UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO

(Bacharelado)

PROTÓTIPO DE SOFTWARE PARA DISPONIBILIZAÇÃO DE UM SINAL DE ÁUDIO EM TEMPO REAL PELA INTERNET

TRABALHO DE CONCLUSÃO DE CURSO SUBMETIDO À UNIVERSIDADE REGIONAL DE BLUMENAU PARA A OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA COM NOME EQUIVALENTE NO CURSO DE CIÊNCIAS DA COMPUTAÇÃO — BACHARELADO

ONI ROGÉRIO PEREIRA JÚNIOR

BLUMENAU, JUNHO/2002

2002/1-58

PROTÓTIPO DE SOFTWARE PARA DISPONIBILIZAÇÃO DE UM SINAL DE ÁUDIO EM TEMPO REAL PELA INTERNET

ONI ROGÉRIO PEREIRA JÚNIOR

ESTE TRABALHO DE CONCLUSÃO DE CURSO FOI JULGADO ADEQUADO PARA OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA DE TRABALHO DE CONCLUSÃO DE CURSO OBRIGATÓRIA PARA OBTENÇÃO DO TÍTULO DE:

BACHAREL EM CIÊNCIAS DA COMPUTAÇÃO

_____

Prof. Francisco Adell Péricas — Orientador na FURB

_____

Prof. José Roque Voltolini da Silva — Coordenador do TCC

BANCA EXAMINADORA

_____

Prof. Francisco Adell Péricas

_____

Prof. Sérgio Stringari

_____

Prof. Antônio Carlos Tavares

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

AGRADECIMENTOS

Acima de tudo a Deus, por todas as bênçãos que recebi para ter a oportunidade de estar aqui, completando mais uma etapa da minha vida.

Aos meus pais, Oni Rogério Pereira e Maria Juliana Pires Pereira, pelo apoio e incentivo nos momentos bons e nos mais difíceis durante todos esses anos.

Ao professor Francisco Adell Péricas, pela atenção e orientação durante o desenvolvimento deste trabalho.

Aos professores, pelo conhecimento que transmitiram durante todas as aulas que freqüentei, conhecimento este que me fará lembrar de cada um durante o exercício de minha profissão, por toda a vida.

A todos aqueles com quem convivi durante todos esses anos na universidade, grandes amigos e alguns deles grandes irmãos, e àqueles que de alguma forma contribuíram para a realização deste trabalho.

RESUMO

O presente trabalho destina-se ao estudo da transmissão e recepção de áudio em tempo real através da Internet. Apresenta considerações sobre a arquitetura Internet e técnicas de transmissão de dados multimídia. O estudo tem como resultado o desenvolvimento de um protótipo de software constituído de dois módulos, um Módulo Servidor responsável pelo envio de sinais de áudio e um Módulo Cliente, responsável pela recepção dos sinais de áudio.

ABSTRACT

The present work is destined to the study of the transmission and audio reception in real time through Internet. It presents considerations on the architecture Internet and techniques of data transmission multimedia. The study has as result the development of a prototype of constituted software of two modules, a Server Module responsible for the sending of audio signs and a Client Module, responsible for the reception of the audio signs.

1 INTRODUÇÃO

Muitos avanços ocorreram no passado em diversas áreas, mas nenhum obteve o impacto sócio-cultural que a Internet provocou e provoca em uma era globalizada como a que tem-se hoje. Essa nova via de comunicação não foi sequer imaginada por aqueles que estudavam os veículos de comunicação por volta de 1950 ou por aqueles que se dedicaram à ficção científica. Sons, vozes e música agora constituem uma parte integrante da Internet.

Segundo Estacio (2001), a Internet evoluiu do simples envio de texto, passando pela troca de imagens e texto, ao conteúdo mais elaborado, chegando agora à transmissão de multimídia. Essa tecnologia evoluiu em função da necessidade do usuário obter informações com rapidez e também devido à deficiência dos estrangulamentos encontrados na rede.

Há algum tempo atrás, para se ouvir som pela Internet, era necessário transferir os arquivos de áudio para o computador. Uma vez que os arquivos estivessem no computador, podia-se ouví-los usando softwares de reprodução de áudio especiais e uma placa de som. Porém, como afirma Gralla (1996), o problema é que estes arquivos eram muito grandes, geralmente acima de 10MB, portanto, levavam horas para serem transferidos ao computador antes que pudessem ser reproduzidos.

Uma utilização bem melhor e mais recente de áudio na Internet chama-se *streaming* de áudio em tempo real. O áudio é tratado de uma forma bem mais inteligente, pois não é necessário que o arquivo inteiro seja carregado para então reproduzí-lo. Ao invés disso, é possível ouvir o áudio enquanto este é carregado para o computador.

A tecnologia *streaming* baseia-se em programas dedicados nas duas pontas da linha – servidor e cliente – permitindo que o áudio seja ouvido na medida em que é recebido. Ao invés de criar um arquivo no disco rígido local, cada trecho do som é transferido para a memória e reproduzido imediatamente. Para a recepção deste sinal de áudio existem hoje softwares especializados neste tipo de transmissão, como o Real Player da Real Networks e o Media Player da Microsoft.

A facilidade de criação de uma “Rádio Virtual” em termos de custo tem induzido às rádios convencionais conectarem-se à Internet e disponibilizarem sua programação, pois tem

sido um diferencial de mercado prático e barato, permitindo que usuários de Internet conectem, por exemplo, uma rádio do Japão estando em um computador no Brasil.

Este trabalho pretende mostrar como funciona a transmissão de difusão de áudio em tempo real entre computadores através de uma rede local ou da Internet. Basicamente, um cliente que está conectado a uma rede solicita a recepção de áudio para seu computador. Este cliente se conecta a um servidor de áudio, que transmite pacotes de dados em formato pré-definido através de um protocolo de comunicação. O cliente então, recebe estes pacotes e os decodifica através de um software específico, chamado tocador de mídia, para então reproduzir o áudio em seu computador.

1.1 OBJETIVOS DO TRABALHO

O objetivo do trabalho é implementar um codificador de áudio que recebe um sinal de áudio externo (de um microfone, por exemplo) e o transferirá pela Internet. Este *streaming* é recebido por um tocador de mídia, que também será implementado.

Os objetivos específicos do trabalho são:

a) capturar um sinal de áudio no servidor;
b) codificar o áudio para o formato *streaming* ;
c) transmitir o *streaming* de áudio pela Internet;
d) receber o *streaming* de áudio no cliente;
e) decodificar e reproduzir o áudio.

1.2 ESTRUTURA DO TRABALHO

Este trabalho está dividido em seis capítulos, os quais são descritos a seguir.

O primeiro capítulo apresenta a introdução e os objetivos a serem alcançados com o desenvolvimento do trabalho.

O segundo capítulo descreve a arquitetura Internet: conceitos, características, modelos e protocolos.

O terceiro capítulo trata de transmissão multimídia na Internet, apresentando aplicações, modos de transmissão de áudio e conceitos sobre codificação e compressão de áudio.

O quarto capítulo trata do desenvolvimento do protótipo, demonstrando as ferramentas utilizadas, especificação e implementação do mesmo.

O quinto capítulo apresenta trabalhos correlatos ao desenvolvimento da pesquisa.

O sexto capítulo finaliza o trabalho, apresentando as conclusões, limitações e sugestões para desenvolvimento de trabalhos futuros.

2 INTERNET

Redes de computadores existem sob diversas formas há muitos anos. Uma das características mais importantes da Internet é que ela é uma “rede de redes”. Isto é, a Internet não conecta apenas computadores isolados, mas redes inteiras de computadores. Essa é a origem do nome: Internet = *inter-network*, “entre redes”. Tecnicamente falando, portanto, a Internet não é uma rede, e sim uma associação de redes que trocam informações seguindo um único padrão Damski (1995).

Para Cyclades (2000) a Internet é um conjunto de redes de computadores interligadas pelo mundo inteiro, que têm em comum um conjunto de protocolos e serviços. Tanto a administração quanto a operação da Internet são descentralizadas, com exceção de algumas tarefas, como a coordenação de pesquisas, estabelecimento de padrões para o funcionamento da rede e a distribuição de endereços e registros de domínios para integração a essa rede. Algumas das instituições responsáveis por estas tarefas são a Internet Society (ISOC), que por meio de debates e publicações procura orientar a pesquisa e utilização da Internet; o Internet Engineering Task Force (IETF), que é um grupo de pesquisadores e técnicos responsáveis pelo desenvolvimento de padrões para o funcionamento da Internet; e o Internet Assigned Numbers Authority (IANA), que é responsável pelo gerenciamento de todos os registros da Internet, de forma direta ou via terceiros. Desses grupos surgem os documentos conhecidos como Request For Comments (RFCs), que apesar de terem sido criados apenas como propostas para padronização, na prática tornaram-se padrões oficiais da Internet.

No Brasil, a instância máxima consultiva é o Comitê Gestor da Internet do Brasil, criado em junho de 1995 por iniciativa dos Ministérios das Comunicações e da Ciência e Tecnologia. A Fundação de Amparo à Pesquisa do Estado de São Paulo (Fapesp) é responsável pelos registros de domínio e endereços de redes cadastradas no Brasil.

2.1 HISTÓRICO DA INTERNET

A Internet surgiu de um projeto da agência norte-americana Advanced Research Projects Agency (ARPA) com o objetivo de conectar os computadores dos seus departamentos de pesquisa. Esta conexão iniciou-se efetivamente em 1969, entre quatro

localidades (Universidades da Califórnia de Los Angeles e de Santa Bárbara, Universidade de Utah e Instituto de Pesquisa de Stanford), e passou a ser conhecida como Arpanet.

