UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO

(Bacharelado)

PROTÓTIPO DE UM *REGISTRAR* PARA UM SISTEMA DE TELEFONIA IP BASEADO NO PADRÃO SIP

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

MARCOS RUBIK

BLUMENAU, JUNHO/2003

2003/1-50

PROTÓTIPO DE UM *REGISTRAR* PARA UM SISTEMA DE TELEFONIA IP BASEADO NO PADRÃO SIP

MARCOS RUBIK

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. Miguel Alexandre Wisintainer

AGRADECIMENTOS

Em primeiro lugar a Deus por ter proporcionado esta oportunidade de estar aqui concluindo mais esta etapa da minha vida.

A meus pais, Mário Rubik e Inêz Batisti Rubik por todo o incentivo e compreensão em todo os momentos deste caminho.

A meu orientador Prof. Francisco Adell Péricas por toda a atenção e incentivo na orientação disponibilizada para o desenvolvimento deste trabalho.

A todos amigos que direta ou indiretamente tiveram participação nos meus estudos durante todo o tempo nesta universidade.

RESUMO

O presente trabalho apresenta o desenvolvimento um protótipo de um *registrar* para o recebimento, armazenamento, interpretação e encaminhamento de chamadas SIP. Foram estudados os componentes que fazem parte de uma rede de Telefonia IP que utilizam o padrão *Session Initiation Protocol* (SIP), como *registrar*, redirecionamento e proxy. Também apresentará considerações sobre protocolos de transmissão de áudio e protocolos de controle e inicialização de chamadas.

ABSTRACT

This work is intended for developing a *registrar* prototype to receive, store, interpret and forward SIP calls. It will be studied components that belong to an IP Telephony, which uses the Session Initiation Protocol (SIP) standard, like *registrar*, redirect and proxy. It also will be presented considerations about audio transmission protocols and call control and initiation protocols.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIAÇÕES

SS7 - *Signaling System 7*

PSTN - *Public Switched Telephony Network*

IP – *Internet Protocol*

VoIP - Voz sobre IP

SIP - *Session Initiation Protocol*

RFC - *Request For Comments*

MMUSIC - Multiparty Multimedia Session Control

IETF - *Internet Engineering Task Force*

DHCP - *Dynamic Host Configuration Protocol*

QoS - *Quality of Service*

LAN - *Local Area Network*

RTP - *Real-Time Transport Protocol*

UDP - User Datagram Protocol

ITU-T – International Telecom Union

FEC - *Forward Error Correction*

IIS - *Internet Integrated Services*

DS - *Differentiated Services*

RSVP - *Resource Reservation Protocol*

ToS - *Type of Sevice*

PCM - *Pulse Code Modulation*

TCP - Transmission Control Protocol

CRC - *Cyclic Redundance Check*

MGCP - *Media Gateway Control Protocol*

SDP - *Session Description Protocol*

CRLF - *Carriage Return, Line Feed*

SSRC - *Synchronization Source*

CSRC - *Contributing Source*

NTP - *Network Time Protocol*

SUMÁRIO

1 INTRODUÇÃO

Os serviços telefônicos estão tendo mudanças significativas a cada década. Na década de 50 a introdução de cabos transatlânticos possibilitou as chamadas internacionais; nos anos 60, as centrais e transmissões digitais melhoraram o sinal de áudio; os serviços de chamadas em espera e discagem por tons foram viabilizados na década de 70 pelas centrais programáveis; o sistema de sinalização em canal comum como o *Signaling System 7* (SS7) possibilitou serviços com números 0800. Na década de 90 marcando definitivamente a trajetória de transmissão e sinalização telefônica analógica a uma infra-estrutura baseada em redes de pacotes (FERNANDES, 2003).

Atualmente, as redes telefônicas *Public Switched Telephony Network* (PSTN), são a ampla maioria das redes telefônicas, não tendo avançado muito em relação aos equipamentos e meios de transmissão. Sua principal característica é estabelecer um circuito entre dois assinantes para que possa haver comunicação (ALENCAR, 1998).

Alencar (1998) define que as redes de computadores são redes baseadas em pacotes e tem tido um grande avanço em relação aos equipamentos de transmissão de dados, possibilitando assim usar esta mesma rede para transferência de voz. Com isto houve um crescente número de aplicações voltadas para a transferência de voz sobre os protocolos de redes, chamadas Voz sobre IP (VoIP).

Segundo Hersent (2002), VoIP é a área que mais cresce no setor de telecomunicações superando a taxa de crescimento de telefonia móvel.

A tecnologia atrás da telefonia IP pode parecer trivial, no entanto Hersent (2002) diz que é muito mais complexa do que a transmissão unidirecional usada na transmissão de TV ou de rádio nas redes de computadores, pois a taxa de transferência entre o locutor e o ouvinte deve permanecer baixa.

A redes VoIP utilizam protocolos de controle de sinalização que tem por função negociar o início de uma transmissão, fim da transmissão, codificação de áudio, localização de usuários, redirecionamento de mensagens, entre outras, que possibilitam a transmissão de voz sobre IP. Para esta função será utilizado o Protocolo de Inicialização de Sessão (*Session Initiation Protocol* – SIP) que está definido na *Request For Comments* (RFC) 3261 do grupo de trabalho *Multiparty Multimedia Session Control* (*MMUSIC*) do *Internet Engineering Task Force* (IETF).

Para tornar o SIP com funcionalidades dinâmicas utiliza-se serviços avançados para tratamento de chamadas onde pode-se implementá-los através de entidades do tipo *proxy*, servidores de redirecionamento SIP e *Registrars*.

Os *proxies* são servidores responsáveis por receber e encaminhar pedidos SIP. Pode ou não mudar os parâmetros da mensagem antes de passar adiante e também pode decidir mandar uma resposta ao cliente gerada através de funções implementadas no *proxy*.

Cabe ao servidor de redirecionamento responder a mensagens de pedido de conexão com respostas da categoria de códigos 3xx (sendo xx códigos de subcategoria que pode ser 00, 01, 02 e 80) indicando a necessidade de uma ação adicional para completar o pedido. O servidor de redirecionamento pode melhorar a escalabilidade dos serviços de distribuição de chamadas.

Para solucionar o problema de um usuário não saber a localização de outro usuário na rede, pois seu endereço pode mudar, por exemplo, devido a uma configuração na rede utilizar *Dynamic Host Configuration Protocol* (DHCP) para fornecer endereços dinamicamente ou um usuário móvel, existe uma entidade denominada *registrar*.

O *registrar* tem por finalidade armazenar e manter atualizados os endereços SIP dos usuários da rede para que possam ser localizados nos lugares que estiverem no momento da requisição. Para manter essa característica e evitar configurações manuais, está definido no SIP um endereço conhecido onde os clientes podem se registrar através de uma mensagem *REGISTER* do SIP. *Registrar* pode ser implementado em conjunção com o servidor de redirecionamento, para redirecionar chamadas para a localização atual do originador da chamada e também atuar com um servidor *proxy*, sendo este o objetivo de desenvolvimento deste trabalho.

Segundo Hersent (2002), os endereços SIP ficam registrados por no máximo uma hora onde se faz necessário que os clientes enviem mensagem de atualização periodicamente.

Para fazer o armazenamento dos endereços SIP em um *Registrar* utiliza-se um banco de dados para tornar possível a gravação, busca e atualização freqüente dos endereços armazenados de forma organizada e eficiente.

Com o crescimento das redes de telefonia IP foram surgindo muitas aplicações que visam solucionar a comunicação nas redes, ou seja, aplicações de interface com o usuário,

sendo que a falta de aplicações que sejam de auxílio as redes VoIP foi o que motivou a estudar e desenvolver este trabalho.

1.1 OBJETIVOS DO TRABALHO

Este trabalho apresenta desenvolvimento um protótipo de um *registrar* integrado a um servidor de redirecionamento e de proxy para o mapeamento de endereços de uma rede de telefonia IP, baseada no padrão SIP, para capturar os pacotes com nome de um cliente da rede, verificar em uma base de dados o endereço IP e encaminhar a mensagem ao seu destinatário utilizando mensagens SIP.

1.2 ESTRUTURA DO TRABALHO

Este trabalho está divido em forma de capítulos descritos a seguir:

Primeiro capítulo expõe na introdução uma justificativa do que originou este trabalho como também uma síntese do que será tratado no desenvolvimento do trabalho. Os objetivos a serem alcançados.

O segundo capítulo explanará a Telefonia IP suas funcionalidades bem como sua aplicação no contexto de redes de computadores.

O terceiro capítulo apresenta os conceitos, padrões e procedimentos utilizados pelo protocolo SIP de Telefonia IP.

O quarto capítulo apresenta a especificação feita para o desenvolvimento do trabalho e apresenta o protótipo com suas funcionalidades.

O quinto capítulo expõe a que conclusão chegou-se após o desenvolvimento do trabalho e algumas sugestões para continuação.

2 TELEFONIA IP

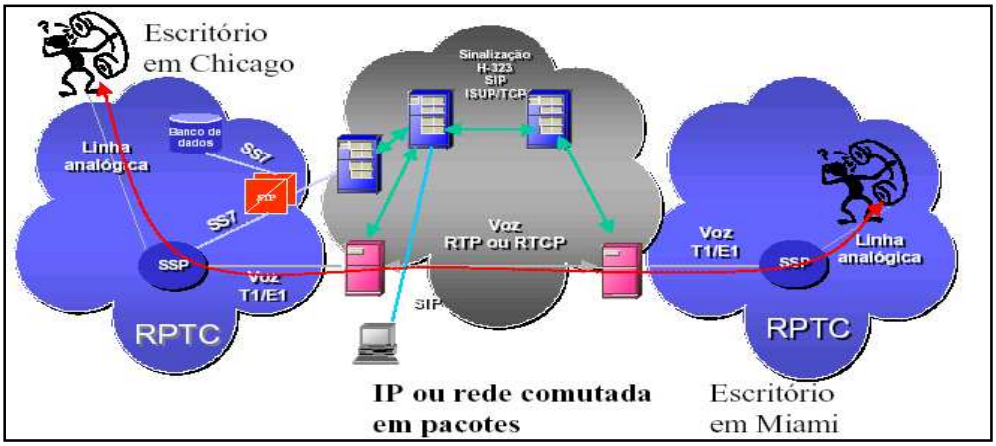
A telefonia convencional *Public Switched Telephony Network* (PSTN) é uma tecnologia antiga e eficiente, pois atinge grande parte da humanidade que possui uma base estável. Contudo as redes de comutação de pacotes, que foram projetadas inicialmente para a transmissão de dados com um grande crescimento nos equipamentos de transmissão de voz, estão cada vez mais sendo utilizadas para a transmissão de Voz sobre IP (VoIP).

A evolução da tecnologia para serviços de transmissão de voz traz consigo outras oportunidades de aplicação para a transmissão de dados.

Segundo Silva (2002), VoIP é uma tecnologia que permite a digitalização e a codificação da voz e o empacotamento dos dados em pacotes IP. Devido aos pacotes passarem por diversos domínios e *routers*, a experiência neste tipo de transmissão na internet mostrou-se ineficaz, pois os parâmetros de *Quality of Service* (QoS) exigidos para este serviço, como atraso e variação deste atraso, não podem ser assegurados. Contudo nas redes locais (LAN), que são relativamente simples, a disponibilização dos recursos torna o VoIP viável.

Com o aumento do número de novas aplicações, da disseminação de microcomputadores pessoais que suportam aplicações multimídia, da maturidade do protocolo IP, da banda de transmissão para o usuário, tornou-se possível a integração entre dados e voz em uma mesma infra-estrutura de rede, contribuindo assim para o VoIP se tornar realidade A Figura 1 mostra um exemplo de uma rede baseada em pacotes.

Figura 1 – Rede baseada em pacotes



Segundo Alves (2002), as redes de telefonia IP são redes baseadas em pacotes, onde um pacote deve ser entendido como uma unidade de dados que é enviado de um emissor a um receptor.

Em uma comunicação de dados quando um pacote não chega ou chega com falhas, ocorre o reenvio do pacote. Na comunicação de voz não existe reenvio já que a transmissão de voz se faz em tempo real, no entanto o que é problemático nas redes de telefonia IP é o atraso. Mas com o progresso tecnológico e novos métodos no tratamento de voz nas redes de dados, o processo de migração de rede de voz em uma única rede está se tornando viável, (ALVES, 2002).

Segundo Souza (2001), é um engano pensar que a telefonia IP é utilizada somente para chamadas de longa distância de baixo custo. Apesar das chamadas de longa distância estarem promovendo o uso de VoIP as companhias estão adotando-o pela facilidade de adição de novos serviços e funcionalidades assim como uma diminuição nos custos de implantação e manutenção com companhias telefônicas.

Como nas empresas existem duas redes diferentes, uma para voz e outra para dados, a possibilidade de juntar as redes fará com que haja uma grande redução de custos (SOUZA, 2001).

2.1 AMBIENTE PARA TELEFONIA IP

Segundo Fernandes (2003), para a transmissão da voz codificada em uma rede com o protocolo IP é importante considerar características como o IP, que por si só não oferece nenhuma garantia de Qualidade de Serviço, caracterizado como tráfego de melhor resultado, agregando-se na formação da solução final, outros protocolos e soluções complementares, para que o resultado seja comparável com o observado na rede de voz convencional.

Para ter uma qualidade desejada pelos usuários de telefonia IP, observam-se vários fatores para a transmissão de voz.

