Instituto de Ciências Matemáticas de São Carlos

ISSN - 0103-2569

DESENVOLVIMENTO E AVALIAÇÃO DE SERVIDORES DE VÍDEO

RUDINEI GOULARTE
ANTONIO MARCOS M. HACHISUCA
EDSON DOS SANTOS MOREIRA
RONALDO C.M. CORREIA

Nº 66

RELATÓRIOS TÉCNICOS DO ICMSC

São Carlos
Fev./1998

DESENVOLVIMENTO E AVALIAÇÃO DE SERVIDORES DE VÍDEO

Rudinei Goularte, Antonio Marcos M. Hachisuca, Edson dos Santos Moreira, Ronaldo C. M. Correia
Departamento de Ciências de Computação e Estatística - ICMSC - USP - São Carlos
ICMSC-USP, Seção de Pós-Graduação, Av. Dr. Carlos Botelho, 1465. Cx. Postal 668 São Carlos - SP
E-mail:{rgoulart, shiro, edson, ronaldo}@icmsc.sc.usp.br

RESUMO

Este relatório discute alguns dos problemas de implementação de um sistema de distribuição de vídeo utilizando tecnologia Internet. A atenção especial é dada ao servidor de vídeo. Além do armazenamento e da distribuição, outro ponto importante é a forma de organizar o acesso aos dados no servidor de modo a obter eficiência e flexibilidade na manipulação dos mesmos. Uma base de metadados é implementada para obter a organização desejada. O relatório apresenta, como ponto principal, testes realizados para a avaliação comparativa de várias alternativas tecnológicas para os vários componentes do sistema. As avaliações incluem a interface de aplicação com a rede e com as bases de dados (verificando o impacto do uso de CORBA e de JDBC), o sistema operacional (NT e Linux), o sistema de armazenamento (SCSI e RAID) e a rede (*ethernet* e uma *switch ethernet* com uma porta FDDI para o servidor). O relatório descreve a metodologia utilizada e os resultados encontrados. Como exemplo de aplicação, o sistema é utilizado em um ambiente educacional onde a editoração e a apresentação são facilitadas.

ABSTRACT

This report discusses some of the problems on the implementation of an Internet-based video delivery system. Special attention is given to the video server. Besides storage and delivery, another important point is how to handle the server's data in a way to provide efficiency and flexibility on the multimedia data access. A metadata-based database is used for that purpose. The report also describes tests carried out to comparatively evaluate several hardware and software tecnological alternatives for the different components of the system. The evaluations include the application interface with the network and the database (verifying the impact of CORBA and JDBC), operating systems (NT an Linux), storage system (SCSI alone and RAID) and network (single-bus ethernet and switched ethernet with a FDDI port for the server). The report describes the metodology used and the results obtained. As an application example,, the system is used in an educational environment where editing and presentation are facilitated.

1. INTRODUÇÃO

A entrega de vídeo através de redes de computadores envolve fatores como armazenamento, entrega, organização dos dados e apresentação. A questão do armazenamento e a questão da entrega de dados de vídeo (e de dados multimídia em geral) em ambientes distribuídos (como a WWW) são pontos chave, uma vez que o fornecimento do material desejado deve atender a uma comunidade de usuários dispostos na rede em plataformas heterogêneas. O grande volume dos dados multimídia e o eventual grande número de usuários realizando acessos ao sistema (simultaneamente) podem levar sistemas convencionais (de armazenamento e entrega) à sobrecarga, comprometendo o atendimento da demanda.

VIDEO-ON-DEMAND

A utilização de servidores de dados multimídia baseados em sistemas VOD (*Video-On-Demand*) [Flu95, Lit94] é uma alternativa para se alcançar a eficiência exigida. Em um sistema VOD os vídeos são armazenados em um servidor de vídeo remoto, ao invés de replicar a coleção de vídeos utilizada (muito grande) em cada cliente. Um sistema VOD pode receber acessos simultâneos de centenas de usuários com diferentes preferências. O sistema deve ser capaz de garantir uma qualidade de serviço satisfatória a todos os clientes. No desenvolvimento de um sistema VOD, pesquisadores têm que resolver alguns problemas. Primeiro, o sistema deve prover mecanismos para o usuário localizar um vídeo desejado. Segundo, o sistema deve armazenar e gerenciar uma grande coleção de dados de vídeos estruturados. Terceiro, o sistema deve transportar um vídeo selecionado para o usuário final e executá-lo [Ber95].

De acordo com o nível de interatividade, os serviços de vídeo podem ser classificados em [Flu95]:

- Serviços *Broadcast* (No-VOD), similar aos canais de televisão, em que o usuário é um participante passivo e não possui nenhum controle sobre a sessão.
- Serviço *Pay-Per-View* (PPV), em que o usuário paga por uma programação específica, similar ao serviço existente em CATV (*Cable TV*).
- Serviço *Quasi Video-On-Demand* (Q-VOD), em que os usuários são agrupados pelo interesse comum que possuem. Os usuários possuem um controle rudimentar, ativado por chaveamento para os diferentes grupos.
- Serviços *Near Video-On-Demand* (N-VOD), em que funções de avançar e voltar são simuladas por transições em um intervalo de tempo discreto (da ordem de cinco minutos). Isto é feito utilizando vários canais que possuem a mesma programação, porém em tempos diferentes.
- Serviço *True Video-On-Demand* (T-VOD), em que o usuário possui total controle sobre a apresentação. O usuário tem todas as funções de VCR (*Vídeo Cassete Recorder*), incluindo avançar, voltar, congelar e posicionar diretamente. T-VOD aloca para cada usuário um canal exclusivo; com isso, o número de usuários do serviço fica limitado pelo número de canais disponíveis. T-VOD necessita de um sinal bidirecional entre o usuário e o controle central. Serviços T-VOD são essenciais para aplicações como jogos interativos, onde todas as ações de um jogador são vistas pelos outros jogadores.

ARMAZENAMENTO

Em sistemas VOD o armazenamento de dados como áudio, vídeo e gráficos (entre outros) é realizado em algum tipo de dispositivo físico (disco, fita, etc.). O armazenamento e o gerenciamento de vídeo digital são difíceis devido à grande demanda de espaço de armazenamento requerida por esse material e, devido ao custo resultante desse armazenamento. Dessa forma, o armazenamento geralmente é realizado com a ajuda de técnicas como RAID (*Redundant Array of Inexpensive Disks*) [Lee94], armazenamento hierárquico [Fed94] ou armazenamento terciário [Lou93] para aumentar a eficiência na gravação e no acesso aos dados.

BUSCA E RECUPERAÇÃO DE DADOS

Outra questão, tão importante quanto o armazenamento dos dados, é a forma de se localizar o material de interesse no repositório. Grande parte dos acessos ao servidor multimídia realizados pelos usuários visa obter informações sobre os dados armazenados, como: tamanho do arquivo, data de criação, tipos de vídeo disponíveis, duração do vídeo, etc. Mesmo com mecanismos poderosos de armazenamento e transmissão, não é uma boa idéia enviar um vídeo completo ao cliente quando se deseja saber apenas o assunto sobre o qual o vídeo trata.

Para localizar um vídeo desejado uma interface amigável deve ser criada, ajudando o usuário a encontrar o vídeo no repositório. O projeto da interface de pesquisa deve considerar quais tipos de pesquisas devem ser suportadas e como as pesquisas devem ser especificadas.

Adicionalmente, para fins de editoração e apresentação de material multimídia, são necessárias ferramentas que aliem interatividade a mecanismos poderosos de busca. A utilização de índices para responder aos tipos de pesquisa necessários é uma boa alternativa, porém dados de mídia contínua (mídias que variam com o tempo como vídeos, animações e áudios) necessitam de novas técnicas de indexação, como a utilização de metadados.

SISTEMA OPERACIONAL

O sistema operacional é fundamental para que um sistema multimídia obtenha bons resultados durante as apresentações. Sistemas operacionais convencionais foram projetados para oferecer um uso justo dos recursos do sistema entre os programas concorrentes, porém, tais sistemas sofrem a falta de serviços de tempo real necessários para o bom desempenho de aplicações multimídia.

Os sistemas operacionais convencionais apresentam problemas para suportar aplicações de mídia contínua. Segundo Buford [Buf94] alguns desses problemas são:

- **Gerenciamento de *Deadline* e Recuperação**. Ao contrário de aplicações de tempo real, muitas aplicações de mídia contínua possuem um *deadline* (prazo de entrega) inerentemente mais suave. Por exemplo, pode-se prosseguir uma videoconferência mesmo que a imagem de vídeo mais recente não possa ser processada a tempo. A perda de um *deadline* não leva ao caos total, porém a notificação da perda é uma informação importante, pois baseada nessa informação a aplicação pode querer mudar o nível de QoS (*Quality of Service*), como diminuir a taxa de *frames* sendo apresentados. Quando a aplicação perde um *deadline* devido à sobrecarga ou a erros de *hardware* ou *software*, o programa do usuário deve estar apto a tomar uma decisão. A decisão geralmente é uma recuperação que, freqüentemente, deve executar com uma prioridade maior que a atividade principal, logo, necessita aumentar sua prioridade primeiro.

• **Latência de Interrupção**. A grande latência das interrupções é uma das razões pelas quais os sistemas operacionais convencionais não suportam muito bem atividades de tempo real. Aplicações de mídia contínua produzem um número freqüente de interrupções no *kernel* resultando em pesadas trocas de contexto. Todo *kernel* não preemptivo provê latência de interrupção muito grande. O processamento dos eventos de tempo real que disparam uma nova *thread* (ou um novo processo) deve esperar até que o processamento atual do *kernel* termine, e isto demora vários milissegundos.