Esse projeto inicial foi colocado à disposição de pesquisadores, o que resultou em uma intensa atividade de pesquisa durante a década de 70, cujo principal resultado foi a concepção do conjunto de protocolos que até hoje é a base da Internet, conhecida como TCP/IP. No início da década de 80, a Arpa iniciou a integração das redes de computadores dos outros centros de pesquisa à Arpanet. Em 1985, a entidade americana National Science Foundation (NSF) interligou os supercomputadores de seus centros de pesquisa, o que resultou na rede conhecida como NSFNET, que em 1986 foi conectada à Arpanet. O conjunto de todos os computadores e redes ligados a esses dois *backbones*[1] passou a ser conhecido oficialmente como Internet.

Em 1990, o backbone Arpanet foi desativado, criando-se em seu lugar o backbone Defense Research Internet (DRI); entre 1991 e 1992, a Advanced Network and Services (ANS) desenvolveu um novo *backbone*, conhecido como ANSNET, que passou a ser o backbone principal da Internet.

A partir de 1993, a Internet deixou de ser uma instituição de natureza apenas acadêmica e passou a ser explorada comercialmente, tanto para a construção de novos backbones como para o fornecimento de serviços diversos em nível mundial.

2.2 A ARQUITETURA INTERNET

A grande característica da arquitetura Internet é a simplicidade de implementação dos seus protocolos, que mesmo assim atendem aos requisitos de interconexão exigidos pela maioria dos sistemas. Os padrões dessa arquitetura não são definidos por entidades internacionais de padronização, e são encontrados em documentos denominados Request For Comments (RFC), que são elaborados e distribuídos pelo Internet Activities Board (IAB).

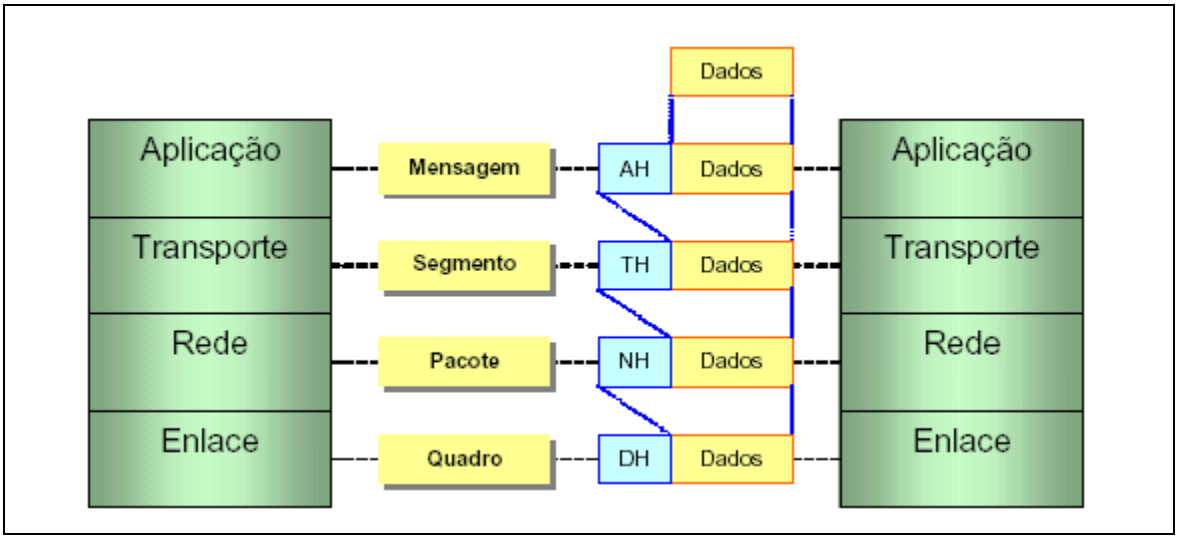
---

[1] Qualquer rede que forme parte de interligações centrais entre redes.

A Internet utiliza a arquitetura TCP/IP, criada nos anos 70, resultado das pesquisas na antiga rede Arpanet. A arquitetura TCP/IP é um conjunto de protocolos usados em redes de computadores. TCP e IP são dois protocolos dessa família e por serem os mais conhecidos, tornou-se comum usar o termo TCP/IP para se referir à família inteira. O Transmission Control Protocol (TCP) é o protocolo da camada de transporte orientado à conexão, que oferece um serviço confiável. Freqüentemente aparece como parte da arquitetura TCP/IP da Internet, mas é um protocolo de propósito geral que pode ser adaptado para ser usado com uma variedade de sistemas. O Internet Protocol (IP) é um protocolo para comunicação de redes Internet. Ele é o responsável pela transmissão de nível *host-to-host*, e é utilizado em dois tipos de estações: *hosts* e *gateways*.

Segundo (CHI1999), a arquitetura TCP/IP, que não especifica a camada física de transmissão de dados, pode ser definida em quatro camadas principais, como apresentado na Figura 1.

FIGURA 1 – DEFINIÇÃO DA ARQUITETURA TCP/IP



Fonte: Péricas (2001)

2.2.1 CAMADA DE ENLACE

Tem como objetivo permitir que pacotes enviados pela camada de rede sejam transportados entre dois computadores. Esta camada pode variar entre computadores interligados em uma mesma rede. Para cada uma dessas possíveis redes existe uma entidade de protocolo associada, com as seguintes funções:

a) propiciar a tradução dos endereços usados na camada de rede para os da tecnologia de rede associada;
b) encapsular os pacotes oriundos da camada de rede no formato da tecnologia de rede associada;
c) extrair os pacotes dos quadros recebidos da rede e encaminhá-los à camada de rede;
d) gerenciar a interface física com a rede.

2.2.2 CAMADA DE REDE

A camada de rede é responsável pelo roteamento. Esta camada é usada para atribuir endereço de rede (IP) ao sistema e rotear a informação para a rede correta. Tem ainda a função de ligação entre as camadas superiores e os protocolos de hardware. Sem essa camada, as aplicações teriam que ser desenvolvidas para cada tipo de arquitetura de rede, como por exemplo Token Ring ou Ethernet. Suas principais funções são:

a) definir para que computador (e por meio de qual interface de rede, no caso de computadores ligados a mais de uma rede) as informações oriundas da camada de transporte deverão ser encaminhadas;
b) definir o destino das informações recebidas por intermédio das interfaces de rede, que podem ser: o reenvio para outra interface, o envio para uma das entidades da camada de transporte, ou simplesmente o seu descarte;
c) agrupar (ou reagrupar) essas informações em unidades de transmissão conhecidas como datagramas, de tamanho compatível com a tecnologia de rede que será usada para transmissão;
d) providenciar a sinalização para a camada de transporte, de condições de erro detectadas no processo de transmissão dos datagramas.

Essa camada possui um protocolo central e alguns protocolos auxiliares:

a) **Internet Protocol (IP):** protocolo central da camada, é responsável pela maioria das funções enumeradas, oferecendo um serviço sem conexão para os protocolos da camada de transporte;
b) **Internet Control Message Protocol (ICMP):** protocolo auxiliar ao protocolo IP, criado para a sinalização de condições de erro;

c) **Internet Group Management Protocol (IGMP):** protocolo auxiliar para o gerenciamento de grupos de multi-difusão.

2.2.3 CAMADA DE TRANSPORTE

Esta camada reúne os protocolos que realizam as funções de transporte de dados fim a fim, e ainda controla o fluxo de dados e efetua processos de verificação e correção de erros. Essa camada possui dois protocolos:

a) **Transmission Control Protocol (TCP):** garante que os dados serão enviados com sucesso, pois realiza transmissões orientadas à conexão, garantindo a entrega dos dados;
b) **User Datagram Protocol (UDP):** transmite os dados de forma sem conexão, não garantindo a entrega dos dados. Esse protocolo substitui o protocolo TCP quando a transferência de dados não precisa estar submetida a serviços como controle de fluxo e congestionamento.

2.2.3.1 O PROTOCOLO TCP

O objetivo do TCP é oferecer aos seus usuários um serviço de transferência confiável de dados, implementando serviços de recuperação de dados perdidos, danificados ou recebidos fora de seqüência e minimizando o atraso de trânsito para transmissão de dados, como citado por Carvalho (1994).

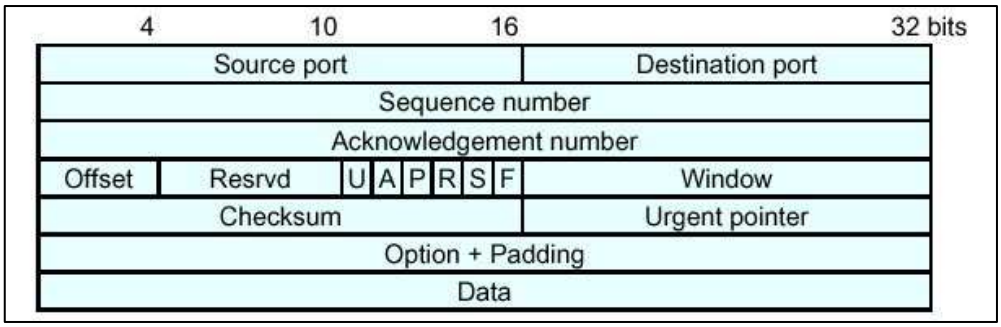
Do ponto de vista de um programa aplicativo, o serviço oferecido pelo TCP/IP tem sete características importantes:

a) orientação à conexão: para haver transmissão de dados, é necessário que primeiro um aplicativo solicite conexão com um destino, para então utilizar a conexão para a transferência dos dados;
b) comunicação ponto a ponto: cada conexão de TCP tem exatamente duas extremidades;
c) confiabilidade completa: o TCP garante que os dados enviados através de uma conexão serão entregues exatamente como foram enviados;

d) comunicação full-duplex: uma conexão de TCP permite que os dados fluam em qualquer direção e permite que um ou outro programa aplicativo envie dados assincronamente;
e) interface de dados: um aplicativo envia uma seqüência contínua de octetos através de uma conexão, ou seja, o TCP não fornece a noção de registros e não garante que os dados serão entregues ao aplicativo receptor em pedaços do mesmo tamanho que foram transferidos pelo aplicativo remetente;
f) partida de conexão confiável: quando dois aplicativos criam uma conexão, o TCP exige que estes concordem com a nova conexão;
g) desligamento de conexão graciosa: um aplicativo pode abrir uma conexão, enviar quantias arbitrárias de dados e então requisitar o fechamento da conexão. O TCP garante a entrega confiável de todos os dados antes de fechar a conexão.