2.1.1 BANDA

Cada codificação de voz usa um tipo de largura de banda diferente e como vários canais de voz compartilharão o mesmo canal digital deve-se alertar ao consumo da banda que pode refletir em uma queda na qualidade de voz. Pode-se observar dois ambientes onde se têm aplicações de voz sobre IP tais como: corporativos e residenciais. O ambiente corporativo pode contar com ligações dedicadas ou à internet ou em sua própria rede onde a velocidade varia entre 64Kbps a 2Mbps. No entanto estes canais também são usados com outras aplicações.

Os ambientes residenciais normalmente são caracterizados pelo acesso discado à internet que atualmente dispõe de uma taxa de 56Kbps.

Compressão de cabeçalhos dos pacotes IP e supressão de silêncio são as técnicas mais usadas para minimizar o requisito de banda (FERNANDES, 2003).

2.1.2 COMPRESSÃO DE CABEÇALHOS DOS PACOTES IP

As aplicações de voz sobre IP utilizam *Real-Time Transport Protocol* (RTP) e o *User Datagram Protocol* (UDP) sendo que em um pacote de voz somente o cabeçalho ocupa 40 bytes e em uma transmissão utilizando a implementação de codificação de voz G.729 com um pacote formado por dois quadros de amostragem terá 20 bytes de informação transmitida. Fica evidente o despropósito na distribuição de bytes úteis e de controle. Adotando-se a técnica descrita na *Request for Commenst* (RFC) 2508 a maioria dos cabeçalhos terão seus tamanhos de 2 ou 4 bytes dependendo do uso do *checksum* pelo UDP ou não. Sendo que após a transmissão do primeiro pacote descomprimido outros pacotes subseqüentes poderão ser transmitidos suprindo seus cabeçalhos sendo remontados no destino.

2.1.3 SUPRESSÃO DE SILÊNCIO

Em uma conversa entre duas pessoas a vários períodos de tempo da chamada que não há conversa. No caso da implementação da codificação G.729 pode-se suprimir esses períodos de tempo, chegando a valores aproximadamente de 25% menores na transmissão.

2.1.4 ATRASO

O tempo que um pacote demora para chegar no destino a partir de sua geração é denominado “atraso”, que não deve ultrapassar valores que sejam aceitáveis para o tipo de transmissão em questão.

Para que se tenha uma transmissão com pequenos atrasos deve-se levar em consideração a disponibilização de recursos em que a aplicação está inserida. Aspectos como a interatividade entre usuários da aplicação devem ser considerados ao avaliar uma transmissão. O tempo entre a geração do pacote e a entrega deve estar entre 200 e 300ms. Nas aplicações onde a informação é transmitida em um único sentido, a Norma G.114 do ITU-T coloca que o intervalo é de 150 e 400ms, mas deve ser avaliado o impacto na qualidade da transmissão, sendo que atraso superior a 400ms é inaceitável (FERNANDES, 2003).

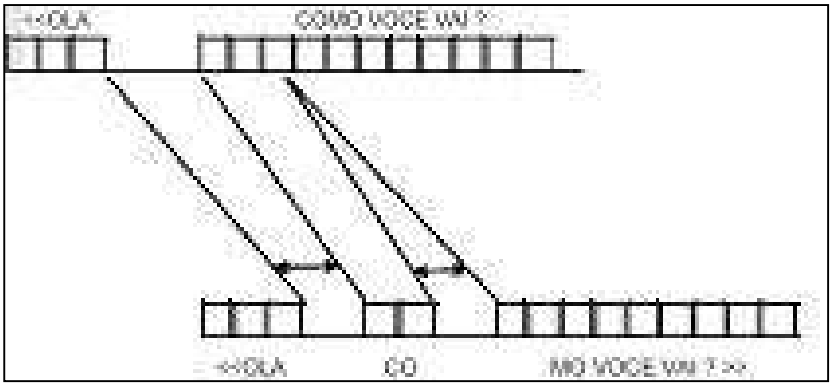
O tempo de propagação na rede, o tempo de transmissão e o processo de codificação são o que caracterizam o atraso total do sistema.

2.1.5 JITTER

A qualidade do sinal de voz recebido também é determinado pela variação do tempo entre chegadas de pacotes consecutivos *jitter* sendo que um tempo de chegada maior mas com uma variação menor caracteriza uma transmissão com melhor qualidade. Diferentes tempos de propagação podem ser causados pelos datagramas terem tomados caminhos diferentes na rede como também teram sofrido congestionamento que obrigue a maiores retardos. Nas entradas dos equipamentos decodificadores são usados *buffers* para que armazenem em uma fila os pacotes que estão chegando para poderem compensar maiores atrasos sempre dentro de um limite determinado pelo tamanho do *buffer*. A Figura 2 mostra a introdução de *jitter* em uma transmissão de voz.

A quantidade de pacotes que podem ser armazenados no *buffer* depende do tamanho dos pacotes, da taxa de transmissão, porque a medida que os pacotes chegam também tem que ser retirados, e do atraso médio da rede.

Figura 2 – Propagação com *Jitter*



2.1.6 TAXA DE PERDAS E ERROS

Segundo Fernandes (2003), em uma rede de telefonia IP, como a transmissão ocorre em tempo real, não há como ter reenvio de pacotes perdidos ou com erros para garantir uma boa qualidade na transmissão. Uma alternativa seria o uso de algoritmos *Forward Error Correction* (FEC), onde o mesmo pacote IP conteria vários quadros de voz implicando um uma redundância de quadro, sendo que só se aplica para codificação que gera pouco atraso, já que a formação de um pacote poderia tornar a solução inviável. Outra exigência seria uma banda de no mínimo 16Kbps.

2.2 QUALIDADE DE SERVIÇO (QOS)

Quando se fala em garantia de *Quality of Service* (QoS) para aplicações de voz sobre IP refere-se a garantia do meio de transmissão com banda suficiente para a transferência do sinal de voz, com um atraso e *jitter* mínimos. Para que seja possível, um conjunto de mecanismos são implementado junto ao protocolo IP, já que ele em sua concepção não foi desenvolvido com esta finalidade.

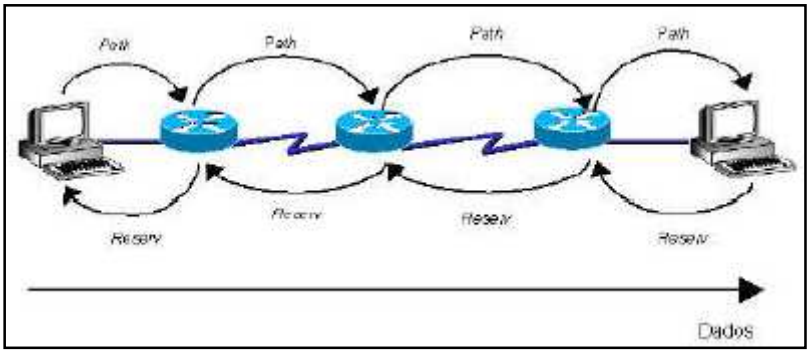
Para prover serviços diferenciados na internet as arquiteturas que provêm estes serviços estão sendo pesquisadas nos últimos anos.O IETF aborda de duas formas diferentes o *Internet Integrated Services* (IIS) e o *Differentiated Services* (DS).

O IIS tem por característica o foco em um fluxo individual de pacotes entre os pontos de origem e destino. Nesta abordagem, cada fluxo pode requisitar diferentes níveis de serviço definindo a banda mínima necessária para a transmissão e o atraso máximo de tolerância. O IIS é composto por quatro componentes básicos:

a) unidade de controle de admissão, que identifica se a rede pode suprir os níveis mínimos necessários;
b) unidade de classificação, que inspeciona os campos dos pacotes para determinar suas classes e o nível de serviço adotado;
c) unidade de *schedule*, que aplica um ou mais mecanismos de gerência de tráfego para garantir que o pacote seja transmitido à rede, a tempo de satisfazer a banda e atrasos adequados ao tipo de fluxo;
d) e o protocolo RSVP, que é o protocolo para reserva recursos.

O *Resource Reservation Protocol* (RSVP) está presente na maioria dos roteadores atuais. Um cliente RSVP pode reservar uma quantidade de banda necessária para prover o tráfego de modo que tenha um baixo atraso para os pacotes de voz. O RSVP é capaz de comunicar a reserva a outros roteadores RSVP como mostrado na Figura 3 onde o cliente manda uma mensagem *path* indicando que quer reservar o recurso e o recepter da mensagem manda uma mensagem *reservation-request* pelo mesmo caminho que a mensagem passou, no entanto esta capacidade tem um impacto direto no desempenho dos roteadores pois cada roteador tem que manter o estado do fluxo. Como roteadores transportam um número muito elevado de fluxo, o processamento desses estados acarreta na sobrecarga dos roteadores.

Figura 3 – Troca básica de mensagens para reserva de recursos



Segundo Fernandes (2003), o DS não tem o foco no tipo de fluxo, mas sim nas aplicações que utilizam níveis de qualidade de serviço semelhante, ou seja, classificam-se os diferentes tipos de tráfegos para determinar que aplicações o utilizarão, seguindo o seguinte grupo de classes:

a) para aplicações que necessitam de atrasos e *jitter* pequenos denomina-se a classe de Serviço *Premium*;

b) para aplicações que necessitam um serviço melhor que o *best-effort* utiliza a classe de Serviço Garantido;
c) e o Serviço Olímpico (subdividido em três subclasses ouro, prata, bronze).

O protocolo IP versão 4 (IPv4) implementa esta classificação através do campo *Type of Sevice* (ToS) do cabeçalho IP. Sendo que os três primeiros bits do campo são usados para identificar a importância do pacote, quanto maior o campo maior a prioridade do pacote.

2.2.1 VOZ

Para que a transmissão da voz seja feita de forma mais eficiente necessita-se que seja feita a digitalização do sinal de voz. Em 1928, por Homer Dudley, tem-se a primeira codificação digital da fala, mas apenas na década de 70 teve uso fora da área militar.

Segundo Fernandes (2003), a técnica de codificação *Pulse Code Modulation* (PCM), modulação por codificação de pulsos, que consiste em 8000 amostragens do sinal de voz contínuo por segundo, representa um valor amostrado de 8bits, o que, para o transmissão de cada canal de voz, implica na necessidade de um canal digital de 64Kbps. Este tipo de codificação possui baixo atraso e pequena complexidade mas requer uma taxa de transmissão elevada.

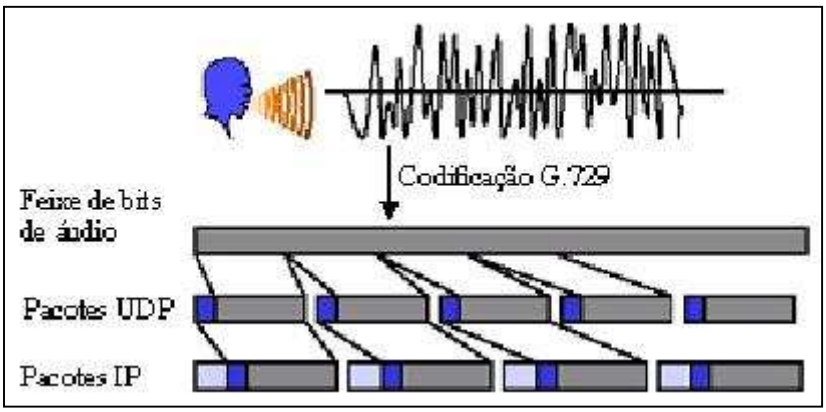
Segundo Souza (2001), a conversação humana é uma forma de onda com freqüências principais na faixa que vai de 300 Hz a 3.4 kHz, com alguns padrões de repetição devido ao timbre de voz e dos fonemas emitidos durante a conversação. O problema da telefonia em geral é reproduzir a qualidade da voz humana em um terminal a distância.

Com a telefonia digital, a voz é codificada em formato digital, que pode ser multiplexado no tempo, de forma a compartilhar o meio de transmissão. Esse fluxo de bits é encapsulado em *datagramas* UDP que são encapsulados em pacotes IP como mostra a Figura 4.

Explorando-se os modelos de codificação de voz, foram desenvolvidas novas técnicas de codificação. Para formatação de quadros após a digitalização, estas técnicas fazem a segmentação de sinal analógico em intervalos periódicos, ou seja, para a composição dos quadros de voz digitalizada, um quadro é composto por informações de voz de um período

mais uma parcela de outro quadro subseqüente. O tempo necessário para coletar as informações do próximo quadro é chamado de *lookahead.* Comparando com a codificação PCM, a taxa requerida é baixa, mas o atraso e a complexidade são elevados.

Figura 4 - Encapsulamento do fluxo de dados (voz) em pacotes IP



Segundo Alves (2002), não é suficiente pegar amostras do sinal digital e colocá-los no pacote que será enviado a rede. Para que seja possível transmitir um pacote de voz numa rede de dados deve-se levar em conta os seguintes fatores:

a) a construção do pacote não é feita em tempo nulo. No caso de amostras de 8Khz é necessário esperar um tempo até que se tenha uma quantidade suficiente para colocar em um pacote de dados. Neste caso está se introduzindo um atraso na comunicação que é proporcional ao tamanho do pacote;

b) nos equipamentos de pacotes numa rede IP (*routers*) existem filas de espera. Se um pacote de voz, sensível às variações de atraso, for colocado numa fila desses equipamentos atrás de um número variável de pacotes que não têm esses requisitos, pode ocorrer variações no atraso (*jitter*), resultando na distorção do sinal;

c) o valor mais comum de normas de digitalização de voz é de 64Kbps. A unidade básica para uma largura de banda de parte dos *routers* é de 64Kbps, com isso os pacotes de voz podem saturar estas ligações, isto sem contar os bits dos cabeçalhos dos pacotes de voz introduzidos pela pilha de protocolos;

d) a norma PCM de 64Kbps é de utilização constante mesmo quando há pausa na conversação, enquanto que redes de dados são concebidas para tratar tráfego com características de rajada. Se o tráfego de silêncio não for enviado caracterizando o

modo rajada pode trazer desconforto ao receptor, ficando uma sensação de que a chamada foi interrompida;

e) como o protocolo TCP/IP é um protocolo orientado à conexão o que caracteriza reenvio de pacotes perdidos, o que em uma conversação em tempo real não pode acontecer, vão ocorrer perdas no diálogo.