• **Gerenciamento de Qualidade de Serviço e Controle de Admissão**. Sistemas operacionais convencionais não possuem um mecanismo para manter um nível de QoS para aplicações de mídia contínua ou esquemas de gerenciamento e prevenção de sobrecarga (através de técnicas de controle de admissão), o que pode acarretar problemas de espera (*delay*) não predizível e *jitter* (descontinuidade no sinal provocada pela perda de um *deadline*) [Buf94, Ste95].

Rede de Comunicação

Similarmente ao armazenamento o transporte e a execução de dados de vídeo requer um certo custo por *stream*, isso porque os dados de vídeo requerem entrega em tempo real e consomem considerável largura de banda de rede quando transmitidos [Ber95].

Em sistemas multimídia distribuídos as redes de comunicação devem dar suporte a um conjunto de serviços e requisitos como: tráfego em tempo real, garantia de alto desempenho e difusão múltipla. Os protocolos existentes como o IPX e o TCP/IP não satisfazem esses requisitos. Eles foram projetados para operar em sub-redes de baixa confiabilidade. Apesar da predominância dos protocolos da família TCP/IP, eles são inadequados para a utilização de multimídia pois: o protocolo de transporte (TCP) não suporta nenhum parâmetro de qualidade de serviço, a verificação de erros por *checksum* e o controle de fluxo através de janelas deslizantes introduzem um *overhead* muito grande na transmissão, existe a ausência de conexões multiponto, não há limite máximo garantido para atraso nem para o *jitter* e não possui a previsibilidade necessária para operar em tempo real. O uso de protocolos TCP/IP é viável para aplicações multimídia quando a sub-rede de comunicação assegura alto desempenho e confiabilidade. Existe uma grande expectativa em torno das redes digitais de serviços integrados em banda larga (BISDN) usando tecnologia ATM [Com95, Cou94].

As limitações apresentadas por sistemas operacionais, redes e protocolos de comunicação convencionais não impossibilitam a construção ou execução de aplicações multimídia distribuídas. Hoje em dia existem diversas dessas aplicações e o que se almeja é o oferecimento de uma qualidade melhor das mesmas. Por exemplo, espera-se que a apresentação de um vídeo em computador tenha a mesma qualidade que a apresentação em televisão, porém, para uma apresentação de vídeos através de uma rede de computadores dificilmente consegue-se uma apresentação em tela cheia; normalmente a janela de apresentação é pequena (120x 260 geralmente). A melhoria na qualidade pode ser alcançada através de arquiteturas, algoritmos e técnicas que compensem ou anulem os problemas mencionados. No entanto, para algumas classes de aplicações essas limitações são perfeitamente aceitáveis (seção 5).

Este relatório está dividido da seguinte forma: a seção 2 mostra a arquitetura que compõe o servidor de vídeo; a seção explica como o acesso as dados está estruturado; a seção 4 apresenta os testes para avaliação de desempenho do servidor; a seção 5 dá um exemplo de ambiente que

utiliza o servidor de vídeo e a seção 6 apresenta as conclusões sobre as avaliações realizadas e sobre as perspectivas do servidor de vídeo.

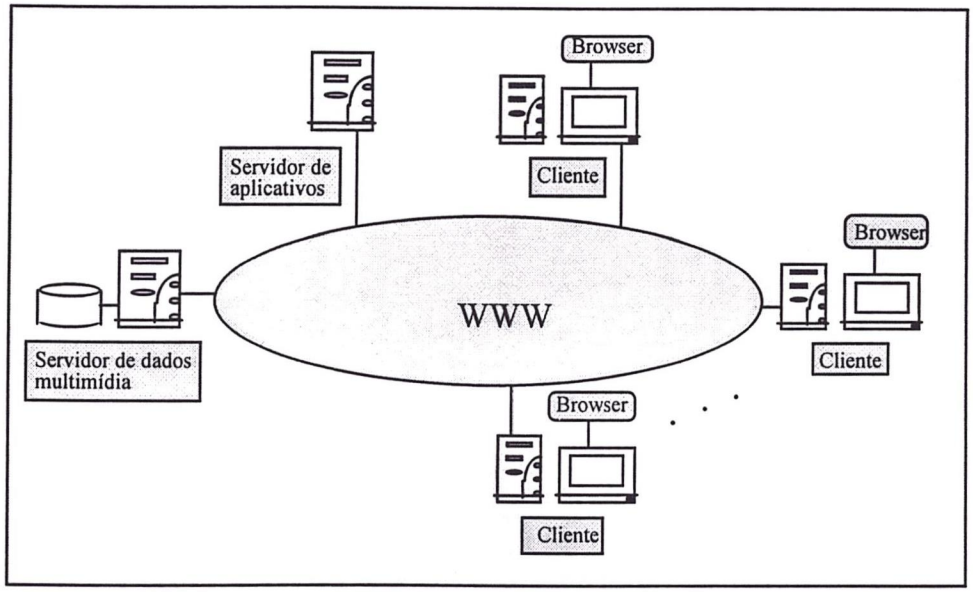

Figura 1 - Visão geral do sistema na WWW.

2 - ARQUITETURA DO SERVIDOR DE VÍDEO

O servidor de vídeo deve atender, sob demanda, a uma comunidade de clientes disposta em um ambiente WWW. Como vídeos são objetos volumosos, eles requerem formas de armazenamento e entrega diferentes das convencionais, que forneçam eficiência no atendimento a vários clientes simultâneos.

O servidor de vídeo utiliza técnica RAID para o armazenamento (figura 2). O RAID consiste na criação de uma pilha onde vários discos são agrupados fazendo com que o sistema operacional os reconheça como um único disco lógico. A tecnologias RAID possui vários níveis (de 0 a 5), sendo que o nível adotado foi o nível 0 (por este apresentar um desempenho melhor em relação aos níveis 1, 2, 3, 4 e 5). O nível 0 funciona da seguinte forma: quando um arquivo é copiado para o disco lógico cada bloco do arquivo é armazenado, paralelamente, em um disco físico. Por exemplo, em um RAID com três discos, três blocos de um arquivo são armazenados paralelamente, um bloco em cada disco físico. A vantagem é uma redução no tempo de armazenamento e leitura dos arquivos armazenados.

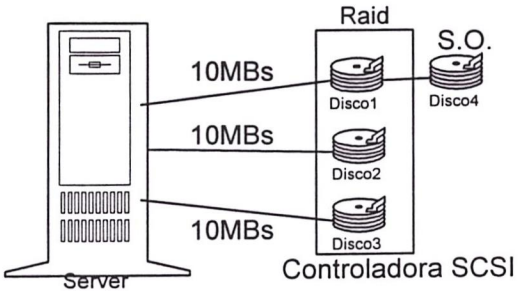

Figura 2 - Sistema de discos do servidor de vídeo.

O *hardware* que compõe o servidor de vídeo está especificado abaixo:

- Microcomputador Pentium/200 MMX.
- 96MB de memória RAM.
- 1 disco SCSI quantum de 2 GB.
- 1 disco SCSI seagate de 2 GB.
- 2 discos SCSI seagate de 4GB.
- placa controladora SCSI Adaptec 3985.

Em um disco SCSI de 2GB foi instalado o sistema operacional, assim como todos os softwares que compõem o servidor. Os arquivos de vídeo encontram-se nos outros três discos SCSI que compõe o disco lógico do sistema RAID nível 0.

Foi escolhida a controladora SCSI Adaptec 3985 por esta possuir uma característica especial em relação às controladoras tradicionais. Ela possui três canais de dados independentes (como se fossem três controladoras SCSI), sendo que cada canal trabalha a uma taxa de transmissão de 10 MB/s, resultando em uma taxa total de 30 MB/s. Dessa forma, cada disco que compõe o RAID foi colocado em um canal diferente, fornecendo uma taxa agregada que pode chegar a 30 MB/s, melhorando o serviço em acessos simultâneos.

O sistema operacional utilizado pelo servidor é o Linux versão 2.0.30, por este ser de domínio público, estar disponível para plataforma Intel e por possuir todas as características necessárias para a execução do *software* servidor, tais como: suporte à linguagem JAVA, suporte a *drivers* JDBC, possuir uma opção de instalação de um servidor httpd (Apache), entre outras.

3 - Organização dos Dados

Para que usuários inexperientes possam ter um acesso fácil ao conteúdo armazenado no servidor é necessário um mecanismo de gerenciamento dos dados armazenados que: facilite o trabalho de editoração fornecendo ferramentas interativas de busca e acesso aos dados e que forneça suporte ao compartilhamento de dados multimídia. A utilização de índices para acesso aos dados fornece interatividade, porém dados multimídia (como vídeo) requerem técnicas de indexação diferentes das convencionais. Quando uma busca por um nome é realizada, o conteúdo a ser indexado são os caracteres ASCII que compõe o nome e esses caracteres representam a informação procurada. Quando se procura pela imagem de um carro, por exemplo, o conteúdo da imagem é composto de valores binários (zeros e uns), mas o que gostaríamos de indexar é a informação semântica que esses valores representam (um carro) e não os valores em si.

A chave para a construção de uma ferramenta de busca interativa eficiente (para os propósitos discutidos) é a implantação de uma base de metadados que contenha os atributos referentes aos vídeos armazenados no servidor. Esses atributos constituem índices para responder às perguntas do usuário. Os índices podem ser: bibliográficos (título, assunto, gênero, diretor, produtor, elenco, etc.); estruturais (formando a hierarquia filme-segmento-cena); de conteúdo (para buscas baseadas em conteúdo ou em palavras-chave associadas a um conteúdo) [Ber95].

