Universidade de Trás-os-Montes e Alto Douro

Mestrado em Tecnologias das Engenharias

sIPtel - Um sistema de IPtel com suporte para vídeo utilizando o protocolo SIP

João Paulo Pereira de Sousa

UTAD, 2003

Universidade de Trás-os-Montes e Alto Douro

sIPtel - Um sistema de IPtel com suporte para vídeo utilizando o protocolo SIP

João Paulo Pereira de Sousa

Licenciado em Engenharia Electrotécnica, Ramo de Electrónica, Instrumentação e Computação pela Universidade de Trás-os-Montes e Alto Douro

Dissertação submetida para satisfação parcial dos
requisitos do grau de Mestre em Tecnologias da Engenharias
(Área de especialização de Comunicações)

Dissertação realizada sob a supervisão do

Prof. Doutor Eurico Manuel Elias de Morais Carrapatoso

Professor Auxiliar do
Departamento de Engenharia Electrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto

Aos meus pais

Resumo

Estudos recentes apontam que o tráfego de dados em breve excederá o tráfego telefónico, se tal já não tiver acontecido. Estes indicadores, juntamente com mais e melhores acessos à Internet, tornam cada vez maior o interesse em transportar voz e vídeo sobre redes de dados.

Neste contexto nasce a Telefonia sobre IP, que oferece através desta infra-estrutura a oportunidade de criar sistemas globais de comunicação multimédia. A redução de custos e a facilidade na implementação de serviços inovadores são argumentos que justificam a forte evolução da IPtel e a tendência eventual de substituir a rede telefónica analógica.

O *Session Initiation Protocol* (SIP), utilizado no desenvolvimento deste serviço, é um protocolo de sinalização e controlo de chamadas entre dois ou mais participantes, que tem ganho uma grande aceitação por parte de empresas ligadas ao mundo das comunicações. Desenvolvido com uma arquitectura normalizada e aberta pela *Internet Engineering Task Force* (IETF), espera-se que o SIP tenha o mesmo impacto no mundo das comunicações IP que o SMTP teve no *e-mail* e o HTTP na *Web*. O SIP anuncia ainda a convergência dos equipamentos e serviços de comunicações, permitindo integrar facilmente serviços de Internet como *Web*, *e-mail*, correio de voz, mensagens instantâneas, colaboração multimédia e presença (informação sobre se um utilizador está ou não disponível para comunicar).

Nesta dissertação é feito um estudo sobre a evolução das diferentes partes que integram o serviço IPtel. São ainda referidas as vantagens na criação de novos serviços e obstáculos a ultrapassar por esta tecnologia de modo a poderem consolidar-se no mercado das comunicações. São apresentados diversos protocolos tipicamente usados na arquitectura protocolar da IPtel e que serão estudados, para suportar a criação do serviço sIPtel.

É feita uma apresentação do serviço de Telefonia sobre IP e é explicada a arquitectura e o funcionamento do protocolo SIP, utilizado para o desenvolvimento da parte de sinalização do sIPtel. É ainda detalhado o desenvolvimento de um serviço que permite a criação, controlo e finalização de sessões de áudio e vídeo entre dois utilizadores através do protocolo SIP e por fim são realizados testes de modo a avaliar a capacidade de interoperabilidade do serviço implementado.

Palavras chave: Telefonia sobre IP, Protocolos, Sinalização, Codificadores de áudio, Codificadores de vídeo, Java.

Abstract

Recent studies point out that data traffic will soon exceed telephone traffic, which probably has already happened. These indicators, together with more and better accesses to the Internet, make it more important to carry transport voice and video on data networks.

In this context, Telephony over IP, which offers the opportunity to create global systems of multimedia communication through this unique infrastructure, is born. The trimming in costs and the easiness to implement innovative services are arguments that justify the strong evolution of IPtel and the growing tendency to substitute the analog telephone network.

The Session Initiation Protocol (SIP), used in the development of this service, is a protocol for creating, modifying and terminating calls between two or more participants, and has won great support from companies connected to the world of communications. Developed with normalized and open architecture by the Internet Engineering Task Force (IETF), it is expected that the SIP will have the same impact on the world of IP communications that SMTP had on e-mail and HTTP on the Web. Thus, SIP foresees the convergence of communication equipments and services, allowing the easy integration of Internet services such as Web, e-mail, voice mail, instant messaging, multimedia collaboration and presence (information whether a user is available to communicate or not).