Contudo existem mecanismos que tentam ultrapassar essas questões:

a) redução dos tamanhos dos pacotes IP;

b) uso de diferentes prioridades e pacotes com diferentes requisitos;

c) recurso de compressão para que o sinal seja com menor débito do que os 64Kbps pois o pacotes de voz ficam com características mais apropriadas para serem transportados;

d) introduzir no lado do receptor ruídos para atenuar o efeito de perda do som quando há pausas no diálogo;

e) utilizar protocolos mais adequados para transmissão de voz como o *Real-Time Transport Protocol* (RTP) que é semelhante ao TCP mas não tem reenvio de pacotes.

2.3 TCP/IP

A arquitetura internet *Transmission Control Protocol/Internet Protocol* (TCP/IP) dá uma ênfase toda especial à interligação de diferentes tecnologias de redes. A idéia é que não existe nenhuma tecnologia que atenda aos anseios de todos os usuários, uns querem uma rede de alta velocidade mas com curto alcance e outros admitem uma rede de baixa velocidade mas com logo alcance, portanto a forma que se pode obter isto é ligar todos os usuários a uma inter-rede.

O *Internet Protocol* (IP) foi projetado para permitir a interconexão de redes de computadores que utilizam a tecnologia de comutação de pacotes. O ambiente das redes consiste em *hosts* conectados entre si e por sua vez conectados a outras redes através de *gateways*. As redes podem variar de redes locais até redes de longa distância.

Segundo Soares (1995), o protocolo IP é um protocolo sem conexão. Tem por função transmitir *datagramas* de dados de um *host* origem para um *host* destino que são localizados através de seus endereços IP. Outro serviço que o IP fornece é a fragmentação e remontagem

de pacotes cujo tamanho ultrapassa o máximo permitido para quadros da camada de acesso ao meio.

Como o serviço oferecido pelo IP é sem conexão, os *datagramas* transmitidos são tratados como unidades independentes, ou seja, cada pacote IP é independente do outro e não é feita nenhuma checagen fim-a-fim ou entre nós intermediários. O único tipo de verificação de erros que é feito é o *Cyclic Redundance Check* (CRC) que garante que as informações estão corretas (SOARES, 1995).

O *Transmission Control Protocol* (TCP) é um protocolo orientado à conexão que fornece um serviço confiável de transferência de dados fim-a-fim.

O TCP interage de um lado com os processos das aplicações e do outro com o protocolo de internet. Tem por função receber chamadas das aplicações semelhantes às chamadas que os sistemas operacionais fazem aos processos de aplicação, como abrir e fechar conexões, e enviar e receber dados de conexões já estabelecidas.

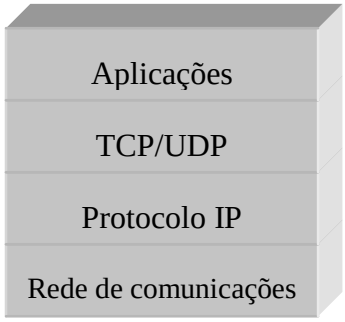
O TCP é capaz de transmitir uma cadeia de dados nas duas direções e geralmente é ele quem decide a hora de transmitir uma cadeia de dados e também de parar de transmitir. Uma exceção é quando recebe uma ordem explícita da aplicação para transmitir imediatamente os dados que estão nos seus buffers.

Segundo Soares (1995), como o TCP não exige um serviço de rede do protocolo de internet confiável para operar, ele garante a recuperação de dados corrompidos, perdidos, duplicados ou entregues fora de seqüência pelo protocolo de internet: garante a qualidade de serviço (QoS) da comunicação.

Para que vários processos possam transmitir simultaneamente, o TCP usa o conceito de porta, ou seja, para cada processo de aplicação que está utilizando o TCP é atribuído uma porta diferente. No entanto, processos de aplicações que são muitos usados como FTP e Telnet são atribuídos a eles portas fixas (SOARES, 1995).

Na Figura 5 pode-se observar o posicionamento do TCP na arquitetura internet TCP/IP.

Figura 5 - Camadas de protocolos da arquitetura internet TCP/IP



Pode-se considerar o TCP/IP como sendo um conjunto de protocolos de comunicação utilizados tanto em redes locais como em redes externas.

Segundo Soares (1995), outra opção de protocolo é o *User Datagram Protocol* (UDP) que fornece um serviço de *datagrama* não-confiável, sendo uma extensão do IP. Tem por função receber os pedidos de transmissão de uma estação origem e os entregar ao IP que é responsável pela transmissão E o processo inverso também ocorre: recebe mensagens do IP e as entrega para aos processos das aplicações.

A arquitetura TCP/IP está dividida em quatro níveis:

a) nível de acesso ao meio: possui os protocolos de nível 1 e 2 do modelo OSI, que carregam a informação entre pontos de uma rede;

b) nível de internet: tem-se o roteamento dos dados na rede efetuado pelo protocolo IP;

c) nível de transporte: aqui atuam os protocolos TCP e UDP que pegam os dados roteados pelo protocolo IP no nível anterior e transmitem para o nível superior no qual tem-se os protocolos de aplicação;

d) nível de aplicação: nesse nível tem-se os protocolos de aplicação, como por exemplo:

a) *File Transfer Protocol* (FTP) protocolo que faz a transferência de arquivos entre computadores;

b) *Simple Mail Transfer Protocol* (SMTP) protocolo de correio eletrônico;

c) *Simple Network Management* (SNMP) protocolo de gerenciamento da rede;

d) *Terminal Emulation* (TELNET) emulação de terminais para acesso a sistemas remotos;

e) *Hipertext Transfer Protocol* (HTTP): protocolo de navegação da internet.

2.4 PADRÕES DE CONTROLE PARA TELEFONIA IP

Para se realizar chamadas telefônicas existem hoje dois padrões que dominam o cenário de telefonia IP, sendo que estes protocolos são responsáveis pelo controle e sinalização que consiste em: localização de usuário, notificação de chamada, notificação de aceite de chamada, início e fim da transmissão e desconexão.

Em sua maioria, os protocolos de sinalização e de controle estão separados da transmissão do conteúdo, pois esses protocolos dependem da aplicação e estão implementados em outros equipamentos com um servidor de localização possibilitando assim a independência entre o emissor e o receptor.

Para garantir a interoperabilidade entre os equipamentos, dois padrões se destacam, o padrão proposto pela ITU-T, H.323 e o padrão proposto pela IETF, SIP.

2.4.1 H.323

Este padrão definido pela ITU-T prevê a implementação de algoritmos que garantam a compatibilidade entre codificadores, conhecidos como *codecs* e *vocodecs*, protocolos para controle de chamadas, estabelecimento de canais de comunicação, negociação de qualidade de serviço, interoperabilidade com outros terminais de telefonia convencional e ISDN e voz sobre ATM.

Existe várias implementações que utilizam H.323 tal como o NetMeeting da Microsoft.

Uma família de diversas funcionalidades especificadas pelo ITU-T é utilizada pelo H.323: H.245 para controle, H.225 para conexão, H.332 para conferências, H.335 para segurança, H.246 para interoperabilidade com o *Real-Time Control Protocol* (RTCP) e a série H.450.x para serviços suplementares.

2.4.1.1 ARQUITETURA DA REDE H.323

Em um ambiente de rede H.323 são definidos alguns elementos para que uma comunicação seja possível. No entanto, se a comunicação é somente entre dois computadores, faz-se necessário somente os terminais H.323.

O terminal H.323 é onde está implementada a aplicação de VoIP que atua como um terminal de voz, já o elemento que fica entre a rede IP e outra rede de telefonia convencional é chamada de *Gateway* H.323. Para que se possa localizar um usuário na rede VoIP utiliza-se o *Gatekepper* que provê o controle de acesso e o mapeamento de endereços. A Unidade de Controle Multiponto (MCU) provê facilidades para três ou mais usuários participares de um grupo de conferência multiponto.

2.4.2 SIP

Alguns padrões relacionados com conferências e telefonia na internet foram desenvolvidos pelo grupo do MMUSIC do IETF que teve seu primeiro standard ratificado e que é o mais usado hoje em dia conhecido como padrão *Session Initiation Protocol* (SIP) e teve sua publicação na RFC 3261 (SILVA, 2002) sendo que posteriormente a IETF passou o desenvolvimento do SIP para um grupo independente para haver uma maior dedicação no trabalho começado.

Segundo Silva (2002), o SIP abrangeu a telefonia IP de uma forma diferente do padrão H.323 sendo que sua aplicação se tornou simples. No entanto serviços oferecidos não se tornaram inferiores, pois já na recomendação inicial estavam incluídos serviços adicionais como transferência de chamada e chamada em espera.

Uma das características do SIP é de não tentar definir qualquer aspecto de comunicação multimídia e preocupar-se com a sinalização, sendo que ele reutiliza algumas características de outros protocolos como os cabeçalhos, erros e regras de codificação do HTTP.

No protocolo SIP as requisições são geradas nas entidades cliente e enviadas a uma entidade receptora ou servidora sendo que esta processa a mensagem e manda de volta. O usuário do sistema final é conhecido como *User Agent Client* (UAC) e o servidor conhecido

como *User Agent Server* (UAS). Uma aplicação de voz sobre IP contém UAC e UAS para receber e responder as mensagens. A Figura 6 ilustra uma operação básica SIP.

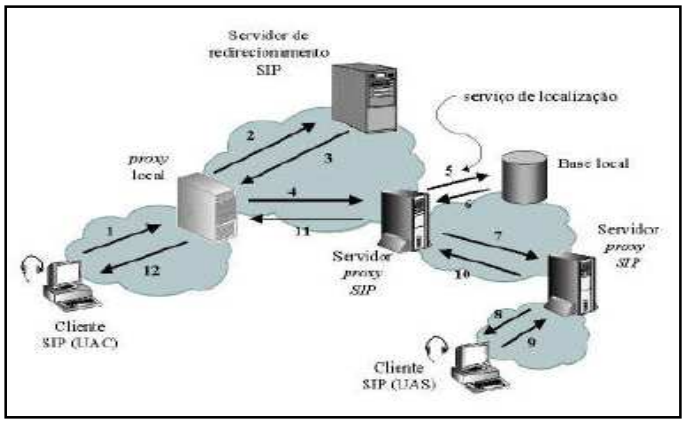
Segundo Oliveira (2001), existem três tipos de servidores espalhados pela rede de VoIP:

a) servidor de registros que recebem requisições sobre a localização corrente de cada usuário;

b) servidor *proxy*, conhecidos como *next-hop* que recebem requisições e enviam-nas para outros servidores ou para os clientes;

c) servidor de redirecionamento também recebe requisições e determina um outro servidor.

Figura 6 - Operação básica do SIP



2.4.3 COMPARAÇÃO ENTRE O SIP E H.323

O H.323 é baseado em protocolos do ITU-T já existentes e tem uma abordagem voltada aos equipamentos terminais.

O SIP é similar ao HTTP e tem uma abordagem voltada aos usuários e serviços integrados na internet.

Segundo Fernandes (2002), na comparação apresentada a seguir serão considerados os aspectos de complexidade de implementação ou funcionamento, expansão funcional, ou seja, facilidade de inclusão de novas funcionalidades, escalabilidade que é a facilidade para aumento da quantidade de elementos interligados e por fim serviços oferecidos.

2.4.3.1 COMPLEXIDADE

A maior complexidade de implementação do H.323 em relação ao SIP se deve ao fato da documentação do H.323 ter 736 páginas contra apenas 128 do SIP que leva o desenvolvedor dedicar muito tempo apenas para o entendimento do funcionamento do H.323.

O SIP trabalha com apenas 37 tipos de cabeçalhos enquanto que o H.323 tem centenas.

O H.323 trabalha com vários protocolos sem uma separação clara, ou seja, esses protocolos são usados por vários serviços. Já no SIP, em uma mesma requisição estão todas as informações necessárias.

2.4.3.2 EXPANSÃO FUNCIONAL

Devido à estrutura textual do SIP, novas características são incluídas de forma fácil e compatível com versões anteriores e novos parâmetros podem ser colocados em qualquer parte de mensagem. No H.323 existem campos predefinidos para essas novas inclusões.

Se um novo “codec” é registrado em um órgão competente, é possível ser suportado pelo SIP, enquanto o H.323 há uma dificuldade na inclusão de novos “codecs” porque eles devem ser padronizados pelo ITU-T.