A utilização desses índices traz a vantagem adicional de prover suporte ao compartilhamento dos dados multimídia armazenados no servidor. Por exemplo, uma página

HTML pode usar um segmento de um vídeo, realizando acessos aos índices para esse segmento, em vez de copiar o segmento desejado em outro arquivo de vídeo.

A seguir temos um modelo físico [Elm94] da base de metadados implantada, composto pelas seguintes tabelas e seus atributos:

documento={ident, nomedoc, datacri, criador, tipo, comentário, instituição, endereco_doc}
indicebib={ident_b, título, resumo, dataentrada, produtor, tamanho, grupo, disciplina, assunto,
tipo_vídeo, tipo_compressão, taxa_frame, tipo_cor, tamanho_frame, tempo}
pessoa={ident_obj_pessoa, nome, sexo, biografia}
objeto={ident_obj_objeto, nomeobj, posição, forma, cor, textura, ação, descrição_obj}
p_chave={palavra, tipoitem, ocorrência}
cena={idcena, id_doc_cena, cena_segmento, tempo_início_cena, tempo_fim_cena,
descrição_cena, objeto_cena, localização_cena, frame_início_cena, frame_fim_cena}
frame={nframe, idcena, descrição_frame}
lista_de_frames={lf_idframe, lf_idobj}
lista_de_cenas={lc_idcena, lc_idobj}
indice_obj={ident_obj, tipo, ndocs, iddoc_obj}
item_frame={palavra_chave, numframe, numcena}
item_pessoa={palavra_chave, nome_pessoa, ocorrência_pessoa}
item_doc={palavra_chave, id_doc_item, ocorrência_doc}
item_bib={palavra_chave, id_bib_item, ocorrência_bib}
item_obj={palavra_chave, id_obj_item, ocorrência_obj}
item_cena={palavra_cena, id_cena_item, ocorrência_cena}
ocor_doc={palavra, nocordoc}
cast={ident_b, nomeator}
elenco_cena={idcena, nome_ator_cena}
obj_cena={id_cena, nome_obj_cena}

Além do *hardware*, o servidor de vídeo é composto por um *software* servidor e um *software* cliente. Para o desenvolvimento dos módulos cliente e servidor do sistema foi adotada a linguagem JAVA, por esta ser multiplataforma (conveniente a ambientes distribuídos baseados em WWW) e por possuir uma vasta biblioteca de funções que facilitam o esforço de programação.

O servidor é responsável por receber requisições (de apresentação de vídeos ou de busca por informações) dos clientes tratá-las e enviar a resposta adequada ao cliente. Os dados sobre os vídeos estão armazenados na base de metadados e o servidor realiza acessos a tais dados através da tecnologia *Java DataBase Connectivity* (JDBC). O JDBC corresponde a uma API de forma que cada fabricante de gerenciadores de bases de dados desenvolve um *driver* JDBC para que seu produto possa ser usado em conjunto com a linguagem JAVA. Por simplicidade, as referências no texto ao *driver* JDBC utilizado (PostgreSQL) serão feitas como JDBC simplesmente. O servidor é implementado como uma aplicação JAVA (JAVA *application*) e está instalado na máquina servidora de aplicativos.

O cliente é implementado como um *applet* JAVA, ficando instalado em cada máquina cliente e sendo apresentado em um *browser*. O cliente inclui funções para cadastro (de vídeos, cenas e *frames*), para busca e possui um *player* para apresentação de vídeos. O *player* é implementado através da utilização do *Java Media Framework* (JMF).

A comunicação cliente/servidor foi desenvolvida utilizando-se a ferramenta Visibroker 3.0, que implementa o padrão CORBA (*Common Object Request Broker*), trazendo as vantagens

da distribuição de objetos [Vin97]. Por simplicidade, as referências no texto à ferramenta Visibroker serão feitas como CORBA simplesmente.

A figura 3 ilustra o funcionamento do servidor de vídeo.

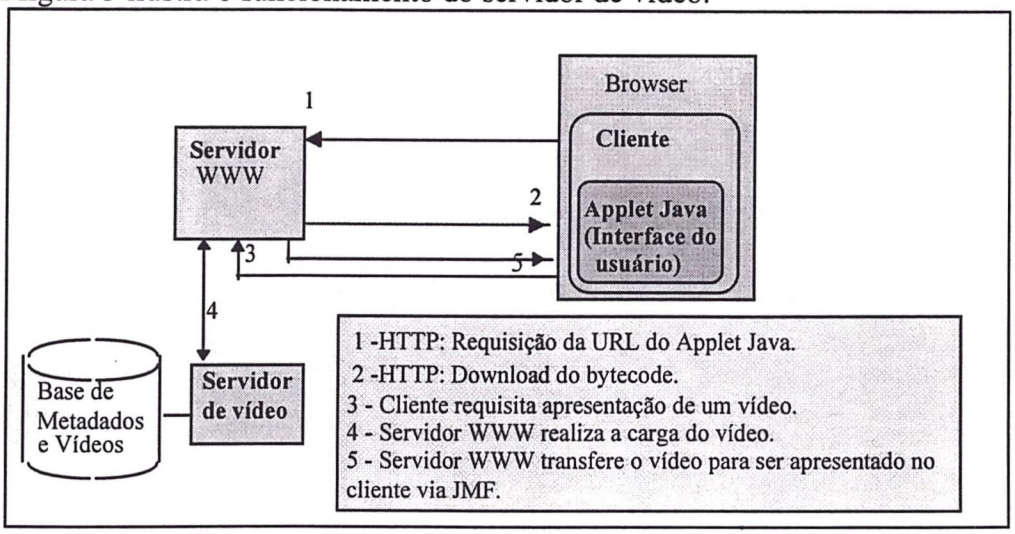

Figura 3 - Funcionamento do servidor de vídeo.

4 - TESTES REALIZADOS

Os testes realizados têm como objetivo detectar os gargalos no sistema servidor a partir de uma visão ampla do mesmo, para que se tenha uma noção clara dos pontos que podem oferecer perda de desempenho. Os testes avaliam o impacto da aplicação (de acesso, distribuição e apresentação), que utiliza tecnologias de distribuição de objetos e de acesso aos dados em ambientes heterogêneos. Os testes avaliam como o sistema operacional influi no desempenho do servidor, qual o desempenho do servidor utilizando RAID comparado a um sistema sem RAID e qual o impacto da rede de comunicação no desempenho do servidor. Com isso, avaliam-se todas as camadas por onde passam os dados durante sua transmissão.

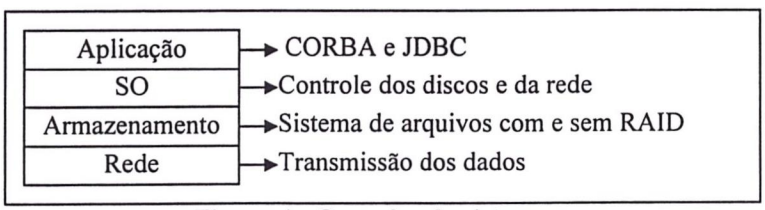

Figura 4 - Camadas do sistema.

4.1 - AVALIAÇÃO DA CAMADA DE APLICAÇÃO

O objetivo da análise da utilização das tecnologias CORBA e JDBC no sistema foi verificar o impacto que as mesmas causam no tempo de resposta às consultas do usuário. A análise baseou-se na comparação dos tempos para a realização de uma consulta pelo sistema através de uma aplicação.

A versão da ferramenta Visibroker (versão 3.0) utilizada, que implementa o padrão CORBA, não é disponível para plataforma LINUX (que é a utilizada pelo servidor de vídeo). Dessa forma, provisoriamente, a base de metadados está instalada em uma máquina diferente da máquina servidora de vídeo.

Para que as comparações pudessem ser válidas, duas versões do sistema (cliente e servidor) foram desenvolvidas. Uma versão foi implementada utilizando-se CORBA na comunicação, a outra foi implementada utilizando-se *sockets* para a comunicação. Ambas as versões foram desenvolvidas como aplicações JAVA, eliminando a presença de um servidor WWW no sistema.

Como ilustra a figura 5, a única diferença entre as duas versões utilizadas para a coleta dos tempos de transmissão é exatamente a presença em uma delas da camada de *software* do padrão CORBA.

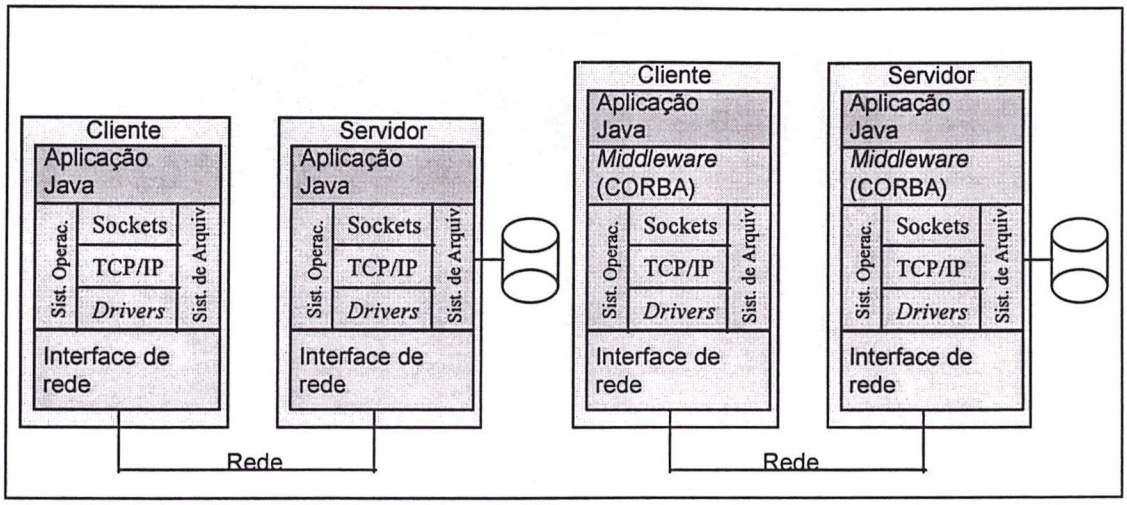

Figura 5 - As duas versões do sistema utilizadas na análise.