In this dissertation the evolution of the different parts that integrate the IPtel service is studied and the advantages of creating new services, together with the obstacles that this technology must overcome so that it consolidates itself in the communications market are referred. Several protocols that are typically used in the architecture protocol of IPtel, which are studied to support the creation of the sIPtel service, are presented.

A presentation of the Telephony over IP service is made and the architecture and the operation of the SIP protocol, used in the development of the sIPtel signaling part, explained. The development of a service that allows the creation, control and finalization of audio and video sessions among two users through the SIP protocol is detailed. Finally tests are carried out in order to evaluate the interoperating capacity of the developed service.

Keywords: Telephony over IP, Protocols, Signaling, Audio Codec, Video Codec, Java.

Résumé

Des études récentes indiquent que le trafic de données dépassera à court terme le trafic téléphonique, si cela n´est pas déjà arrivé. Ces indicateurs associés avec plus et accès meilleurs a l´Internet augmentent l´envie de transporter la voix et le vidéo sur les réseaux de données.

Dans ce contexte apparaît la Téléphonie sur le IP, qui offre avec cette infrastructure l´opportunité de créer des systèmes globaux de communication multimédia. La réduction des dépenses et la facilitée de développer des services innovateurs sont des arguments qui justifient la forte évolution de IPtel et la tendance de substituer le réseaux téléphonique analogique.

Le *Session Initiation Protocol* (SIP), utilisé pour développer ce service, est un protocole de signalisation et contrôle d´appels entre deux participants ou plus, qui a gagné un grand support de la part des entreprises relationnées au monde des télécommunications. Le SIP a été développé avec une architecture normalisée et ouverte par *Internet Engenieering Task Force* (IETF), et on espère qu´il ait le même impact sur le monde des communication IP que le SMTP a eux pour le mail et le HTTP pour le *Web*. Le SIP annonce encore la convergence des équipements et des services de communication, et permet d´intégrer facilement les services Internet comme *Web*, mail, courrier de voix, messages instantanés, collaboration multimédia et présence (information sur si un utilisateur est disponible ou non).

Dans cette maîtrise, est réalisée une étude sur l´évolution des différentes parties qui intègrent le service IPtel, nous faisons référence aux avantages de la création de nouveaux services et obstacles a dépasser par cette technologie pour quelle puisse se consolider dans le marché des communications. On présente plusieurs protocoles utilisés couramment dans l´architecture des protocoles de IPtel qui sont le centre des études et la base pour la création de services sIPtel. On fait la présentation du service Téléphonie sur IP et on explique aussi l´architecture et le fonctionnement du protocole SIP, utilisé pour développer la partie de la signalisation du sIPtel. Un service qui permet la création, le contrôle et la finalisation de session audio et vidéo entre deux utilisateurs à travers le protocole SIP est en suit présenté. Finalement on réalise des tests pour pouvoir évaluer la capacité d'interopérabilité du service développé.

Mots-clé: Téléphonie sur IP, Protocoles, Signalisation, Codificateurs audio, Codificateurs vidéo, Java.

Agradecimentos

Expresso a minha sincera gratidão ao orientador, Prof. Doutor Eurico Carrapatoso, pela sua partilha de conhecimento e permanente disponibilidade durante a realização deste trabalho.

Agradeço ao José Oliveira pelo seu apoio e esclarecimentos durante o desenvolvimento deste serviço.

Ao Pedro João pelos seus conselhos preciosos, dados na iniciação deste trabalho.

Aos meus amigos com os quais convivi durante este último ano, pelos seus momentos de descontracção proporcionados.

Por fim, um eterno agradecimento aos meus Pais, à minha família e à Raquel a quem devo o maior suporte e incentivo para a conclusão deste trabalho.