2.4.3.3 ESCALABILIDADE

Os servidores ou *gateways* SIP podem trabalhar nos modos *stateful* ou *stateless*, sendo que no segundo caso os servidores recebem e encaminham as requisições não mantendo nenhum tipo de informação, pois as mensagens possuem informações suficientes para garantir que a mensagem seja enviada corretamente. O H.323 é *stateful*, ou seja, ele mantém todo o controle do estado da chamada durante toda a duração, em um ambiente onde pode haver muitas chamadas simultaneamente implicando em problemas de performance.

2.4.3.4 SERVIÇOS

Os dois protocolos oferecem serviços bastante parecidos. As facilidades de transferência, conferências e encaminhamento de chamadas são entendidos como serviços.

Medidas de simulações de dois processos foram feitas entre alguns locais dos Estados Unidos e indicados os percentuais de dias onde ocorreu mais de 1% de rejeição de chamadas

(Tabela 1), sendo que foi estabelecido o tempo máximo de 2 segundos de espera (FERNANDES, 2002).

Tabela 1 - Percentual de dias onde mais de 1% das chamadas não foram completadas.

		Boston	Chicago	West Coast	Washington	Colorado
New York	SIP	20,3	77,2	32,3	9,1	15,4
	H.323	28,2	94,7	40,0	20,0	18,5
Boston	SIP		1,6	31,5	0,0	5,4
	H.323		1,6	31,5	0,0	10,8
Chicago	SIP			34,3	5,2	28,6
	H.323			34,3	6,9	61,4
West Coast	SIP				33,3	45,3
	H.323				36,7	57,3
Washington	SIP					6,6
	H.323					6,6

3 SIP

O protocolo *Session Initiation Protocol* está descrito na RFC 3261 do IETF, sendo este um protocolo de sinalização para criar, modificar e terminar sessões de confêrencias multimídia ou chamadas telefônicas na internet.

O IETF define vários protocolos escalonáveis e em tempo real que formam a arquitetura de comunicações e o SIP é apenas um deles. Sendo os principais o *Session Description Protocol* (SDP), *Real-Time Transport Protocol* (RTP), *Real-Time Control Protocol* (RTCP) e *Media Gateway Control Protocol* (MGCP).

Segundo Fernandes (2002) o SIP é um protocolo de controle referente à camada de aplicação de modelo de referência *Open System Interconection* (OSI). Localização de usuários, estabelecimento de chamadas, suporte a *unicast* ou *multicast* e administração na participação de chamadas são funcionalidades que o SIP possui.

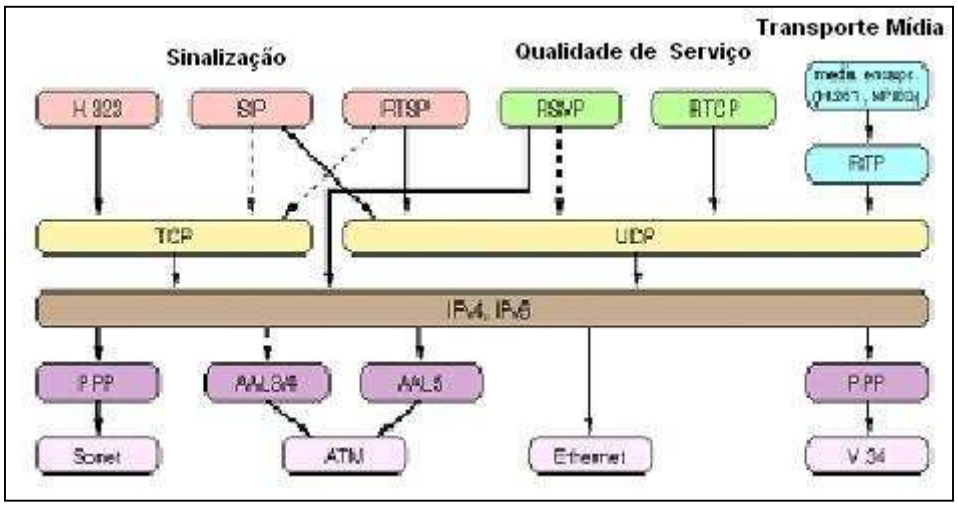
O SIP possui a possibilidade de participação de um usuário em terminal H.323 via *gateway* e é um protocolo cliente-servidor similar ao HTML no tocante à sintaxe e semântica das estruturas empregadas com campos explicitamente descritos.

Um conjunto de campos de cabeçalho que descrevem a chamada, seguida por uma mensagem que descreve a sessão que será utilizada na chamada, que normalmente é o SDP, consiste em uma requisição SIP. A Figura 7 mostra uma visão geral da arquitetura dos protocolos.

Uma das características do SIP é de não definir o protocolo a ser usado nas comunicações multimídia, sendo que ele é combinado com outro protocolo IETF para criar uma solução completa que, por exemplo, se a aplicação usar características de tocar e gravar, pode ser usado o RTSP que provêm funcionalidades de *voicemail* (OLIVEIRA, 2002).

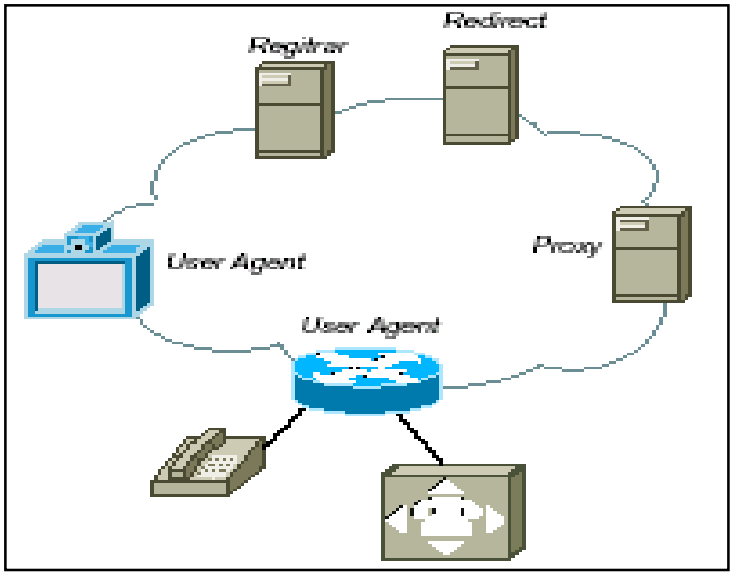
As requisições SIP são geradas por um cliente e enviadas ao servidor que processa a requisão e manda uma resposta ao cliente. Um cliente SIP é formado por dois módulos obrigatórios, um chamado de *User Agent Client* (UAC) que gera as requisições e outro chamado de *User Agent Server* (UAS) que é responsável por responder as chamadas.

Figura 7 - Arquitetura dos protocolos.



Também em uma arquitetura SIP existem três tipos de servidores, ou seja, um servidor de registro que armazena as localizações dos usuários, um servidor *proxy* que recebe as requisições e manda para um próximo servidor que saiba a localização do usuário e um servidor de redirecionamento que também recebe requisições, mas retorna o endereço do servidor para onde a requisição deve ser encaminhada. A Figura 8 representa os componentes que fazem parte de uma rede SIP.

Figura 8 - Componentes SIP



3.1 MENSAGENS SIP

Segundo Hersent (2002), as mensagens SIP são codificadas usando a sintaxe de mensagem do HTTP, sendo que o conjunto de caracteres é o ISO 10646 com codificação de 8 bits e as linhas são terminadas com *Carriage Return, Line Feed* (CRLF).

As mensagens SIP trafegam por padrão pela porta 5060 sendo que o usuário pode escolher em qual porta deseja receber as mensagens.

Os dois tipos de mensagens SIP, *requests* (pedidos) e *responses* (respostas), compartilham um mesmo formato.

As respostas possuem alguns campos de cabeçalhos que estão tanto nos pedidos como nas respostas tais como:

a) Call-ID: serve para coincidir os pedidos com as respostas correspondentes como nos pedidos *REGISTER* e *OPTIONS*. Para os pedidos de *INVITE* e *REGISTER* ajuda a detectar cópias duplicadas. Para cada nova chamada deve-se gerar um novo Call-ID sendo que a primeira parte do Call-ID deve usar um nome único para cada computador e a segundo um endereço IP para tornar a máquina globalmente única;

b) Cseq: cada pedido deve ter um campo Cseq que deve ser um campo composto por um número sem sinal. Para cada chamada o número o campo Cseq é incrementado com a exceção da chamada ser uma retransmissão da chamada anterior, onde o servidor deve copiar o valor de Cseq para o pedido;

c) From: este campo deve estar em todos os pedidos e respostas sendo que ele contém um nome opcional a ser mostrado e o endereço do originador do pedido;

d) To: este campo indica o destino pretendido de um pedido. Ele é simplesmente copiado nas respostas. Um *tag* (etiqueta) pode ser usado quando houver mais de um destinatário (caso de um *helpdesk*) para distinguir as respostas de pontos finais diferentes;

e) Via: este campo é usado para armazenar o caminho de um pedido SIP para permitir que os servidores intermediários possam retransmitir as respostas para o mesmo caminho. Cada proxy adiciona seu endereço no campo Via;

f) Encrypton: este campo indica que parte do cabeçalho e o corpo da mensagem podem estar criptografados;

g) Content-Type: este campo indica o tipo de mídia que está sendo utilizado no corpo da mensagem;

h) Content-lenght: contém o número de bytes do corpo da mensagem;

i) Contact: indica no caso de registro onde o cliente deseja ser contatado.

Na Figura 9 pode-se observar a estrutura das mensagens SIP de *request* e *response*.

Figura 9 - Estrutura da mensagem SIP

3.1.1 PEDIDOS SIP

Os pedidos SIP são gerados pelo UAC e enviados para o UAS. Esses pedidos podem ser:

- ACK: um pedido ACK é enviado pelo cliente para confirmar que ele recebeu uma resposta do servidor;
- BYE: um pedido BYE é enviado para um agente de origem ou de um agente de destino para interromper uma chamada;
- CANCEL: esse pedido deve ser enviado quando se quer interromper a transmissão antes de receber uma resposta do servidor;
- INVITE: é usado para iniciar uma chamada;

- OPTIONS: é um pedido enviado a um servidor para saber as capacidades, sendo que o servidor pode enviar de volta uma lista de métodos e até em alguns casos pode enviar as capacidades de algum usuário requisitado;
- REGISTER: um cliente pode registrar um ou mais endereços de sua localização.

Os pedidos SIP possuem alguns campos adicionais para especificar características específicas adicionais, tais como:

- Accept: este cabeçalho opcional indica quais tipos de mídia são aceitáveis na resposta;
- Accept-Language: indica as linguagens preferidas pelo originador da chamada;
- Expires: para uma mensagem *REGISTER* esse campo indica quanto tempo o registro será válido. Para uma mensagem *INVITE* pode ser usado para limitar a duração de buscas;
- Priority: indica a prioridade do pedido;
- Record-Route: é usado por alguns *proxies* para adicionar ou atualizar o campo de cabeçalho se quiser fazer parte do caminho de todas as mensagens de sinalização;
- Subject: campo de texto livre que deve fornecer alguma informação sobre a natureza da chamada.

3.1.2 RESPOSTAS SIP

As respostas SIP na primeira linha sempre contêm um código de status e uma frase de justificativa inteligível e a maior parte do cabeçalho é copiada da mensagem de pedido.

Foram definidas seis categorias de código de status para pedidos e respostas SIP que dependem do primeiro dígito como mostrado na Tabela 2 sendo que esta classificação torna mais fácil a identificação dos códigos.

Tabela 2 - Categorias de códigos de status

1xx – Informativo		Pedido continuando a processar o pedido
	100 180 181 182	Tentando Chamando A chamada está sendo retransmitida Colocado na fila
2xx – Sucesso		A ação foi recebida, atendida aceita com sucesso
	200	OK

3xx – Redirecionamneto		Uma ação adicional deve ser tomada para completar o pedido
	300	Múltiplas escolhas
	301	Movido permanentemente
	302	Movido temporariamente
	380	Serviço alternativo
4xx – Erro de Cliente		O pedido contém sintaxe inválida ou não pose ser efetuado neste servidor
	400	Pedido inválido
	401	Não autorizado
	402	Necessário pagamento
	403	Proibido
	404	Não encontrado
	405	Método não permitido
	406	Não aceitável
	407	Necessária autenticação do *proxy*
	408	Tempo para o pedido esgotado
	409	Conflito
	410	Não mais presente
	411	Necessário fornecer o comprimento
	413	Corpo da mensagem de pedido muito grande
	414	URI do pedido muito grande
	415	Tipo de mídia não suportado
	420	Extensão inválida
	480	Temporariamente não disponível
	481	Transação ou *leg* de chamada não existe
	482	*Loop* (laço) detectado
	483	Excesso de *hops* (segmento)
	484	Endereço incompleto
	485	Ambíguo
5xx		Erro de servidor
	500	Erro interno ao servidor
	501	Não implementado
	502	*Gateway* inválido
	503	Serviço não disponível
	504	Tempo esgotado no *gateway*
	505	Versão SIP não suportada
6xx		Falha global
	600	Ocupado em todos os lugares
	603	Declínio
	604	Não existe em lugar nenhum
	606	Não aceitável

3.2 ESTABELECENDO UMA CHAMADA

Para estabelecer uma chamada um cliente SIP chama um outro ponto final SIP enviando uma mensagem de pedido *INVITE*. Nessa mensagem possui informações que o ponto de origem pode suportar para que o ponto de destino estabeleça a conexão de mídia, solicitada e também o endereço onde o ponto de origem deseja receber esses dados de mídia sendo que essa codificação escolhida aparece como parte do cabeçalho RTP.