Os compiladores utilizados foram o JDK 1.1.3 e o VBJ 3.0 (parte do Visibroker 3.0), sendo que o VBJ foi utilizado apenas para gerar os *stubs* e os *skeletons* necessários para o estabelecimento da comunicação cliente/servidor segundo o padrão CORBA. O JDK foi utilizado para a geração dos *bytecodes* do cliente e do servidor de ambas as versões.

A *Java Virtual Machine* fornecida pelo JDK foi utilizada para a execução dos programas das duas versões do sistema.

Os testes foram realizados utilizando-se duas máquinas Sun SparcStation 5 com 32 MB de memória RAM, executando o sistema operacional Solaris 2.5, conectadas a uma rede *Ethernet* de 10 Megabits (figura 6).

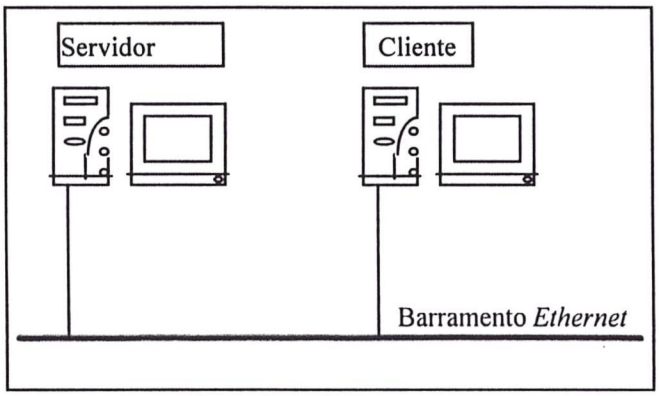

Figura 6 - Rede isolada para testes.

Foram realizadas as seguintes consultas à base de metadados:

select * from cena;

select * from cena, documento;

select * from cena, documento,frame;

select * from cena, documento,frame,objeto;

Essas consultas retornam todos os atributos das tabelas indicadas após a cláusula **from**. As *strings* contendo as respostas às consultas têm, respectivamente, tamanhos de 588 *bytes*, 1328 *bytes*, 13008 *bytes* e 19728 *bytes*.

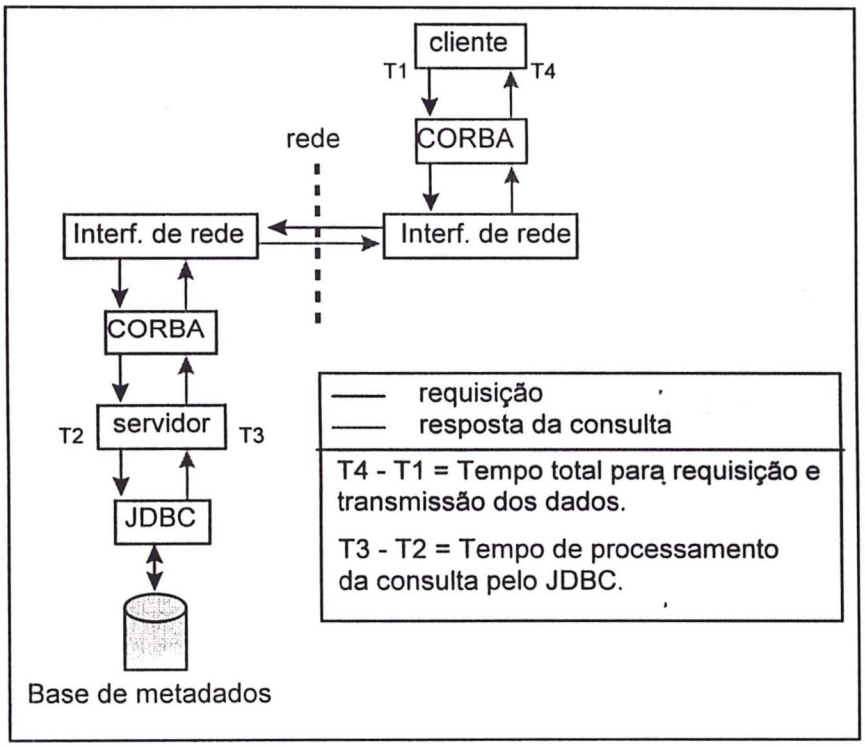

Figura 7 - Pontos de coleta dos tempos.

A figura 7 ilustra a coleta de uma amostra feita com a versão que utiliza CORBA na comunicação. Cada coleta registra o tempo em que a requisição foi enviada (T1) e o tempo em que o cliente obteve a resposta da consulta (T4). T1 e T4 são obtidos pelo relógio da máquina cliente. Na máquina servidora, foi registrado o momento em que a requisição foi passada ao JDBC (T3) e o momento em que o JDBC retorna a resposta da consulta ao servidor (T4). Para cada consulta de teste foram realizadas 30 coletas de tempo com um cliente, 30 com dois clientes e 30 com três clientes. Nas coletas com dois e três clientes as requisições eram feitas simultaneamente a partir de uma única máquina, e cada cliente enviou 30 requisições.

O mesmo procedimento de coleta foi utilizado na versão do sistema que utiliza *sockets* na comunicação.

Após a coleta dos dados, obteve-se a média dos tempos de resposta de uma consulta, em cada caso (tamanho de mensagem e número de clientes), e dos tempos de processamento do JDBC, tempos esses medidos em milissegundos. As médias foram anotadas desprezando-se as casas decimais.

Na análise de desempenho do JDBC buscou-se medir, em média, qual a porcentagem do tempo total de uma consulta que o JDBC ocupava.

A tabela 1 sumariza os resultados obtidos onde a coluna "JDBC" representa a média do tempo (em milissegundos) de processamento do JDBC para cada número de clientes em cada tamanho de mensagem. As colunas "Tempo Total" representam as médias dos tempos (em milissegundos) entre a requisição de uma consulta e a obtenção da resposta, para cada tamanho de mensagem em cada versão do sistema de teste (*socket* e CORBA). As colunas "%" indicam a taxa porcentual procurada.

Tabela 1 - Porcentagem do Tempo Total gasto pelo JDBC.

Mensagem de 588 *bytes*					
		Socket		**CORBA**	
	JDBC	**Tempo Total**	**%**	**Tempo Total**	**%**
1 cliente	725	1232	58	1320	54
2 clientes	832	1356	61	1547	53
3 clientes	969	1696	57	1736	55
Mensagem de 1328 *bytes*					
		Socket		**CORBA**	
	JDBC	**Tempo Total**	**%**	**Tempo Total**	**%**
1 cliente	838	1367	61	1467	57
2 clientes	1016	1659	61	1746	58
3 clientes	1281	2051	62	2776	46
Mensagem de 13008 *bytes*					
		Socket		**CORBA**	
	JDBC	**Tempo Total**	**%**	**Tempo Total**	**%**
1 cliente	1069	3629	29	3945	27
2 clientes	1731	3990	43	6570	26
3 clientes	2730	7653	35	12315	22
Mensagem de 19728 *bytes*					
		Socket		**CORBA**	
	JDBC	**Tempo Total**	**%**	**Tempo Total**	**%**
1 cliente	1623	5210	31	5723	28
2 clientes	2536	7308	34	7792	32
3 clientes	3739	9463	39	19213	19

Os resultados apontados na tabela 1 (e na figura 8) mostram que para mensagens pequenas o JDBC ocupa uma parte substancial do tempo total para a realização de uma consulta, ficando entre 57 e 62% na versão *socket* e entre 46 e 58% na versão CORBA.

Observa-se que aumentando o tamanho da mensagem a porcentagem diminui. Isso porque o tempo total para processamento de uma consulta (tempo entre o cliente requisitar a consulta e obter a resposta) aumenta a uma taxa maior que o tempo de processamento da consulta pelo JDBC no servidor. Por exemplo, o tempo de processamento do JDBC para três clientes com mensagem de 1328 *bytes* foi de 1281 milissegundos e aumentou para 3739 milissegundos com três clientes e mensagem de 19728 *bytes*, um aumento de 291%. Considerando-se o mesmo número de clientes e os mesmos tamanhos de mensagens, o tempo total da versão CORBA aumentou de 2776 para 19213, um aumento de 692% (a versão *socket* aumentou 461%).

Com isso conclui-se que para os propósitos do sistema, que utilizará mais freqüentemente mensagens pequenas, o *driver* JDBC utilizado constitui um "gargalo".

