ser adotada para diminuir os tempos relativos ao sistema de disco é a que utiliza "cache" de disco. Entretanto, esta solução pode apresentar uma série de inconvenientes, como os descritos no item 5.2.

Dessa forma, a estrutura utilizando três discos de 12 ms pode ser considerada a mais adequada para que um servidor de arquivos apresente bom desempenho quando submetido aos níveis de carga de trabalho considerados nas simulações, sem trazer efeitos inconvenientes ao servidor.

Salienta-se que, mesmo para níveis de carga de trabalho mais altos, a técnica que utiliza vários discos com acessos apenas a áreas parciais dos mesmos deve continuar sendo a mais

adequada para a obtenção de um servidor de arquivos com as qualificações descritas acima (apesar dos custos adicionais de implantação). Dependendo dos níveis dessa carga, um número maior de discos pode ser necessário, exigindo a utilização de um processador, ou de um conjunto destes, que possa manter o nível de processamento quando gerenciando todos os discos de forma concorrente.

5.4 Considerações Finais

Este capítulo apresentou a análise de alguns procedimentos que podem ser aplicados para a obtenção de servidores de arquivos com alto desempenho. Para auxiliar na realização dessas análises foi utilizado, como referência, o modelo de simulação do servidor de arquivos do sistema distribuído TRICE, descrito no capítulo quatro. Os resultados mais importantes das simulações e as observações relevantes sobre cada técnica considerada foram apresentados aqui.

O sistema de simulação foi extremamente útil para demonstrar qual recurso do servidor limitava o seu desempenho e a partir daí qual a medida mais adequada a ser adotada para diminuir aquelas limitações.

No item 5.2 são analisadas algumas técnicas que podem ser aplicadas para a obtenção de servidores de arquivos com alto desempenho; suas vantagens e desvantagens, bem como resultados de simulações considerando essas técnicas, são apresentadas e discutidas. A seção 5.2.3 compara as técnicas abordadas. Baseado nos resultados das simulações e nas vantagens e desvantagens de cada técnica considerada nas análises, concluiu-se que a técnica utilizando vários discos em que os acessos são feitos apenas a áreas parciais dos mesmos é a mais adequada para ser aplicada em servidores de arquivos, a fim de obter sistemas com grande desempenho e alta confiabilidade.

De acordo com o modelo de simulação, para que esses servidores venham a apresentar níveis de desempenho da ordem dos obtidos nas simulações torna-se necessário que o gerenciamento dos discos seja feito de forma concorrente.

O item 5.3 mostra que a estrutura composta pelos recursos sugeridos (um processador com fator 5 para a velocidade e três discos com tempo médio de acesso a dados da ordem de 12 ms, com acessos a áreas parciais dos mesmos) é a mais adequada para que um servidor de arquivos apresente grande desempenho, além de alta confiabilidade, quando submetido aos níveis de carga de trabalho relevados nas simulações. Observou-se que, mesmo para cargas ainda mais altas, a consideração da técnica que utiliza vários discos deve continuar sendo a mais adequada. Observação também é feita quanto a possível necessidade de se utilizar um maior número de discos para cargas muito altas, exigindo a adoção de um processador ou de um conjunto destes que possa manter o nível de processamento quando gerenciando todos os discos de forma concorrente.

Por fim, salienta-se que, embora as conclusões obtidas a partir das análises aqui realizadas sejam relativas ao servidor de arquivos do sistema TRICE, essas conclusões podem ser estendidas a outros servidores. Ainda, os modelos e os programas considerados nos processos de simulação realizados foram validados utilizando técnicas direcionadas a isso, como as descritas no capítulo quatro.

CAPÍTULO SEIS

EXEQUIBILIDADE DE IMPLEMENTAÇÃO DE UM SERVIDOR DE ARQUIVOS

No capítulo anterior foi apresentada uma técnica que pode ser utilizada para a implementação de servidores de arquivos com alto desempenho e grande confiabilidade. Esta técnica considera o uso de múltiplos discos onde os acessos, realizados em operações normais, são feitos apenas em áreas parciais dos dispositivos. O objetivo deste capítulo é demonstrar que é possível implementar a estrutura proposta para tais servidores, tanto a nível de "hardware" como de "software".

No item 6.1 são feitas considerações quanto a configuração para um servidor de arquivos a ser projetado seguindo a técnica citada e quanto a necessidade de que os acessos aos discos utilizados sejam gerenciados de forma concorrente pelo processador do servidor.

O item 6.2 mostra que é possível construir um servidor de arquivos seguindo a técnica de múltiplos discos com acessos a áreas parciais. São citados exemplos de recursos (processadores e linguagens de programação) disponíveis comercialmente, a partir dos quais tal servidor pode ser implementado. Também neste item é apresentado um exemplo de como construir um módulo, composto de processador e vários discos, que pode servir de base para o servidor considerado. No final, são descritos algoritmos a partir dos quais o sistema programacional do servidor de arquivos poderia ser implementado.

6.1 A Configuração do Servidor de Arquivos

No capítulo anterior foi apresentada uma técnica que pode ser utilizada com o objetivo de implementar servidores de arquivos (ou de modificar servidores existentes) a fim de obter sistemas com alto desempenho. Esta técnica, que considera a utilização de um subsistema composto por vários discos, onde os acessos aos mesmos em operações normais são feitos apenas em áreas parciais de cada um dos recursos envolvidos, apresenta uma série de vantagens sobre outras técnicas que visam atingir o mesmo objetivo. Uma das principais vantagens é que grande benefício a nível de desempenho pode ser obtido sem introduzir problemas quanto a consistência das informações (como acontece nos esquemas que consideram "caches" de disco).

Além da obtenção de grande desempenho, a técnica descrita possibilita a construção de servidores de arquivos de alta confiabilidade. Embora essa técnica não necessite que diferentes discos mantenham as mesmas informações, os arquivos podem ser replicados sobre vários discos, aumentando consideravelmente a segurança dos dados com relação a falhas desses dispositivos. Os efeitos negativos da replicação sobre o desempenho do servidor podem ser bastante atenuados se cada escrita for executada apenas em um disco e posteriormente sobre outros, nos momentos em que o servidor não tiver solicitações pendentes.

Assim, a aplicação da técnica de múltiplos discos com acessos a áreas parciais na construção de servidores de arquivos deve proporcionar a obtenção de sistemas com grande desempenho e alta confiabilidade.

Deve-se salientar que, de acordo com o modelo (descrito no capítulo cinco) utilizado nas simulações, cujos resultados demonstram a viabilidade e a eficiência da abordagem discutida acima, os discos devem ser gerenciados diretamente pelo processador do servidor. Entretanto, para o total aproveitamento das vantagens oferecidas a partir da adoção dessa técnica, principalmente com relação ao desempenho do servidor de arquivos, os acessos aos discos devem ocorrer de forma concorrente. Esse gerenciamento concorrente significa que as transferências de informações entre o processador e cada disco devem ser realizadas de modo independente, sem interferir nas operações de comunicação com os outros discos. Assim, além de força de

processamento suficiente para não limitar o desempenho do servidor de arquivos, o processador deve apresentar características que possibilitem o gerenciamento concorrente de tarefas (no caso específico, acessos a discos) de forma eficaz, proporcionando transferências de informações rápidas e seguras.

Desse modo, a estrutura básica de um servidor de arquivos, a ser implementado seguindo a técnica de múltiplos discos com acessos a áreas parciais, deve conter, além dos discos, um processador capaz de executar concorrentemente suas tarefas. Um outro fator que deve ainda ser analisado é a capacidade de processamento do processador adotado.

6.2 Exequibilidade de Implementação

A partir da descrição da configuração básica exigida para um servidor de arquivos, a ser projetado de acordo com a técnica de acessos parciais, torna-se necessário verificar a viabilidade de sua implementação a nível de "hardware" e de "software". Inclui-se aí a preocupação quanto a processador e a linguagem de programação que podem ser usados em tal implementação.

6.2.1 Exemplos de Recursos Disponíveis

Foi descrito no item 6.1 que o processador, a ser utilizado na estrutura proposta para o servidor de arquivos, deve apresentar características que possibilitem a execução concorrente de tarefas e deve ter velocidade de processamento suficiente para executar eficientemente os tipos de operações solicitadas, sem se tornar o fator limitante do desempenho do servidor. Quanto a linguagem de programação a ser considerada, esta deve ter características que não degradem o desempenho do processador utilizado e deve possuir mecanismos direcionados a execução concorrente de processos.

A seguir são apresentados exemplos de processadores e de linguagens de programação que podem ser utilizados para a implementação de um servidor de arquivos como o proposto.

O Processador Transputer

Os processadores Transputer [IN89a, IN89b], que fazem parte da família de componentes INMOS, são desenvolvidos com tecnologia RISC e utilizados na área de processamento paralelo. Esse processador pode executar tarefas de forma concorrente com muita eficiência. Os componentes mais recentes da família Transputer (linhas T800 e T9000) são microprocessadores de 32 bits, compostos de unidade de ponto flutuante de 64 bits, memória interna ("on-chip") de acesso rápido (o T800 possui 4 Kbytes de RAM estática e o T9000 tem 16 Kbytes) e barramento largo e multiplexado para dados e endereços (por exemplo, no T800 este barramento tem largura de 32 bits).

Características marcantes desses processadores são as suas conexões ("links") seriais bi-direcionais de comunicação, que permitem transferências sincronizadas de dados com outro Transputer ou subsistemas de modo rápido e seguro (os T800 e T9000 possuem quatro conexões físicas). Cada conexão de comunicação é composta por um canal serial de entrada e por um de saída, e está ligada à unidade central do processador através de uma interface de conexão. As interfaces de conexão são subsistemas autônomos para acesso direto a memória (DMA) que gerenciam as transferências de dados pelas conexões de forma independente. Como resultado dessa arquitetura, um processador Transputer pode simultaneamente executar processos internos e se comunicar sobre todas as conexões seriais [IN89a, IN89b]. Através dessas interfaces de conexão, vários dispositivos externos podem ser controlados de forma concorrente.

Outro ponto importante a salientar sobre os processadores Transputer é que estes possuem "hardware" específico embutido que providencia o chaveamento de processos de forma muito rápida [BU88, IN89a, IN89b]. O chaveamento de contexto, atividade que envolve a parada de execução de um processo e execução de outro, que em muitos processadores são tarefas gerenciadas por "software" (consumindo tempo extra do processador), é executado no Transputer por "hardware". O chaveamento de processos é um tipo de tarefa executada com muita frequência em servidores de arquivos, que devem atender pedidos originados em diversas estações de trabalho

(o chaveamento de processos em servidores pode ser observado na descrição do funcionamento geral de tais sistemas apresentada nos capítulos anteriores).

A partir da adoção de um Transputer como o processador para o servidor de arquivos proposto, os discos podem ser interfaceados em suas conexões seriais e dessa maneira gerenciados de forma concorrente.

O Processador Digital de Sinais

Um recurso muito utilizado em ambientes para processamento paralelo e outras aplicações em tempo-real é o processador digital de sinais (DSP) da TEXAS Instruments [LI87, PA88a, TE91]. A geração TMS320C4x da família TMS320 de DSP's é composta por processadores de 32 bits que possuem unidade de ponto flutuante de 40 bits, um co-processador de acesso direto a memória (DMA) para controle de canais de entrada e saída, memória interna de acesso rápido (dois blocos de 4 Kbytes de RAM, cada um suportando duplo acesso em um único ciclo), memória "cache" de instruções interna (até 128 instruções de 32 bits para seções de códigos mais repetidas podem ser armazenadas) e barramentos internos largos de dados e endereços (sete barramentos internos de 32 bits separados permitem a execução paralela de buscas de instruções de programas, de acessos a dados e de acessos DMA).

Uma característica importante do DSP é o co-processador de DMA. Este gerencia seis canais de comunicação externa independentes, que podem ler ou escrever em qualquer lugar dentro do espaço de memória de forma totalmente separada da unidade central de processamento (UCP). Através desses canais, de oito linhas de dados, informações externas podem ser transferidas concorrentemente com a operação normal da UCP; isso permite o interfaceamento de periféricos sem reduzir o desempenho do processador [TE91].

De acordo com o exposto, também um DSP pode ser utilizado como o processador do servidor de arquivos proposto e assim gerenciar discos interfaceados aos seus canais de comunicação de forma concorrente.

Linguagens Para Programação Concorrente

Atualmente, muitas linguagens de programação estão disponíveis para utilização em sistemas onde se tem a necessidade de execução concorrente de processos. Dois tipos de linguagens concorrentes podem ser considerados: as extensões de linguagens sequenciais e as linguagens desenvolvidas para aplicações concorrentes. As extensões (por exemplo, Pascal e C) apresentam a vantagem de serem linguagens básicas já conhecidas pelo usuário e portanto de mais fácil apredizagem. No entanto, normalmente essas linguagens não apresentam a eficiência e facilidade de ativação e comunicação entre processos concorrentes apresentados por linguagens desenvolvidas especificamente para programação concorrente (por exemplo, Occam2 e Ada).