Para o ponto de destino indicar que está aceitando uma chamada ele responde com uma mensagem OK e o ponto de origem indica que recebeu a mensagem com uma mensagem ACK.

Isso mostra a simplicidade do SIP para estabelecer uma chamada sendo que com uma ida e uma volta de mensagens e mais uma ida e uma volta de negociação do canal de mídia é possível estabelecer um canal de comunicação.

3.3 NEGOCIAÇÃO DO *CODEC*

Para a negociação de *codec,* um terminal de origem envia uma mensagem de INVITE passando um codificador de áudio. Se o receptor não suportar essa codificação de áudio, talvez por não ter a largura de banda necessária ou por não ter o codificador que é requisitado, ele envia uma mensagem “606 Not Acceptable” e seleciona uma lista de *codecs* que ele suporta e por sua vez o terminal de origem envia novamente uma mensagem INVITE com um codificador que o terminal de destino pode aceitar.

Segundo Hersent (2002), o SIP não define nenhum tipo de *codec* que deve ser usado para estabelecer uma chamada.

3.4 FINALIZANDO UMA CHAMADA

Se em algum momento da sessão alguma das partes quiser finalizar a chamada, ela envia um pedido do tipo BYE e inverte os campos TO e FROM do cabeçalho. Mesmo que os fluxos de mídia não são mostrados, as mensagens incluem os cabeçalhos obrigatórios.

3.5 REJEITANDO UMA CHAMADA

Se por algum motivo um cliente não pode atender à chamada, seja porque ele não deseja atender ou está em outra conversação, existem mensagens que expressam essa condição. O originador de uma mensagem pode tentar localizar um destino em outros lugares como, por exemplo, se um usuário não se encontra no terminal no momento, ele pode indicar onde ele está e as mensagens que chegam para ele podem ser enviadas para outro destino. Se o usuário não desejar ser encontrado em nenhum lugar ele pode responder a uma requisição de mensagem com uma resposta do tipo “600 Ocupado em todos os lugares”.

3.6 SDP

O SIP utiliza o protocolo *Session Description Protocol* (SDP) para definir uma sintaxe padrão para o tipo de transmissão de áudio que será usada. Este protocolo inclui as seguintes funções:

- fluxo de mídia: o SDP leva informações sobre o número e o tipo de cada fluxo de mídia já que em uma sessão pode haver vários fluxos;
- endereços: para garantir a independência para cada fluxo, é indicado o endereço do destinatário, seja *unicast* ou *multicast*;
- portas: para cada fluxo, a porta UDP para recepção e/ou envio é indicada;
- tipo de conteúdo: define o formato de mídia que pode ser usado na sessão;
- origem: para poder contatar o responsável pela chamada nas sessões do tipo *broadcast*.

O protocolo SDP consiste em várias linhas "<type>=<value>" que podem ser legíveis aos usuários sendo que os nomes dos campos e os atributos usam caracteres ASCII facilitando assim a programação e a depuração.

Tabela 3 – Exemplo de uso dos campos SDP

```
v = 0
o = sergiool 87728 8772 IN IP4
15.164.10.1
s = Ola, senhores!
u = http://www.dcc.ufmg.br/~sergiool
e = sergiool@dcc.ufmg.br
c = IN IP4 150.164.10.1
b = CT:64
t = 3086272736 0
k = clear:manhole cover
m = audio 3456 RTP/AVP 96
a = rtpmap:96 VDVI/8000/1
m = video 3458 RTP/AVP 31
m = application 32416 udp wb
a = orient:portrait
```

Na tabela 3 identifica os campos do protocolo SDP no qual o "v" indica a versão da sessão, a linha "o" apresenta o conjunto de valores para identificar a sessão que inclui

endereço IP, portas utilizadas e usuário, o endereço de e-mail e a URL são informados para mais informações sobre a sessão e são indicados no campo “u”. Endereço para a sessão é indicado no campo “c” e a linha “b” indica a largura de banda, a linha “t” o tempo de início e de fim. A linha “k” traz a chave de criptografia para a sessão, a linha “m” indica o tipo de fluxo de mídia, o número de porta para o fluxo, o protocolo e a lista de tipos de conteúdo, a linha “a” representa um conteúdo.

3.7 VISÃO GERAL DO RTP E O RTCP

Segundo Hersent (2002), o receptor tem que levar em conta o *jitter* numa transmissão de dados em tempo real que usa multiplexação estatística.

Para que os receptores pudessem compensar a perda de *jitter* e a perda de seqüência dos pacotes introduzidos na rede IP, foi projetado o *Real-Time Transport Protocol* (RTP) que pode ser usado para qualquer fluxo de dados em tempo real definindo o modo de formatar pacotes IP e incluem informação sobre o tipo de dados transportado, *timestamps* e número de seqüências.

Junto com o RTP pode-se usar o *Real-Time Control Protocol* (RTCP) que é um protocolo de controle de transmissão em tempo real e é usado para transportar algum retorno sobre a qualidade da transmissão e também transportar algumas informações a respeito da identidade dos participantes.

O comportamento da rede IP não é influenciado pelo uso do RTP e RTCP que não controlam a qualidade de serviço, sendo que a rede pode perder o serviço, inserir atraso ou perder os pacotes RTP. O RTP possibilita para o receptor compensar o *jitter* por meio de controle de buffer e seqüenciamento dos pacotes tendo mais informações da rede para poder tomar medidas de correção.

O RTP e RTCP podem ser usados acima de qualquer protocolo, mas eles são usados em cima do UDP uma vez que os dados precisam ser transportados com uma latência muito baixa. O RTP costuma ser associado a uma porta do UDP de número par e o RTCP é associado a próxima porta ímpar do UDP.

3.7.1 RTP

O RTP é usado em uma rede que introduz *jitter* e pode tirar a seqüência dos pacotes para transmitir dados de áudio e vídeo.

Para que o cliente possa gerenciar as chegadas dos pacotes de uma forma correta, o RTP usa o número de seqüência, que em uma aplicação que esteja reproduzindo o áudio pode definir um *buffer* para armazenar os pacotes que chegam em uma ordem correta antes de reproduzir. Se na hora de reproduzir o áudio o pacote não tiver chegado, a aplicação pode optar por copiar o último pacote que foi reproduzido e repeti-lo até chegar a vez do próximo ou ainda usar algum esquema de interpolação definido pelo *codec* de áudio.

O *timestamp* que é o campo usado para guardar informações sobre o tipo de dados é usado em uma aplicação de vídeo que permite, por exemplo, deduzir qual é parte de tela é descrita pelo pacote IP, no entanto devido a problemas de seqüenciamento ainda usa-se o pacote para construir a parte da imagem que ele descreve.

Como cada formato do pacote RTP de informação em tempo real é livre usa-se o *payload type* (tipo de *payload*) no cabeçalho de cada pacote RTP para distinguir um formato em particular sem que seja necessário analisar o conteúdo do *payload*.

Define numa sessão RTP uma associação de participantes em que cada participante usa dois endereços. No caso do UDP são duas portas para cada sessão sendo uma para o fluxo RTP e uma para o RTCP.

A fonte de sincronização, *Synchronization Source* (SSRC), é uma fonte de fluxo RTP identificada por 32 bits no cabeçalho RTP. Uma mesma referência de tempo e de seqüenciamento é usada com um SSRC nos pacotes RTP.

Fonte contribuinte, *Contributing Source* (CSRC), é usada quando o fluxo RTP é resultado de uma combinação de fluxos feita por um *mixer* (misturador) RTP sendo que cada fluxo possui um CSRC que é adicionado a lista de CSRC´s que forma um SSRC de todo o fluxo.

O formato *Network Time Protocol* (NTP) é uma maneira padrão de formatar um *timestamp* escrevendo o número de segundos passados desde 01/01/1900 com 32bits para a parte inteira e 32 bits para a parte decimal.

A Figura 10 mostra a estrutura de um pacote RTP sendo que dois bits são reservados para a versão do RTP, o bit de *padding* (P) indica se o *payload* sofreu enchimento para fins de alinhamento e se tiver sofrido enchimento o último octeto guarda a quantidade que foi acrescentada, um bit de extensão X que indica a presença de extensões após eventuais CSRCs do cabeçalho fixo, o contador de CSRC (CC) que indica a quantidade de CSRCs e um bit de marcador (M) que é definido pelo perfil do RTP.

Figura 10 - Pacote RTP

<table>
<tr><td colspan="2">V=2</td><td>P</td><td>X</td><td colspan="2">CC</td><td>M</td><td colspan="4">Tipo de payload</td><td colspan="8">Número de seqüência</td></tr>
<tr><td colspan="16">Timestamp</td></tr>
<tr><td colspan="16">Identificador de fonte de sincronização (SSRC)</td></tr>
<tr><td colspan="16">Identificador de fonte contribuinte (CSCR)</td></tr>
<tr><td colspan="8">Depende de perfil</td><td colspan="8">Tamanho</td></tr>
<tr><td colspan="16">Dados</td></tr>
<tr><td>0</td><td>2</td><td>4</td><td>6</td><td>8</td><td>10</td><td>12</td><td>14</td><td>16</td><td>18</td><td>20</td><td>22</td><td>24</td><td>26</td><td>28</td><td>30</td></tr>
</table>

3.7.2 RTCP

Pacotes de controle relativos a uma sessão RTP são transmitidos de tempos em tempos para os participantes de uma transmissão, sendo que esses pacotes incluem informações a respeito dos participantes em suas fontes de fluxo individuais. Uma forma de transmitir essas informações é através do uso de *sender reports* usados pelos transmissores e *receiver reports* usados pelos receptores.

Um problema com a transmissão freqüente desses pacotes é que a medida que aumenta o número de participantes de uma transmissão *multicast* aumenta também o número de pacotes RTCP trafegando na rede. Esse problema não ocorre com o RTP porque os participantes não falam todos ao mesmo tempo.

Como em uma transmissão o número de participantes é conhecido por todos, pode-se controlar a emissão de pacotes RTCP limitando-os estritamente ao necessário, no entanto o

RTP define que 5% da largura de banda da sessão pode ser usado para pacotes RTCP, dos quais por sua vez, um quarto se destina aos transmissores que enviam informações destinadas a sincronização.

Segundo Hersent (2002), há vários tipos de pacotes RTCP para cada tipo de informação, tais como:

- *Sender Reports* (SR), possui informações de transmissão e recepção para transmissores ativos;
- *Receiver Reports* (RR), contém informações de recepção para ouvintes que não sejam também transmissores ativos;
- *Source Description* (SDES), descrevem vários parâmetros de fonte;
- BYE, utilizado para um participante abandonar uma conferência;
- APP é utilizado pela aplicação para funções específicas.

Um pacote UDP pode conter vários pacotes RTCP que possuem informações suficientes para serem decodificados sendo que esse empacotamento diminui o *overhead* gasto.

Cada pacote SR contém três seções obrigatórias, sendo que a primeira contém o contador de relatórios de recepção (RC), que possui o tamanho de 5 bits para determinar o número de relatórios; tipo de *payload* (PT) que tem o objetivo de evitar que se misture com um pacote RTP; para representar o tamanho do SR utiliza-se um campo de 16 bits e o SSRC do originador (HERSENT, 2002).

Informações a respeito do fluxo RTP enviado para o transmissor são encontrados na segunda seção que define um *timestamp* NTP que se dá no instante do envio do relatório. O campo para a contagem de pacotes do transmissor do início da transmissão possui um tamanho de 32 bits que é o mesmo tamanho do campo para a contagem de octetos do *payload* do transmissor.

Segundo Hersent (2002), a terceira seção possui, para cada fonte que o transmissor teve conhecimento, um conjunto de blocos de relatórios de recepção sendo que o SSRC_n é o identificador da fonte a qual ele está se referindo, a fração perdida tem um tamanho de 8 bits e

é dada pelo cálculo de pacotes recebidos dividido pelos pacotes esperados vezes 256, tendo um campo que acumula o número de pacotes perdidos desde o início da transmissão, um campo de 32 bits em que armazena o valor estendido do número de seqüência que se divide nos 16 bits mais significativos armazenando a quantidade de vezes que os ciclos atingiram o número máximo e os outros 16 bits contém o último número da seqüência. Para fazer a estimativa da variância do tempo entre chegadas utiliza-se o campo *jitter* entre chegadas.

Um pacote *Receiver Report* (RR) difere-se apenas no campo *Payload Type* (PT) e na segundo seção relativa ao transmissor que não está presente.