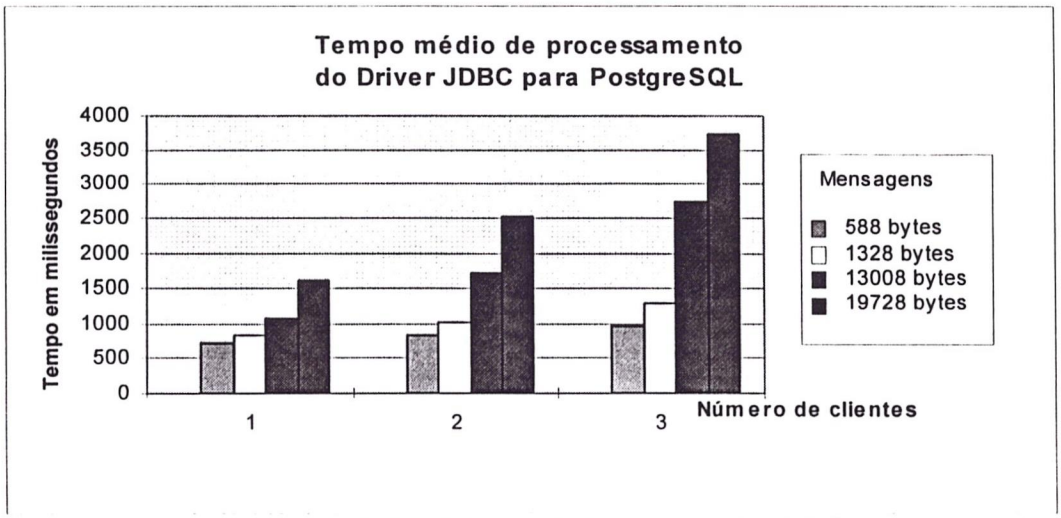

Figura 8 - Gráfico do tempo médio de processamento do JDBC.

Com relação à utilização de CORBA, analisando-se os dados obtidos e comparando-se o desempenho das duas versões de teste do sistema, chegou-se aos resultados contidos na tabela 2. Esta tabela mostra as médias do tempo total gasto pelo sistema entre o cliente requisitar uma consulta e obter a resposta (*Socket* e CORBA), subtraídas do tempo gasto pelo JDBC para o processamento das consultas. A coluna *socket* mostra o tempo médio de transmissão das mensagens, sem computar o tempo de processamento do JDBC no servidor. A coluna CORBA mostra o tempo médio da transmissão das mensagens, também sem computar o tempo de processamento do JDBC, mas incluindo o tempo gasto pela camada CORBA. A diferença entre essas duas colunas (contida na coluna milissegundos) é o atraso medido em milissegundos imposto pela utilização de CORBA no sistema. A coluna "%" mostra em termos porcentuais o quanto a versão de teste do sistema utilizando CORBA foi mais lenta que a versão utilizando *sockets*.

Tabela 2 - Comparação de desempenho entre as versões *Socket* e CORBA.

Mensagem de 588 *bytes*				
	Socket	**CORBA**	**%**	**Milissegundos**
1 Cliente	526	576	9	50
2 Clientes	534	704	31	170
3 Clientes	551	942	70	391
Mensagem de 1328 *bytes*				
	Socket	**CORBA**	**%**	**Milissegundos**
1 Cliente	548	607	10	59
2 Clientes	566	807	42	241
3 Clientes	609	1656	171	1047
Mensagem de 13008 *bytes*				
	Socket	**CORBA**	**%**	**Milissegundos**
1 Cliente	2570	2866	11	296
2 Clientes	2894	4203	45	1309
3 Clientes	4359	10149	132	5790
Mensagem de 19728 *bytes*				
	Socket	**CORBA**	**%**	**Milissegundos**
1 Cliente	3609	4077	12	468
2 Clientes	3956	6071	53	2115
3 Clientes	5059	16138	218	11079

Como o sistema de teste foi implementado em duas máquinas (uma servidora e uma cliente) deve-se considerar o atraso imposto pela máquina cliente ao processar o envio e o recebimento de requisições concorrentes (2 e 3 clientes).

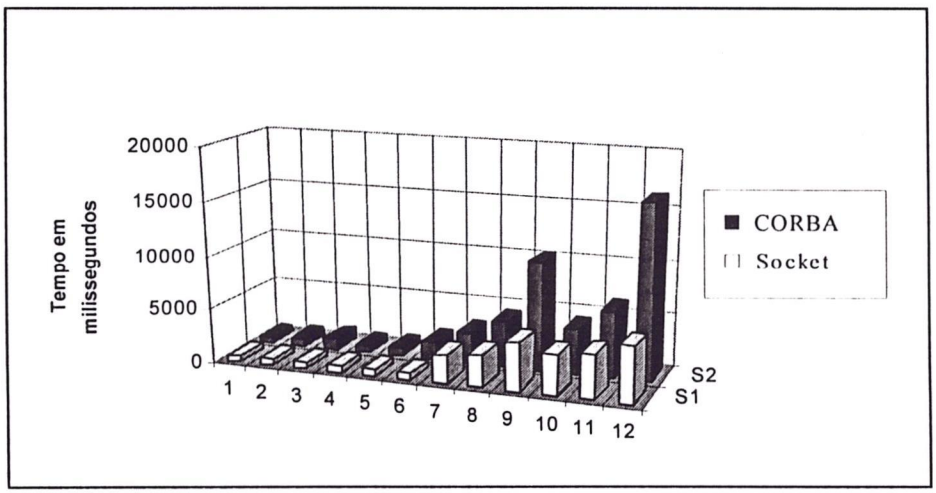

Figura 9 - Gráfico comparativo do desempenho entre as versões CORBA e *Sockets*.

Na figura 9 as colunas de 1 a 3 representam as mensagens de 588 *bytes*, as colunas de 4 a 6 representam as mensagens de 1328 *bytes*, as colunas de 7 a 9 representam as mensagens de 13008 *bytes* e as colunas de 10 a 12 representam as mensagens de 19728 *bytes*.

Os testes mostraram que a utilização de CORBA impõe um atraso na comunicação cliente/servidor, como esperado (figura 9). Em termos porcentuais as diferenças são consideráveis (tabela 2), porém o tempo que o usuário final terá que esperar a mais por uma

resposta não é tão significativo para os tamanhos de mensagens que serão mais utilizados no sistema (entre 100 e 5000 *bytes*). Por exemplo, um usuário que requisitar uma consulta cujo tamanho da mensagem de resposta seja de 13008 *bytes* terá que esperar 0,296 segundos (em média) a mais se utilizar CORBA na comunicação cliente/servidor.

Tabela 3 - Desvio padrão das amostras.

Mensagem de 588 *bytes*			
	CORBA (Desv. Padrão; Média)	***Socket* (Desv. Padrão; Média)**	**JDBC (Desv. Padrão; Média)**
1 cliente	186; 1320	172; 1232	104; 725
2 clientes	289; 1547	304; 1356	219; 832
3 clientes	524; 1736	331; 1696	327; 969
Mensagem de 1328 *bytes*			
	CORBA (Desv. Padrão; Média)	***Socket* (Desv. Padrão; Média)**	**JDBC (Desv. Padrão; Média)**
1 cliente	137; 1467	127; 1367	95; 838
2 clientes	177; 1746	373; 1659	285; 1016
3 clientes	1074; 2776	689; 2051	560; 1281
Mensagem de 13008 *bytes*			
	CORBA (Desv. Padrão; Média)	***Socket* (Desv. Padrão; Média)**	**JDBC (Desv. Padrão; Média)**
1 cliente	605; 3945	187; 3629	98; 1069
2 clientes	856; 6570	709; 3990	953; 1731
3 clientes	3250; 12315	2893; 7653	1305; 2730
Mensagem de 19728 *bytes*			
	CORBA (Desv. Padrão; Média)	***Socket* (Desv. Padrão; Média)**	**JDBC (Desv. Padrão; Média)**
1 cliente	476; 5723	329; 5210	158; 1623
2 clientes	2190; 7792	848; 7308	1241; 2536
3 clientes	7693; 19213	2216; 9463	1874; 3739

4.2 Testes com SOs Diferentes - Linux e Windows/NT

Metodologia Aplicada

Antes de apresentar os resultados do teste com o sistema operacional, vale descrever a metodologia utilizada. Vale também ressaltar que essa metodologia foi empregada em todos os testes, com exceção da avaliação da utilização das tecnologias CORBA e JDBC. Para a realização dos testes de desempenho do servidor de vídeo o ambiente ilustrado na figura 10 foi montado.

O servidor de vídeo segue a especificação da seção 2 (figura 2). Os clientes são microcomputadores Pentium de 166 MHz, com 32 MB de memória RAM e discos rígidos de 1.7 GB (IDE), com placas de rede *Fast-Ethernet* 3Com PCI 10/100 Mbps.

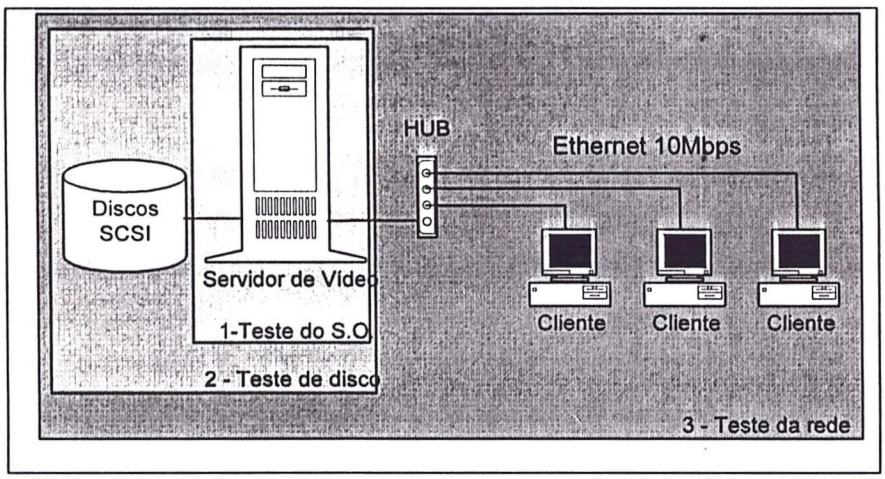

Figura 10 – Ambiente de testes.

Os testes consistiram na transferência de dois arquivos, um de 7MB e outro de 81MB, de forma local e de forma remota. Para a transferência dos arquivos foi utilizado o programa FTP por este fornecer a taxa e o tempo de transferência de um arquivo e, por estar disponível para as plataformas utilizadas.