A linguagem Ada [BA89a, BR90, LA92, WE83] é uma linguagem de alto nível criada sob patrocínio do Departamento de Defesa dos Estados Unidos em meados da década de 70. Ada é uma linguagem desenvolvida para a programação concorrente de sistemas. A unidade básica de concorrência em Ada é um módulo chamado tarefa (*task*). Cada tarefa consiste de duas partes: uma especificação e um corpo. Uma tarefa possui entradas (*entry*) que podem ser solicitadas por outras tarefas. A sincronização entre tarefas ocorre quando uma delas solicita uma entrada e a outra tarefa aceita a entrada, estabelecendo um ponto de comunicação que é efetuado utilizando a técnica de "rendezvous". A partir desse ponto é estabelecido o processo de comunicação entre as tarefas.

Occam2 [BU88, IN89a, JO88], projetada pela INMOS Corp., é uma linguagem de propósito geral com mecanismos específicos para programação de sistemas concorrentes. Um sistema programado em Occam2 pode ser descrito como uma coleção de processos (*process*) sequenciais, executados concorrentemente, que se comunicam através de canais unidirecionais. Esses processos podem ser executados de forma concorrente em um único processador ou em paralelo em diferentes processadores. Os canais são implementados como dispositivos de entrada e saída para comunicação entre processos executados em diferentes processadores. Para processos rodando de forma concorrente em um único processador, os canais são mapeados como variáveis

na memória interna. Occam 2 foi desenvolvida em paralelo com o processador Transputer, visando sua aplicação em sistemas concorrentes.

A linguagem **Pascal Concorrente** [AN83, HA75, HA94, ST92] é uma extensão do Pascal Sequencial com características que permitem concorrência. Seus principais aspectos são *processos* e *monitores*, e mais as declarações *init*, *delay* e *continue*, além do tipo de dado *queue*. No Pascal Concorrente, *processo* é um subprograma sequencial que pode ser executado simultaneamente com outros processos. Um *monitor* atua como um mecanismo para englobar dados compartilhados entre processos e para a sincronização de acesso aos dados. A declaração *init* processoX aloca espaço de memória para as variáveis locais do processoX e inicia sua execução. Quando um processo requer um recurso que está ocupado, o monitor pode executar um comando *delay*, que coloca o processo na estrutura *queue*. A declaração *continue* faz com que o processo em *queue* seja removido para execução.

Da mesma forma que Pascal, outras linguagens sequenciais apresentam versões direcionadas para a execução concorrente de tarefas; alguns exemplos são C Paralelo [ST92], Modula [WI85] e Prolog [SH86, SH89].

6.2.2 Um Exemplo de "Hardware"

Na seção anterior é citado dois exemplos de processadores que podem ser utilizados na estrutura de um servidor de arquivos a ser implementado seguindo a técnica de múltiplos discos com acessos a áreas parciais: o Transputer T800 e o DSP da Texas. Segundo as observações feitas naquela seção, se for utilizado como o processador do servidor proposto o T800, este deve desempenhar de forma eficaz as funções a ele atribuídas.

Como já salientado, o processador a ser considerado na estrutura do servidor de arquivos proposto deve gerenciar de forma concorrente os discos a ele conectados. Tendo isso em vista, para a utilização de um Transputer T800 na estrutura desse servidor, deve ser providenciada uma forma de interfaceamento dos discos a tal processador, de modo que a comunicação entre o T800 e cada um desses dispositivos seja feita de forma independente com relação aos outros

discos. Uma maneira de se alcançar este objetivo é através do interfaceamento dos discos às conexões seriais do processador Transputer.

Para demonstrar a viabilidade da construção de um servidor de arquivos utilizando um processador Transputer e vários discos, seguindo a técnica de acessos parciais, foi implementado um subsistema para permitir o interfaceamento entre uma conexão serial do Transputer T800 e um disco rígido (a consideração do T800 é devido a sua disponibilidade). Esse subsistema é descrito a seguir.

O Subsistema de Interfaceamento

A interface entre uma conexão serial de um processador Transputer e um disco rígido deve ser implementada como um subsistema capaz de providenciar as transferências de dados com segurança e rapidez.

Parte desse subsistema deve providenciar a conversão dos dados seriais do Transputer em dados paralelos a serem enviados ao disco e vice-versa. Para simplificar o subsistema considerado, a interface pode ser implementada utilizando o circuito adaptador de conexão serial C012. O circuito adaptador C012 [IN89a, IN89b], outro componente da família Transputer, permite o fácil interfaceamento de um processador Transputer com processadores comuns ou outros subsistemas. O C012 converte dados seriais de uma conexão do Transputer em dados paralelos, e vice-versa, e controla os protocolos de comunicação exigidos pelos processadores Transputer, simplificando razoavelmente o projeto de tal tipo de interface.

O circuito C012 é dotado de um porto serial (para ligação com uma conexão serial de outro componente Transputer) e de um porto paralelo de 8 bits (para interfaceamento com outros subsistemas) e possui registradores de controle e de dados para o seu gerenciamento. A operação desse circuito é controlada através de seu porto paralelo, pela leitura e escrita em seus registradores.

A construção do subsistema para conexão do T800 a um disco pode ser realizada utilizando um controlador "inteligente" (um microprocessador ou microcontrolador) para gerenciar a interface, oferecendo uma solução relativamente simples e de baixo custo [MI90, MI92]. Entretanto, esta solução apresenta dois inconvenientes:

- o desenvolvimento de "software": para rodar no dispositivo controlador, para gerenciar tanto o circuito adaptador C012 quanto o dispositivo periférico;
- desempenho do subsistema: o uso de um controlador inteligente pode provocar atrasos na transferência de dados, que dependendo da aplicação podem prejudicar o desempenho do sistema global.

A fim de eliminar a preocupação quanto ao desenvolvimento do programa para rodar no componente controlador da interface e principalmente minimizar os atrasos provocados por tal dispositivo, pode-se substituir este controlador por um subsistema capaz de gerenciar toda a interface. Esse subsistema pode ser composto por circuitos lógicos básicos (flip-flops, portas AND, OR, etc.), reduzindo ainda mais os custos de construção da interface.

Apesar do projeto que não considera a inclusão de um controlador inteligente na interface se tornar mais complexo que o projeto que inclui tal recurso, a ativação de todos os sinais necessários para o controle e para a leitura/escrita de registradores envolvidos (tanto do C012 quanto do dispositivo periférico) é feita por "hardware". Isso pode trazer grandes benefícios ao desempenho do sistema, uma vez que os atrasos verificados com a inclusão do controlador, em que o controle era feito por "software", podem ser reduzidos. Deve-se salientar que, dessa forma, o controle da interface pode ser feito pelo próprio Transputer, sem a intervenção de controladores intermediários.

Assim, a opção escolhida, para a construção de uma interface entre uma conexão serial do Transputer T800 e um disco rígido, foi utilizar um circuito adaptador C012 e componentes lógicos básicos. O diagrama de blocos para o interfaceamento é apresentado na figura 6.1.

Figura 6.1 - Diagrama de blocos do interfaceamento T800 - disco IDE.

Para a implementação do subsistema de interfaceamento foi considerado o disco com interface padrão IDE (Integrated Drive Electronics) [AL91, LU94, RI95] por dois motivos: disponibilidade de tal tipo de disco e facilidade de interfaceamento e controle de seus registradores. O padrão IDE de interface para disco rígido se destaca pela alta taxa de transferência que possibilita, além de facilidade de implementação e modos de operação e controle. A controladora IDE [AL91, WD90] possui vários registradores (o de estado, comando e de endereçamento em disco) através dos quais é programada pelo seu hospedeiro trocando informações de comando, de estado e de dados. Todos os registradores têm tamanho de 8 bits, exceto o de dados que é de 16 bits.

O Projeto e Operação da Interface

A interface projetada utiliza um circuito adaptador C012 e um subsistema lógico composto de registradores, "buffers" e portas lógicas básicas. A figura 6.2 apresenta o diagrama de blocos da interface projetada.

Figura 6.2 - Diagrama de blocos da interface projetada.

A conexão serial do circuito adaptador C012 (pinos LkIn e LkOut) deve ser ligada a uma conexão serial de um processador Transputer; o barramento (ou porto) paralelo (BP) de 8 bits do C012 é interfaceado aos registradores R1 a R5 do subsistema lógico (os registradores R1 a R5 são circuitos integrados TTL de 8 bits).

O registrador R1 tem a função de armazenar e enviar informações de controle e de endereçamento à interface IDE e está ligado (através de um buffer) aos pinos de controle da IDE. Os outros registradores são interfaceados 2 a 2 ao barramento de dados (de 16 bits) da controladora IDE (R2/R3 para enviar dados a IDE e R4/R5 para receber dados da IDE). Quatro buffers isolam, quando necessário, R2/R3 da controladora e R4/R5 do adaptador C012.

As transferências entre o processador Transputer e a interface projetada são realizadas byte a byte (esta é a forma de transferência, através das conexões seriais, utilizada pelos componentes Transputer). As comunicações entre a interface projetada e a controladora IDE podem ser realizadas no formato de um byte (no caso de informações para controle e endereçamento) ou de dois bytes (quando transferindo dados). Um byte enviado pelo Transputer ao C012 é escrito em um dos registradores R1 (para controle), R2 ou R3 (ambos para dados), para

posterior transferência para a IDE. Um byte lido da IDE é escrito no registrador R4 ou R5 que, transferido ao C012, é enviado ao processador.

A interface funciona sob o comando do processador Transputer. Um processo de leitura ou de escrita em um registrador da IDE é inicializado quando o C012 recebe um byte do Transputer e o armazena em R1. Esse processo é controlado pelos circuitos do subsistema lógico de acordo com o valor do byte em R1, que descreve o tipo de ação a ser executada. Estes circuitos são ativados pelos sinais InInt e OutInt do adaptador C012. O pino InInt quando em nível alto sinaliza que um byte foi recebido do Transputer e está pronto para ser lido através do barramento paralelo. O pino OutInt em nível alto sinaliza quando o C012 está pronto para receber um byte pelo porto paralelo, para retransmiti-lo ao processador.

O byte em R1 é composto por três bits para o endereçamento de um dos sete registradores da IDE, por quatro bits para a ativação dos sinais de controle da IDE (dois para os pinos de seleção da controladora e dois para os pinos IOW e IOR) e de um bit para especificar se a transferência entre a interface e a controladora deve ocorrer em 8 ou 16 bits. Num processo de leitura, o byte de R1 é transmitido para a IDE e para o subsistema lógico a fim de que a informação lida de um dos registradores da controladora seja armazenada em R4 (para informações de 8 bits), e em R5 (se a informação for de 16 bits), com posterior transferência do(s) byte(s) para o C012. Em um processo de escrita na IDE, um byte para R1 e um byte para R2 (dados de 8 bits), e também um para R3 (se a informação for de 16 bits), são transmitidos pelo Transputer, para posterior escrita num registrador da IDE.

Assim, os protocolos de comunicação entre o T800 e a interface projetada seguem os seguintes formatos:

- (1) um byte de saída e um ou dois bytes de entrada, no caso de leitura na controladora IDE;
- (2) três bytes de saída, quando escrevendo no registrador de dados da IDE; ou
- (3) dois bytes de saída, quando escrevendo nos outros registradores da IDE.

A interface descrita foi implementada em "wire-wrap". O projeto detalhado da interface é apresentada na figura B.1 do apêndice B.

Testes Realizados com a Interface

Para demonstrar o funcionamento e a eficiência da interface implementada, foram realizados vários testes utilizando uma configuração composta por um módulo Transputer T800 e por um sistema de disco padrão IDE, além da interface construída. O módulo Transputer usado é constituído de um processador T800 de 20 Mhz e de 2 Mbytes de memória RAM. O disco utilizado nos testes foi um Maxtor de 127 Mbytes, com tempo médio de busca do cilindro de aproximadamente 15 ms, com velocidade de 3551 r.p.m.

O módulo Transputer foi acoplado a um computador hospedeiro (modelo PC), a partir do qual o ambiente de desenvolvimento, utilizado para a implementação e a execução do programa que rodou no T800, foi inicializado (o ambiente usado foi o TDS - Transputer Development System da INMOS [IN90], sendo o programa implementado em linguagem Occam2). Os periféricos do hospedeiro eram utilizados pelo módulo Transputer como dispositivos para a entrada e a saída de dados do sistema.