3.8 ENTIDADES SIP

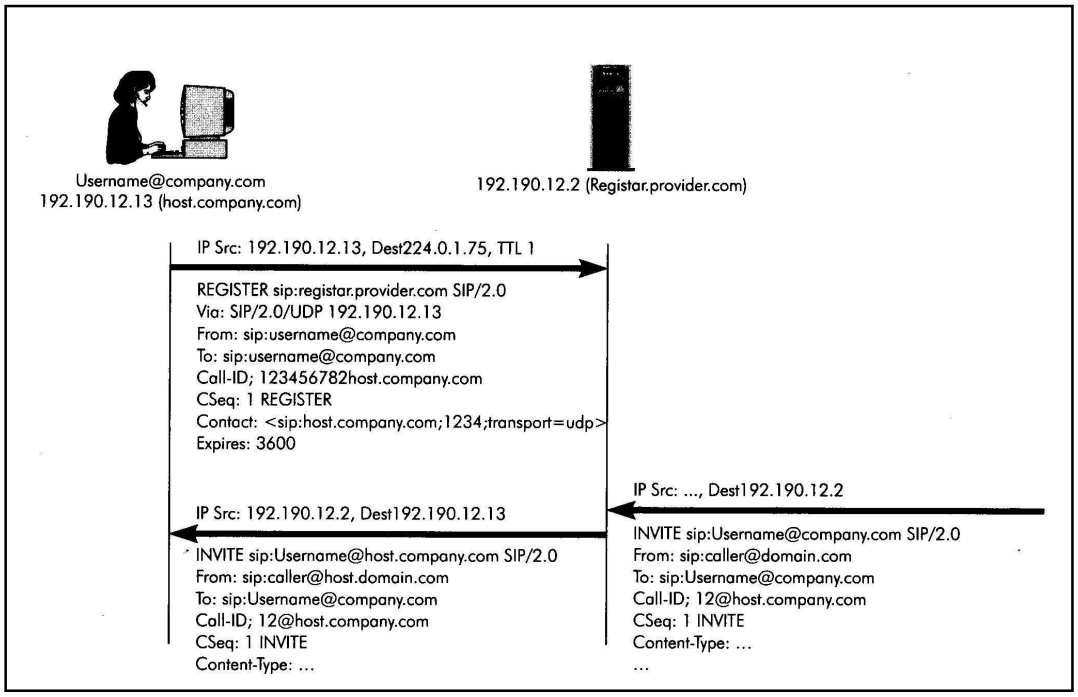
Entidades SIP são componentes de apoio a rede de voz sobre IP que tem por função o mapeamento de usuários.

3.8.1 REGISTRAR

Segundo Hersent (2002), o *Registrar* é um servidor que aceita os pedidos REGISTER, sendo que ele também pode implementar outras funções como a de um *proxy*.

Para que a localização de um usuário seja conhecida por todos os usuários da rede, utilizamos o *Registrar*. O IP do usuário pode mudar devido a várias circunstâncias como o usuário usar uma conexão discada que fornece endereços dinâmicos, participar de uma rede que utiliza o serviço de DHCP que também fornece número IP dinamicamente ou mesmo se tratar de um usuário móvel, sendo para isto necessário manter o mapeamento dos endereços SIP e endereços IP.

Figura 11 – Troca de mensagens SIP com Registrar



A Figura 11 demonstra a troca de mensagens entre um cliente SIP e o Registrar mostrando a mensagem REGISTER enviado ao Registrar para processar o registro sendo que quando chega uma mensagem ao Registrar ela é processa e manda para o seu destinatário.

Um cliente SIP não necessita fazer configurações manuais, pois o SIP mantém um endereço conhecido chamado *all SIP server* possibilitando ao cliente registrar seu endereço IP através de uma mensagem REGISTER *multicast*.

No momento os clientes SIP não podem aprender o endereço de um servidor SIP ou saber se algum servidor *Registrar* aceitou o registro, pois eles não são capazes de responder a mensagem REGISTER *multicast* sendo esta uma limitação da própria definição do SIP.

Se o endereço do *Registrar* for conhecido, o cliente tem a opção de contatar o servidor através de mensagem REGISTER *unicast*.

Um usuário deve mandar a mensagem REGISTER periodicamente para atualizar seu estado, pois o registro tem um valor padrão de uma hora, sendo que esse valor pode ser definido no campo de cabeçalho “Expires”.

3.8.2 PROXY

Um servidor *proxy* pode passar uma mensagem adiante sem alterar ou pode alterar alguns campos do cabeçalho, pois ele atua de um lado como servidor , recebendo mensagens, e por outro lado como cliente enviando mensagens, sendo que ele pode gerar uma resposta localmente e enviar para o transmissor (HERSENT, 2002).

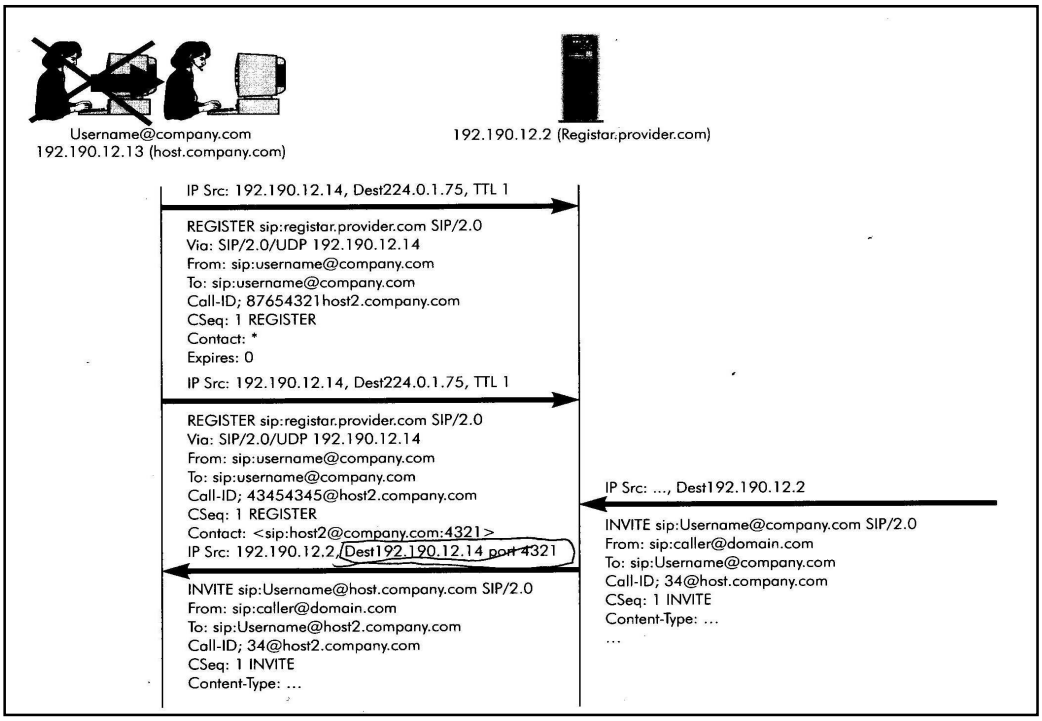
Se uma mensagem deve seguir o mesmo caminho da ida na volta, talvez por motivo de tarifa ou controle de *firewall,* os *proxys* colocam informações nos cabeçalhos para que a mensagem saiba onde é a origem.

Em uma transação SIP, se for utilizado TCP, ele próprio faz o controle para onde deve ir a resposta. No entanto, se for utilizado o UDP, deve incluir informações adicionais no cabeçalho para permitir que o receptor saiba para onde mandar a resposta.

Os cabeçalhos “Via” do SIP servem para controle por onde a mensagem está passando ajudando a evitar *loops* (laços) de roteamento, pois por cada *proxy* que a mensagem passa é verificado se o endereço dele está na lista “Via” sendo que se o endereço não estiver na lista do *proxy* SIP ele o adiciona e quando o *proxy* repassa uma resposta ele faz o serviço inverso retirando o endereço da lista.

Se o *proxy* deve rotear não apenas pedidos e respostas, mas todas as mensagens o cabeçalho Via não é suficiente então os *proxies* fazem uso do cabeçalho “Record Route”. Isto ocorre porque os clientes SIP adicionam um campo no cabeçalho que permite que os servidores respondam diretamente para os clientes. Fazendo uso do campo “Record Route” os *proxies* incluem seus endereços SIP na primeira posição do cabeçalho fazendo com que eles estejam no caminho de todas as mensagens.

Figura 12 – Registro do cliente e o encaminhamento da mensagem



A Figura 12 demonstra o Registro do cliente e o encaminhamento da mensagem para o cliente.

As mensagens roteadas pelos *proxies* SIP contém informações suficientes para permitir que os *proxies* sejam *stateless* (sem-estado), ou seja, não é necessário que eles monitorem todo o fluxo que passa por eles.

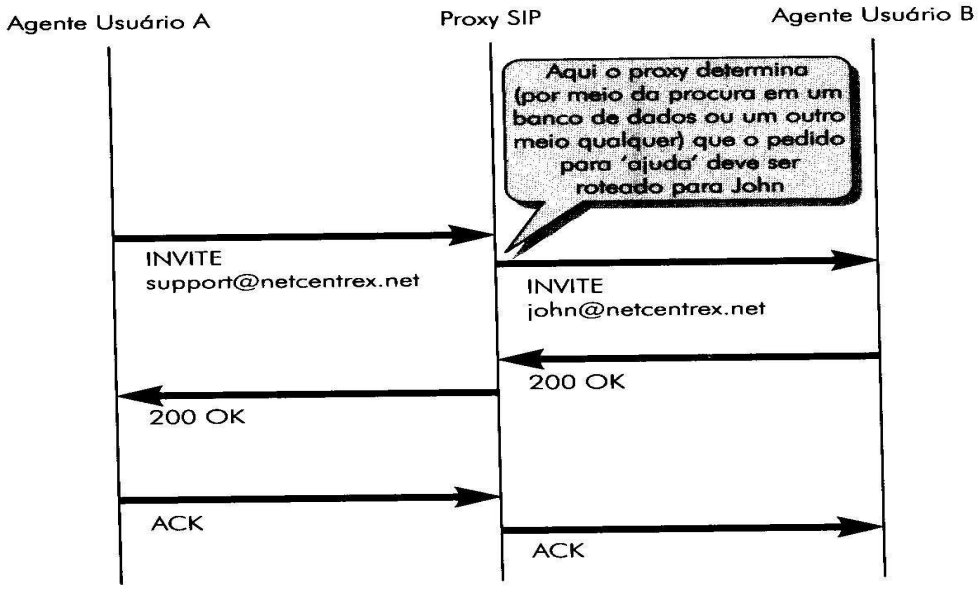
3.8.3 SERVIDOR DE REDIRECIONAMENTO

Um servidor de redirecionamento é uma ferramenta útil para melhorar o escalonamento de servidores de distribuição de chamadas ou servidores agentes de chamadas, pois pode distribuir chamadas entre grupos de servidores secundários permitindo assim um equilíbrio de carga. Por exemplo, uma chamada feita para um determinado endereço pode ser atendida por um outro servidor da rede. Isto se faz possível através do campo “Contact” onde ele identifica a partir do domínio do endereço um servidor secundário da rede (HERSENT, 2002).

Um pedido INVITE pode originar no servidor de redirecionamento algumas respostas tais como:

- múltiplas escolhas (300) indica que o cliente pode ser encontrado em vários lugares sendo que a resposta indicaria os endereços;
- movido permanentemente (301) indica que o cliente não pode ser encontrado no endereço pedido e que deve-se tentar contatar em outra localização possível passando no campo “Contact” da resposta possíveis destinos;
- movido temporariamente (302): indica que para um cliente que esta em outra localização mas por tempo limitado;
- serviço alternativo (380) além de dizer uma nova localização do usuário também adiciona as capacidades que o usuário tem para transmitir. Assim, o cliente, ao gerar um pedido para o usuário, adiciona as capacidades que o usuário suporta evitando assim uma possível retransmissão.

Figura 13 – Encaminhamento de mensagem entre o cliente e o Proxy



A Figura 13 mostra a troca de mensagem entre o Registrar e o cliente onde o cliente manda uma mensagem para o Registrar que processa e identifica que o cliente esta em outro lugar enviando mensagem para o cliente *Moved* indicando o domínio do cliente e o endereço IP.

Para direcionar as chamadas para a localização atual do usuário o servidor de redirecionamento pode ser usado em conjunto com o *Registrar* que também pode atuar como um sistema de distribuição de chamada.

4 DESENVOLVIMENTO DO PROTÓTIPO DE SOFTWARE

No desenvolvimento do protótipo criou-se um sistema que forneça suporte aos sistemas de VoIP para que os mesmos possam contatar usuários de outros sistemas de VoIP sem a necessidade de conhecer o endereço IP. Abrangeu-se neste protótipo a possibilidade do protótipo enviar a mensagem que chega a ele diretamente para seu destino como também responder ao usuário que esta contatando as mensagens pertinentes a sua requisição.

4.1 REQUISITOS PRINCIPAIS DO PROTÓTIPO DE SOFTWARE

O protótipo de software deve ser responsável por:

a) receber mensagens dos sistemas de uma rede de Voz sobre IP;

b) interpretar as mensagens;

c) interagir com o banco de dados;

d) registrar os endereços dos usuários;ler

e) registrar endereços gerando uma resposta para os clientes;

f) enviar as mensagens para seu destino.

Para a recepção, e interpretação e geração de mensagens SIP foi seguido o padrão determinado pelo IETF descrita na RFC 3261.

As funcionalidades implementadas no protótipo têm por objetivo receber uma mensagem de um cliente, interpretar a mensagem recebida, gerando uma mensagem e enviando para um destinatário ou gerando um registro no banco de dados.

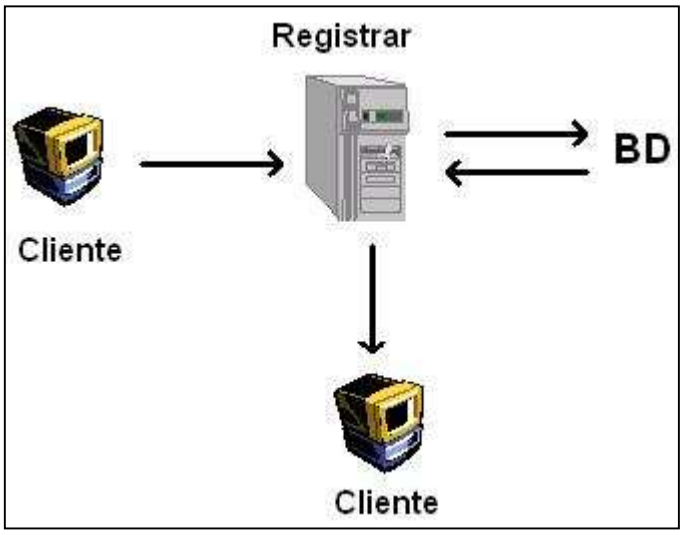
4.2 ESPECIFICAÇÃO

O desenvolvimento do software especifica as funcionalidades pré-definidas para o protótipo. Na Figura 14 mostra-se o cliente registrando seu endereço no registrar.