Os testes realizados foram os seguintes:

- Teste 1 - transferência de um arquivo de 7MB.
- Teste 2 - dois acessos concorrentes à mesma cópia do arquivo de 81MB.
- Teste 3 - três acessos concorrentes à mesma cópia do arquivo de 81MB.
- Teste 4 - dois acessos concorrentes a três cópias diferentes do arquivo de 81MB.
- Teste 5 - três acessos concorrentes a três cópias diferentes do arquivo de 81MB.

DESEMPENHO DO SISTEMA OPERACIONAL

O objetivo desse teste é comparar o desempenho do servidor de vídeo (com e sem a tecnologia RAID) em dois sistemas operacionais diferentes, Linux (versão *Slakware* 3.3 *kernel* 2.0.30) e Windows/NT 4.0. Os dois SOs foram instalados no servidor de vídeo. Foram realizadas transferências de arquivos seguindo a metodologia descrita acima.

Os testes realizados compararam o desempenho do servidor, em plataforma Linux, quando esta utiliza e quando não utiliza RAID. Também comparam o desempenho do servidor em plataformas diferentes (Windows/NT e Linux) sem utilizar RAID. Nesse teste não foi possível comparar o desempenho do servidor entre os dois sistemas operacionais utilizando a tecnologia RAID, já que o Windows/NT não possui suporte para a placa Adaptec 3985 (controladora de RAID). Para os testes sem a tecnologia RAID empregou-se um (único) disco rígido SCSI quantum de 4.0 GB e uma placa controladora SCSI Adaptec 2940AU. A tabela 4 mostra os resultados obtidos com os testes, que expressam a taxa de transferência dos arquivos em *KiloBytes* por segundo.

Tabela 4 - Desempenho do servidor com SOs diferentes.

Teste	WindowsNT (kbytes/ seg)	Linux (kbytes/ seg)	Linux c/ RAID (kbytes/ seg)
1	1372	2780	3530
2	745	1917	2687
3	522	1482	2138
4	571	898	1483
5	416	660	948

Conclui-se que o sistema operacional Windows/NT, em sua configuração padrão, apresenta um desempenho inferior ao sistema operacional Linux no atendimento a requisição locais de arquivos (tabela 4 e figura 10). Outra constatação é que a utilização da tecnologia RAID (de forma local) no gerenciamento do sistema de arquivos obtém um desempenho superior em relação do sistema de arquivos tradicional (único disco).

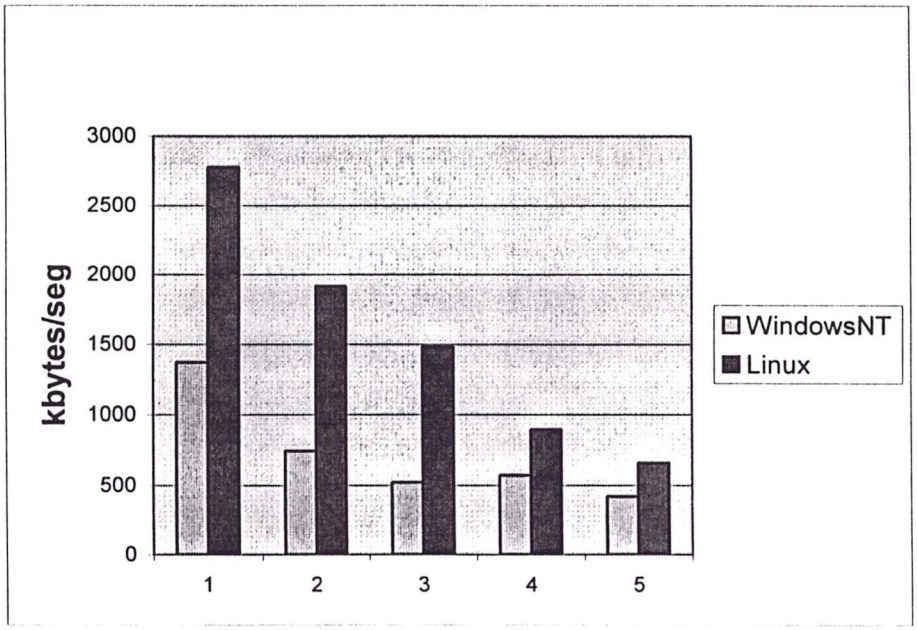

Figura 11 - Desempenho do servidor com sistemas operacionais diferentes.

4.3 - TESTES COM A REDE

O teste anterior apontou um desempenho melhor do servidor de vídeo, em ambiente local, com a plataforma Linux. Como os sistemas operacionais em questão possuem formas diferentes de gerenciar o envio de informações pela rede, buscou-se verificar se o Linux continuaria oferecendo um melhor serviço quando a transferência de arquivos ocorresse de forma remota.

Utilizaram-se duas configurações de rede diferentes. Em uma configuração o servidor e as estações clientes estão conectados através de uma rede *Ethernet* de 10Mbps (figura 12), na outra é utilizada uma rede FDDI de 100Mbps conectando o servidor e os clientes através de um *Switch Ethernet* de 10 Mbps (figura 13). A configuração do servidor utilizada nesse teste foi a mesma configuração sem RAID do teste anterior.

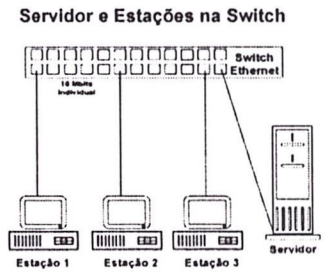

Figura 12 - Rede Ethernet para testes.

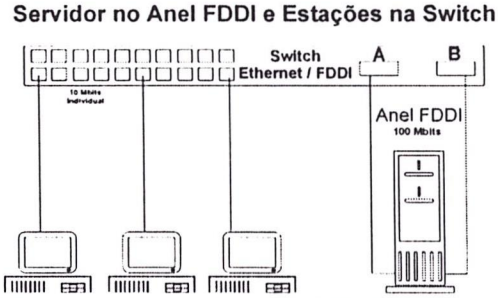

Figura 13 - Rede FDDI+*Switch ethernet* para testes.

Os resultados obtidos dos testes sobre a rede *Ethernet* são mostrados na tabela 5 e na figura 14. Os resultados dos testes sobre a rede FDDI são mostrados na tabela 6 e a figura 15. A figura 16 compara o porcentual de utilização da rede *Ethernet* entre os dois sistemas operacionais.

Tabela 5 - Desempenho sobre rede *Ethernet*.

Teste	WindowsNT (kybtes/ seg)	Linux (kbytes/ seg)
1	783	1069
2	524	569
3	420	426
4	553	568
5	380	386

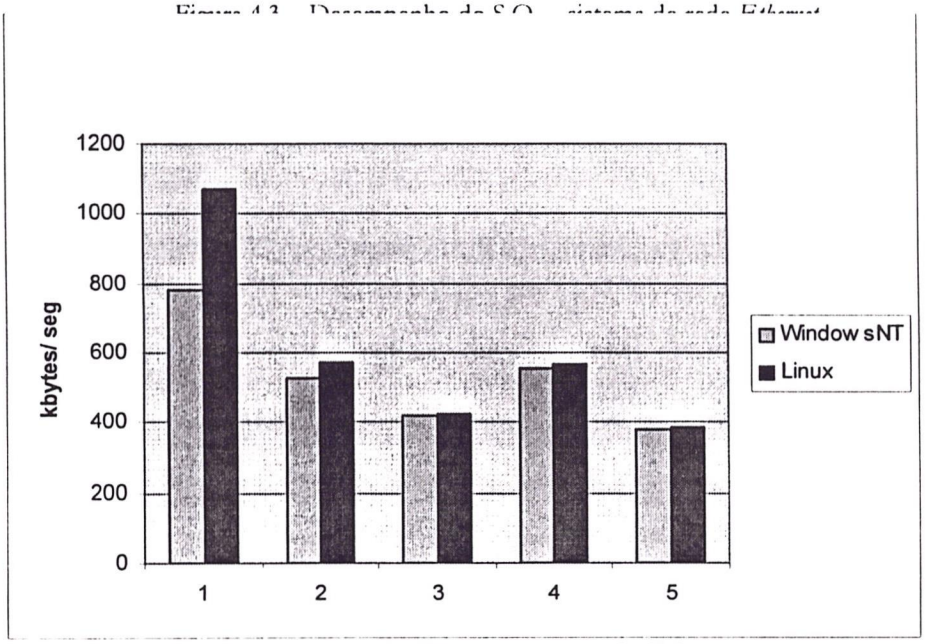

Figura 14 - Desempenho do SO sobre uma rede *Ethernet*.

Tabela 6 - Desempenho sobre rede FDDI.