Índice

Lista de figuras

Lista de tabelas

Lista de acrónimos

ADPCM	Adaptive Differential Pulse Code Modulation
ANTD	Advanced Networking Technologies Division
API	Application Programming Interface
ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
ATM	Asynchronous Transfer Mode
AVP	Audio/Video Profile
CD	Compact Disk
CIF	Common Interchange Format
CLNP	Connection Less Network Protocol
CNAME	Canonical Name
CVSD	Continuously-Variable Slope Differential
DHCP	Dynamic Host Configuration Protocol
Diff-Serv	Differentiated Services
DNS	Domain Name System
DVI	Digital Voice Incorporated
FEC	Forward Error Correction
GSM	Global Systems for Mobile Communications
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IIS	Internet Integrated Services
IMA	Interactive Multimedia Association
IP	Internet Protocol

IPSEC	Internet Protocol Security
IPtel	IP Telephony
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IPX	Internetwork Packet Exchange
ISDN	Integrated Services Digital Network
ISP	Internet Service Provider
ITU	International Telecommunication Union
JMF	Java Media Framework
JPEG	Join Photographic Experts Groups
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LTP	Long-Term Predictor
MADCAP	Multicast Address Dynamic Client Allocation Protocol
Mbone	Multicast BackBone on the Internet
MC	Multipoint Controller
MCU	Multipoint Control Unit
MEGACO	Media Gateway Control
MGCP	Media Gateway Control Protocol
MIME	Multipurpose Internet Mail Extensions
MMUSIC	Multiparty Multimedia Session Control
MOS	Mean Opinion Score
MP	Multipoint Processor
MPEG	Moving Pictures Experts Groups
NIST	National Institute of Standards and Technology
NTV	Network Voice Protocol
OSI	Open Systems Interconnection
PBX	Private Branch Exchange
PPP	Point-to-Point Protocol
PSTN	Public Switched Telephone Network
PT	Payload Type
PVP	Packet Video Protocol
QCIF	Quarter Common Interchange format

QoS	Quality of Service
RAS	Registration Admission and Status
RFC	Request for Comments
RNAP	Resource Negotiation and Pricing Protocol
RPE	Regular Pulse Excitation
RR	Receivers Reports
RSTP	Real-Time *Stream* Protocol
RSVP	Resource Reservation Protocol
RTCP	Real Time Control Protocol
RTP	Real-Time Transport Protocol
SAP	Session Announcement Protocol
SCTP	*Stream* Control Transmission Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SMTP	Simple Mail Transfer Protocol
SR	Sender Reports
SRP	Scable Reservation Protocol
SSRC	Synchronization source
TCP	Transmission Control Protocol
Tftp	Trivial File Transfer Protocol
TIA	Telecommunication Industries Association
TLS	Transport Layer Security
ToS	Type of Service
TRIP	Telephony Routing over IP
TTL	Time To Live
TU	Transaction User
UA	User Agent
UAC	User Agent Client
UAS	User Agent Server
UDP	User Datagram Protocol
UML	Unified Modeling Language
URI	Universal Resource Identifier
URL	Uniform Resource Locator

USFS	United States Federal Standards
VoIP	Voice over IP
Web	World Wide Web
YESSIR	Yet another Sender Session Internet Reservations
TPC	Technical Program Committee

Capítulo 1

Introdução

Nos últimos anos diversas investigações têm sido realizadas na área das aplicações multimédia na Internet, as quais permitem a transmissão de áudio, vídeo e dados. Este tema tem despertado o interesse de profissionais da área de redes de computadores e dos utilizadores em geral, justificado pela promessa de serviços de grande utilidade como telefonia, videoconferência e ensino à distância, entre outros.

Com o emergir da Internet, verifica-se uma expansão na utilização da ligação do computador à rede para explorar a convergência de serviços como o áudio e vídeo juntamente com o tráfego de dados. Estes serviços têm como objectivo remover barreiras de tempo e distância e poupar dinheiro, bem como aumentar a eficiência na utilização dos recursos.

Durante anos, companhias e organizações utilizaram um conjunto limitado de serviços de comunicação, como o telefone e o fax, suportados pela rede telefónica tradicional e uma rede comutada de pacotes para o transporte de dados. Actualmente têm a oportunidade de utilizar a rede IP como única infra-estrutura para as comunicações, permitindo a integração de novos serviços que possibilitam novas formas de comunicação e de condução de negócios, e a redução de custos, tornando-as mais competitivas.