O formato do segmento TCP, segundo Commer (2001), está representado na Figura 2.

FIGURA 2 – FORMATO DO SEGMENTO TCP



Fonte: Commer (2001)

Quando um computador envia um segmento, os campos ACKNOWLEDGMENT NUMBER e WINDOW se referem aos dados que chegam: o ACKNOWLEDGMENT NUMBER especifica o número de seqüência dos dados que foram recebidos e o WINDOW especifica quanto espaço de buffer adicional está disponível para mais dados. O campo SEQUENCE NUMBER se refere aos dados sendo enviados, fornecendo o número de seqüência para os dados sendo transportados no segmento. O receptor usa o número de seqüência para reordenar segmentos que chegam fora de ordem e para computar um número de confirmação. O campo DESTINATION PORT identifica que o programa aplicativo no computador receptor que deve receber os dados, enquanto que o campo SOURCE PORT

identifica o programa aplicativo que enviou os dados. Finalmente, o campo CHECKSUM contém a soma de verificação que cobre o cabeçalho de segmento TCP e os dados.

2.2.3.2 O PROTOCOLO UDP

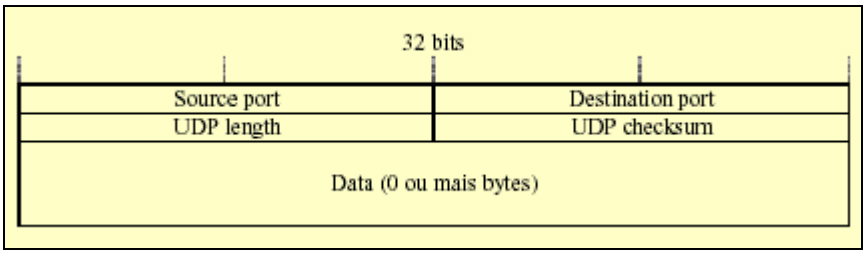
Existem situações em que o dispositivo de origem não precisa da garantia da entrega de dados ao dispositivo de destino. Nestes casos, o TCP é substituído pelo User Datagram Protocol (UDP), que é um protocolo sem conexão, ou seja, não necessita estabelecer uma conexão entre origem e destino antes de enviar os dados. Este protocolo não verifica nem se o dispositivo está *on-line*.

O UDP fornece um serviço de transmissão sem conexão, não-confiável, usando IP para transportar mensagens entre máquinas. Usa o IP para transportar mensagens, porém acrescenta a habilidade de distinguir entre múltiplos destinos em um mesmo host, segundo Commer (1997). Um programa aplicativo que usa UDP assume a responsabilidade de lidar com o problema de confiabilidade, inclusive perda de mensagem, duplicação, retardo, transmissão defeituosa e perda de conectividade.

Cada mensagem UDP é chamada de datagrama do usuário e consiste de duas partes: um cabeçalho e uma área de dados. A estrutura do cabeçalho consiste em quatro campos referentes à porta que originou a mensagem, à porta de destino da mesma, ao tamanho e à soma de verificação.

A Figura 3 ilustra o segmento UDP:

FIGURA 3 – SEGMENTO UDP



Fonte: Carvalho (1994)

Os campos *source port* (porta de origem) e *destination port* (porta de destino) identificam os números das portas de serviço do UDP utilizadas em uma comunicação. O campo porta de origem é opcional, quando usado, especifica a porta para a qual as respostas devem ser enviadas; se não utilizado, o campo deve estar zerado. O campo *UDP length* (tamanho da mensagem) contém o número total de octetos do datagrama UDP, incluindo o cabeçalho e os dados do usuário. O campo *checksum* (soma de verificação) é opcional, não precisa ser utilizado em todos os datagramas; se houver um valor zero neste campo, indica que ele não deve ser tratado. Porém, como citado por Carvalho (1994), no caso do IP o *checksum* é calculado somente sobre o cabeçalho de IP, e se não for utilizado o *checksum* do UDP, nenhuma verificação é realizada sobre os dados do usuário.

2.2.4 CAMADA DE APLICAÇÃO

A camada de aplicação é responsável por permitir que aplicações possam se comunicar através de hardware e software de diferentes sistemas operacionais e plataformas, no processo chamado de cliente-servidor. A aplicação cliente geralmente está em um equipamento mais simples, a qual envia requisições à aplicação servidora, que tem capacidade para atender várias requisições diferentes de clientes diferentes simultaneamente.

As aplicações são desenvolvidas utilizando-se algum tipo de protocolo de aplicação, dependendo do seu propósito. Alguns exemplos de protocolos de aplicação:

a) *Simple Mail Transfer Protocol* (SMTP): utilizado para envio e recebimento de mensagens de correio eletrônico. É definido nos RFC 821 e 822;
b) *File Transfer Protocol* (FTP): permite a transferência de arquivos entre um computador local e um servidor remoto, sendo bastante utilizado para envio e recebimento de arquivos pela Internet. O protocolo FTP é definido no RFC 959;
c) *Simple Network Management Protocol* (SNMP): este protocolo permite coletar dados sobre o status dos dispositivos de rede permitindo ao administrador da rede monitorar o funcionamento do sistema. É definido nos RFC 1155 a 1158 e 1213;
d) *Hyper Text Transfer Protocol* (HTTP): é o protocolo utilizado pela *World Wide Web* (WWW) para transmitir páginas da Internet. Este protocolo é definido no RFC 1945.

3 MULTIMÍDIA NA INTERNET

Redes multimídia são redes destinadas a transmitir sinais multimídia onde interoperam ambientes e aplicações heterogêneas, em diferentes plataformas, permitindo aos usuários compartilharem áudio, vídeo e voz.

Segundo Ross (2001), nos últimos anos ocorreu um explosivo avanço no desenvolvimento de aplicações de rede com o intuito de transmitir áudio e vídeo pela Internet. Novas aplicações, conhecidas como aplicações de transmissão de conteúdo multimídia, como por exemplo, telefonia IP, rádio via Internet, teleconferências, mundos virtuais, sites multimídia, ensino à distância, são anunciadas diariamente.

Porém, estes serviços diferem significativamente das aplicações orientadas à dados, como a Web (texto e imagem), correio eletrônico, transferência de arquivos. Particularmente, aplicações multimídia são muito sensíveis a atrasos na transmissão, variações que podem ocorrer nesses atrasos e ocasionais perdas de dados. Essas diferenças determinam que a rede orientada à transmissão de dados não é adequada para as aplicações multimídia. Assim, busca-se alternativas para que a arquitetura Internet possa oferecer o suporte exigido por estas aplicações multimídia.

3.1 APLICAÇÕES MULTIMÍDIA DE REDE

Segundo Fluckiger (1995), existem dois tipos de aplicações multimídia na rede, aquelas aplicações unidirecionais, ou seja, somente um dos sistemas transmite dados, enquanto o outro somente recebe, e aplicações bidirecionais, onde os dois sistemas transmitem dados de um para outro. Outro tipo de aplicação multimídia consiste em uma comunicação ponto a ponto, ou seja, dois sistemas participam de uma sessão, diferente de uma comunicação ponto a multiponto, onde múltiplos sistemas são envolvidos.

Dois conceitos de aplicações multimídia são conhecidos hoje, um chamado de difusão e outro chamado de multi-difusão. Abaixo, algumas características de cada um desses conceitos:

f) na transmissão por difusão, a informação é transmitida de um ponto a todos os pontos que possam receber essa informação, como o sinal de TV ou de rádio;

g) na transmissão por multi-difusão, a informação é transmitida de um ponto a somente um conjunto de destinatários que possam receber esse sinal. Essa técnica pode ser implementada de duas maneiras: multi-difusão para grupos fechados, ou seja, somente um grupo pré-definido de destinatários pode receber informação. Este grupo é definido pelo sistema que envia os dados; e multi-difusão para grupos abertos, onde qualquer sistema que possa receber esses dados conecta-se ou desconecta-se a qualquer momento.

Para o perfeito funcionamento de aplicações multimídia de rede, duas características são importantes: a consideração de tempo e a tolerância à perda dos dados. As aplicações multimídia são extremamente sensíveis a atrasos na transmissão, refletindo diretamente na recepção destes dados no cliente (falhas no áudio, espaços “em branco” dentro de uma música, por exemplo). Quanto à tolerância à perda dos dados, estas perdas podem ocasionar pequenas falhas na reprodução do áudio/vídeo, porém muitas vezes pode-se ocultar total ou parcialmente essas falhas. Segundo Ross (2001), para estas aplicações longos atrasos tornam-se inoportunos, mas não causam danos aos dados, e a integridade na transferência dos dados é de extrema importância.

Existem hoje três categorias de transmissão de áudio pela Internet, as quais serão vistas a seguir.