Utilizando a configuração descrita, foram testados vários comandos reconhecidos pela interface padrão IDE, como por exemplo a leitura e a escrita de setores (no caso, de tamanho 512 bytes), posicionamento das cabeças de leitura/escrita na trilha desejada do disco e verificação da localização destas cabeças [AL91, WD90]. A partir desses testes, pôde-se comprovar o correto funcionamento da interface implementada.

A fim de verificar o desempenho da interface construída, foram realizados alguns testes com o objetivo de obter os valores para as taxas de transferências entre o processador T800 e o sistema de disco IDE considerado. As conexões seriais do T800 e do adaptador serial C012 podem ser programadas, por "hardware", para velocidades de 5, 10 e 20 Mbits/s [IN89b]. Nos testes realizados, tais conexões foram programadas para a velocidade de 20 Mbits/s (o que corresponde a uma taxa efetiva de 1740 Kbytes/s [IN89b]) para se poder observar a maior taxa de

transferência permitida pela interface construída. Observa-se que, devido aos protocolos de comunicação exigidos pelos processadores Transputer, para a transferência de um byte é necessário o envio de mais 3 bits de controle junto aos 8 de dados, totalizando 11 bits por byte transferido.

Para que os valores obtidos para essas taxas refletissem apenas os tempos referentes as transferências de dados entre o Transputer e a controladora do disco, sem considerar os tempos relativos a acessos a disco e a transferências entre a controladora e o disco, foram testados comandos direcionados a leitura e a escrita no "buffer" de setores da controladora [AL91, WD90], minimizando assim a influência das características do dispositivo considerado.

Os programas utilizados nesses testes para rodar no T800, escritos em linguagem Occam2, seguiram o algoritmo mostrado a seguir, no qual se observa os pontos onde ocorreram as tomadas dos tempos de transferências (utilizando os canais de tempo, "TIMER", disponíveis na linguagem Occam2 [BU88]). Os programas utilizados nos testes são apresentados no apêndice C.

Início

enviar parâmetros para programação da IDE;

se for leitura da IDE

obter t1;

leitura de setor(es) na controladora IDE;

obter t2;

se for escrita na IDE

obter t1;

escrita de setor(es) na controladora IDE;

obter t2;

tempo decorrido = t2 - t1;

Fim.

Os valores médios para as taxas de transferências obtidos nos testes realizados são os seguintes:

- escrita em disco: 1208 Kbytes/s;
- leitura em disco: 1422 Kbytes/s.

Embora os tempos relativos aos acessos a disco e as transferências entre este dispositivo e a sua controladora não influenciem os resultados obtidos nos testes, outros fatores afetam significativamente esses resultados. Dentre estes fatores tem-se os tempos para ativações do adaptador C012 [IN89b] e dos componentes do subsistema lógico da interface projetada [TE87] e principalmente a velocidade de transferências das conexões seriais do T800 e do adaptador C012.

A influência das conexões seriais pode ser verificada analisando-se o tempo de transferência. Para a velocidade de 20 Mbits/s, o tempo de transferência de cada bit é de 50 nanossegundos (ns). Com isso, o tempo entre inícios de transferências de bytes de dados entre o T800 e o adaptador C012 deveria ser de 550 ns (para transferir um byte são necessários 11 bits, devido aos protocolos de comunicação). Entretanto, o tempo mínimo para a transferência de um byte do T800 para o C012 é de 745 ns [IN89b]. Isso ocorre porque o menor tempo exigido para a ativação interna do C012 é de 95 ns e o tempo gasto no envio de cada sinal de reconhecimento (composto de 2 bits) pelo C012 ao T800 (para habilitar o início de outra transferência) é de 100 ns. Então, daquele tempo mínimo total de 745 ns, 650 ns são dependentes exclusivamente da velocidade de comunicação das conexões seriais, o que mostra a grande influência dessa velocidade nas taxas de transferências obtidas utilizando a interface implementada.

Deve-se observar que, para esse tempo mínimo entre inícios de envios de bytes (de 745 ns), a máxima taxa de transferência do T800 para o C012 é de aproximadamente 1340 Kbytes/s, valor pouco superior ao resultado obtido nos testes com a interface construída, que é de 1208 Kbytes/s. No sentido contrário, do C012 para o T800, o tempo mínimo entre inícios de transferências de bytes é de 645 ns, resultando em uma taxa máxima de transferência de aproximadamente 1550 Kbytes/s, também pouco superior ao valor de 1422 Kbytes/s encontrado

nos testes. Assim, os valores das taxas de transferências medidas podem ser considerados bastante razoáveis quando comparados com as máximas taxas permitidas pela ligação T800 - C012, pois aqueles valores representam mais de 90% dessas taxas máximas. Isso mostra a eficácia da interface implementada com relação aos recursos utilizados na sua construção (o adaptador C012) e nos testes (o processador T800).

Através do exposto, mostrou-se uma forma de construir um subsistema que permite o interfaceamento de discos a conexões seriais de processadores Transputer. A partir da replicação da interface desenvolvida, um Transputer pode se comunicar com vários discos de forma independente através de suas conexões seriais, possibilitando o gerenciamento concorrente desses dispositivos.

Assim, um sistema composto por um processador Transputer T800 (com sua memória RAM externa associada) e vários discos, interfaceados às conexões seriais do T800, constitui um módulo básico que pode ser usado para a implementação de um servidor de arquivos projetado de acordo com a técnica de múltiplos discos com acessos parciais.

Como descrito anteriormente, as taxas de transferências obtidas com a utilização da interface construída são fortemente influenciadas pela velocidade de comunicação das conexões seriais dos dispositivos Transputer envolvidos. Se aquelas taxas não forem consideradas satisfatórias para a implementação do servidor de arquivos proposto, outros componentes com portos de comunicação de maior velocidade podem ser considerados, através da implementação de uma interface similar à descrita anteriormente. Nesse caso, o circuito adaptador de conexões seriais C101 e o processador T9000 (ambos da família Transputer), podem ser utilizados.

A Capacidade de Processamento do T800

No capítulo cinco concluiu-se que um servidor de arquivos, estruturado com um processador com fator 5 de velocidade relativa ao do MC68020 e três discos com tempo médio de acesso de 12 ms (gerenciados concorrentemente pelo processador), nos quais os acessos são

realizados apenas em áreas parciais dos dados, deve apresentar alto desempenho se for submetido a níveis de serviço como os considerados nas simulações realizadas.

Com relação a velocidade do processador T800, deve-se salientar sua alta eficiência no chaveamento de processos, tanto devido ao seu "hardware" quanto a possibilidade de sua aplicação junto a linguagem Occam, desenvolvida especificamente para programação concorrente.

Em [SA90] é feita uma comparação entre os tempos de execução de alguns processos, escritos em linguagem Occam2, num processador Transputer T800 e os tempos de execução dos mesmos tipos de processos, escritos em BCPL (a linguagem em que foi escrito o código do servidor de arquivos do TRICE), num MC68020 (o processador original daquele servidor). Descreve-se que foram realizados vários tipos de testes considerando operações frequentemente executadas por servidores de arquivos, como as operações de chaveamento de processos, implementações de listas, operações de procuras (como as realizadas em diretórios) e operações gerais (associações, testes, operações aritméticas e lógicas). A conclusão em [SA90] é que o conjunto T800-Occam2 tem velocidade superior a 17 vezes a do conjunto MC68020-BCPL quando executando os tipos de operações descritos acima. Isso significa que o Transputer T800, executando processos escritos em Occam2, pode ser usado para compor a estrutura proposta para o servidor de arquivos, pois a sua velocidade (17 vezes) excede o fator de 5 para a velocidade do processador, valor encontrado a partir das simulações.

Quanto ao número de discos que podem ser interfaceados a um Transputer T800, observa-se que este processador possui quatro conexões seriais, a partir das quais os três discos sugeridos para a estrutura do servidor de arquivos, se este for submetido a níveis de serviço equivalentes aos considerados nas simulações, podem ser conectados.

Assim, o processador Transputer T800 pode ser utilizado para se obter um servidor de arquivos que possibilite o gerenciamento concorrente de três discos de forma eficaz e segura, e com níveis de desempenho da ordem dos atingidos nas simulações.

Deve-se salientar que, como observado no capítulo cinco, o número necessário de discos para que o servidor de arquivos sempre apresente alto desempenho depende dos níveis de carga de trabalho aos quais é submetido; ou seja, quanto maior for o serviço exigido do servidor, maior deve ser a quantidade de discos necessários. Uma vez que cada processador T800 possui quatro conexões seriais, um único T800 não será suficiente para permitir o interfaceamento, através de suas conexões seriais, de uma quantidade de discos superior a quatro. Uma solução para isso é utilizar um conjunto de processadores T800, permitindo que um maior número de discos possa então ser interfaceado às conexões seriais disponíveis nesse sistema de processamento. Observa-se ainda que, além de permitir o gerenciamento concorrente de muitos discos, essa solução possibilita a execução paralela de outros tipos de tarefas dentro do sistema de processamento (o que não era possível com um único elemento processador), devendo elevar os níveis de desempenho do servidor de arquivos.

Os Efeitos da Utilização do T800 e de Occam2

As utilizações do Transputer e de Occam2 devem proporcionar ganhos ao desempenho do servidor de arquivos ainda maiores que aquele obtido considerando um processador com velocidade 5 vezes a do recurso original (estipulado como uma boa capacidade de processamento, através das simulações). Isso ocorre porque a velocidade do T800, executando processos escritos em Occam2, é superior a 17 vezes a do processador MC68020 (o processador original dos servidor de arquivos), rodando processos em BCPL.

A fim de verificar os benefícios que poderiam ser obtidos no desempenho do servidor de arquivos com as adoções do Transputer T800 e de Occam2, foram realizadas novas simulações considerando o fator de 17 para a velocidade do processador no modelo que relevou o uso da técnica de múltiplos discos com acessos a áreas parciais dos mesmos. A figura 6.3 apresenta os tempos médios de resposta obtidos nas simulações considerando processadores com velocidades 5 vezes e 17 vezes a do recurso original.

Figura 6.3- Tempos de Resposta para Diferentes Processadores

A partir desses resultados observa-se que a consideração no modelo de um fator de 17 para a velocidade do processador produziu reduções superiores a 20% nos tempos médios de resposta para todos os níveis de carga de trabalho com relação aos valores resultantes da consideração do processador com fator 5 de velocidade, mostrando outro benefício que pode ser obtido com a utilização do processador Transputer T800. Observar também que esses valores representam reduções de no mínimo 50% para cargas leves e de até 84% para altas cargas quando comparados aos resultados obtidos com o modelo base que considera o uso de apenas um disco.

6.2.3 Um Exemplo de Algoritmo

Na seção 6.2.2 mostrou-se a exequibilidade de implementação, a nível de "hardware", de um servidor de arquivos a ser projetado considerando a técnica de múltiplos discos com acessos parciais. O passo seguinte é mostrar que a implementação desse servidor também é possível a nível de "software". Para isso, é apresentado a seguir um algoritmo geral a partir do qual é possível desenvolver um sistema programacional que gerencie de forma concorrente todo o

"hardware" do servidor de arquivos a ser construído, permitindo o tratamento independente dos vários discos utilizados:

Execução Concorrente { de processos }

Início

Processo Entrada/Saída;

Processo Processador;

Processo Disco;

Fim.

Neste algoritmo geral, três processos seriam executados de forma concorrente pelo processador do servidor de arquivos. O processo *Entrada/Saída* receberia solicitações dos usuários e as colocaria na fila do processador; solicitações atendidas pelo sistema de disco também seriam retornadas a esse processo e assim inseridas naquela fila. As solicitações aguardariam nessa fila a desocupação do processador; quando tal recurso estivesse livre, a solicitação da frente da fila seria enviada ao processo *Processador*, que cuidaria da execução das solicitações. O processo *Entrada/Saída* também estaria incumbido de enviar respostas aos clientes.

O processo *Processador* receberia solicitações do processo *Entrada/Saída* e providenciaria as execuções das mesmas no processador do servidor de arquivos. O processamento de cada solicitação continuaria sem interrupção até o seu final ou até que um acesso a disco se tornasse necessário. Se o processamento da solicitação chegasse ao seu final, uma resposta seria enviada ao processo *Entrada/Saída*. Se um acesso a disco fosse necessário, a solicitação seria enviada ao processo *Disco*, para que tal acesso fosse providenciado.

O processo *Disco* estaria incumbido de providenciar acessos aos discos utilizados no servidor de arquivos e trocar dados com os mesmos. Ao receber uma solicitação do processo *Processador*, o processo *Disco* determinaria em qual disco do sistema estariam as informações necessárias. A solicitação de acesso seria, então, colocada na fila do disco selecionado, para

aguardar pela liberação do mesmo. As solicitações atendidas seriam retornadas ao processo *Entrada/Saída* para que o processador do servidor continuasse a sua execução.