A convergência de serviços tem sido tentada por várias vezes na história das comunicações, mas apenas a Internet oferece a primeira expectativa real na unificação de todos os serviços de comunicações numa única rede e num único sistema terminal [Schulzrinne, 2000].

Ao contrário da telefonia tradicional, as comunicações IP permitem para além do transporte de voz, a integração de vídeo, dados e de novos serviços como *chat*, mensagens instantâneas e *Web*, numa única infra-estrutura e num único serviço, o IPtel. Um dos exemplos mais comuns da integração da *Web* com as chamadas de voz e/ou vídeo é o serviço "*click to dial*", em que por exemplo, é iniciada uma chamada telefónica para o serviço de apoio ao cliente durante

uma compra de um produto num site electrónico, para o esclarecimento de uma dúvida.

Actualmente existem dois protocolos para as comunicações em tempo real, o SIP da *Internet Engineering Task Force* (IETF) e o H.323 da *International Telecommunications Union* (ITU). Estes dois protocolos são utilizados para encaminhamento, sinalização e controlo de chamadas e outros serviços suplementares. O H.323 é um protocolo já estabelecido e largamente utilizado devido principalmente a ter dado provas da sua capacidade e à interoperabilidade com a rede telefónica pública comutada. O SIP é um protocolo recente que promete escalabilidade, flexibilidade e facilidade na criação de serviços. Embora não sejam interoperáveis estes dois protocolos têm seguido uma tendência de aperfeiçoamento do seu funcionamento, aprendendo um com o outro e aumentando as suas semelhanças cada vez que é publicada uma nova versão.

O protocolo SIP é caracterizado por ser um protocolo simples, flexível e escalável, integrando-se facilmente em aplicações de Internet, devido à semelhança com protocolos da *Web* e correio electrónico. Actualmente existe uma grande discussão sobre qual das aproximações irá ter mais popularidade no futuro, parecendo existir uma maior vantagem para o SIP.

1.1 Objectivos

O objectivo deste trabalho era o desenvolvimento de um serviço independente da plataforma, que disponibilizasse a comunicação em tempo real de áudio e vídeo, utilizando o protocolo de sinalização SIP para o estabelecimento e controlo das sessões dos vários meios de informação. Para atingir o objectivo proposto iria ser necessário realizar um conjunto de tarefas auxiliares:

- Fazer um estudo geral desta tecnologia;
- Analisar protocolos da família IP utilizados no serviço IPtel;
- Analisar serviços disponibilizados nesta tecnologia e como implementar esses serviços;
- Estudar e definir componentes necessários para o desenvolvimento do sIPtel.

Após a criação do sIPtel previa-se ainda a realização de vários testes ao nível da interoperabilidade, os quais incluiriam também a execução de ensaios com outras aplicações

SIP.

1.2 Estrutura da Dissertação

Esta dissertação está estruturada em seis capítulos, incluindo esta introdução, no qual foi exposto o contexto deste trabalho e os principais objectivos que encaminharam o trabalho desenvolvido.

No segundo capítulo é apresentada a evolução histórica da Telefonia sobre IP e das diversas partes que a compõem. De seguida é analisada a sua arquitectura e os seus componentes, assim como as vantagens e obstáculos a ultrapassar em relação à mais directa concorrente, a telefonia tradicional. É ainda descrita a arquitectura protocolar da IPtel, sendo abordados os protocolos de sinalização, codificadores de áudio e vídeo e o protocolo RTP para o transporte destes dois tipos de meios. Finalmente são discutidas soluções que permitem satisfazer requisitos de qualidade de serviço ao nível do transporte de meios em tempo real na rede IP.

No terceiro capítulo é feita a especificação do serviço telefonia sobre IP a desenvolver. Inicialmente são considerados os requisitos necessários ao nível da sinalização e dos meios. Em seguida é feito um estudo detalhado do protocolo de sinalização SIP e da codificação dos meios, que regeram o desenvolvimento do sIPtel.

O quarto capítulo abrange todo o processo de implementação do serviço. São apresentadas as funcionalidades disponibilizadas pelo sIPtel e, posteriormente, efectuada uma descrição da arquitectura do sistema ao nível do desenvolvimento, funcionamento e composição.