3.1.1 STREAMING DE ÁUDIO ARMAZENADO

Nesta classe de aplicações, clientes requisitam arquivos de áudio armazenados em um servidor, podendo estes clientes interromper temporariamente, avançar ou retroceder a recepção de áudio. Estas aplicações podem ser subdivididas em duas classes:

a) *Stored Media* (mídia armazenada): os dados multimídia são pré-gravados e armazenados em um servidor. Como resultado, o usuário pode interromper, retroceder, avançar ou navegar dentro do conteúdo multimídia;
b) *Streaming:* Na maioria das aplicações de áudio armazenado, o cliente reproduz o áudio poucos segundos após começar a receber o arquivo do servidor. Este recurso permite que o cliente ouça o trecho de áudio que recebeu enquanto recebe outros trechos de áudio do servidor. Esta técnica evita que o arquivo de áudio seja recebido por completo no cliente para depois ser reproduzido. Existem muitas

aplicações *streaming* hoje no mercado, como por exemplo, o Real Player da Real Networks e o Windows Media Player, da Microsoft.

3.1.2 STREAMING DE ÁUDIO EM TEMPO REAL

Este tipo de aplicação é similar às tradicionais transmissões de rádio, com a diferença de que estas transmissões são feitas via Internet. Neste caso, o cliente recebe o áudio transmitido ao vivo de um servidor localizado em qualquer lugar do mundo.

Como o áudio não é armazenado, o cliente não pode realizar funções como retroceder ou avançar, porém, vários clientes podem estar conectados a um mesmo servidor recebendo o mesmo sinal de áudio. Esta recepção pode ocorrer de forma ponto a ponto, ou seja, um cliente faz uma conexão ponto a ponto com o servidor, assim cada cliente tem sua própria conexão com o servidor, ou de forma de multi-difusão, onde todos os clientes conectados a um servidor compartilham o mesmo áudio transmitido.

3.1.3 STREAMING DE ÁUDIO INTERATIVO EM TEMPO REAL

Aqui os clientes podem utilizar o áudio para se comunicarem em tempo real. Exemplos de aplicações que utilizam áudio interativo em tempo real são os softwares de telefonia via Internet e de teleconferências.

3.2 TRANSMISSÃO DE ÁUDIO EM TEMPO REAL PELA INTERNET

Segundo Liu (2000), transmitir áudio pela Internet não é uma tarefa fácil. Há três dificuldades principais:

a) aplicações multimídia requerem muita largura de banda. O hardware disponível hoje não oferece tal disponibilidade de banda a baixo custo;
b) grande parte das aplicações multimídia ocorrem em tempo real, e com os congestionamentos de rede e/ou perda de dados na transmissão, acaba prejudicando a reprodução de áudio no cliente, ocasionando falhas ou até a não reprodução deste áudio;
c) arquivos multimídia são geralmente grandes. Somente o aumento da largura da banda não resolveria o problema, pois há a limitação no tamanho do buffer no

computador do cliente. Portanto, se os dados forem enviados em grande quantidade, o buffer irá estourar, ocasionando perda de pacotes e baixa qualidade na reprodução. Se os dados forem enviados em pouca quantidade, a reprodução não acontecerá.

Como o aumento da largura de banda não solucionaria o problema da transmissão de dados multimídia em tempo real, e com milhões de usuários conectados a uma rede que possui largura de banda limitada, os atrasos na transmissão são imprevisíveis. Segundo Liu (2000), para resolver estes problemas é necessário que se implemente uma transmissão multimídia de qualidade aceitável.

A utilização de protocolos de tempo real para a transmissão de dados multimídia torna-se um fator necessário no momento em que a Internet utiliza cada vez mais esse tipo de transmissão.

3.3 CODIFICAÇÃO E COMPRESSÃO DE ÁUDIO

Segundo Ross (2001), para haver transmissão de áudio através da Internet, este sinal deve ser codificado e comprimido. A necessidade de codificação deve-se ao fato de que computadores transmitem bits, então o áudio a ser transmitido deve estar representado na forma de bits, e por sua vez a compressão é importante porque dados de áudio não comprimidos requerem muito espaço para armazenamento e largura de banda; na digitalização do áudio são removidas as redundâncias e o tamanho dos arquivos tende a reduzir.

Um sinal de áudio analógico é digitalizado seguindo as seguintes etapas:

a) o sinal analógico de áudio é inicialmente amostrado a uma taxa pré-determinada, como por exemplo 44.000 amostras por segundo;
b) cada uma das amostras é arredondada para um número finito de valores. Esta operação é chamada de quantização. O número finito de valores, chamado de valor de quantização, é tipicamente uma potência de 2, como por exemplo quantização de 256 valores;
c) cada um dos valores de quantização é representado por um número fixo de bits. Por exemplo, se temos 256 valores de quantização, então cada valor é representado

por 1 byte. Cada amostra é convertida para esta representação, e o conjunto dessas amostras forma a representação digital do sinal.

A técnica de compressão mais conhecia e utilizada atualmente é a **MPEG Layer 3**, popularmente conhecida como MP3. Esta técnica fornece qualidade de CD ao sinal de áudio, e a degradação do áudio na conversão é muito pequena. Outro aspecto do arquivo MP3 é que quando particionado em pedaços menores possibilita a reprodução de cada um desses pedaços. Este formato de arquivo permite que os arquivos MP3 de música possam ser transmitidos em tempo real através da Internet.

4 TRABALHOS CORRELATOS

A tecnologia de transmissão de áudio em tempo real pela Internet é relativamente nova, porém podem ser encontrados trabalhos tanto de caráter científico quanto comercial relacionados a este tema. Através de pesquisas na Internet pode-se encontrar trabalhos que implementam sistemas de transmissão de áudio em tempo real, alguns deles realizados por profissionais na área de Internet assim como por universitários com o objetivo de adquirir conhecimento sobre esta nova área da Internet.

A seguir podem ser citados alguns destes trabalhos:

a) Projeto LANBRETAS (*LAN Based Real Time Audio Systems*): projeto que envolve a pesquisa e desenvolvimento de aplicações de Correios Eletrônicos Multimídia, sistemas de Teleconferência. Informações disponíveis em <http://www.inf.puc-rio.br/telemidia/www/fsm2_lan.html>.

b) AREAL (*Áudio Rendering Engine and Library*): biblioteca com um conjunto de ferramentas para desenvolvimento de aplicações de transmissão de áudio em tempo real para computadores com processador PENTIUM rodando o sistema operacional Windows, disponível somente na linguagem C/C++. Disponível em <http://www.shout.net/~mhamman/papers/icmc98.htm>.

Comercialmente, as tecnologias de áudio mais conhecidas são as disponibilizadas pela Microsoft, com o Windows Media Player (http://www.microsoft.com) e pela Real Networks, através do RealPlayer (http://www.real.com).

5 DESENVOLVIMENTO DO PROTÓTIPO DE SOFTWARE

Para o desenvolvimento do protótipo de software, foi necessário a criação de um sistema que possibilite a captura e transmissão de sinais de áudio através da Internet. Para isto, observou-se a necessidade de criar um protótipo de software como servidor, que enviará dados de acordo com as requisições dos clientes, e um protótipo de software como cliente, que faz uma requisição ao servidor, estabelece uma conexão e trata o áudio recebido.

5.1 REQUISITOS PRINCIPAIS DO PROTÓTIPO DE SOFTWARE

O protótipo de software é composto por dois sub-sistemas, um responsável pelo envio do sinal de áudio através da Internet e outro sistema que irá conectar-se ao primeiro, recebendo o sinal de áudio à medida que o áudio é enviado. O sistema responsável pelo envio é denominado Módulo Servidor e o sistema que receberá o áudio é denominado Módulo Cliente.

O Módulo Servidor possui os seguintes requisitos:

a) tratar as requisições do cliente, como solicitação de conexão, recepção de áudio, encerramento de recepção de áudio;
b) aceitar ou não a solicitação do cliente para a recepção do áudio;
c) transmitir o sinal de áudio para o cliente. O servidor irá capturar o áudio, gerando *streams* de áudio com tamanho pré-definido, enviando estes através da Internet.

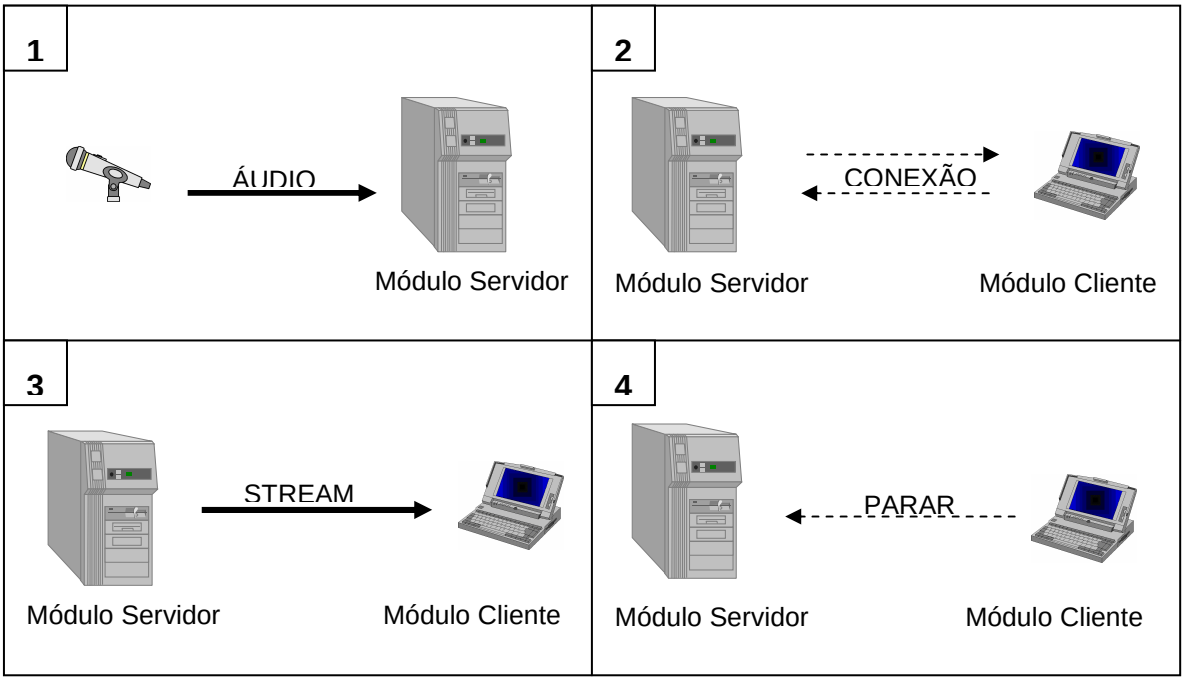
O Módulo Cliente possui os seguintes requisitos:

a) requisitar uma conexão com um servidor de áudio para a recepção do sinal. O computador deverá estar conectado a uma rede local, ou à Internet através de conexão *dial-up*, para a comunicação com o servidor;
b) reproduzir o sinal de áudio, reagrupando os *streams* de áudio enviados pelo Módulo Servidor, e também iniciar e encerrar a recepção de áudio.

5.2 ESPECIFICAÇÃO DO PROTÓTIPO DE SOFTWARE

O desenvolvimento deste protótipo de software pressupõe a implementação de dois módulos, o Módulo Servidor e o Módulo Cliente para a transmissão do sinal de áudio através da Internet. Para se ter uma idéia geral do funcionamento do protótipo de software, a Figura 4 apresenta quatro etapas principais, que vão desde a captura de sinais de áudio pelo Módulo Servidor até a recepção destes sinais de áudio pelo Módulo Cliente.

FIGURA 4 – DEMONSTRAÇÃO DO FUNCIONAMENTO DO PROTÓTIPO



O Quadro 1 descreve as etapas ilustradas na Figura 4, que vão desde a captura do áudio até a sua disponibilização no cliente.

QUADRO 1 – ETAPAS DA TRANSMISSÃO E RECEPÇÃO DE ÁUDIO

Sequência	***Descrição***
1	O servidor recebe o áudio de um microfone conectado à placa de som do computador.
2	Um cliente que deseja receber o áudio estabelece uma conexão com o servidor. Estabelecida a conexão, o servidor irá iniciar o envio do áudio para o cliente.
3	O servidor envia *streaming* de áudio para os clientes conectados.

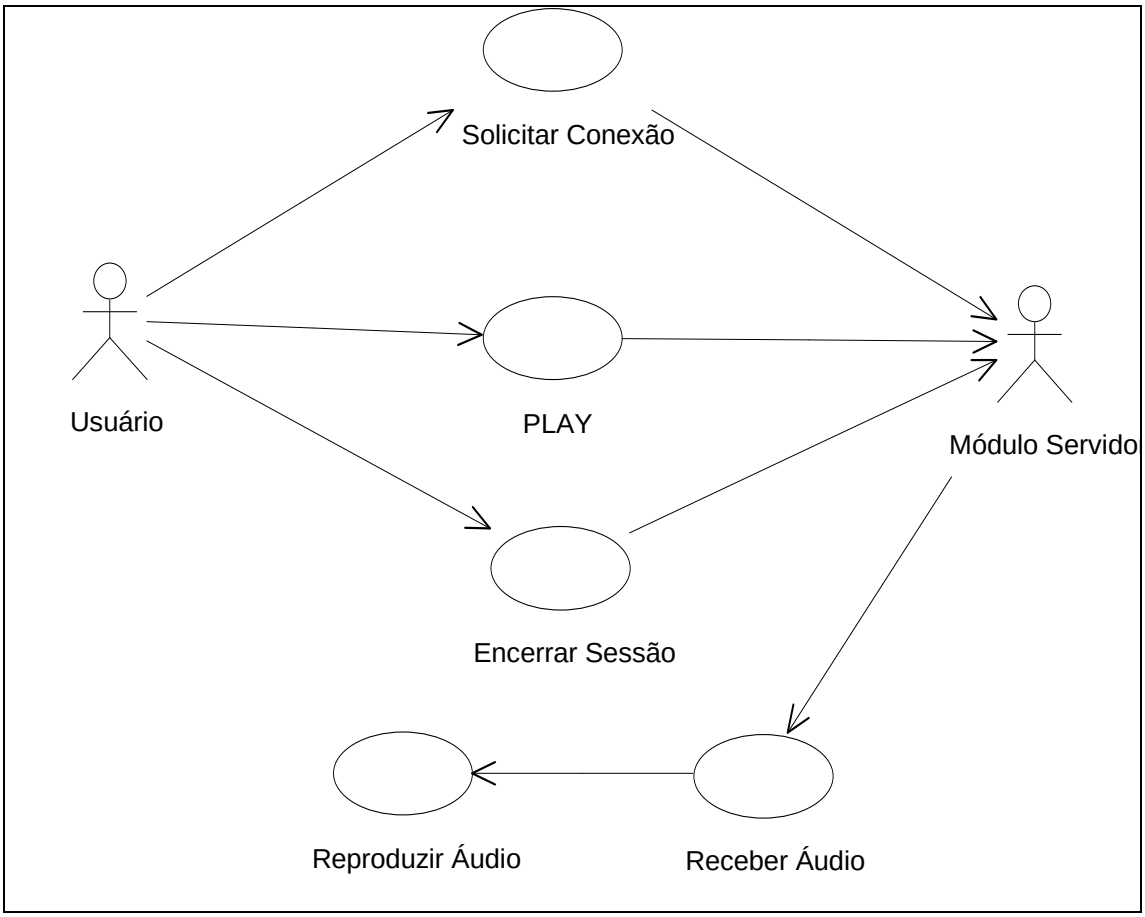
4	Quando um cliente desejar encerrar a transmissão, este envia uma solicitação de desconexão, onde o servidor encerrará a conexão e excluirá o cliente da lista de clientes.

A especificação do protótipo foi realizada utilizando o diagrama de casos de uso e o diagrama de classes do *Unifield Modeling Language* (UML).

5.2.1 DIAGRAMAS DE CASO DE USO

Os diagramas de caso de uso foram especificados utilizando-se a ferramenta Rational Rose. A Figura 5 demonstra o diagrama de caso de uso do Módulo Cliente, onde pode-se definir as funções do mesmo.

FIGURA 5 – DIAGRAMA DE CASO DE USO – MÓDULO CLIENTE



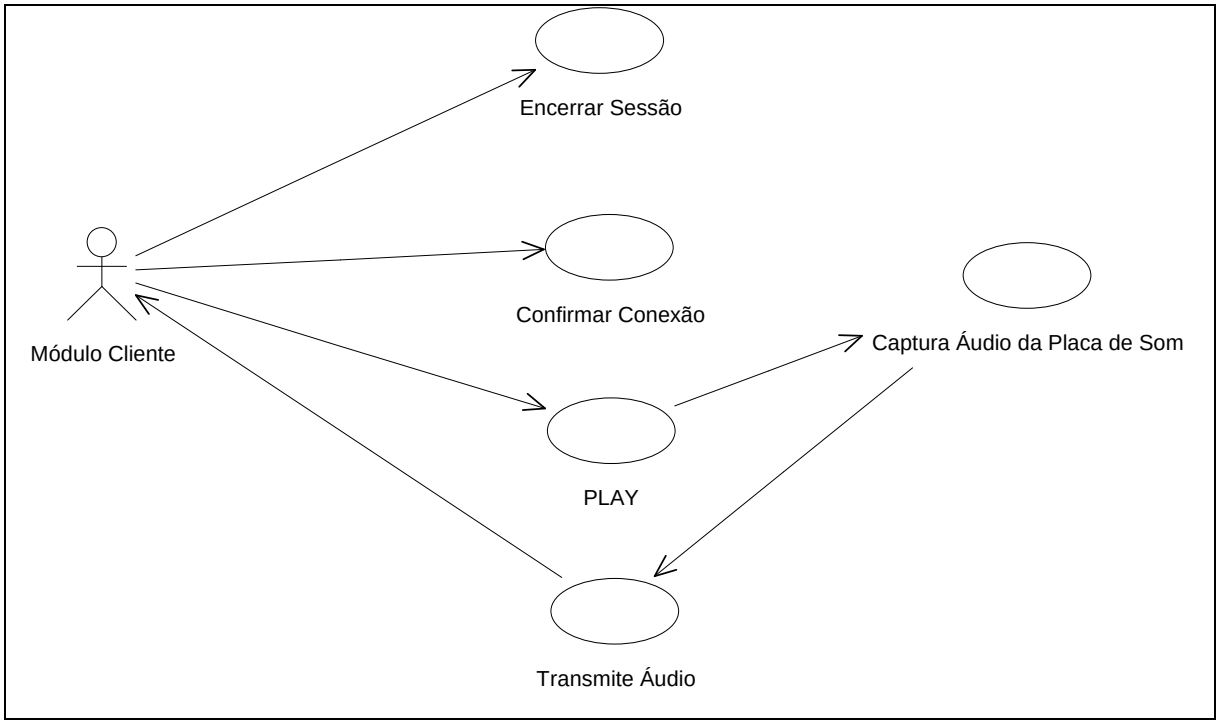
O Quadro 2 descreve os casos de uso do Módulo Cliente, indicando para cada um destes casos o nome, o ator que inicia a ação e uma breve descrição de cada caso de uso.