As implementações dos três processos citados poderiam seguir os algoritmos apresentados a seguir.

Processo Entrada/Saída

{ funções: receber solicitações dos usuários e do sistema de disco, e colocá-las na fila do processador; enviar respostas aos usuários. }

Execução Concorrente { de sequências de comandos }

Início

Para Sempre

Recebe *solicitação*; { de um usuário ou do sistema de discos }

Coloca *solicitação* na fila do processador;

Para Sempre

Recebe pedido de *nova solicitação* do processo *Processador*;

Verifica *próxima solicitação* na fila do processador;

Envia *próxima solicitação* para o processo *Processador*;

Para Sempre

Recebe *resposta* do processo *Processador*;

Envia *resposta* ao usuário;

Fim.

Processo Processador

{ função: executar a solicitação, até o final da mesma ou até que acesso a disco seja necessário }

Início

Para Sempre

Envia pedido de *nova solicitação* ao processo *Entrada/Saída*;

Recebe *próxima solicitação* do processo *Entrada/Saída*;

Processamento da solicitação até interrupção;

Verifica se é final de processamento:

Sim: envia *resposta* ao processo *Entrada/Saída*;

Não: envia *solicitação* ao processo *Disco*;

Fim.

Processo Disco

{ funções: seleccionar o disco ao qual fazer acesso e trocar dados com o disco seleccionado }

Execução Concorrente { de sequências de comandos }

Início

Para Sempre

Recebe *solicitação* do processo *Processador*;

Selecciona o disco para fazer acesso;

Envia *solicitação* para o subprocesso *disco-selec1*; { $I = 1$ a N }

{ Início do subprocesso *disco-selec1* }

Para Sempre

Recebe *solicitação*;

Coloca *solicitação* na fila do disco1;

Para Sempre

Verifica *próxima solicitação* na fila do disco1;

Determina local dos dados no disco1;

Envia comando de posicionamento ao disco1;

Transfere dados com o disco1;

Retorna *solicitação* ao processo *Entrada/Saida*;

{ Final do subprocesso *disco-selec1* }

{ Subprocesso *disco-selec2* }

.

.

{ Subprocesso *disco-selecN* }

Fim.

Nos algoritmos anteriores, o comando *Execução Concorrente* significa que os processos ou sequências de comandos devem ser executados de forma concorrente dentro do servidor de arquivos, e o comando *Para Sempre* significa um laço infinito de execução dos comandos englobados. No processo *Disco*, os subprocessos *disco-selec2* a *disco-selecN* são similares a *disco-selec1*, e N é o número de discos considerados no projeto do servidor de arquivos.

Seguindo esses algoritmos, um sistema programacional poderia ser implementado utilizando alguma linguagem de programação que possua ferramentas adequadas voltadas para a execução concorrente de processos, como as exemplificadas na seção 6.2.1

6.3 Considerações Finais

O objetivo deste capítulo é mostrar que a implementação de servidores de arquivos de alto desempenho, projetados de acordo com a técnica de múltiplos discos com acessos apenas a áreas parciais, é exequível tanto a nível de "hardware" como de "software".

A fim de mostrar a viabilidade de implementação desse servidor a nível de "hardware", foi descrito o desenvolvimento de um módulo, composto por um processador (com sua memória externa associada) e vários discos, que pode servir de base para o desenvolvimento do referido servidor. Um tipo de processador que pode ser considerado para compor tal módulo é o Transputer T800 e uma maneira de interfacear os discos ao T800, para que esses dispositivos possam ser gerenciados de forma concorrente, é conectá-los ao Transputer através de subsistemas de interfaceamento similares ao descrito neste capítulo. Esse módulo-base também pode ser utilizado em um servidor de arquivos existente, desempenhando as atividades relativas ao gerenciamento do sistema de disco desse servidor.

O subsistema descrito aqui permite que um disco padrão IDE possa ser interfaceado a uma conexão serial de qualquer processador Transputer, proporcionando transferências de dados de forma correta e eficaz (como demonstrado pelos testes descritos neste capítulo). Outra vantagem do subsistema implementado é que esse pode ser conectado a um Transputer em qualquer local dentro de uma rede de processadores, onde exista a necessidade de acesso rápido a disco, aumentando a flexibilidade do sistema global.

Quanto ao "software" para gerenciar o servidor de arquivos, descreve-se que foram apresentados neste capítulo algoritmos a partir dos quais se torna possível desenvolver um sistema programacional. Esse "software" deve possibilitar o gerenciamento de forma concorrente de todo o "hardware" do servidor de arquivos a ser construído, permitindo que os vários discos considerados sejam tratados de modo independente. Tal sistema programacional poderia ser implementado utilizando alguma das linguagens de programação descritas como exemplos na seção 6.2.1, pois aquelas possuem ferramentas apropriadas para a execução concorrente de processos.

CAPÍTULO SETE

CONSIDERAÇÕES FINAIS, CONCLUSÕES E PROPOSIÇÕES

7.1 Considerações Finais

Compartilhamento de recursos é uma das características fundamentais em muitos sistemas computacionais distribuídos baseados em redes locais. Dentre os vários recursos que podem estar disponíveis, o servidor de arquivos é um dos mais importantes, uma vez que esse recurso é intensamente utilizado pelos clientes. Um servidor de arquivos frequentemente se torna o recurso mais exigido, passando a ser o ponto de estrangulamento do sistema. Assim, desempenho é um tópico extremamente importante a ser considerado no projeto de servidores de arquivos.

A técnica de simulação foi usada neste trabalho para auxiliar na análise dos efeitos das aplicações de alguns métodos direcionados a obtenção de servidores de arquivos com alto desempenho. Especial ênfase é dada ao método apresentado nesta tese: utilização de múltiplos discos, onde os acessos são realizados apenas a áreas parciais dos mesmos. Simulação foi escolhida devido a sua alta flexibilidade e baixo custo, permitindo a avaliação de projetos alternativos para o sistema modelado em um modo relativamente fácil.

A seguir, são enfatizados os aspectos mais relevantes do trabalho apresentado nesta tese.

A revisão bibliográfica apresentada no capítulo dois baseia-se em uma ampla bibliografia e apresenta os conceitos importantes dentro do contexto de servidores de arquivos baseados em redes locais de computadores. Alguns pontos são amplamente discutidos e merecem destaque devido a sua influência na implementação de servidores de arquivos:

- . recursos rápidos: a utilização de sistemas de processamento e de armazenamento eficazes trazem grandes benefícios ao desempenho geral do servidor de arquivos; esse ponto merece atenção especial no projeto de servidores de arquivos;
- . utilização de "cache": através da utilização de "cache" muitos acessos a disco são evitados, podendo reduzir significativamente o tempo de resposta do servidor; os problemas relacionados a consistência das informações podem ser tratados utilizando várias técnicas;
- . múltiplos discos: a implementação de um servidor de arquivos com múltiplos discos possibilita aumento de capacidade de armazenamento, maior taxa de transferência e maior confiabilidade do sistema;
- . segurança: controlar acessos não autorizados aos dados é uma característica fundamental em servidores de arquivos; evitar a leitura de ou a escrita em um arquivo pode ser conseguido usando diferentes técnicas; por exemplo, lista de controle de acesso, capacidade e codificação;
- . confiabilidade: a adoção de disco de boa qualidade, a produção de cópias de dados e a utilização de transações atômicas são formas de obter um servidor de arquivos de alta confiabilidade.

Além desses conceitos, exemplos de servidores de arquivos e de sistemas de arquivamento implementados são apresentados na revisão; destaque especial é dado a projetos de servidores de arquivos.

Também no capítulo dois é feita uma revisão sobre modelamento para simulação de sistemas computacionais. São descritas algumas técnicas que podem ser utilizadas em análises de desempenho desses tipos de sistemas. Foi salientado que a técnica de simulação é bastante

adequada para análises de projetos alternativos, onde diferentes abordagens são consideradas (caso do trabalho desenvolvido), a um custo relativamente baixo. Os aspectos principais que devem ser considerados no modelamento para a simulação de um sistema computacional também foram ressaltados. Um ponto fundamental para se obter uma simulação confiável é a caracterização completa e fiel para o sistema a ser simulado.

Para a construção do modelo de simulação utilizado neste trabalho foram consideradas as características de um ambiente de referência existente, o sistema distribuído baseado em rede local TRICE. As características e os aspectos mais importantes (tipo, composição, organização, funcionamento geral, etc.), relevantes ao trabalho desenvolvido, foram apresentados no capítulo três. Especial ênfase é dada ao servidor de arquivos desse sistema.

O uso do ambiente do sistema TRICE se justifica pelo fato do seu servidor de arquivos apresentar características bastante adequadas para o desenvolvimento das pesquisas realizadas durante este trabalho, e também devido a disponibilidade de farto material [SA90b] relativo a aquele sistema. Esse material consiste de dados que caracterizam o servidor de arquivos desde a carga imposta por uma estação de trabalho até os tempos e operações envolvidos na execução das requisições recebidas pelo servidor.

O modelo e o programa de simulação, utilizados nesta tese para avaliação de desempenho de servidores de arquivos, foram apresentados no capítulo quatro. O modelo criado representa, de forma simplificada, as funções de interesse do servidor de arquivos de referência.

Mesmo sendo um modelo simplificado, foi possível a obtenção de resultados bastante satisfatórios (precisos e confiáveis) nas simulações realizadas, além da criação de um programa de simulação (implementado em GPSS/H) simples e flexível. Essas características possibilitam modificações rápidas na estrutura do programa de simulação, permitindo testes de várias alternativas para o projeto do servidor simulado. Além disso, cada simulação pode ser

executada em tempo bastante curto (aproximadamente um minuto em um computador 486 - 66Mhz), possibilitando a realização de um grande número de simulações.

O modelo de simulação foi validado utilizando diferentes técnicas. Inicialmente, foram executadas simulações controladas, a fim de se obter resultados previsíveis ou conhecidos. A etapa final de validação foi realizada através de comparações entre os resultados obtidos nas simulações e os dados coletados junto a um servidor de arquivos real, obtendo-se diferenças sempre menores que 5%.

O capítulo cinco descreve as análises de alguns procedimentos que podem ser aplicados na implementação de servidores de arquivos visando a obtenção de sistemas com alto desempenho. O modelo e o programa de simulação do servidor de arquivos do sistema TRICE, citados anteriormente, foram utilizados como ferramentas na realização dessas análises.

Através dos resultados de simulações, pôde-se observar que o sistema de disco era o fator limitante do desempenho do servidor de arquivos, mesmo quando discos bastante rápidos foram considerados. Assim, todos os esforços foram concentrados sobre o sistema de disco, a fim de diminuir os tempos relativos a esse. Duas formas básicas de diminuir o tempo de disco foram analisadas: a redução no número de acessos a disco e o aumento no número de discos.

Durante a discussão dessas análises foi apresentada uma forma alternativa de reduzir os tempos relativos ao sistema de armazenamento secundário de um servidor de arquivos: a utilização de um sistema composto por múltiplos discos, nos quais os acessos, durante operações normais de trabalho, são realizados apenas a áreas parciais de dados. Nesse esquema, os movimentos das cabeças de leitura/escrita dos discos são restritos a áreas parciais dos mesmos; sendo assim, o tempo médio total de acesso a dados é reduzido, diminuindo o tempo médio de resposta de um servidor de arquivos assim construído.

Baseado nos resultados das simulações e nas vantagens e desvantagens proporcionadas por cada uma das técnicas analisadas, concluiu-se que a técnica de múltiplos discos

com acessos parciais apresenta uma série de vantagens sobre as outras. Além de possibilitar a implementação de um servidor de arquivos com alto desempenho, a técnica de acessos parciais possibilita a obtenção de um sistema com grande confiabilidade, conseguida a partir da replicação de arquivos em vários discos. Ainda, o alto desempenho que pode ser obtido com o uso de tal técnica ocorre sem introduzir problemas de consistência de informações, como ocorre com as técnicas que utilizam "caches" de disco.

Para o total aproveitamento das vantagens oferecidas a partir da aplicação da técnica de múltiplos discos com acessos apenas a áreas parciais na construção de um servidor de arquivos, principalmente com relação ao desempenho desse servidor, os acessos aos discos devem ser realizados de forma concorrente. Esse gerenciamento concorrente significa que as transferências de informações entre o processador principal do servidor e cada disco devem ser realizadas de modo independente, sem interferir nas comunicações com outros discos. Assim, além de velocidade de processamento suficiente para não se tornar o fator limitante do desempenho do servidor de arquivos, o processador principal deve apresentar características que possibilitem o gerenciamento concorrente de tarefas de forma eficaz, providenciando transferências rápidas e seguras de informações com os discos.