Figura 14 – Registrar Cliente

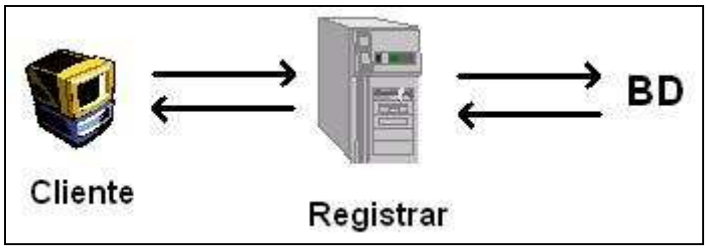
A Figura 15 mostra que após o Registrar receber uma mensagem ele faz uma consulta ao Banco de Dados buscando o endereço IP do destinatário e encaminha a mensagem para o mesmo.

Figura 15 – Encaminhamento da mensagem SIP



A Figura 16 demonstra a funcionalidade do redirecionamento de mensagens onde o cliente manda uma mensagem para o Registrar sendo que este, após fazer uma consulta no banco de dados, envia uma resposta para o cliente que o contatou indicando para qual endereço o cliente deve mandar a mensagem.

Figura 16 – Redirecionamento da mensagem



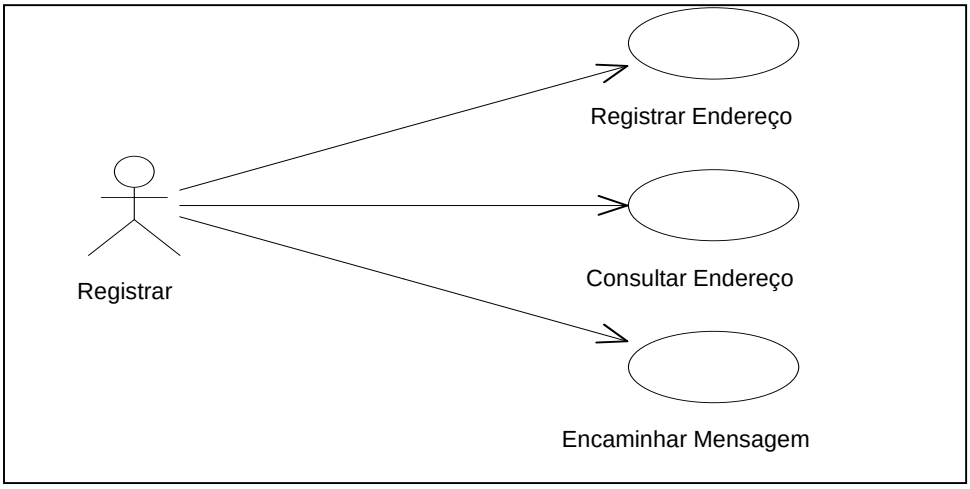
4.2.1 DIAGRAMAS DE CASO DE USO

Segundo Furlan (1998), a modelagem de casos de uso fornece situações de aplicação onde não é o objetivo entender como os casos de uso são implementados ou como é o funcionamento interno. Utiliza-se a modelagem de casos de uso para capturar necessidades de um novo sistema como também para desenvolver novas versões de um sistema.

Os casos de uso têm como propósito descrever os requerimentos funcionais do sistema de maneira consensual entre usuários e desenvolvedores de sistemas como fornecer uma descrição consistente e clara sobre as responsabilidades que devem ser cumpridas pelo sistema (FURLAN, 1998).

A Figura 17 representa os casos de uso que fazem parte do protótipo onde definem-se as funções do sistema. Os diagramas de casos de uso foram especificados utilizando a ferramenta Rational Rose.

Figura 17 – Diagrama de Caso de Uso



O Quadro 1 mostra a descrição de cada caso de uso com o autor e nome dos casos.

Quadro 1 – Descrição dos casos de uso

Caso de Uso	Ator	Descrição
Registrar endereço	Registrar	Quando o servidor recebe uma mensagem de um cliente da rede para ser registrado ele extrai o nome, domínio e endereço IP do cliente registrando-o no banco de dados. Caso o usuário deseje ser contatado em outro local ele deve manda uma mensagem

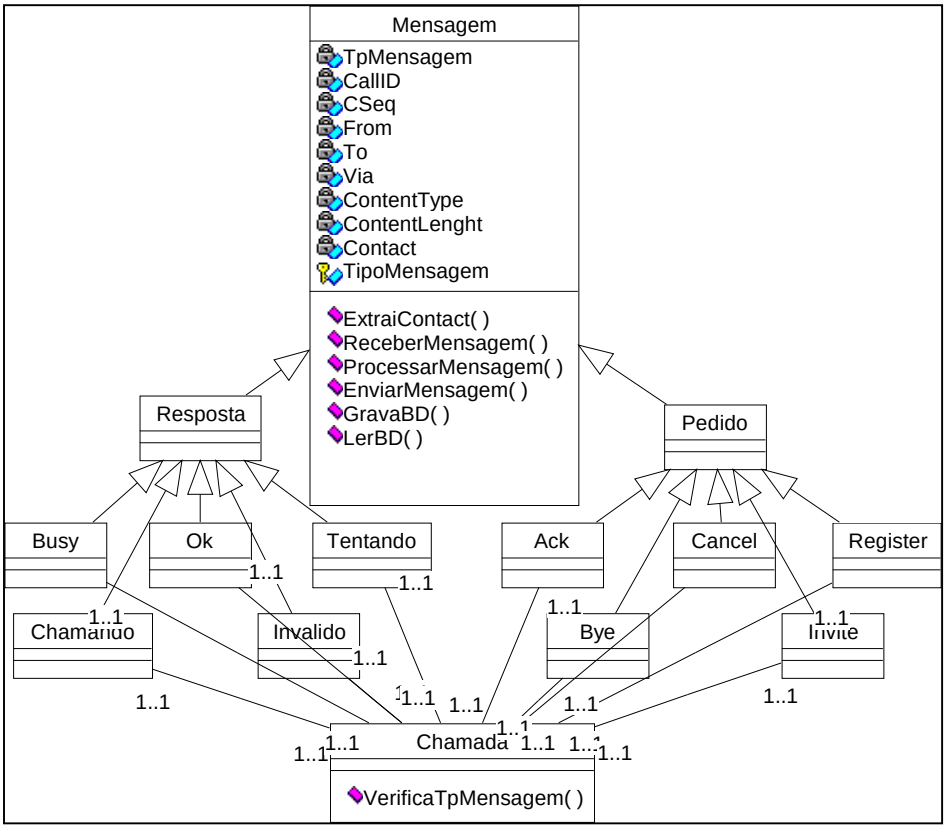
		contendo o domínio e endereço IP.
Consultar endereço	Registrar	Ao chegar uma mensagem no servidor para um determinado destinatário, o servidor faz uma consulta ao banco de dados verificando qual o IP do usuário de destino e mandando a mensagem para o mesmo.
Encaminhar Mensagem	Registrar	Quando o cliente mandar uma mensagem para o servidor e o servidor verificar que o usuário está em outro lugar, o servidor gera uma resposta para o cliente mandando o endereço que o usuário pode ser contatado.

4.2.2 DIAGRAMA DE CLASSES

Segundo Furlan (1998), o diagrama de classes é representado por uma estrutura lógica e estática mostrando uma coleção de elementos declarativos de modelo, como classe, tipos e seus respectivos conteúdos e relações. Estruturar atributos e operações em classes é fundamental para o trabalho de modelagem através do enfoque da orientação a objeto.

Na Figura 18 pode-se observar através da representação no diagrama de classes a estrutura de classes do sistema, bem como seus eventos e atributos.

Figura 18 – Diagrama de Classes



Para o caso de uso Registrar Endereço mostrado na Figura 17 é representado no diagrama de classes da seguinte forma: o método da classe Chamada verifica qual é o tipo de mensagem e cria um objeto para o tipo de mensagem chamando o método ReceberMensagem da classe Mensagem sendo que este chama o método ProcessarMensagem que verifica que o tipo da mensagem é de registro chamando a função GravaBD que irá grava o nome do usuário, domínio, endereço IP e se a mensagem é de redirecionamento.

O caso de uso Consultar Endereço será efetuado no método ProcessarMensagem quando é verificado que a mensagem é para ser encaminhada para outro usuário é chamado a função LerBD que verifica o domínio do usuário e o endereço IP.

O caso de uso Encaminhar Mensagem será executado quando a mensagem for de redirecionamento após ter verificado o endereço do destinatário é chamado o método Enviar Mensagem que monta o buffer e encaminha para o destino.

4.3 IMPLEMENTAÇÃO DO PROTÓTIPO DE SOFTWARE

O protótipo do software foi desenvolvido utilizando-se a ferramenta de desenvolvimento Borland C++ Builder 5 e utilizando seus componentes para envio e receptação de mensagens e a interação com o banco de dados para armazenar e ler os registros gravados em uma tabela.

Para o envio e recebimento de mensagens foi utilizado componente TNMUDP disponível na ferramenta C++ Buider 5.

No tratamento da mensagem que chega no buffer foi criado uma Classe mensagem que possui um método VerificaTpMensagem onde verifica-se qual é o tipo da mensagem, para cada tipo de mensagem existe uma classe sendo que após ter verificado que tipo de mensagem chegou é criado um objeto da determinada classe.

As classes do tipo de mensagens são classes que herdam os métodos da classe base então após ter sido criado um objeto de uma classe é chamado o método ReceberMensagem da classe base chamada Mensagem que irá fazer um *parser* do buffer, ou seja, irá extrair do buffer os campos da mensagem.

Após ter extraído os campos o método ReceberMensagem irá chamar o método ProcessarMensagem onde para cada tipo de mensagem terá um processamento por exemplo: se chegar uma mensagem do tipo REGISTER esse método irá gravar no banco de dados o nome do cliente que é extraído do campo From, domínio do cliente que é extraído do campo Contact e o endereço do que é verificado quando se recebe uma mensagem.

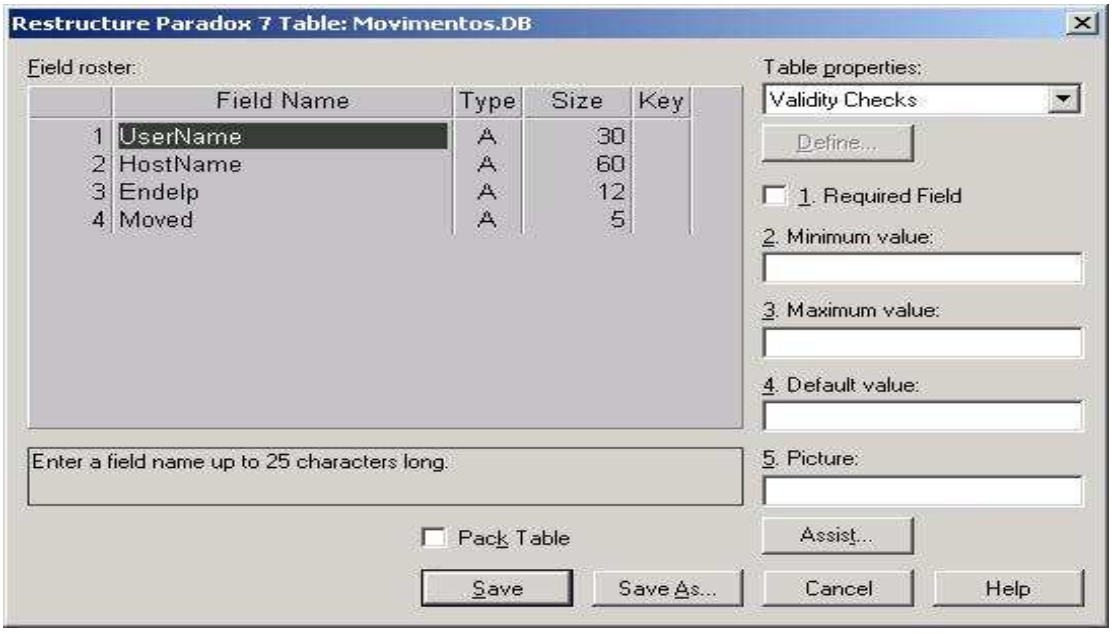
Se a mensagem for do tipo INVITE o método ProcessarMensagem irá chamar uma função que lê o banco de dados e verifica qual é o endereço IP e o domínio da pessoa de destino e chama o Método EnviarMensagem que irá montar a mensagem e mandar para o destino.

4.3.1 TABELA DE ENDEREÇOS SIP

Para armazenar os registros de uma forma consistente se fez uso de uma tabela que foi criada usando o banco de dados Paradox onde, quando uma mensagem de registro chegava de um cliente o sistema armazenava o nome do cliente, domínio onde se encontra ou onde ele deseja ser encontrado, endereço IP onde ele se encontra ou deseja ser encontrado e uma variável para armazenar um “flag” que indica se o registro é redirecionamento.