QUADRO 2 - DESCRIÇÃO DOS CASOS DE USO – MÓDULO CLIENTE

Caso de Uso	***Ator que inicia a ação***	***Descrição***
Solicitar Conexão	Usuário	O cliente envia uma requisição de conexão ao servidor, indicando que deseja receber áudio.
Encerrar Sessão	Usuário	O cliente pode a qualquer momento encerrar a recepção de áudio, enviando um pedido de desconexão ao servidor.
PLAY	Usuário	Ao ser efetivada a conexão, o cliente passa a receber o áudio.
Receber Áudio	Módulo Servidor	Uma vez que o cliente esteja preparado para a recepção de áudio, este começa a receber os sinais de áudio do servidor.
Reproduzir áudio	Módulo Servidor	O áudio recebido é reproduzido no cliente.

A Figura 6 mostra o diagrama de caso de uso do Módulo Servidor, que demonstra as funções do servidor na transmissão do áudio.

FIGURA 6 – DIAGRAMA DE CASO DE USO – MÓDULO SERVIDOR



O Quadro 3 descreve os casos de uso do Módulo Servidor, indicando nome, ator e descrição de cada caso.

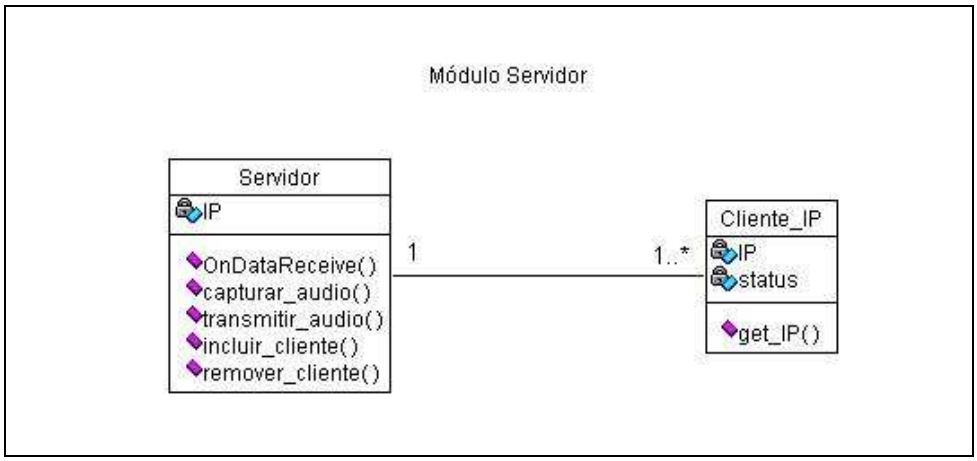
QUADRO 3 - DESCRIÇÃO DOS CASOS DE USO – MÓDULO SERVIDOR

Caso de Uso	***Ator que inicia a ação***	***Descrição***
Captura Áudio da Placa de Som	Módulo Cliente	Um microfone é conectado a uma placa de som existente no computador servidor. Este microfone irá transmitir o áudio para uma placa de som, que irá fazer a captura do áudio analógico (por exemplo, a voz humana).
Transmite Áudio	Módulo Cliente	No momento em que o áudio é capturado do microfone, ocorre a transmissão do áudio através da rede.
Encerrar Sessão	Módulo Cliente	O servidor recebe um pedido de desconexão e finaliza a conexão com este cliente, removendo o seu endereço IP da lista de destinatários.
Confirmar Conexão	Módulo Cliente	O servidor recebe um pedido de conexão, indicando que o cliente deseja receber áudio. O servidor inclui o endereço IP do cliente na lista de destinatários para que possa iniciar a transmissão de áudio.
PLAY	Módulo Cliente	O servidor recebe o comando PLAY e inicia a transmissão de áudio para o cliente.

5.2.2 DIAGRAMA DE CLASSES

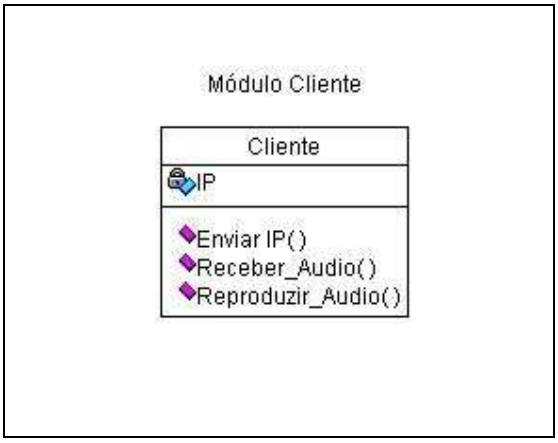
A Figura 7 mostra o diagrama de classes desenvolvido para especificar o protótipo do Módulo Servidor.

FIGURA 7 – DIAGRAMA DE CLASSES – MÓDULO SERVIDOR



A Figura 8 mostra o diagrama de classes do Módulo Cliente, desenvolvido para especificar o protótipo.

FIGURA 8 – DIAGRAMA DE CLASSES – MÓDULO CLIENTE



5.3 IMPLEMENTAÇÃO DO PROTÓTIPO DE SOFTWARE

O protótipo de software foi implementado utilizando a ferramenta de desenvolvimento Borland Delphi 5, além de componentes específicos para a captura de áudio no Módulo Servidor e para a transmissão e recepção de *streams* de áudio pela Internet. Estas ferramentas serão descritas nos tópicos a seguir.

5.3.1 BORLAND DELPHI 5

Segundo Cantu (2000), o Borland Delphi 5 ainda é a melhor combinação de programação orientada a objetos e programação visual para Windows. Sua abrangência para desenvolvimento de softwares como aplicações de banco de dados, cliente/servidor, múltiplas camadas, intranet ou Internet.

O Delphi apresenta algumas vantagens, como a programação em formulários orientada a objetos, compilador rápido, suporte a bancos de dados, integração com a programação Windows e sua tecnologia de componentes.

5.3.2 COMPONENTE TAUDIO

Para que o áudio pudesse ser extraído da placa de som, foi utilizado o componente *freeware* TAudio, disponível na Internet em <http://www.torry.net/audio.htm>. Este componente acessa os dispositivos de áudio da placa de som do computador, utilizando os serviços de áudio desta placa, como por exemplo, extrair áudio da placa de som e tocar o áudio em um alto-falante.

O componente TAudio identifica a placa de som utilizada através do método *Query*, que extrai as informações da placa de som, como por exemplo nome do produto, versão, dispositivos de controle de volume, entre outros. O Quadro 4 apresenta a implementação da captura do sinal de áudio no Módulo Servidor.

QUADRO 4 - CAPTURA DO ÁUDIO NO MÓDULO SERVIDOR

```
procedure TForm1.Audio1Record(Sender: TObject; LP, RP: Pointer;  BufferSize: Word);
var  LeftStream: TMemoryStream;
Begin
   if (LP <> nil) and (Mensagem <> 'Parar') then  begin
     LeftStream :=TMemoryStream.Create;
     LeftStream.Write(LP^,BufferSize);
     NMStrmServidor.Host:= Mensagem;
     NMStrmServidor.PostIt(LeftStream);
     LeftStream.Free;
   end;
End;
```

5.3.3 COMPONENTE TNMSTRM

O componente TNMStrm é utilizado para enviar *streams* de áudio através de uma intranet ou Internet, sendo utilizado com outro componente, o TNMStrmServer, que será visto adiante. Este componente faz parte dos componentes padrão da ferramenta Borland Delphi 5.

Seu funcionamento consiste em indicar ao componente um *host* válido e uma porta específica para receber os *streams* de áudio, podendo também ser enviado o nome do usuário que enviou os streams, para fins de identificação. Os *streams* então são enviados através do método *PostIt*, que retorna um OK se os *streams* foram recebidos com sucesso pelo cliente.

5.3.4 COMPONENTE TNMSTRMSERVER

Este componente é utilizado para receber os *streams* de áudio enviados pelo componente TNMStrm. Quando o componente recebe um *stream*, o evento *OnMSG* é chamado, e neste evento são implementadas as funções para o tratamento destes *streams* que são recebidos. O Quadro 5 apresenta o a implementação da recepção dos *streams* de áudio no Módulo Cliente.