No quinto capítulo são apresentados, testes realizados de modo a avaliar o nível de interoperabilidade através de critérios de avaliação definidos pelo *Technical Program Committee* (TPC) e também ensaios de comunicação com outros *softwares* SIP que foram efectuados.

Finalmente no sexto capítulo são apresentadas as conclusões relativas aos objectivos propostos e discutidas perspectivas de trabalho futuro.

Capítulo 2

Telefonia sobre IP

O objectivo deste capítulo é apresentar as funcionalidades básicas de um serviço IPtel. Este capítulo está dividido em sete partes. A primeira parte faz uma breve alusão à evolução das diferentes partes que integram o serviço IPtel. A segunda parte faz uma abordagem à arquitectura IPtel, referenciando as principais diferenças com a arquitectura encontrada na telefonia tradicional. A terceira parte descreve os principais componentes que constituem o núcleo de um serviço IPtel. A quarta e quinta parte descrevem respectivamente as vantagens e desvantagens desta tecnologia, confrontando-a diversas vezes com o serviço de telefonia tradicional. A sexta parte enuncia um conjunto de protocolos que permitem proporcionar diferentes características da tecnologia IPtel. Por fim é abordada a qualidade de serviço, um aspecto de elevada relevância para o sucesso desta tecnologia.

2.1 História da IPtel

A necessidade de reduzir os custos por parte das empresas e organizações de comunicação tem sido o grande impulsionador para o desenvolvimento da IPtel. A convergência de transportar voz, vídeo e dados sobre a mesma infra-estrutura oferece uma oportunidade para a redução dos custos de comunicação e aparecimento de novos serviços.

O serviço telefónico actual é em grande parte disponibilizado pela Rede Telefónica Pública Comutada (*Public Switched Telephone Network* - PSTN). Esta rede foi projectada para a comunicação em tempo real de voz síncrona com Qualidade de Serviço (*Quality of Service* - QoS) garantida. Quando uma chamada telefónica é iniciada é estabelecido um circuito reservado *full-duplex* restrito a dois interlocutores. Assim que a chamada é finalizada esse circuito reservado é destruído e a linha fica novamente livre para outras comunicações.

Os esforços para transportar áudio em redes de pacotes iniciam-se na década de 70 por Danny Cohen. Este relata uma experiência de transmissão de voz em pacotes e em tempo real entre o USC/ISI (*University of Southern California/Information Sciences Institute*) e o *MIT's Lincoln Lab* [Schulzrinne, 2002a]. As amostras de áudio eram comprimidas utilizando o codificador *Continuously-Variable Slope Differential* (CVSD) e o transporte dos pacotes de áudio era feito com o protocolo *Network Voice Protocol* (NTV), o primeiro protocolo de Internet para transportar voz em pacotes especificado formalmente em 1977 por Danny Cohen [RFC 741, 1977]. O seu trabalho continuou com o melhoramento da qualidade oferecida pela rede de comutação de pacotes em comparação com as redes de comutação de circuitos no que diz respeito a problemas de entrega assíncrona, altas taxas de perda de pacotes, latências elevadas e *jitter* (variação entre o tempo em que o pacote é esperado e o tempo em que é recebido, isto é, o pacote foi recebido antes ou depois do esperado) [Rosenberg, 2001]. Em 1992, a *Internet Engineering Task Force* (IETF) realiza a primeira *audiocast* através da *Multicast Backbone on the Internet* (MBone), a partir de San Diego [Casner, 1992]. Em 1992, Henning Schulzrinne começa a desenvolver o *Real-Time Transport Protocol* (RTP), de modo a normalizar uma camada de transporte para meios em tempo real, sendo este protocolo publicado em 1995 como *IETF Proposed Standard*.

Depois de transportar dados e áudio em pacotes, apenas faltava o vídeo para completar o transporte dos três elementos essenciais para um ambiente de conferência multimédia em redes de comutação de pacotes. É R. Cole que em 1981 propõe o *Packet Video Protocol* (PVP), um protocolo para o transporte de vídeo em pacotes. Em 1992, após o IETF ter realizado a primeira difusão de áudio, é feita a partir de Boston através da Mbone a primeira difusão de áudio e vídeo simultaneamente, utilizando as aplicações *vat* e DVC respectivamente [Umair, 2002].