Teste	WindowsNT (kybtes/ seg)	Linux (kbytes/ seg)
1	1039	1069
2	1060	1075
3	1000	1065
4	986	1064
5	779	1008

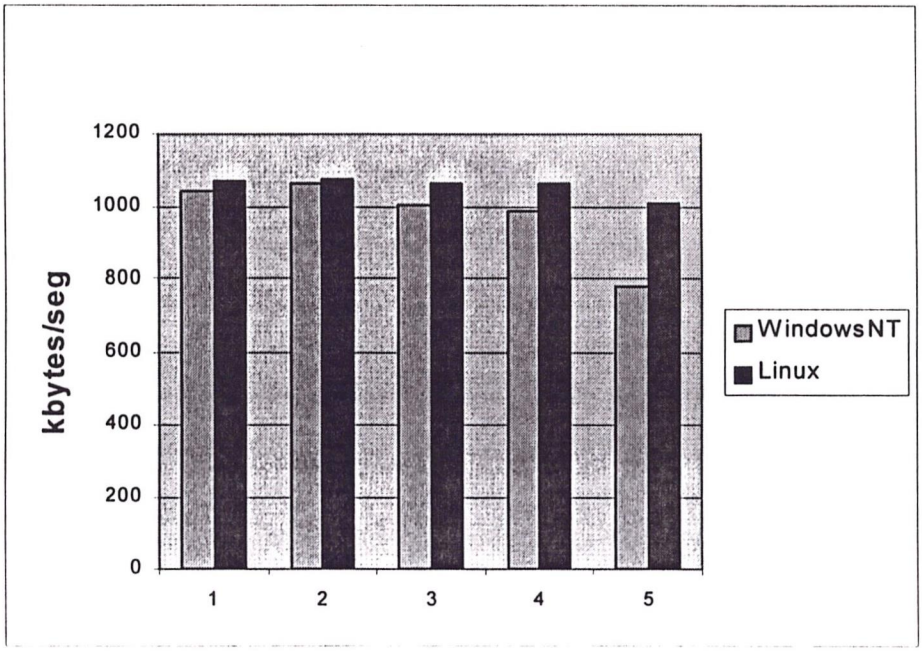

Figura 15 – Desempenho do S.O. – sistema de rede FDDI + Switch *Ethernet*.

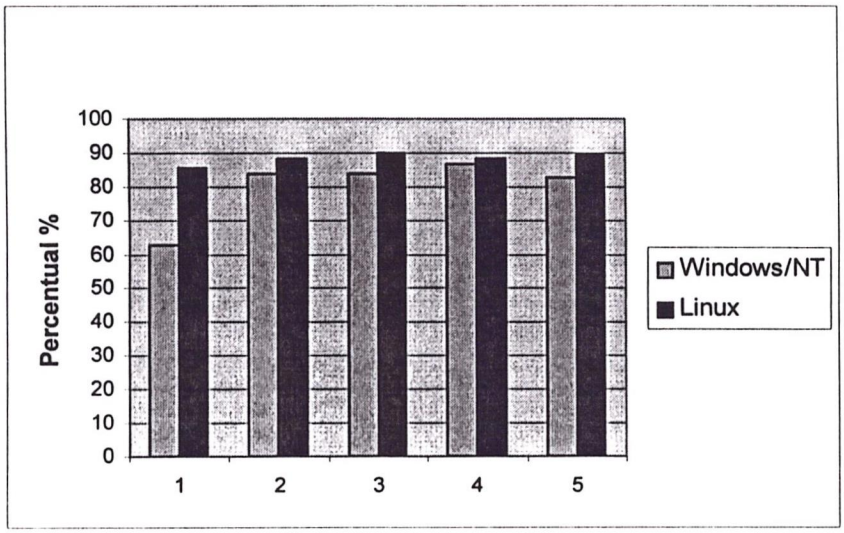

Figura 16 – Porcentual de utilização da rede *Ethernet*.

O Windows/NT apresentou um desempenho inferior ao Linux na transferência dos arquivos nos dois casos analisados (*Ethernet* e FDDI).

Outra conclusão é que a rede *Ethernet* se torna um gargalo para um número médio de clientes (figura 14). Analisando a figura 16 podemos observar que no teste 1, o gargalo é o sistema de rede do Windows/NT, já nos testes seguintes o gargalo é a rede *Ethernet* (10Mbps), já que a sua utilização chega a 90%, que para a rede *Ethernet* é uma taxa alta. No sistema operacional Linux, o gargalo sempre é a rede.

Na figura 15 podemos concluir que o gargalo inicialmente é a rede *Ethernet* (10Mbps), porém para um número maior de clientes, o gargalo se torna o sistema de arquivo, concluímos também que gerenciamento de acesso ao disco do WindowsNT é menos eficiente que do Linux.

4.4 - AVALIAÇÃO DO ACESSO REMOTO AO DISCO VIA ETHERNET

Quando o sistema RAID é instalado em um computador, ele promove mudanças na forma de acesso e armazenamento dos dados, modificando o sistema de arquivos.

O objetivo desse teste foi comparar o desempenho do servidor de vídeo, sobre uma rede de computadores, ao se utilizar um sistema de arquivo tradicional (único disco) e utilizando o sistema de arquivo com a tecnologia Raid-0. O mesmo teste já foi realizado de maneira local (descrito na seção 4.2) e os resultados estão na tabela 4. Os testes foram realizados sobre o sistema operacional Linux. O servidor foi configurado com a controladora SCSI Adaptec 3985 (controladora de RAID) e o disco lógico RAID foi montado sobre os três discos rígidos SCSI quantum (1 de 2GB e 2 de 4GB). Os resultados obtidos são mostrados na tabela 7 e na figura 17.

Tabela 7 - Desempenho do servidor sobre ethernet.

Teste	Um único disco SCSI (kbytes/seg)	Tecnologia RAID (kbytes/ seg)
1	1064	1069
2	579	569
3	427	426
4	559	568
5	388	386

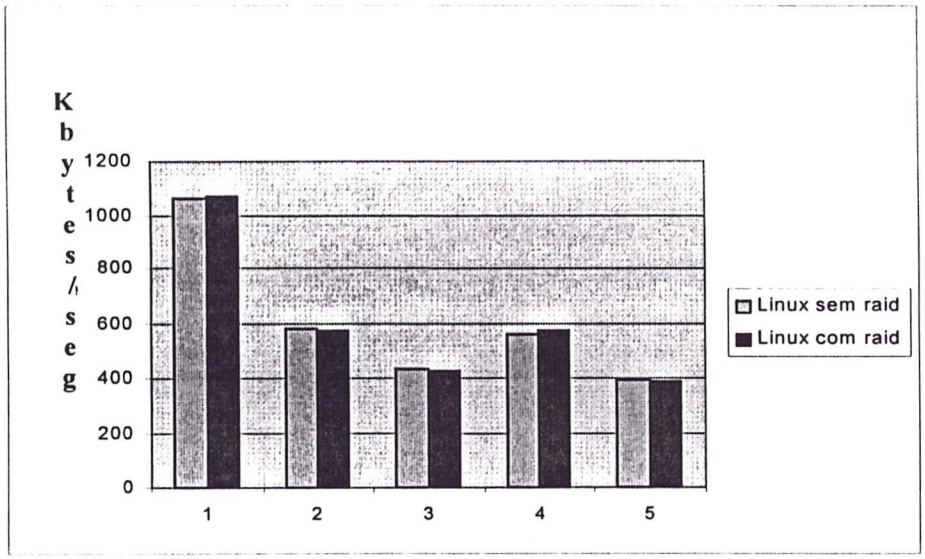

Figura 17 – Desempenho do Sistema de Arquivo sobre uma rede *Ethernet* (10Mbps).

Conclui-se que o gargalo do sistema é a rede *Ethernet* (10Mbps). Tanto o sistema de arquivo tradicional quanto no sistema de arquivo com tecnologia RAID apresentaram resultados semelhantes. A utilização da rede neste teste foi de aproximadamente 90%, que é alta para a rede *Ethernet* (10Mbps).

4.5 - AVALIAÇÃO DO ACESSO REMOTO AO DISCO VIA FDDI

Este teste visa medir o desempenho ao se utilizar um sistema de arquivo tradicional e utilizando o sistema de arquivo com a tecnologia Raid-0 sobre uma rede onde o servidor está conectado através de uma rede FDDI (100Mbps) e as estações estão conectados através de uma switch *Ethernet* (10Mbps). Os testes foram realizados sobre o Sistema Operacional Linux e com a controladora SCSI Adaptec 3985, o disco lógico RAID foi montado sobre os três discos rígidos SCSI quantum (1 de 2GB e 2 de 4GB), placa de rede digital FDDI DEFPA-AA. Os resultados obtidos são mostrados na tabela 8 e na figura 18.

Tabela 8 - Desempenho do servidor sobre rede FDDI.

Teste	Um único disco SCSI (kbytes/seg)	Tecnologia RAID (kbytes/ seg)
1	1069	1078
2	1075	1076
3	1065	1063
4	1064	1061
5	1008	1067

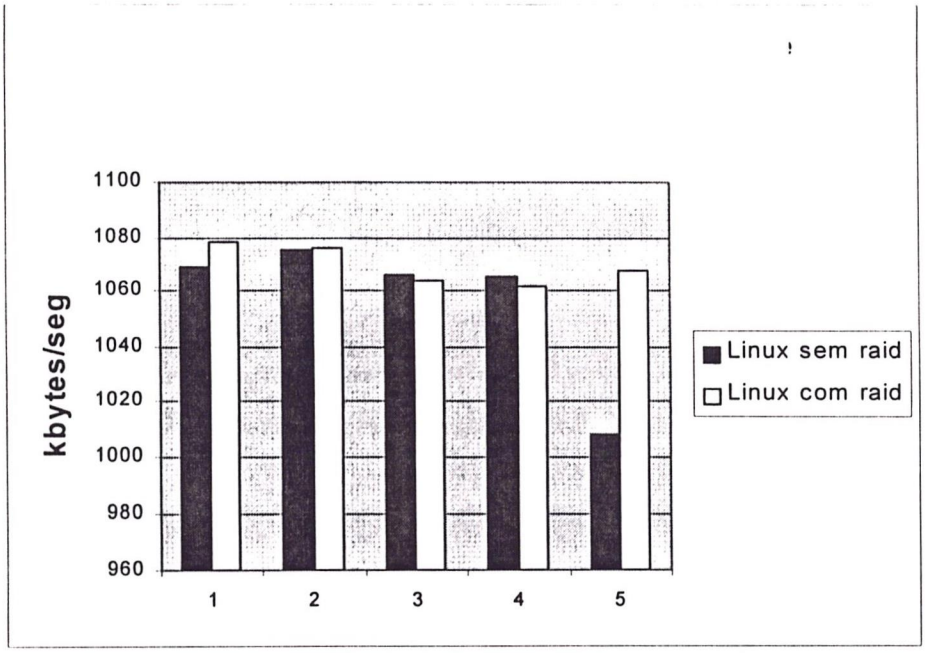

Figura 18 – Desempenho do Sistema de Arquivo sobre uma rede FDDI e switch *Ethernet*.