Segundo as observações anteriores, a estrutura básica exigida para aquele servidor de arquivos deve ser composta por vários discos e um processador capaz de gerenciar esses dispositivos, e todas as tarefas a ele atribuídas, de forma concorrente. Entretanto, torna-se necessário verificar a viabilidade de implementação daquele servidor. A sugestão apresentada nesta tese, após a análise de alguns recursos disponíveis, é a utilização de um Transputer interfaceado diretamente aos discos através de suas conexões seriais (um dos fatores que pesaram sobre a escolha do Transputer foi a sua disponibilidade).

O interfaceamento de uma conexão serial do Transputer com um disco é realizado através de um subsistema especialmente criado para tal fim durante este trabalho. A avaliação desse subsistema mostra seu funcionamento correto e a sua eficiência (a interface implementada

possibilita a obtenção de taxas de transferência superiores a 90% das velocidades máximas permitidas pela ligação T800 - C012).

A nível de "software", foram desenvolvidos algoritmos que poderiam ser utilizados na implementação de um sistema programacional para o gerenciamento de forma concorrente de todo o servidor de arquivos, possibilitando comunicações independentes entre o processador e cada disco. Algumas das linguagens de programação que poderiam ser utilizadas são discutidas. Uma dessas linguagens é Occam, que apresenta uma série de ferramentas voltadas para a programação concorrente de processos. Occam é bastante adequada para ser utilizada em conjunto a um Transputer, uma vez que essa linguagem foi desenvolvida em paralelo com tais processadores.

7.2 Conclusões

As principais conclusões obtidas a partir do desenvolvimento deste trabalho são apresentadas neste item.

O Ponto de Estrangulamento de Servidores de Arquivos

A partir de resultados de simulações, modelando um servidor de arquivos real composto por um processador e um disco, observou-se que a taxa de utilização do disco é sempre maior que a do processador, mesmo em uma situação em que o disco mais rápido do mercado e um processador relativamente lento estariam sendo usados no servidor, caracterizando o disco como o gargalo do sistema.

Assim, se o disco é o fator limitante do desempenho do servidor de arquivos, esforços devem ser concentrados sobre esse recurso a fim de reduzir os tempos relativos ao sistema de armazenamento.

Utilização da Técnica de Múltiplos Discos com Acessos Parciais

Uma técnica alternativa para implementar servidores de arquivos baseados em rede local de computadores, visando obter sistemas com alto desempenho, foi proposta e analisada neste

trabalho. Essa técnica tem por objetivo reduzir os tempos relativos ao sistema de armazenamento secundário de um servidor de arquivos. Tal redução é obtida através da utilização de um sistema composto por múltiplos discos, nos quais os acessos, durante operações normais de trabalho, são realizados apenas a áreas parciais de dados. Isso faz com que o tempo médio de acesso a um determinado bloco de dados seja reduzido com relação ao tempo que seria gasto se os acessos ao disco fossem feitos de forma convencional (em que as cabeças de leitura/escrita podem fazer acessos a toda a área do disco em operações normais de trabalho).

A referida técnica pode ser utilizada em qualquer ambiente onde vários discos estão, ou podem estar, disponíveis, especialmente naqueles em que exista a necessidade de diminuição dos tempos de acessos a tal tipo de dispositivo, mesmo quando os mais rápidos discos já estiverem sendo utilizados.

Uma vez que vários discos estariam disponíveis a partir da aplicação da técnica de acessos parciais, arquivos poderiam ser replicados, sendo suas cópias distribuídas entre os dispositivos de armazenamento. Isso aumentaria sensivelmente a segurança dos dados armazenados nos discos com relação a falhas de dispositivos, proporcionando a construção de um servidor de arquivos com grande confiabilidade.

Salienta-se que as altas velocidades no armazenamento e na recuperação de informações, que podem ser obtidas a partir da utilização da técnica apresentada nesta tese, é o resultado da melhora de desempenho do disco de forma isolada. Algumas outras técnicas (como RAID) proporcionam altas velocidades através do aumento nas taxas de transferências do sistema de arquivamento como um todo. A justificativa para as utilizações dessas técnicas é que muitas aplicações atuais trabalham com arquivos grandes e que se esses arquivos forem divididos em blocos, sendo os blocos distribuídos entre vários discos, a taxa total de transferência do sistema de arquivos pode ser aumentada consideravelmente. Mesmo que esses tipos de técnicas sejam usados em sistemas de arquivos, a técnica de acessos parciais pode ser usada em conjunto, uma vez que arquivos grandes são divididos em blocos menores e assim o tempo total de transferência de cada

bloco pode continuar a sofrer forte influência dos tempos de posicionamento das cabeças de leitura/escrita dos discos.

A Interface Projetada

Durante a realização deste trabalho foi projetado e implementado um subsistema que permite o interfaceamento direto de um disco (padrão IDE) a uma conexão serial de um processador Transputer. A criação dessa interface fez parte do processo que tinha como objetivo demonstrar que a implementação de um servidor de arquivos, seguindo a técnica de múltiplos discos com acessos parciais as áreas de dados, era viável.

Embora nos testes da interface construída tenha sido utilizado o processador T800, observa-se que tal interface é bastante flexível, no sentido de que pode ser conectada a qualquer processador da família Transputer para este ter acesso direto a um sistema de disco. Para isso, basta uma simples ligação entre uma conexão serial do Transputer considerado e aquela interface por meio de um par de fios trançados.

Uma forma comum de utilização de computadores Transputer é associá-lo a outro computador, chamado hospedeiro (por exemplo, um IBM - PC). Através do hospedeiro, o computador Transputer tem acesso a periféricos (teclado, vídeo, discos, etc.). Um "software", que roda no hospedeiro, gerencia as transferências de dados entre o Transputer e os periféricos. No entanto, a interveniência do hospedeiro introduz atrasos nos processos de troca de informações entre o Transputer e, por exemplo, um disco, atrasos que muitas vezes podem ser tornar indesejáveis. O subsistema implementado neste trabalho é uma forma de evitar a passagem dos dados pelo hospedeiro, fazendo com que o computador Transputer tenha acesso direto a um disco.

Outro ponto a salientar é que a interface desenvolvida (associada a um disco) pode ser conectada a um processador em qualquer lugar dentro de uma rede de processadores Transputers, onde haja a necessidade de acesso rápido a disco.

Resumo das Contribuições Principais

Baseado no exposto neste capítulo, pode-se citar como contribuições principais deste trabalho:

- . **A Técnica de Múltiplos Discos com Acessos Parciais:** essa técnica é primariamente direcionada para aumentar o desempenho de um servidor de arquivos, a partir da redução do tempo de acesso a disco; além disso, permite que arquivos sejam replicados, aumentando consideravelmente a confiabilidade do servidor;
- . **Obtenção de um Sistema para Simulação de Servidores de Arquivos:** a implementação desse sistema de simulação permite a avaliação de desempenho de servidores de arquivos, existentes ou apenas projetados, e a verificação da influência dos seus recursos; possibilita também a avaliação de diferentes abordagens para a sua implementação, através de rápidas modificações na estrutura do modelo;
- . **A Interface Para Processadores Transputer:** a interface implementada permite a ligação direta de uma conexão serial de processadores Transputer a um disco IDE; essa é uma forma alternativa de acesso a disco por computadores Transputer, que em muitos casos faz acessos a discos de um hospedeiro.

7.3 Proposições

Algumas alternativas para a continuação do trabalho apresentado nesta tese são as seguintes:

- . **Implementação de um Protótipo para o Servidor de Arquivos Proposto:** a técnica de múltiplos discos com acessos parciais foi avaliada através de simulação, mostrando-se bastante eficaz. O próximo passo seria a implementação de um protótipo para um servidor de arquivos baseado nessa técnica. A nível de "hardware", tal protótipo poderia seguir as características básicas citadas, ou seja, a utilização de um módulo Transputer e de um subsistema de interfaceamento para disco, similar ao descrito. Quanto ao "software", uma linguagem bastante

adequada para os fins é Occam, que poderia ser usada na implementação do sistema programacional de controle do servidor seguindo os algoritmos básicos também apresentados nesta tese;

- . **Utilização da Técnica em Servidores de Arquivos Existentes:** servidores de arquivos compostos de vários discos poderiam ter seus "softwares" modificados para a incorporação das características da técnica de múltiplos discos com acessos parciais. O novo "software" poderia seguir os algoritmos apresentados;
- . **Avaliação de Projetos Alternativos:** o sistema de simulação desenvolvido é bastante flexível, permitindo rápidas alterações para a avaliação de diferentes estruturas para um servidor de arquivos. Assim, esse sistema de simulação pode ser utilizado para a verificação de abordagens diferentes das discutidas, como o uso de múltiplos conjuntos processador - discos e outras estratégias de "cache" de disco (por exemplo, "cache" distribuído).

Referências Bibliográficas e Bibliografia Geral

- [AC86] Acetta,M., Baron,R., Golub,D., Rashid,R., Tevanian,A., Young,M. - Math: A New Kernel Foundation for UNIX Development. Proceedings of the USENIX Conference, pp.93-112, 1986.
- [AL94] Almasi,G.S. - Highly Parallel Computing, 2a. edição. The Benjamin Cummings, 1994.
- [AL91] Alford,R.C. - The IDE Hard Disk Drive Interface. Byte, n.3, pp.317-324, 1991
- [AN83] Andrews,G.R., Schneider,F.B. - Concepts and Notations for Concurrent Programming. ACM Computing Survey, vol.15, n.1, pp.3-43, 1983.
- [BA89a]Banks,J., Carson,J.S., Sy,J.N. - Getting Started with GPSS/H. Wolverine Software Corporation, 1989.
- [BA89b]Barnes,J.G.P. - Programming in Ada, 3a. edição. Addison Wesley, 1989.
- [BA91] Baker,M.G., Hartman,J.H., Kupfer,M.D., Shirriff, K.W., Ousterhout,J.K. - Measurements of a Distributed File System. Proceedings of the 13th Symposium on Operating Systems Principles (SOSP), Asilomar, CA, october 1991.
- [BE84] Bernstein,P.A., Goodman,N. - An Algorithm for Concurrency Control and Recovery in Replicated Distributed Databases. ACM Transaction Database Systems, vol.9, n.4, pp.596-615, 1984.

- [BI80] Birrell, A.D., Needham,R.M. - A Universal File Server. IEEE Transaction on Software Engineering, vol.SE-6, n.5, pp.450-453, 1980.
- [BI85] Birrell,A.D. - Secure Communications Using Remote Procedure Calls. ACM Transactions on Computer Systems, vol.3, n.1, pp.1-14, 1985.
- [BR85] Brown, M.R. , Kolling,K.N., Taft,E.A. - The Alpine File System. ACM Transaction on Computer Systems, vol.3, n.4, pp.261-293, 1985.
- [BR90] Bryan,D.L., Mendal,G.O. - Exploring Ada, vol.1. Prentice-Hall, 1990.
- [BU88] Burns,A. - Programming in Occam2. Addison-Wesley Publ.Co, 1988.
- [CH91] Cheng,H.C., Sheu,J.P - Design and Implementation of a Distributed File System. Software - Practice and Experience, vol.21, n.7, pp.657-675, 1991.
- [CO94] Couloris,G. , Dollimore,J., Kindberg,T. - Distributed Systems: Concepts and Design, 2a. edição. Addison-Wesley,1994.
- [DA85] Davicev,D., Burkhard,W.A. - Consistency and Recovery Control for Replicated Files. Operating Systems Reviews, vol.19, n.5, pp.87-96, 1985.
- [FR85] Fridich,M., Older,W. - Helix: The Architecture of the XMS Distributed File System. IEEE Software, vol.12, n.5, pp.21-29, 1985.
- [GA94] Ganger,G.R., Worthington, B.L., Hou,R.Y., Patt,Y.N. - Disk Arrays: High-Performance, High- Reliability Storage Subsystems. IEEE Computer, vol.27, n.3, pp.30-37, 1994.

- [GR84] Graube,M., Mulder,M.C. - Local Area Networks. IEEE Computer, vol.17, n.10, pp.242-247, 1984.
- [HA75] Hansen,B. - The Programming Language Concurrent Pascal. IEEE Transaction Software and Engeneering, n.6, pp.199-206, 1975.
- [HA88a]Hac,A., Jin,X., Soo,J.H. - Algorithms for Replication in a Distributed System. IEEE Software, pp.206-214, 1988.
- [HA88b]Harker,S. - The Use of Prototyping and Simulation in the Development of Large-Scale Applications. The Computer Journal, vol.31, n.5, 1988.
- [HA92] Hartman,J.H., Ousterhout, J.K. - Zebra: A Striped Network File System. Proceedings of the USENIX Workshop on File Systems, 1992.
- [HA94] Hansen,B. - The Programming Language SuperPascal. Software - Practice and Experience, vol.24, n.5, pp.467-483, 1994.
- [HO78] Hoare,C.A.R. - Communicating Sequential Processes. Communication of ACM, vol.21, n.8, pp.666-677, 1978.
- [HO81] Holler, E. - Distributed Systems: Architecture and Implementation. Verlag, 1981.
- [HO88] Howard,J.H., Kazar,M.L., Menees,S.G., Nichols,D.A., Satyanarayanan,M., Sidebothan,R.N. - Scale and Performance in a Distributed File System. ACM Transaction on Computer Systems, vol.6, n.1, pp55-81, 1988.