Em 1995, Steve McCanne e Van Jacobson [McCane, 1995] desenvolveram a *vic*, uma aplicação que utiliza o codificador normalizado H.261. Ainda nesse ano foi surgiu outra aplicação, o *CU-SeeMe* [Dorcey, 1995], que foi dos primeiros protótipos de videoconferência disponíveis na Internet. Inicialmente para *MacOs* e depois para *Windows*, este protótipo utilizava um processo responsável pela distribuição de sinais pelos vários intervenientes da conferência.

Em 1996, é publicada pela *International Telecommunications Union* (ITU) a primeira versão da recomendação H.323 [H.323, 1996]. Inicialmente projectada para as LANs e dedicada à

realização de conferências, a H.323 é uma recomendação para a comunicação de áudio, vídeo e dados. Esta recomendação tem como objectivo a definição de protocolos ou a utilização de protocolos já existentes e a definição de procedimentos para as comunicações multimédia. Nesse ano é também prestado pela *Delta Three* o primeiro serviço comercial de Telefonia sobre IP, seguindo-se a *Net2phone*, *iBasis* e *Telematrix* [Schulzrinne, 2000].

Ainda em 1996 a *Microsoft* lança o seu primeiro sistema de conferência sobre redes de pacotes. O *Microsoft NetMeeting v1.0*, inicialmente sem vídeo, que foi incorporado meses mais tarde na versão v2.0b2, utilizava os protocolos recentes T.120 para a conferência de dados e H.323 para videoconferência, ambos da ITU.

Em Fevereiro de 1999, o protocolo SIP foi aceite como norma, pelo IETF como um protocolo de sinalização para a criação, modificação e finalização de sessões com um ou mais participantes.

Nos últimos anos, com o estabelecimento da Internet, as primeiras conferências empresariais marcam a transição da utilização de redes de pacotes para o tráfego de voz como experiências de laboratório, para o mundo dos serviços empresariais. A Telefonia sobre IP (*IPtel – IP Telephony*) é também designada como Voz sobre IP (*VoIP – Voice over IP*) ou ainda Telefonia sobre Internet (*Internet Telephony*). A Telefonia sobre IP definida como a comunicação multimédia entre dois ou mais participantes, requer uma parte de sinalização de modo a que um dos intervenientes que deseja comunicar encontre o outro e o avise da sua intenção. Esta funcionalidade é referida como Sinalização da IPtel. A necessidade de sinalização distingue a IPtel do *stream* multimédia principalmente no controlo e estabelecimento das sessões, introduzindo um conjunto de características ao nível da sinalização (como por exemplo o encaminhamento e localização de utilizadores), inexistentes em serviços da Internet como *broadcast* e *streaming* a pedido. Schulzrinne e Rosenberg [1998] definem a sinalização na IPtel como a criação, o controlo e a finalização de chamadas, entendendo por chamada uma associação entre aplicações que pode ser activada e desactivada. Exemplos de uma chamada podem ser uma sessão telefónica entre duas partes, uma conferência multimédia ou um jogo multi-utilizador. O uso comum do termo telefonia IP não deve ser entendido somente como transporte de voz, mas também como transporte de outros tipos de meios como vídeo e dados.

Este novo serviço permite a troca de pacotes entre dois ou mais intervenientes através da rede, utilizando protocolos da Internet e o intercâmbio da informação necessária para controlar essa

troca. No chamador a voz é capturada por um microfone e o vídeo é obtido por uma câmara de vídeo sendo este sinais geralmente digitalizados. Em seguida são codificados e encapsulados em pacotes que são enviados através da rede com a utilização de protocolos de Internet. Do outro lado, esses pacotes são desencapsulados e descodificados, o sinal digital é convertido em sinal analógico e reproduzido em altifalantes enquanto o vídeo é enviado para o ecrã.

Nos dias de hoje, devido ao crescente desempenho dos computadores pessoais e à rápida expansão da Internet, tem-se assistido a um enorme crescimento no número de aplicações que podem transmitir e receber multimédia através de um PC ligado à Internet. É então necessário garantir a interoperabilidade entre as aplicações; questões como protocolos e arquitecturas têm claramente um papel de grande relevância nesta tecnologia.