QUADRO 5 - RECEPÇÃO DO ÁUDIO NO MÓDULO CLIENTE

```
procedure TForm1.NMStrmServClienteMSG(Sender: TComponent;  const sFrom: String; strm: TStream);
Begin
  if (bufferNum > 0) then  begin
    BufLStream[bufferNum].Clear;
    BufLStream[bufferNum].CopyFrom(strm,strm.Size);
    bufferNum:= bufferNum + 1;
    if (bufferNum > 10) then  bufferNum:= 1;
  end;
  if (playNum = 0) and (bufferNum = 4) then
  begin
    playNum:= 1;
    Memo1.Lines.Add('tocando áudio...');
    Audio1.Player.Play(BufLStream[playNum],nil,0);
    playNum:= playNum + 1;
    if (playNum > 10) then  playNum:= 0;
  end;
End;
```

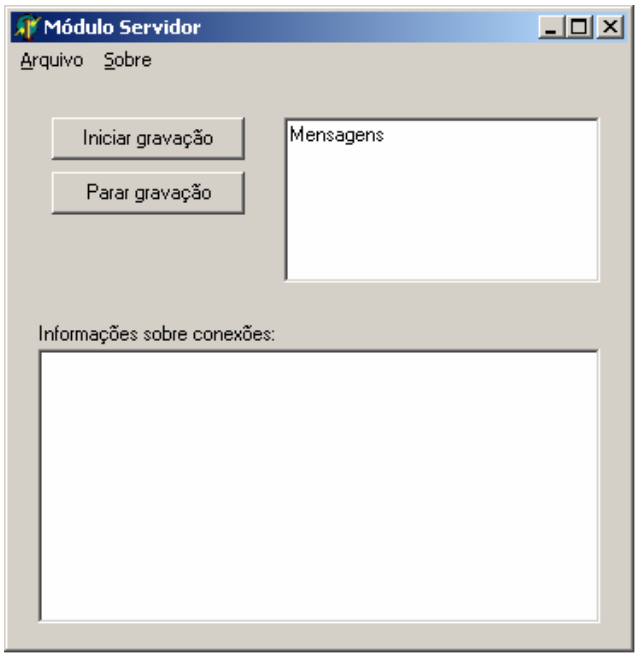
5.4 FUNCIONAMENTO DO PROTÓTIPO DE SOFTWARE

O protótipo de software possui dois módulos, o **Módulo Servidor** e o **Módulo Cliente**. O Módulo Servidor deve ser executado em um computador que esteja conectado a uma rede qualquer (uma LAN ou mesmo a Internet), e que possua uma placa de som para capturar o áudio através de um microfone, que estará conectado a essa placa de som. O Módulo Cliente deve ser executado em um computador que possa se conectar a uma rede através de uma placa de rede (em uma LAN) ou mesmo através de uma placa de fax modem (na Internet) e também uma placa de som, onde estejam conectadas caixas acústicas para que o usuário possa ouvir o áudio enviado pelo servidor. Não há a necessidade de o cliente possuir um microfone, pois a sua função é somente receber o áudio.

5.4.1 O MÓDULO SERVIDOR

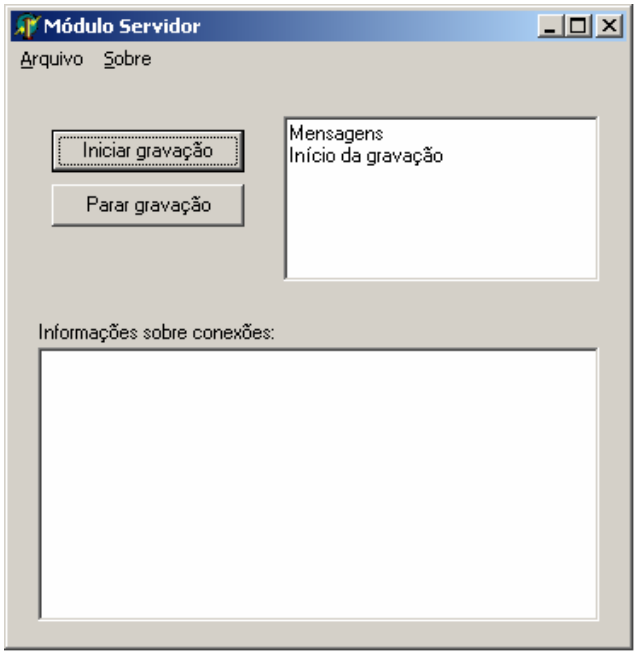
O protótipo de software servidor, neste trabalho chamado de **Módulo Servidor**, é responsável pelo envio de áudio aos clientes conectados a ele. A Figura 10 apresenta a interface do Módulo Servidor. Esta interface constitui-se de um menu, com as opções **Arquivo**, onde estão disponíveis as funções dos botões e a opção **Sobre** que mostra uma janela com informações sobre o protótipo. Existem também dois botões, **Iniciar gravação** e **Parar gravação**. Ao lado destes botões há uma caixa de mensagens, que lista dados relativos à captura do áudio. Abaixo, uma outra caixa de mensagens lista as conexões que são feitas durante a transmissão do áudio.

FIGURA 9 - TELA DO PROTÓTIPO DO SERVIDOR



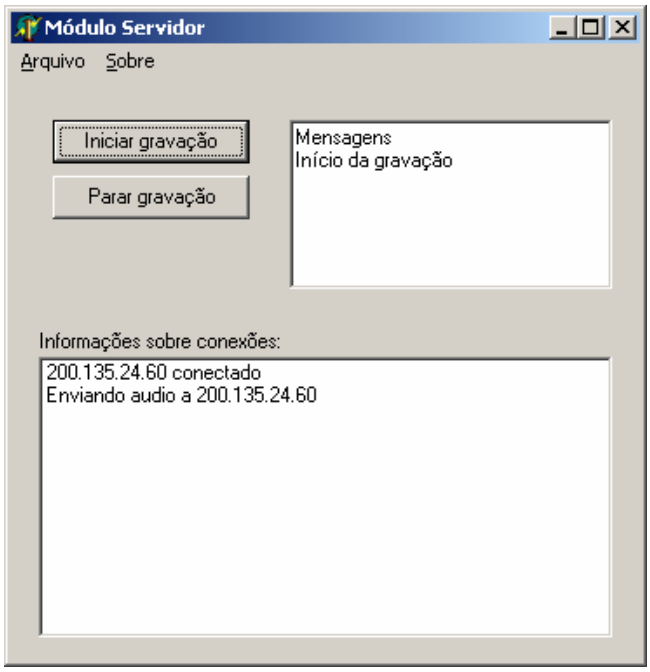
Para iniciar a captura de áudio, deve-se clicar no botão **Iniciar gravação**, onde será gerada uma mensagem na caixa de texto indicando o início da captura de áudio. Esta ação é demonstrada na Figura 11.

FIGURA 10 - INÍCIO DA CAPTURA DE ÁUDIO



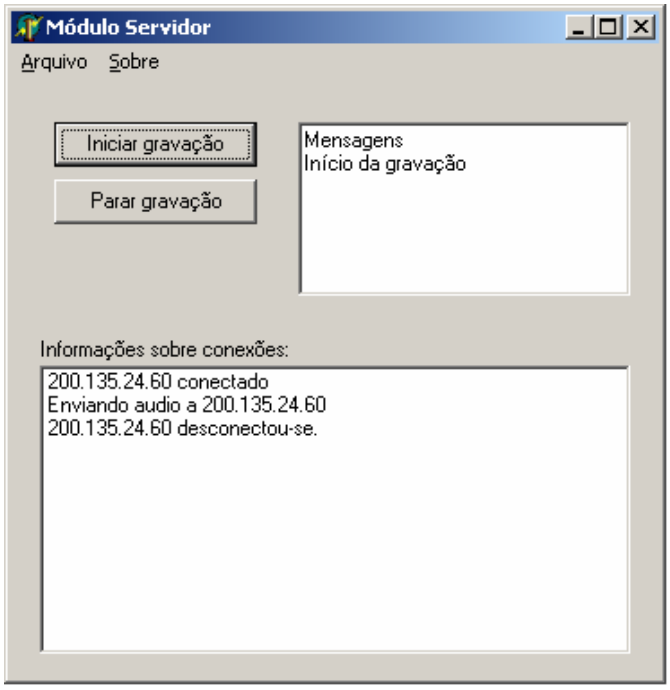
No momento em que algum cliente conectar-se ao software servidor, uma mensagem é exibida na caixa de mensagens inferior, indicando o endereço IP do cliente que solicitou recepção de áudio. Uma vez estabelecida a conexão, uma outra mensagem é gerada, identificando que o cliente iniciou a recepção de áudio. A Figura 12 apresenta esta ação.

FIGURA 11 - CONEXÃO E INÍCIO DE ENVIO DE ÁUDIO A UM CLIENTE



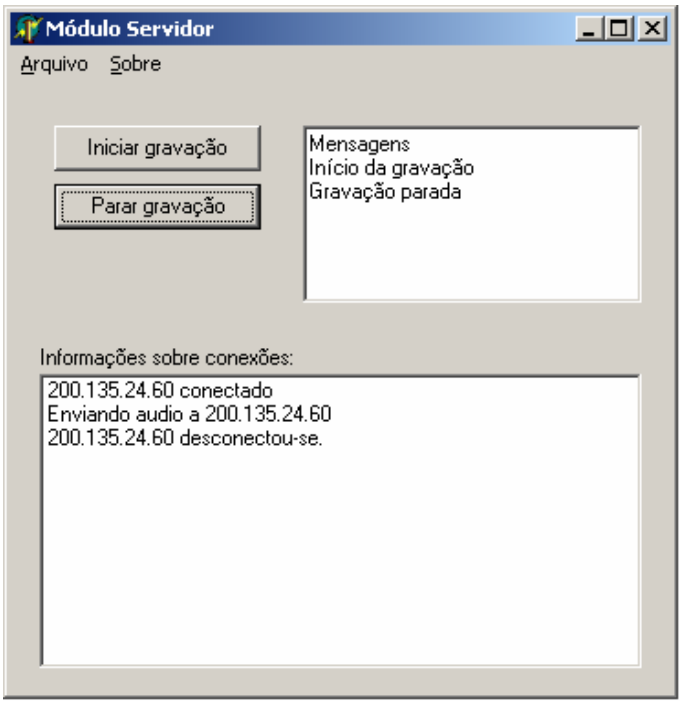
Quando um cliente desconecta-se do servidor, uma mensagem é gerada na caixa de mensagens inferior, indicando que o cliente específico encerrou a recepção de áudio. A Figura 13 demonstra esta ação.

FIGURA 12 - ENCERRAMENTO DE ENVIO DE ÁUDIO AO CLIENTE.



Por fim, quando deseja-se encerrar a captura de áudio, clica-se no botão **Parar gravação**. Neste momento, se houver clientes conectados ao servidor, estes serão automaticamente desconectados pelo servidor. A Figura 14 demonstra esta ação.