- [HW85] Hwang,K., Briggs, F.A. - Computer Architeture and Parallel Processing. McGraw-Hill, 1985.
- [IN88] Inmos - Transputer Instruction Set: A Compiler Writer's Guide. Prentice-Hall, 1988.
- [IN89a] Inmos - The Transputer Applications Notebook: Systems and Performance. Inmos, 1989.
- [IN89b] Inmos - Transputer Databook. Inmos, 1989.
- [IN90] Inmos - Transputer Development System, 2a. edição. Prentice Hall, 1990.
- [KA94] Karedla,R. , Love,J.S., Wherry, B.G. - Caching Srategies to Improve Disk System Performance. IEEE Computer, vol.27, n.3, pp.38-46, 1994.
- [KI91] Kirner,C. - Arquiteturas de Sistemas Avançados de Computação. Anais da Jornada EPUSP/IEEE em Sistemas de Computação de Alto Desempenho, pp.307-353, 1991.
- [KO78] Kobayashi,H. - Modeling and Analysis - An Introduction to Systema Performance Evaluation Methodology. Addison Wesley, 1978.
- [JA89] Jalics,P.J., McIntyre,D.R. - Caching and Other Disk Access Avoidance Techniques on Personal Computer. Communications of the ACM, vol.32, n.2, pp.246-255, 1989.
- [JO88] Jones,G. and Goldsmith, M. - Programming in OCCAM2. Prentice-Hall, 1988.
- [LA86] Lazowska,E.D., Zahorjan,J.- File Access Performance of Diskless Workstations. ACM Transactions on Computer Systems, Vol.4, no.3, pp.238-268, 1986.

- [LA92] Lander,L.C., Mitra,S., Singhvi,N., Piatkovski,T.F. - The Elaboration Order Problem of Ada. Software - Practice and Experience, vol.22, n.5, pp.391-417, 1992.
- [LE90] Levy,E., Silberschatz,A. - Distributed File Systems: Concepts and Exemples. ACM Computing Surveys, vol.22, n.4, pp.321-374.
- [LI87] Lin,K., Frantz,G.A., Simar,R. - The TMS320 Family of Digital Signal Processors. Proceeding IEEE, pp.1143-1159, 1987.
- [LI93] Lilja,D.J. - Cache Coherence in Large-Scale Shared-Memory Multiprocessors: Issues and Comparisons. ACM Computing Surveys, vol.25, n.3, pp.304-338, 1993.
- [LU94] Luhn,R. - Big Hard Drives: 300MB and Counting. PC World, june, pp.203-209, 1994.
- [MA80] Maryansky,F.J. - Digital Computer Simulation. Haydem Book Company, 1980.
- [MA87] MacDougall,M.H. - Simulation Computer Systems - Techniques and Tools. The MIT Press, 1987.
- [MA93] Martin,J.L. - Travels with Gopher. IEEE Computer, vol.26, n.5, pp.84-87, 1993.
- [ME85] Meister,B., Svobodova,L. - File Transfer in Local-Area Networks: A Performance Study. Proceeding International Conference of Distributed Computing Systems, pp.338-349, 1985.
- [MI82] Mitchell,J.G. - File Servers for Local Area Networks. Local Area Networks - Lectures Notes, Canterbury, march/april, 1982.

- [MI90] Milway,M., Gorton,I., Fulcher,J., Zelinsky,A. - Interfacing Transputer Links to External Devices. Microprocessors and Microsystems, december, p.644-652, 1990.
- [MI92] Miller,B. - The Inside Story on Hard-Disk Standards. Radio-Electronics, june, p.63-66, 1992.
- [MU85] Mullender,S.J. - A Distributed File Service Based on Optimistic Concurrency Control. ACM Operating System Review, vol.19, n.5, pp.51-62, 1985.
- [MU91] Muller,K., Pasquale, J. - A High Performance Multi-Structured File System Design. ACM Operating Systems Reviews, vol.25, n.5, pp.56-67, 1991.
- [MU93] Mullender,S.J. - Distributed Systems, 2a. edição. ACM Press Frontier Series, Addison Wesley, 1993.
- [NE88] Nelson,M.N., Welch,B.B., Ousterhout,J.K. - Caching in the Sprite Network File System. ACM Transaction on Computer Systems, vol.6, n.1, pp.134-154, 1988.
- [OU85] Ousterhout,J.K., Da Costa,H., Harrison,D., Kunze,J.A., Kupfer,M., Thompson,J.G. - A Trace-Driven Analysis of the 4.2 BSD UNIX File System. Proceeding of the 10th Symposium on Operating Systems Principles, ACM, pp.15-24, 1985.
- [OU88] Ousterhout,J.K., Cherenon,A.R., Dougliis,F., Nelson,M.N., Welch,B.B. - The Sprite Network Operating System. IEEE Computer, vol.21, n.2, pp.23-36, 1988.
- [OU89] Ousterhout,J.K., Doudlis, F. - Beating the I/O Bottleneck: A Case for Log-Structured Fill Systems. Operating Systems Reviews, january, pp.11-28, 1989.

- [PA88a] Papamichalis,P., Simar Jr.,R - The TMS320C30 Floating-Point Digital Signal Processor. IEEE Micro, n.8, pp.13-29, 1988.
- [PA88b] Patterson,D.A., Gibson,G., Katz,R.H. - A case for Redundant Arrays of Inexpensive Disks (RAID). SIGMOD ACM, june, 1988.
- [PO85] Popek, G., Walker,B. - The LOCUS Distributed System Architecture. Cambrige MA: MIT Press, 1985.
- [PR90] Przybylki,S. - The Performance Impact of Block Sizes and Fetch Strategies. Proceedings of 14th Annual International Symposium on Computer Architecture, 1990.
- [RI80] Richards,M., Strevens,C.W. - BCPL: The Language and Its Compiler. Cambridge University Press, Cambridge, 1980.
- [RI95] Riofrio, M. - Cheaper by the Gigabyte. PC World, march, p.215-226, 1995.
- [RO88] Rozier,M., Abrossimov,V., Armand,F., Boule,I., Gien,M., Guillemont,M., Kaiser,C., Leonard,P., Neuhauser,W. - Chorus Distributed Operating Systems. Computing Systems Journal, vol.1, n.4, pp.305-370, 1988.
- [RO91] Rosenblum,M., Ousterhout,J.K. - The Design and Implementation of a Log-Structured File System. Proceeding of the 13th Symposium on Operating Systems Principles, Asilomar, CA, october, 1991.
- [SA84] Sargent, R.G. - A Tutorial on Verification and Validation of Simulation Models. Proceedings of the 1984 Winter Simulation Conference. Editado por S. Sheppard, U. Pooch, D. Pegden, pp.115-121.

- [SA85a] Satyanarayanan,M., Howard,J.H., Nichols,D.A., Sidebotham,Spector,A.Z., West,M.J. - The ITC Distributed File System: Principles and Design.Proceeding of the 10th Symposium on Operating Systems Principles, Orcas Island, WA, december, pp.35-50, 1985.

- [SA85b] Saydan,T. - Process Oriented Languages. Simuletter, vol.16, n.2, april, 1985.

- [SA90a] Santana, R.H.C. - Performance Avaluation of LAN-Based File Servers. Tese de PhD, Universidade de Southampton, Inglaterra, 1990.

- [SA90b] Santana, R.H.C. et al. - Trice - A Transparent Multi-LAN Distributed Computing System, XVI Conferência Latino-Americana de Informática, Assunção, Paraguay, 1990

- [SE94] Seagate - Product Overview. Seagate, march, 1994.

- [SH86] Shapiro,E. - Concurrent Prolog: A Progress Report. IEEE Computer, august, pp.44-58, 1986.

- [SH89] Shapiro,E. - The Family of Concurrent Logic Programming Language. Computing Surveys, vol.21, n.3, pp.412-510, 1989.

- [SH88] Sharma,R., Rose,L.I. - Modular Design for Simulation. Software - Practice and Experience, vol.18, n.10, 1988.

- [SH80] Shock,J.F., Hupp,J.A. - Measured Performance of an Ethernet Local Network. Communications ot the ACM, vol.23, n.12, pp.711-720, 1980.

- [SM82] Smith,A.J. - Caches Memories. ACM Computing Surveys, vol.14, n.3, pp.473-530, 1982.
- [SN92] Snow,C.R. - Concurrent Programming. Cambridge University Press, 1992.
- [SO92] Soares,L.F.G. - Modelagem e Simulação Discreta de Sistemas. Editora Campus, 1992.
- [ST87] Stus,J.M. - Considerações Sobre o Desenvolvimento por Protótipos. XX Congresso Nacional de Informática, São Paulo, pp.1044-1047, 1987.
- [ST92] Stevenson,D.E., Ammons,L.K., Crosmum,W.G., Jackson,A., Raj,G.L. - A Vector C and Fortran Compiler for the FPS T-series: Experiences with Compiling To Occaml. Software - Practice and Experience, vol.22, n.5, pp.371-390, 1992.
- [SV84] Svobodova,L. - File Servers for Network-Based Distributed Systems. ACM Computing Surveys, vol.16, n.4, pp.353-398, 1984.
- [TA85] Tanenbaum,A.S., Van Renesse, R. - Distributed Operating Systems. ACM Computing Surveys, vol.17, n.4, pp.419-470, 1985.
- [TA92] Tanenbaum,A.S. - Modern Operanting Systems. Prentice-Hall, 1992.
- [TA95] Tanenbaum,A.S. - Distributed Operanting Systems. Prentice-Hall, 1995.
- [TE87a] Terry,D.B. - Caching Hints in Distributed Systems. IEEE Transactions on Software Engineering, vol. SE13, n.1, pp.48-54, 1987.
- [TE87b] Texas Instruments - TTL Logic DataBook. Texas Instruments Incorporated, 1987.

[TE91] Texas Instruments - TMS320C4x - User's Guide. Texas Instruments Incorporated, 1991.

[WD90] Western Digital - 4.0 Technical Reference Manual. Western Digital Corporation, 1990.

[WE83] Wegner,J., Smolka,S.A. - Process, Tasks and Monitors: A Comparative Study of Concurrent Programming Primitives. IEEE Transaction on Software Engineering, vol.SE8, n.3,pp.73-84, 1983.

[WE86] Welch,B, Ousterhout,J. - Prefix Tables: A Simple Mechanism for Locating Files in a Distributed System. Proceedings of the 6th Conference on Distributed Computer Systems, Cambridge, pp.184-189, 1986.

[WE90] Weicker,R.P. - An Overview of Common Benchmarks. IEEE Computer, december, pp.65-75, 1990.

[WI85] Wirth,N. - Programming in Modula-2, 3a. edição. Springer Verlag, 1985.

[WO86] Woodside,C.M., Tripathi,S.K. - Optimal Allocation of File Servers in Local Network Environment. IEEE Transaction on Software Engineering, vol.SE-12, n.8, pp.844-848, 1986.

Anexo A

Distribuições Utilizadas no Sistema de Simulação

Figura A.1 - Distribuição dos Tempos Entre Solicitações - Intervalos de 100 ms.

Figura A.2 - Distribuição dos Tempos Entre Solicitações - Intervalos de 10 ms.

Figura A.3 - Distribuição dos Tipos de Solicitações.

Tabela A.1 - Probabilidade Para Cada Solicitação de Controle.

Tipo de Solicitação	Probabilidade (%)
fim de leitura	28,47
criar arquivo	2,74
eliminar arquivo	2,17
abrir arquivo	31,63
fechar arquivo	30,47
previlégio	0,21
renomear	0,14
criar diretório	0,02
eliminar diretório	0,01
mudar diretório	1,02
informações	0,09
eliminar entrada	0,01
eliminar saída	0,01
retroceder	0,35
cota	0,02
finalizar entrada	2,63

Figura A.4 - Distribuição do Número de Blocos Para os Diretórios.

Figura A.5 - Distribuição do Número de Blocos Para os Arquivos Ligados.

Figura A.6 - Distribuição do Número de Blocos Para os Arquivos Contíguos.

Anexo B

Diagramas da Interface

T800 - IDE

Figura B.1 - Diagrama de Blocos da Interface T800 - IDE.

Figura B.2 - Diagrama Esquemático da Interface T800 - IDE.