Concluímos que o gargalo do sistema inicialmente é a rede *Ethernet* (10Mbps), porém com um número médio de clientes, demonstrou que o sistema de arquivo tradicional se torna o gargalo, o gráfico demonstra também que o gargalo sempre é a rede no sistema de arquivo com tecnologia RAID.

5. Exemplo de Utilização do Servidor

Como exemplo de utilização do sistema servidor de vídeo tem-se o ambiente Interland [Cas97], que é composto por aplicações de autoria e de apresentação hipermídia, implementadas pensando-se nos usuários finais: professores e alunos. Além disso, outras aplicações de apoio ao professor, tais como registro do comportamento dos alunos, devem estar disponíveis.

Um componente interessante desse ambiente, que pode ser utilizado em diversas situações de treinamento, é a sessão de vídeo controlada. A função desse componente é permitir a apresentação de um trecho de vídeo (eventualmente, mais de um) para diversos alunos ao mesmo tempo e, além disso, permitir que o professor tenha total controle sobre a execução do mesmo.

Um caso de uso desse componente ocorre na situação em que o professor deseja apresentar um vídeo e parar em posições predeterminadas, ou não, para explicar um assunto correlacionado e depois continuar a execução. Se, por exemplo, um aluno solicitar um detalhamento sobre um ponto específico do vídeo, o professor pode marcar um trecho do mesmo através de seu console e executar novamente esse trecho em todas as máquinas que estão assistindo a sessão, em um subconjunto dessas máquinas ou apenas na máquina do aluno que solicitou a informação.

Em termos de autoria, o professor deverá apenas escolher esse componente e fornecer parâmetros como o trecho de vídeo a ser utilizado; e a ferramenta de edição será responsável pela inclusão dos *tags* e *scripts* necessários na página HTML para execução do componente.

Em termos de apresentação, os alunos deverão carregar a página contendo o componente e este último encarregar-se-á, transparentemente, de tarefas como: registro com o servidor de controle e início da transferência do trecho do vídeo do servidor apropriado para a máquina local. Alternativamente, em sistemas não baseados em páginas como a WWW, o professor pode ativar o programa de apresentação controlada de vídeo nas máquinas dos alunos através de comandos remotos.

Restrições quanto à utilização desse componente em aplicações voltadas para a Internet certamente são aplicáveis, uma vez que a infra-estrutura existente ainda não fornece suporte para entrega de dados em tempo real e garantia de qualidade de serviço. No entanto, a utilização de vídeo em redes internas (*intranets*) é uma realidade.

6 - Conclusões

Os testes realizados mostram que é perfeitamente possível a implementação de sistemas de distribuição de vídeo via tecnologias de rede Internet. Para contornar os problemas oriundos da precariedade dos protocolos TCP/IP e ethernet para a transmissão de mídia contínua, nota-se o aparecimento de várias tecnologias que possibilitam o uso de um sistema de armazenamento remoto de vídeo e sua apresentação local com razoável qualidade **sem a utilização de sincronismo** durante todo o processo (p. ex. Vxtreme, RealVideo, CU-Seeme e o padrão JMF).

Desta forma, a questão principal torna-se o provimento de largura de banda necessária para a aplicação. A maioria dos problemas inseridos pela rede e pelos sistemas operacionais são resolvidos através de esquemas avançados de bufferização.

Este trabalho avaliou várias opções de tecnologias "convencionais", mostrando configurações possíveis de serem utilizadas para o serviço de armazenamento e distribuição de vídeo. Em termos gerais, a única peça não-convencional utilizada pelo sistema é uma placa especial SCSI, com 3 canais. A utilização de RAID é alcançada via software padrão embutido no Linux.

A tabela 4 mostra que a configuração melhor testada consegue entregar 7,060 MBytes por segundo (2 vezes 3,530, pois cada transferência envolve dois acessos ao disco). Este número é altamente dependente da configuração (CPU, discos, controladora, RAM, etc). Tem-se notícia de plataformas Linux provendo taxas de acesso médias de 27 MBytes por segundo (dual pentium 300 MHz, 1 GBytes de memória; 8 discos SCSI 4 GBytes e controladora DPT ultrawide).

Apesar do sistema poder ser utilizado na "grande rede", prevemos uma demanda maior para serviços como este em redes locais (empresas e escolas). Vários esquemas diferentes de rede podem ser utilizados para a distribuição. Os testes nesta área mostram o ganho que se pode ter quando se utiliza switches que possuem portas de alto desempenho (no caso foi utilizado FDDI) para conexão do servidor à switch. Por exemplo, supondo-se que uma aplicação necessitasse de 600 KBytes por segundo (padrão MPC3) e que esta fosse utilizada simultaneamente por 20 máquinas, a quantidade agregada de dados transmitidos seria de 12 MBytes por segundo (96 Mbits por segundo). Uma configuração possível de rede seria a conexão das 20 estações numa switch ethernet com porta FDDI para o servidor. Neste caso, precisaríamos de um servidor melhor configurado que o utilizado nos testes, com custo adicional relativamente baixo (vide, na figura 16 que, com 3 máquinas acessando o sistema, já se nota a limitaçÃo do acesso ao disco no caso **sem** RAID. Se o teste tivesse sido continuado com mais máquinas clientes, chegaríamos também a notar o limite do sistema **com** RAID). Situações de demanda maior de largura de banda precisaria de diferente arquitetura de rede, pois no caso descrito a conexão FDDI está próxima do limite de sua capacidade. Uma troca de FDDI por ATM a 155 Mbps poderia prover uma capacidade adicional ao sistema.

Os testes de medida de performance no acesso a bases de dados e à rede também são reveladoras, principalmente por mostrarem que o acesso via JDBC impõe um delay razoável. No processo de apresentação do vídeo este delay não é significativo, pois aparece apenas na fase inicial (requisição) da transmissão do vídeo Vale salientar aqui que deve-se levar em consideração que implementações de CORBA e de JDBC podem variar muito em termos de performance. Uma avaliação comparativa entre várias implementações mostrou-nos ser uma tarefa interessante para o futuro.

Finalmente, devemos ressaltar que, apesar de não avaliado quantitativamente, a organização do sistema, dividido em uma base de dados de títulos (servidor de metadados) e uma base de dados de conteúdo torna o processo de edição e apresentação de vídeo bastante eficiente e flexível.

REFERÊNCIAS BIBLIOGRÁFICAS

[Ber95] Berger, D. **Video-on-Demand Metadata Query Interfaces**. Tese de metrado, Universidade da Califórnia, Berkeley, 1995.

[Buf94] Buford, J. F. K. **Multimedia Systems**. Addison-Wesley, 1994.

[Cas97] Castro, M.Alice.S; Goularte, R.; Reami, E. R.; Moreira, E.D.S. **Infra-Estrutura de Suporte a Editoração de Material Didático Utilizando Multimídia**. Revista Brasileira de Informática na Educação, n. 1, pp. 61-70, setembro, 1997.

[Com95] Comer, D. E. **Internetworking Whit TCP/IP**. Vol. 1, Prentice-Hall, 1995.

[Cou94] Couloris, G. F.; Dollimore, J. **Distributed Systems**. Addison-Wesley Publishing Company, 1994.

[Elm94] Elmasri, R; Navathe, S. B. **Fundamentals of Database Systems**. The Benjaming/Cummings Publishing Company, Inc, 1994.

[Fed94] Federighi, C.; Rowe, L. A. **A Distributed Hierarchical Manager for a Video-on-Demand System**. Anais do Symp. On Elec. Imaging Sci. & Tech., pp. 185-197, fevereiro, 1994.

[Flu95] Fluckiger, F. **Understanding Networked Multimedia Applications and Technology**. Hemel Hempstead - UK, Prentice-Hall, 1995.

[Lee94] Lee, E. K.; Chen, P. M.; Hartman, J. H.; Drapeau, A. L. C.; Miller, E. L.; Katz, R. H.; Gibson, G. A.; Patterson, D. A. **RAID-II: A Scalable Storage Architecture for High-bandwidth Network File Service**. Anais do 21st International Symposium on Computer Architcture, pp. 234-244, abril, 1994.

[Lit94] Little, T.D.C.; Venkatesh, D. **Client Server Metadata Management for the Delevery of Movies in a Video-on-Demand System**. Anais do 1st Intl. Workshop on Services in Distributed and Netwoked Enviroments, pp. 11-18, junho, 1994.

[Lou93] Lougher, P.; Shepherd, D. **The Design of a Storage Server for Continuous Media**. The Computer Journal, vol. 36, n. 1, pp. 32-42, 1993.

[Mat94] Mattison, P. **Practical Digital Video with Programming Examples in C**. Wiley Professional Computing, 1994.

[Mor95] Moreira, E. S.; Nunes, M. G. V.; Pimentel, M. G. C. **Design Issues for a Distributed Hypermedia-based Tutoring System (HyDTS)**. Anais do International Conference on Computer Application in Industry, pp. 108-113, dezembro, 1995.

[Ste95] Steinmetz, R.; Narstedt, K. **Multimedia: Computing, Communications and Applications**. Prentice-Hall, 1995.

[Vin97] Vinoski, S. **CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments**. IEEE Communications Magazine, vol. 14, n. 2, fevereiro, 1997.