Embora a IPtel esteja em implantação no mercado das comunicações, a ITU apresentou em 2001 valores que enunciam um forte crescimento do tráfego mundial gerado por este serviço (Figura 2.1), justificando assim a sua importância nas comunicações.

Diversos factores têm contribuído para o desenvolvimento da IPtel, um dos principais e talvez o maior é o factor económico. A actual corrida a esta tecnologia e a competição entre empresas proporcionam um ambiente favorável ao aparecimento de novos serviços, ao rápido desenvolvimento de aplicações e a uma pressão na redução de custos na utilização da IPtel.

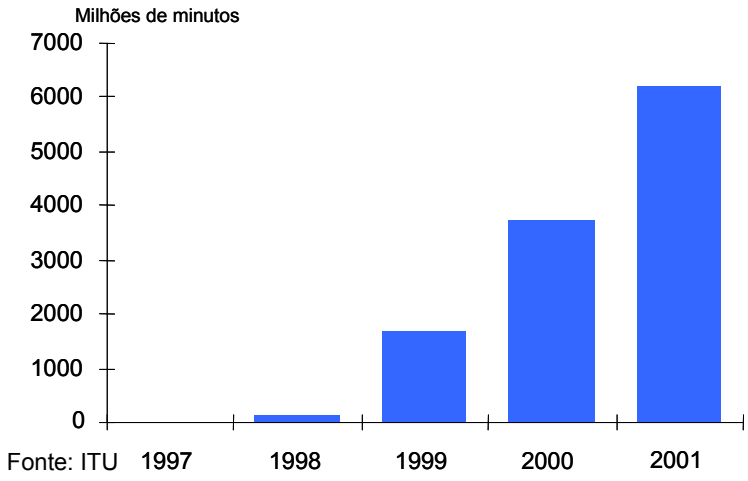


Figura 2.1 – Tráfego de Telefonia sobre IP

Além da sua principal característica, a comunicação por voz, a IPtel possibilita a convergência de serviços como vídeo, fax, mensagens instantâneas, *chat*, presença entre outros, bem como a utilização de múltiplos dispositivos (telefones IP, PCs, Palmtops e telemóveis). Entenda-se

por presença a capacidade de um utilizador poder informar a sua disponibilidade para comunicar com outras pessoas. Este serviço pode ser encontrado em *softwares* de alguma notoriedade como o ICQ ou *Windows Messenger*.

2.2 Arquitectura da IPtel

A principal diferença entre a IPtel e a rede telefónica tradicional (PSTN) manifesta-se na arquitectura de comutação: enquanto a PSTN é uma rede de comutação de circuitos, a rede IP é uma rede de comutação de pacotes. Esta particularidade permite que numa rede IP dois dispositivos troquem diferentes tipos de informação sem necessitarem de estar directamente conectados e sem reserva de recursos, sendo características de localização e encaminhamento da responsabilidade dos protocolos. O mesmo não se passa numa rede PSTN, onde é estabelecido um circuito físico entre dois dispositivos, reservando um canal *full-duplex* para cada sessão de conversação, independentemente da existência ou não de tráfego de voz.

A Figura 2.2 mostra os três tipos de dispositivos que Lennox e Schulzrinne [Lennox, 2000] identificaram numa rede IPtel: terminais, *gateways* (permitem interligar duas redes que não usem a mesma tecnologia de comunicação) e servidores de sinalização. Os terminais permitem executar os serviços, como por exemplo fazer e receber chamadas. Estes dispositivos terminais na rede IP são entendidos como dispositivos inteligentes possuindo total controlo sobre o estado da chamada, ao contrário dos telefones tradicionais que apenas reagem a comandos de uma central controladora, reflectindo uma arquitectura mestre-escravo.