Figura B.2 - Diagrama Esquemático da Interface T800 - IDE (continuação).

Apêndice C

Listagens dos Programas

Figura C.1 - Programa Para Testes da Interface T800-IDE: Escrita em Disco.

```

PROC transferencia(CHAN OF ANY vai, volta, entra, sai)
  -- Para escrever setor(es) da IDE - 28.04.94

  VAL num.reg IS 512:
  [num.reg]BYTE dados:
  BYTE aux, aux1, aux2, r7, w0,w1,w2,w3,w4,w5,w6,w7:
  INT i:

  SEQ
    volta ? aux      -- inicializa processo

    r7 := BYTE(215)  -- para ler o reg. 1F7 da IDE

    w0 := BYTE(48)   -- para escrever no reg. 1F0 da IDE
    w1 := BYTE(177)
    w2 := BYTE(178)
    w3 := BYTE(179)
    w4 := BYTE(180)
    w5 := BYTE(181)
    w6 := BYTE(182)
    w7 := BYTE(183)

    sai ! w1          -- pre-compensacao
    aux := BYTE(00)
    sai ! aux
    sai ! aux

    sai ! w2          -- n. de setores a transferir
    i := 1
    aux1 := BYTE(01)
    sai ! aux1
    sai ! aux1

    sai ! w3          -- a partir do setor xx
    aux1 := BYTE(10)
    sai ! aux1
    sai ! aux1

    sai ! w4          -- cilindro low
    aux := BYTE(166)
    sai ! aux
    sai ! aux

    sai ! w5          -- cilindro high
    aux := BYTE(03)
    sai ! aux
    sai ! aux

    sai ! w6          -- drive/cabeca
    aux1 := BYTE(160) -- = A0h --> 1(mod0 ECC) 01(512B/setor)
                    -- 0(drive master) 0000(cabeca logica)
    sai ! aux1
    sai ! aux1

    sai ! w7          -- comando
    aux1 := BYTE(48) -- escrever
    sai ! aux1
    sai ! aux1

  SEQ j = 0 FOR 512
    dados[j] := 'A'

```

```

WHILE i <> 0
  SEQ
  aux2 := BYTE(88)
  WHILE aux1 <> aux2 -- espera IDE ficar pronta
    SEQ
    sai ! r7
    entra ? aux1
    entra ? aux

    sai ! w0
    sai ! dados -- envia dados para a IDE
    i := i - 1

  aux := BYTE(51) -- char 3
  vai ! aux -- termina processo com 1 byte
:
$USE linkaddr
CHAN OF ANY vai, volta, entra, sai:

PROCESSOR 0 T800
  PLACE volta AT link0.in :
  PLACE vai AT link0.out:
  PLACE entra AT link2.in :
  PLACE sai AT link2.out:
  transferencia(vai, volta, entra, sai)

;em 28/04/94.
;Para escrita na IDE.
;
DOSSEG
.MODEL SMALL
.STACK 100h
.CODE
;
  mov ax, 02 ;para OCCAM iniciar processo
  mov dx, 0151h
  out dx, al
;
;
  mov bx, 01
retl:
  mov dx, 0152h
nao_pronto: ;espera T800 ficar pronto
  in al, dx
  test al, 01
  jz nao_pronto
;
  mov dx, 0150h
  in al, dx ;recebe de OCCAM.
  mov dl, al
  mov ah, 2 ;escreve na tela.
  int 21h
;
  dec bx
  jnz retl
;
;
  mov ah, 4ch ;termina programa.
  int 21h
END

```


Figura C.2 - Programa Para Testes da Interface T800-IDE: Leitura em Disco.

```

PROC transferencia(CHAN OF ANY vai, volta, entra, sai)
  -- Para ler setor(es) da IDE

  VAL num.reg IS 512:
  [num.reg]BYTE dados:
  BYTE aux, aux1, aux2, r0, r7, w1,w2,w3,w4,w5,w6,w7:
  INT i:

  SEQ
    volta ? aux      -- inicializa processo

    r0 := BYTE(80)   -- para ler do reg. 1F0 da IDE
    r7 := BYTE(215)

    w1 := BYTE(177)  -- para escrever no reg. 1F1 da IDE
    w2 := BYTE(178)
    w3 := BYTE(179)
    w4 := BYTE(180)
    w5 := BYTE(181)
    w6 := BYTE(182)
    w7 := BYTE(183)

    sai ! w1         -- pre-compensacao
    aux := BYTE(00)
    sai ! aux
    sai ! aux

    sai ! w2         -- n. de setores a transferir
    i := 2
    aux1 := BYTE(02)
    sai ! aux1
    sai ! aux1

    sai ! w3         -- a partir do setor xx
    aux1 := BYTE(01)
    sai ! aux1
    sai ! aux1

    sai ! w4         -- cilindro low
    aux := BYTE(00)
    sai ! aux
    sai ! aux

    sai ! w5         -- cilindro high
    aux := BYTE(00)
    sai ! aux
    sai ! aux

    sai ! w6         -- drive/cabeca
    aux1 := BYTE(160)
    sai ! aux1
    sai ! aux1

    sai ! w7         -- comando
    aux1 := BYTE(32)
    sai ! aux1
    sai ! aux1

    WHILE i <> 0
      SEQ
        aux1 := BYTE(00)
        aux2 := BYTE(88)
        WHILE aux1 <> aux2
          SEQ

```

```

    JNE .11
    entra ? aux1
    entra ? aux

    sai ! r0
    entra ? dados    -- recebe dados da IDE

    vai ! dados      -- envia dados para tela
    i := i - 1
:
#USE linkaddr
CHAN OF ANY vai, volta, entra, sai:

PROCESSOR 0 T800
PLACE volta AT link0.in :
PLACE vai   AT link0.out:
PLACE entra AT link2.in :
PLACE sai   AT link2.out:
transferencia(vai, volta, entra, sai)

;em 28/04/94.
;Para fazer a transferencia da IDE para a tela.
;
DOSSEG
.MODEL SMALL
.STACK 100h
.CODE
;
mov ax, 02    ;para OCCAM iniciar processo
mov dx, 0151h
out dx, al
;
;
mov bx, 1024
retl:
mov dx, 0152h
nao_pronto:    ;espera T800 ficar pronto
in al, dx
test al, 01
jz nao_pronto
;
mov dx, 0150h
in al, dx      ;recebe de OCCAM.
mov dl, al
mov ah, 2      ;escrever na tela.
int 21h
;
dec bx
jnz retl
;
;
mov ah, 4ch    ;terminar programa.
int 21h
END

```

Figura C.3 - Programa de Simulação Original.

```

SIMULATE

*      Em 26.06.95                                D1CAC.GPS
*      Simulacao de UM processador, UM disco e CACHE - media de 10 simulacoes
*      Utilizando tempo medio de SEEK
*      Considera 83% dos acessos a disco p/ leitura e 17% p/ escrita

*      Definicao de Variaveis
REAL      &DELAY, &PRSP, &SEEK, &TDISC, &TPROC, &T512, &DIV, &WSSP
INTEGER    &I, &J, &WWS

LET        &WSSP=1.0                                Velocidade da WS
LET        &PRSP=1.0                                Velocidade do processador
LET        &SEEK=29.400                             Tempo medio de SEEK
LET        &T512=0.341                             Tempo p/ transferir 512 bytes
LET        &TDISC=&SEEK+&T512                       Tempo total de disco
LET        &I=0                                     No. de WS ocupadas
LET        &WWS=1                                   Proxima WS solicitante

*      Definicao de Funcoes

*      Determina numero de acessos a disco
NACCES FUNCTION RN(2),D20
0.189,0/0.853,1/0.871,2/0.932,3/0.939,4/0.946,5/0.949,6/0.951,7/
0.976,8/0.987,9/0.991,10/0.994,11/0.995,12/0.9977,14/0.9984,18/
0.9990,23/0.9995,32/0.9997,41/0.9999,68/1.00,105

*      Calcula o tempo de execucao no processador
TEMPO FUNCTION RN(3),D12
0.304,1.4/0.36,1.9/0.367,3.1/0.733,3.8/0.735,4.5/0.907,6.1/0.95,7.6/
0.954,8/0.956,10.9/0.957,15.6/0.9966,17.4/1.00,23

*      Tempo entre solicitacoes
TIR FUNCTION RN(4),D53
0.023,7/0.158,15/0.181,25/0.22,35/0.261,45/0.291,55/0.316,65/0.346,75/
0.376,85/0.406,95/0.431,105/0.456,115/0.492,125/0.533,135/0.574,145/
0.608,155/0.638,165/0.666,175/0.702,185/0.73,195/0.76,205/0.785,215/
0.805,225/0.818,235/0.827,245/0.86,275/0.8673,302/0.8743,309/
0.8793,319/0.8833,329/0.8867,339/0.8917,354/0.9005,384/0.9105,429/
0.9245,479/0.929,529/0.9321,579/0.9351,654/0.9421,854/0.9509,1104/
0.9604,1454/0.9727,2404/0.9774,3604/0.98,4504/0.9804,4949/0.9829,5494/
0.9847,6494/0.9888,8494/0.9898,10494/0.9948,15994/0.9962,25994/
0.997,36327/1.00,80000

GENERATE    0,,,,,2PB                                Geracao de solicitacoes
GATE LR     GERACAO                                Impede nova geracao se GERACAO=SET
ASSIGN      WS, &WWS, PB                             Determina WS solicitante
BLET        &WWS=&WWS+1                             Proxima WS solicitante
BLET        &I=&I+1                                   Incrementa o no. de WS ocupadas
TEST E      &I,1,CONT                               Testa ocupacao do sistema
LOGIC S     GERACAO                                Sinaliza q/ todas WS estao ocupadas
CONT        BLET &DELAY=FN(TIR)
TEST L      &DELAY,300,BUR1
BLET        &DELAY=&DELAY/&WSSP
TRANSFER    ,BURST
BUR1        TEST L &DELAY,1100,BURST                Solicitacao dentro de burst novo
BLET        &DIV=&WSSP-(((&DELAY-275)/829)*(&WSSP-1))

```

BURST	BLET TRANSFER ADVANCE	&DELAY=&DELAY/&DIV ,BURST &DELAY	Atrasa entrada da solic. no sistema
	ASSIGN	ACES, FN(NACCES), PB	Calculo do n. de acessos a disco
REPR	QUEUE QUEUE SEIZE DEPART	SYSQ PROCQ PROCES PROCQ	Entrada das solicitacoes no sistema Inicio de espera na fila-processador Captura do processador Fim de espera na fila do processador
SEGUE	BLET ADVANCE RELEASE	&TPROC=FN(TEMPO) (&TPROC/1.5)/&PRSP PROCES	Calculo do tempo de processamento Processamento da solicitacao Liberacao do processador
	TEST NE TRANSFER	PB(ACES), 0, FIM .17, , ESCR	Fazer acesso a disco? Se nao --> FIM 17% p/ escrita
* Para leitura:	TRANSFER	.845, LEIT, CACH	Dados em cache? Se sim --> CACH
* Para escrita:	TRANSFER	.000, , CACH	Dados em cache? Se sim --> CACH
ESCR	TRANSFER	.000, , CACH	Nao. Espera na fila do disco
LEIT	QUEUE SEIZE DEPART ADVANCE RELEASE TRANSFER	DISCQ DISCO DISCQ &TDISC DISCO , ATUA	Captura do disco Fim de espera na fila do disco Processamento em disco Liberacao do disco Transfere p/ atualizacao
CACH	SEIZE RELEASE ASSIGN SEIZE TRANSFER	CACHE CACHE ACES-, 1, PB PROCES , SEGUE	Acesso ao cache Libera o cache Atualiza ACES Recaptura PROCES Continua processamento
ATUA	ASSIGN TRANSFER	ACES-, 1, PB , REPR	Atualiza ACES Retorna p/ reproprocessamento
FIM	DEPART BLET BLET LOGIC R TERMINATE	SYSQ &WWS=PB(WS) &I=&I-1 GERACAO 1	Saida das solicit. do sistema Libera p/ novas geracoes Atualiza n. solicit. terminadas
*	GPSS/H Control Statements		
	DO	&J=1, 10	
	CLEAR		
	LET	&I=0	
	LET	&WWS=1	
	START	50000	Termina apos S solic. atendidas
	ENDDO		
	END		