A Figura 19 demonstra a estrutura da tabela onde ficam armazenados os endereços dos usuários que fazem parte de uma rede de VoIP.

Figura 19 – Registros de Endereços SIP.



4.4 FUNCIONAMENTO DO PROTÓTIPO DE SOFTWARE

Como o protótipo é um software de suporte a uma rede de Voz sobre IP, para seu funcionamento necessita-se de dois ou mais Clientes na rede para que tenha interação com o Registrar, seja mandando mensagens de registro de endereço como também mandando mensagens para outros clientes da rede. O protótipo deve receber a mensagem, verificar o endereço do destino e mandar para o destino ou gerar uma resposta para o cliente.

Para que o software entre em funcionamento basta colocá-lo em execução quando aparecerá a tela mostrada na Figura 20.

Figura 20 – Tela do Software Registrar.

Estando com o programa em execução, ao receber uma mensagem, chamará uma função ReceiverData que cria um *buffer* onde armazena os dados recebidos. Após ter recebido os dados o programa cria um objeto do tipo Chamada passando o buffer, Endereço IP de origem e o componente UDP como mostrado na

Figura 21 – Recebendo um buffer de um cliente.

```
void __fastcall Tfrmregistrar::UdpDataReceived(TComponent *Sender,
      int NumberBytes, AnsiString FromIP, int Port)
{

        char cp[1024];
        int recLen;
        //Verifica se número de bytes recebido é maior que zero
        if (NumberBytes > 0)
        {
                //Recebe o buffer de mensagem
                Udp->ReadBuffer(cp,1024,recLen);
                //Cria um objeto do tipo chamada
                Chamada teste(AnsiString(cp),FromIP,Udp);
        }
}
```

Tendo criado o objeto Chamada, o programa passa para a função VerificaTpChamada onde essa função tem por objetivo extrair a primeira linha do *buffer* e verificar qual é o tipo da mensagem para posteriormente criar um objeto do tipo da mensagem que chegou como mostrado na Figura 22.

Figura 22 – Verifica tipo da mensagem

```
void __fastcall Chamada::VerificaTpMensagem(AnsiString buffer,AnsiString PIpSource,
                                            TNMUDP *PUdp)
{
    char *p = buffer.c_str();
    AnsiString TipoMensagem,buff;

    TipoMensagem = GetLine(1, buffer);     // tem a primeira linha
    while ((*p != '\n') && (*p != '\0'))  // pula a primeira linha
      p++;
    p++;
    buff = p;  // tem da segunda linha em diante
    //Verifica o tipo da mensagem e cria um objeto para cada tipo
    if(TipoMensagem.Pos("INVITE") != 0)
       Invite ChamInvite(TipoMensagem,buff,PIpSource,PUdp);
    else if(TipoMensagem.Pos("REGISTER") != 0)
       Register ChamRegister(TipoMensagem,buff,PIpSource,PUdp);
    else if(TipoMensagem.Pos("ACK") != 0)
       Ack ChamAck(TipoMensagem,buff,PIpSource,PUdp);
    else if(TipoMensagem.Pos("BYE") != 0)
       Bye ChamBye(TipoMensagem,buff,PIpSource,PUdp);
    else if(TipoMensagem.Pos("CANCEL") != 0)
       Cancel ChamCancel(TipoMensagem,buff,PIpSource,PUdp);
}
```

Criado o objeto do tipo da mensagem, o programa chama um método Receber Mensagem da classe base que é classe Mensagem onde é extraída da mensagem todos os respectivos campos como mostrado na Figura 23.

Figura 23 – Método receber mensagem.

```
void __fastcall Mensagem::ReceberMensagem(AnsiString TipoPedido,AnsiString buffer,
                                          AnsiString PIpSource,TNMUDP *PUdp)
{
    TpMensagem = TipoPedido;
    CallID = GetLine(1, buffer);
    Cseq = GetLine(2, buffer);
    Contact = GetLine(3, buffer);
    From = GetLine(4, buffer);
    To = GetLine(5, buffer);
    Via = GetLine(6, buffer);
    ContentType = GetLine(7, buffer);
    ContentLen = GetLine(8, buffer);

    ProcessarMensagem(PIpSource,PUdp);
}
```

Após ter inicializado os atributos da classe Mensagem o programa verifica qual objeto foi criado sendo que se foi um objeto Register o programa chama uma função onde registra no banco de dados o nome do cliente, domínio do cliente, endereço IP, e se o registro é do tipo redirecionamento, o qual observa-se na Figura 24.

Figura 24 – Gravando mensagem no banco de dados

```
void __fastcall Mensagem::GravarBD(TNMUDP *Udp,AnsiString PIpSour)
{
        Tab->Open(); //Abre a tabela
        Tab->First();//Vai para primeiro registro
        bool found = false;
        while (!found && !Tab->Eof){//Procura um usuário
          found = (Tab->FieldByName("UserName")->AsString == tempCon);
          if (!found)
            Tab->Next();
        }
        if (!found){ //Se não achou insere o usuário
            Tab->Insert();
            Tab->FieldByName("UserName")->AsString = tempCon;
        }
        else       //se achou atualiza os dados
            Tab->Edit();
        Tab->FieldByName("HostName")->AsString = ExtraiContact();
        if (PIpSour == tempVia){
            Tab->FieldByName("EndeIp")->AsString = PIpSour;
            Tab->FieldByName("Moved")->AsString = "";
        }
        Tab->Post();//Grava na tabela
}
```

Caso a mensagem recebida seja do tipo INVITE o programa irá executar uma função para buscar o endereço IP do cliente destino como mostra a Figura 25. Após ter encontrado o endereço IP do destino, a mensagem é montada novamente e mandada para o mesmo podendo ser visto na Figura 26, caso em que o programa busca o endereço IP se o campo da tabela “Moved” estiver como “true”, significando que o programa deve gerar uma resposta para o cliente que enviou a requisição, passando no cabeçalho Via o endereço IP, onde o cliente pode ser contatado e no cabeçalho Contact, em qual domínio deve ser contatado, para que o cliente possa enviar a mensagem diretamente para o destinatário.

Figura 25 – Ler registro do banco de dados.

```
AnsiString __fastcall Mensagem::LerBD(TNMUDP *Udp,AnsiString PIpSour)
{
   Que->Open();
   switch (Flag){
    case tipoInvite:if (Que->FieldByName("Moved")->AsString != "True"){
                        TpMensagem = "INVITE "+tempCon+"@"+Que->FieldByName("HostName")->A
                        To = "To: "+tempCon+"@"+Que->FieldByName("HostName")->AsString+"\r
                        Via = "Via: SIP/2.0/UDP "+Que->FieldByName("EndeIp")->AsString;+"\
                        retorno = Que->FieldByName("EndeIp")->AsString;}
                    else {
                        TpMensagem = "SIP/2.0 302 Movido temporariamente\n";
                        To = "To: "+tempCon+"@"+Que->FieldByName("HostName")->AsString+"\r
                        Via = "Via: SIP/2.0/UDP "+Que->FieldByName("HostName")->AsString+"
                        retorno = PIpSour; }
                    break;
    case tipoAck: TpMensagem = "ACK "+tempCon+"@"+Que->FieldByName("HostName")->AsString+"
                    retorno = Que->FieldByName("EndeIp")->AsString;
                    break;
    case tipoBye: TpMensagem = "BYE "+tempCon+"@"+Que->FieldByName("HostName")->AsString+"
                    retorno = Que->FieldByName("EndeIp")->AsString;
                    break;
}   }
```

Figura 26 – Enviar Mensagem

```
void __fastcall Mensagem::EnviarMensagem(AnsiString PIpSour,TNMUDP *PUdp)
{
    AnsiString Buffer = TpMensagem+CallID+Cseq+Contact+From+To+Via+ContentType+ContentL

    PUdp->LocalPort = 5060;
    PUdp->RemoteHost = PIpSour;
    PUdp->RemotePort = 5060;
    PUdp->SendBuffer(Buffer.c_str(),Buffer.Length()+1,Buffer.Length()+1);
}
```

4.5 TESTES E VALIDAÇÃO

Os testes realizados no Registrar foram executados com o suporte de um software para envio e recebimento de mensagens que segue o padrão de clientes SIP.

Neste software o usuário pode se registrar, quando será enviado ao servidor uma mensagem REGISTER, informando no campo *Contact* qual o domínio que deseja ser contatado. Se o cliente desejar ser contatado em outro local será informado na mensagem no campo *Via* o endereço IP onde ele deseja ser contato.

Quando o cliente efetuar uma ligação deve escolher o destinatário que será chamado quando será enviado ao Registrar a mensagem. Caso receba de volta uma mensagem *Moved* significa que o cliente que deseja contatar está em outro lugar: então nesta mensagem deve-se pegar o endereço IP, que está no campo *Via*, do novo local e o domínio, que está no campo *Contact*, gerando uma nova mensagem diretamente para o destino.

A Figura 27 mostra a tela do simulador onde pode ser escolhido o destinatário que deseja chamar.

Figura 27 – Tela do Simulador

Se o cliente que está chamando aceitar a conexão, o software receberá uma mensagem “200 OK” à qual deve-se responder para o cliente um ACK mostrando que entendeu a resposta.

Quando um cliente receber uma chamada de conexão, poderá escolher se aceita a conexão. Caso aceite, deve enviar uma mensagem “200 OK”; caso não aceite, deve enviar uma mensagem de “Ocupado”.

Todas as mensagens que um cliente envia para outro irão passar pelo Registrar para que as interprete e tome uma ação de acordo com o tipo de mensagem. Apenas quando o cliente receber uma mensagem do Registrar indicando que o destinatário está em outro lugar, o software cliente irá transmitir a mensagem diretamente para o destinatário.

5 CONCLUSÕES

No decorrer do desenvolvimento do trabalho confirmou-se a complexidade do tema Voz sobre IP onde seria necessário, para um perfeito entendimento do assunto um estudo completo sobre cada componente que envolve uma rede de Voz sobre IP.

O Registrar como um dos componentes que fazem parte de VoIP se mostrou bastante complexo. Pelo assunto VoIP ser bastante recente houve dificuldade em encontrar material de referência e especialmente quanto ao Registrar, a dificuldade foi maior. No entanto apesar de toda a dificuldade, o tema abordado se mostrou muito interessante a medida que foi sendo compreendido e desenvolvido.

No decorrer do desenvolvimento do trabalho verificaram vários pontos que foram importantes para a implementação e que contribuíram para um melhor entendimento do assunto que inicialmente dava um outro foco a proposta do trabalho.

Em relação aos objetivos propostos foram alcançados com sucesso. Como se optou em desenvolver um trabalho baseado no padrão de mensagens SIP, tornou-se relativamente simples de implementar, pois teve como maior preocupação manter o padrão das mensagens geradas pelo software dentro do padrão especificado pela IETF.

Na implementação outro ponto importante foi a própria programação dos eventos principais do protótipo onde teve que se utilizar componentes oferecidos pela ferramenta com o objetivo de comunicar-se com outro componente da rede e também em fazer a interação com o banco de dados.

Com relação às ferramentas utilizadas não houve dificuldade maior para a implementação.

O estudo e desenvolvimento do trabalho foram de grande importância devido a poucas referências sobre o tema, por ser uma tecnologia nova, citando o uso do protocolo SIP, mas que vem crescendo e acompanhando a evolução de própria Internet, já que as principais aplicações se voltam para este fim.

5.1 EXTENSÕES

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

a) implementar o padrão de mensagens usando a biblioteca oSIP;

b) integrar o protótipo a uma plataforma VoIP comercial baseada em SIP.

REFERÊNCIAS BIBLIOGRÁFICAS

ALENCAR, Marcelo Sampaio de. **Telefonia digital**. São Paulo: Érica, 1998.

ALVES,Victor Manuel Golçalves.**Apresentação e análise da *Ip Telephony***, Porto, out 2002. Disponível em: http://www.inescn.pt/~jneves/feup/mrsc-2001/sm/trabalhos.html. Acesso em: 02 jun. 2002.

FERNANDES, Nelson Luiz Leal.**Voz sobre Ip**:Uma visão geral,Rio de Janeiro, fevereiro,[2003]. Disponível em: <http://www.ravel.ufrj.br/publicacoes/tipos.php?IdTipo=4>. Acesso em: 01 jun 2003.

FURLAN, Jose Davi. **Modelagem de objetos através da UML-The Unifield Modeling Language**. São Paulo: Makron Books, 1998. 329 p.

HERSENT, Olivier; GURLE, David; PETIE, Jean Pierre. **Telefonia IP**. São Paulo: Addison, 2002.

OLIVEIRA, Sérgio. Telefonia IP para ambientes móveis usáveis: Simpósio Brasileiro de Redes de Computadores, 19., 2001, Florianópolis. **Anais...** Florianópolis: UFSC, 2001. p. 542-558.

SILVA, Arlindo Maia da; RAMOS, Luís.**Serviços de Multimídia**, Porto, fev 2002. Disponível em: Acesso em: 10 mar. 2002.

SOARES, Luiz Fernando G; LEMOS, Guido; COLHER, Sérgio. **Redes de computadores:** das LANs, MANs e WANs às redes ATM. 2. ed. Rio de Janeiro: Campus, 1995.

SOUZA, José Marcio de. Protótipo de um sitema de VoIP(Voz sobre IP). 2001. 62 f. Trabalho de Conclusão de Curso(Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.