FIGURA 13 - FIM DA CAPTURA DE ÁUDIO



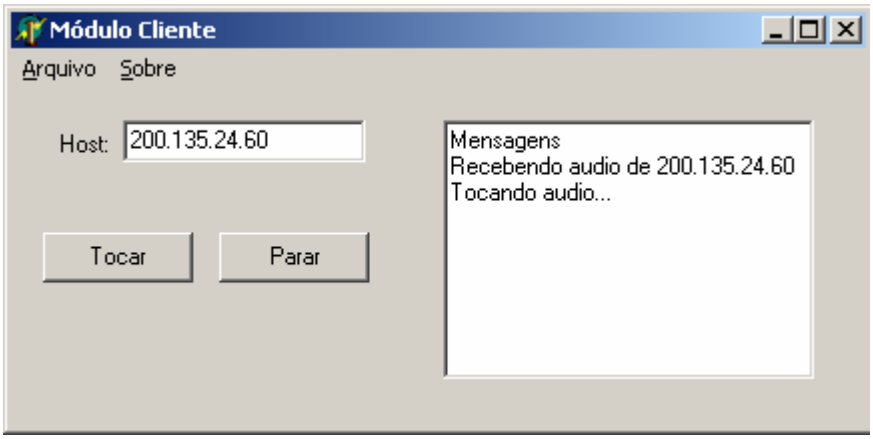
5.4.2 O MÓDULO CLIENTE

O protótipo de software cliente, chamado de **Módulo Cliente**, tem sua interface apresentada na Figura 15. A interface apresenta um menu, que possui as opções **Arquivo**, com as funções dos botões e a opção **Sair**, e uma caixa de texto, onde o usuário digita o endereço IP do computador servidor ao qual o cliente quer conectar-se. A caixa de mensagens ao lado repassa informações ao usuário referente à recepção de áudio, e os botões **Tocar** e **Parar** iniciam e encerram a recepção de áudio no software cliente.

FIGURA 14 - TELA DO PROTÓTIPO DO CLIENTE

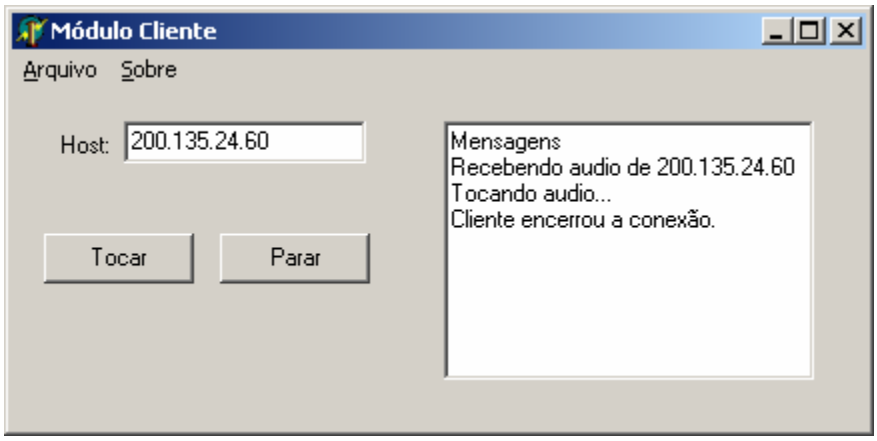
Para iniciar a recepção de áudio, o usuário clica no botão **Tocar**, o qual enviará uma solicitação de conexão ao servidor, que por sua vez enviará o áudio para este cliente. Uma mensagem é gerada na caixa de mensagens, indicando o início da recepção de áudio, como mostrado na Figura 16.

FIGURA 15 - INÍCIO DA RECEPÃO DE ÁUDIO



Quando o usuário desejar encerrar a recepção de áudio, este clica no botão **Parar**, que finalizará a conexão com o servidor e irá gerar uma mensagem na caixa de mensagens, indicando o fim da recepção, como mostrado na Figura 17. Note-se que se o servidor encerrar a transmissão de áudio para um cliente, uma mensagem também será gerada na caixa de mensagens, desta vez indicando que o servidor encerrou a transmissão.

FIGURA 16 - ENCERRAMENTO DE RECEPÇÃO PELO SOFTWARE CLIENTE



6 CONCLUSÕES

Durante o desenvolvimento do trabalho, pôde-se constatar que a transmissão de áudio em tempo real pela Internet é um assunto bastante recente, com problemas ainda a serem resolvidos, como por exemplo a estrutura da rede, ainda não preparada para atender aplicações de transmissão de informação em tempo real.

Porém, o problema mais importante constatado durante o desenvolvimento deste trabalho foi a utilização de componentes especializados baseados na arquitetura TCP, impossibilitando o uso do protocolo UDP, mais adequado para este tipo de transmissão por não exercer rígido controle sobre pacotes de dados, ficando a cargo do desenvolvedor implementar os controles necessários. A grande maioria das aplicações de transmissão de áudio multimídia utiliza o protocolo UDP na camada de transporte e um protocolo de tempo real na camada de aplicação, pois os pacotes que são enviados e porventura perdidos durante uma transmissão não são reenviados, havendo o descarte desses pacotes de dados perdidos, o que na arquitetura TCP/IP não é aceitável.A utilização de um protocolo de tempo real não foi possível devido à não existência de um conjunto de bibliotecas disponível na linguagem Borland Delphi 5 que desse suporte a esse tipo de protocolo.

Este trabalho procurou apresentar as características da transmissão de áudio em tempo real pela Internet, com o objetivo de desenvolver um aplicativo que atendesse às expectativas do trabalho.

Apesar de algumas técnicas de compressão de áudio terem sido abordadas, estas não foram utilizadas neste trabalho, por não ser este o objetivo principal deste estudo, porém, para novos estudos nesta área é importante que o item seja relevado.

O desenvolvimento deste trabalho utilizou os componentes NMStrm e NMStrmSrv que, apesar de se basearem no protocolo TCP, se mostraram bastante adequados para o envio de *streaming* pela Internet. É interessante ressaltar que, apesar do nome estar invertido, o componente NMStrm deve ser utilizado no Módulo Servidor e o componente NMStrmSrv deve ser utilizado no Módulo Cliente.

O software Rational Rose, utilizado para a especificação do protótipo, mostrou-se adequado para o uso, restringindo-se somente à especificação e modelagem do mesmo, não

sendo utilizado para a geração de código. A ferramenta Borland Delphi 5 apresentou algumas restrições, como a falta de bibliotecas específicas para a transmissão de dados através de protocolos de tempo real, o que dificultou a implementação do protótipo.

Com o desenvolvimento do protótipo de software, que gera um sinal de áudio em um servidor e o envia através da Internet a um cliente conectado em tempo real, o objetivo do trabalho foi alcançado.

6.1 LIMITAÇÕES

Como não foram utilizadas técnicas de compressão de áudio, o protótipo desenvolvido consome muita largura de banda, tornando limitada a qualidade da transmissão deste tipo de dados através da Internet.

6.2 SUGESTÕES

Procurando aprimorar os resultados obtidos com o protótipo, sugere-se:

a) implementar protocolos de tempo real para serem utilizados com a ferramenta Borland Delphi, seguindo as especificações contidas no RFC1885;
b) utilizar algoritmos de compressão, que melhorariam a performance de envio e recepção do áudio e, portanto, sua qualidade.

REFERÊNCIAS BIBLIOGRÁFICAS

(CAR1994) CARVALHO, Tereza C. Melo de Brito. **Arquitetura de redes de computadores OSI e TCP/IP.** São Paulo, Makron Books, 1994.

(CAN2000) CANTU, Marco. **Dominando o Delphi 5:** a bíblia. São Paulo: Makron Books, 2000.

(CHI1999) CHIOZZOTTO, Mauro e SILVA, Luis Antonio P. **TCP/IP tecnologia e implementação**. São Paulo: Érica, 1999.

(COM1997) COMMER, Douglas E., STEVENS, David L. **Internetworking with TCP/IP:** client-server programming and applications. New Jersey: Prentice-Hall, 1997.

(COM2001) COMMER, Douglas E. **Redes de computadores e Internet**. 2. ed. Porto Alegre: Bookman, 2001.

(CYC2000) CYCLADES BRASIL. **Guia Internet de conectividade.** 6. ed. São Paulo: Editora SENAC, 2000.

(DAM1995) DAMSKI, José Carlos; VALENTE, André. **Internet:** guia do usuário brasileiro. São Paulo: Makrom Books, 1995.

(EST2001) ESTACIO, Alessander. **Rádio, PC e TV**, [S.l], [2001?]. Disponível em: <http://www.eletroequip.com.br/alex13.asp>. Acesso em: 01 mar. 2002.

(FLU1995) FLUCKIGER, François. **Understanding networked multimedia:** applications and tecnology. Londres: Prentice Hall Inc., 1995.

(GRA1996) GRALLA, Preston. **Como funciona a Internet.** São Paulo: Quark Editora, 1996.

(LIU2000) LIU, Chunlei. **Multimedia Over IP: RSVP, RTP, RTCP, RTSP**. Ohio, 2000. Disponível em: <http://www.cis.ohio-state.edu/~jain/cis788-97/ip_multimedia/index.htm>. Acesso em: 24 abr. 2002.

(PER2001) PÉRICAS, Francisco A.. **Redes de computadores:** conceitos e arquiteturas. Blumenau, [2001?]. Disponível em: <http://www.inf.furb.br/~pericas>. Acesso em: 16 set. 2001.

(ROS2001) ROSS, Keith W. e KUROSE, James F. **Computer Neworking:** a top-down approach featuring the Internet. Boston: Addison Wesley, Inc., 2001.