Figura C.4 - Programa de Simulação Usando Dois Discos: Considerando Acessos a Disco de Forma Convencional.

```

SIMULATE

*      Em 26.06.95                                D2CAI.GPS
*      Simulacao de UM processador, DOIS discos e CACHE.
*      Considerando tempo medio de SEEK.
*      Considera 83% dos acessos a disco p/ leitura e 17% p/ escrita
*      Considera que os discos possuem informacoes iguais

*      Definicao de Variaveis
REAL      &DELAY,&PRSP,&SEEK,&TDISC,&TPROC,&T512,&DIV,&WSSP
INTEGER    &I,&WWS

      LET      &WSSP=40.0                      Velocidade da WS
      LET      &PRSP=5.0                      Velocidade do processador
      LET      &SEEK=12.000                  Tempo medio de SEEK
      LET      &T512=0.050                  Tempo p/ transferir 512 bytes
      LET      &TDISC=&SEEK+&T512            Tempo total de disco1
      LET      &I=0                          No. de WS ocupadas
      LET      &WWS=1                        Proxima WS solicitante

*      Declaracao de STORAGE
STORAGE    S(DISCO),2

*      Definicao de Funcoes

*      Determina numero de acessos a disco
NACCES FUNCTION RN(2),D20
0.189,0/0.853,1/0.871,2/0.932,3/0.939,4/0.946,5/0.949,6/0.951,7/
0.976,8/0.987,9/0.991,10/0.994,11/0.995,12/0.9977,14/0.9984,18/
0.9990,23/0.9995,32/0.9997,41/0.9999,68/1.00,105

*      Calcula o tempo de execucao no processador
TEMPO FUNCTION RN(3),D12
0.304,1.4/0.36,1.9/0.367,3.1/0.733,3.8/0.735,4.5/0.907,6.1/0.95,7.6/
0.954,8/0.956,10.9/0.957,15.6/0.9966,17.4/1.00,23

*      Tempo entre solicitacoes
TIR FUNCTION RN(4),D53
0.023,7/0.158,15/0.181,25/0.22,35/0.261,45/0.291,55/0.316,65/0.346,75/
0.376,85/0.406,95/0.431,105/0.456,115/0.492,125/0.533,135/0.574,145/
0.608,155/0.638,165/0.666,175/0.702,185/0.73,195/0.76,205/0.785,215/
0.805,225/0.818,235/0.827,245/0.86,275/0.8673,302/0.8743,309/
0.8793,319/0.8833,329/0.8867,339/0.8917,354/0.9005,384/0.9105,429/
0.9245,479/0.929,529/0.9321,579/0.9351,654/0.9421,854/0.9509,1104/
0.9604,1454/0.9727,2404/0.9774,3604/0.98,4504/0.9804,4949/0.9829,5494/
0.9847,6494/0.9888,8494/0.9898,10494/0.9948,15994/0.9962,25994/
0.997,36327/1.00,80000

      GENERATE 0,,,,,2PB                      Geracao de solicitacoes
      GATE LR   GERACAO                      Impede nova geracao se GERACAO=SET
      ASSIGN WS,&WWS,PB                     Determina WS solicitante
      BLET      &WWS=&WWS+1                  Proxima WS solicitante
      BLET      &I=&I+1                      Incrementa o no. de WS ocupadas
      TEST E    &I,70,CONT                  Testa ocupacao do sistema
      LOGIC S   GERACAO                      Sinaliza q/ todas WS estao ocupadas
CONT        BLET      &DELAY=FN(TIR)

```

	TEST L	&DELAY,300,BUR1	Solicitacao dentro do burst antigo?
	BLET	&DELAY=&DELAY/&WSSP	Sim
	TRANSFER	,BURST	
BUR1	TEST L	&DELAY,1100,BURST	Solicitacao dentro de burst novo
	BLET	&DIV=&WSSP-(((&DELAY-275)/829)*(&WSSP-1))	
	BLET	&DELAY=&DELAY/&DIV	
BURST	ADVANCE	&DELAY	Atrasa entrada da solic. no sistema
	ASSIGN	ACES, FN(NACCES), PB	Calculo do n. de acessos a disco
	QUEUE	SYSQ	Entrada das solicitacoes no sistema
REPR	QUEUE	PROCQ	Inicio de espera na fila do processad
	SEIZE	PROCES	Captura do processador
	DEPART	PROCQ	Fim de espera na fila-processador
SEGUE	BLET	&TPROC=FN(TEMPO)	Calculo do tempo de processamento
	ADVANCE	(&TPROC/1.5)/&PRSP	Processamento da solicitacao
	RELEASE	PROCES	Liberacao do processador
	TEST NE	PB(ACES),0,FIM	Fazer acesso a disco? Se nao --> FIM
	TRANSFER	.17,,ESCR	17% p/ escrita
* Para leitura:	TRANSFER	.845,DSC,CACH	Dados em cache? Se sim --> CACH
* Para escrita:	TRANSFER	.000,,CACH	Dados p/ o cache? Se sim --> CACH
ESCR	TRANSFER	,CACH	Escreve 100% no cache? Sim --> CACH
*ESCR	TRANSFER		
DSC	QUEUE	DISCQ	Espera na fila dos discos
	ENTER	DISCO	Captura 1 de 2 discos
	DEPART	DISCQ	Deixa fila dos discos
	ADVANCE	&TDISC	Processamento no disco
	LEAVE	DISCO	Liberacao de 1 disco
	TRANSFER	,ATUA	Transfere p/ atualizacao
CACH	SEIZE	CACHE	Acesso ao cache
	RELEASE	CACHE	Libera o cache
	ASSIGN	ACES-,1,PB	Atualiza ACES
	SEIZE	PROCES	Recaptura PROCES
	TRANSFER	,SEGUE	Continua processamento
ATUA	ASSIGN	ACES-,1,PB	Atualiza ACES
	TRANSFER	,REPR	Retorna p/ reprocessamento
FIM	DEPART	SYSQ	Saida das solicit. do sistema
	BLET	&WWS=PB(WS)	
	BLET	&I=&I-1	
	LOGIC R	GERACAO	Libera p/ novas geracoes
	TERMINATE	1	Atualiza n. solicit. terminadas
*	GPSS/H Control Statements		
	START	50000	Termina apos S solic. atendidas
	END		

Figura C.5 - Programa de Simulação Usando Dois Discos: Considerando Acessos a Disco de Forma Parcial.

```

SIMULATE

*      Em 26.06.95                                D2CAD.GPS
*      Simulacao de UM processador, DOIS discos e CACHE.
*      Considerando tempo medio de SEEK.
*      Considera 83% dos acessos a disco p/ leitura e 17% p/ escrita
*      Considera que os discos possuem informacoes diferentes

*      Definicao de Variaveis
REAL      &DIV,&WSSP
REAL      &DELAY,&PRSP,&SEEK,&TDISC1,&TDISC2,&TPROC,&T512
INTEGER    &I,&WWS

      LET      &WSSP=40.0                      Velocidade da WS
      LET      &PRSP=5.0                      Velocidade do processador
      LET      &SEEK=8.000                    Tempo medio de SEEK
      LET      &T512=0.050                    Tempo p/ transferir 512 bytes
      LET      &TDISC1=&SEEK+&T512            Tempo total de disco1
      LET      &TDISC2=&TDISC1                Tempo total de disco2
      LET      &I=0                          No. de WS ocupadas
      LET      &WWS=1                        Proxima WS solicitante

*      Definicao de Funcoes

*      Determina numero de acessos a disco
NACCES FUNCTION RN(2),D20
0.189,0/0.853,1/0.871,2/0.932,3/0.939,4/0.946,5/0.949,6/0.951,7/
0.976,8/0.987,9/0.991,10/0.994,11/0.995,12/0.9977,14/0.9984,18/
0.9990,23/0.9995,32/0.9997,41/0.9999,68/1.00,105

*      Calcula o tempo de execucao no processador
TEMPO FUNCTION RN(3),D12
0.304,1.4/0.36,1.9/0.367,3.1/0.733,3.8/0.735,4.5/0.907,6.1/0.95,7.6/
0.954,8/0.956,10.9/0.957,15.6/0.9966,17.4/1.00,23

*      Tempo entre solicitacoes
TIR FUNCTION RN(4),D53
0.023,7/0.158,15/0.181,25/0.22,35/0.261,45/0.291,55/0.316,65/0.346,75/
0.376,85/0.406,95/0.431,105/0.456,115/0.492,125/0.533,135/0.574,145/
0.608,155/0.638,165/0.666,175/0.702,185/0.73,195/0.76,205/0.785,215/
0.805,225/0.818,235/0.827,245/0.86,275/0.8673,302/0.8743,309/
0.8793,319/0.8833,329/0.8867,339/0.8917,354/0.9005,384/0.9105,429/
0.9245,479/0.929,529/0.9321,579/0.9351,654/0.9421,854/0.9509,1104/
0.9604,1454/0.9727,2404/0.9774,3604/0.98,4504/0.9804,4949/0.9829,5494/
0.9847,6494/0.9888,8494/0.9898,10494/0.9948,15994/0.9962,25994/
0.997,36327/1.00,80000

      GENERATE 0,,,,,2PB                      Geracao de solicitacoes
      GATE LR GERACAO                        Impede nova geracao se GERACAO=SET
      ASSIGN WS,&WWS,PB                      Determina WS solicitante
      BLET &WWS=&WWS+1                      Proxima WS solicitante
      BLET &I=&I+1                          Incrementa o no. de WS ocupadas
      TEST E &I,70,CONT                     Testa ocupacao do sistema
      LOGIC S GERACAO                      Sinaliza q/ todas WS estao ocupadas
CONT      BLET &DELAY=FN(TIR)
      TEST L &DELAY,300,BUR1
      BLET &DELAY=&DELAY/&WSSP
      TRANSFER ,BURST

```

BUR1	TEST L	&DELAY,1100,BURST	Solicitacao dentro de burst novo
	BLET	&DIV=&WSSP-(((&DELAY-275)/829)*(&WSSP-1))	
BURST	BLET	&DELAY=&DELAY/&DIV	
	ADVANCE	&DELAY	Atrasa entrada da solic. no sistema
	ASSIGN	ACES, FN(NACCES), PB	Calculo do n. de acessos a disco
REPR	QUEUE	SYSQ	Entrada das solicitacoes no sistema
	QUEUE	PROCO	Inicio de espera na fila-processador
	SEIZE	PROCES	Captura do processador
	DEPART	PROCO	Fim de espera na fila do processador
SEGUE	BLET	&TPROC=FN(TEMPO)	Calculo do tempo de processamento
	ADVANCE	(&TPROC/1.5)/&PRSP	Processamento da solicitacao
	RELEASE	PROCES	Liberacao do processador
	TEST NE	PB(ACES), 0, FIM	Fazer acesso a disco? Se nao --> FIM
	TRANSFER	.17,,ESCR	17% p/ escrita
*	Para leitura:		
	TRANSFER	.845,,CACH	Dados em cache? Se sim --> CACH
	TRANSFER	.50,DSC1,DSC2	50% p/ DISCO1 e 50% p/ DISCO2
*	Para escrita:		
ESCR	TRANSFER	.000,,CACH	Dados p/ o cache? Se sim --> CACH
*ESCR	TRANSFER	,CACH	Escreve 100% no cache? Sim --> CACH
	TEST E	Q(DISCQ1),Q(DISCQ2),DIFE	Filas iguais? Se nao --> DIFE
	TRANSFER	.50,DSC1,DSC2	Sim; escreve em qq disco
DIFE	TEST L	Q(DISCQ1),Q(DISCQ2),DSC2	Se fila do disco1 eh menor, usa disco
DSC1	QUEUE	DISCQ1	Inicio de espera na fila do disco1
	SEIZE	DISCO1	Captura do disco1
	DEPART	DISCQ1	Fim de espera na fila do disco1
	ADVANCE	&TDISC1	Processamento no disco1
	RELEASE	DISCO1	Liberacao do disco1
	TRANSFER	,ATUA	Transfere p/ atualizacao
DSC2	QUEUE	DISCQ2	Inicio de espera na fila do disco2
	SEIZE	DISCO2	Captura do disco2
	DEPART	DISCQ2	Fim de espera na fila do disco2
	ADVANCE	&TDISC2	Processamento no disco2
	RELEASE	DISCO2	Liberacao do disco2
	TRANSFER	,ATUA	Transfere p/ atualizacao
CACH	SEIZE	CACHE	Acesso ao cache
	RELEASE	CACHE	Libera o cache
	ASSIGN	ACES-, 1, PB	Atualiza ACES
	SEIZE	PROCES	Recaptura PROCES
	TRANSFER	,SEGUE	Continua processamento
ATUA	ASSIGN	ACES-, 1, PB	Atualiza ACES
	TRANSFER	,REPR	Retorna p/ reproprocessamento
FIM	DEPART	SYSQ	Saida das solicit. do sistema
	BLET	&WWS=PB(WSS)	
	BLET	&I=&I-1	
	LOGIC R	GERACAO	Libera p/ novas geracoes
	TERMINATE	1	Atualiza n. solicit. terminadas
*	GPSS/H Control Statements		
	START	50000	Termina apos S solic. atendidas
	END		