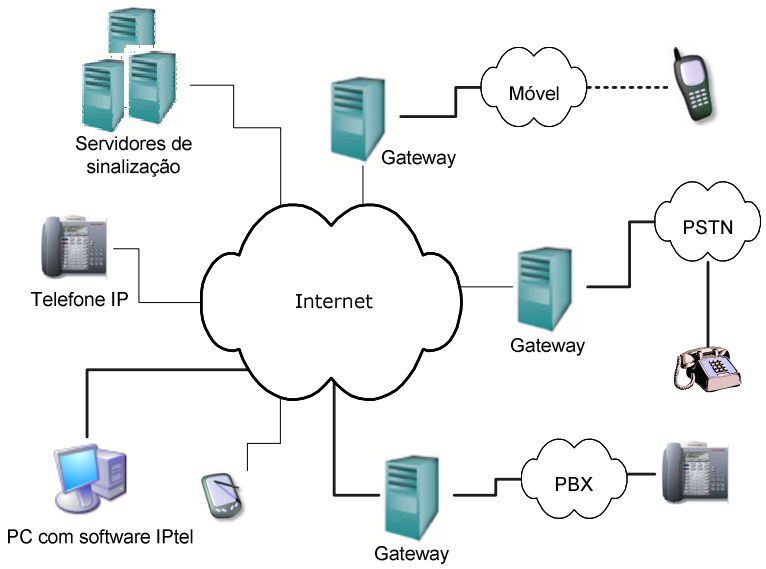


Figura 2.2 – Componentes de uma arquitectura IPtel

São exemplos de dispositivos terminais, computadores com *software* IPtel, telefones IP e faxes IP, entre outros. Na Tabela 2.1 [Schulzrinne, 2000], encontram-se algumas das funcionalidades de um telefone IP.

Função	Requisito	Protocolo
Nível de rede	Obrigatório	IPv4 ou IPv6
Controlo IP	Obrigatório	ICMP
Subscrição *multicast*	Opcional	IGMP
Tradução de endereço	Obrigatório	ARP
Auto-configuração	Opcional	DHCP
Transporte	Obrigatório	UDP, TCP
Tradução do nome	Opcional	DNS
Transporte de meios	Obrigatório	RTP
Sinalização	Obrigatório	SIP, H.323
Reserva de recursos	Opcional	RSVP, Diff-Serv
Serviço de directórios	Opcional	LDAP, whois
Actualização de *software* dinamicamente	Opcional	Tftp
Alocação de endereços *multicast*	Opcional	MADCAP
Anúncio de sessão	Opcional	SAP

Tabela 2.1 – Protocolos utilizados por um telefone IP

As *gateways* são dispositivos opcionais numa rede IPtel, sendo apenas usadas quando existe necessidade de interligar duas redes que não usem a mesma tecnologia de comunicação. Providenciam mecanismos para a interligação entre redes telefónicas diferentes, permitindo que utilizadores de diferentes tecnologias possam comunicar entre si de um modo transparente. Para os outros dispositivos da telefonia IP, as *gateways* não são muito diferentes

dos terminais pois tais como estes, iniciam e respondem à sinalização assim como recebem e transmitem meios. As *gateways* têm então a função de tradução entre terminais com diferentes formatos de transmissão, localização ou procedimentos de comunicação, e codificação de meios.

Os servidores de sinalização funcionam ao nível da aplicação, controlando o encaminhamento das mensagens de sinalização. Estes servidores disponibilizam serviços de localização do utilizador, mantendo informação sobre onde pode ser encontrado de modo a encaminhar os pedidos de sinalização para a localização actual do utilizador. São também responsáveis pelos serviços de tarifação e controlo de admissão, nos casos de ser necessária autorização para utilização do sistema.

2.3 Componentes de um serviço IPtel

O serviço IPtel deve ter em consideração vários aspectos: como o transporte de informação multimédia; a sinalização para o estabelecimento e controlo de chamadas; a interoperabilidade com outras redes; a qualidade de serviço; e a utilização em grande escala. Rosenberg identificou pelo menos cinco componentes que constituem o núcleo do serviço da IPtel e são necessários para a sua implementação [Rosenberg, 2001].

- Transporte: é responsável pelo transporte de multimédia entre dois pontos numa rede IP. Cuida da resolução de problemas inerentes ao processo de transporte, recuperação relativamente à congestão e perdas de pacotes, minimização do *jitter* e do atraso de pacotes. O transporte em tempo real de áudio e vídeo ponto a ponto é feito através do protocolo RTP [RFC 1889, 1996];
- Controlo de transporte: é responsável pela administração e controlo do procedimento do protocolo de transporte. Recorrendo ao protocolo RTP para o transporte dos meios, o controlo de transporte é efectuado com o protocolo *Real Time Control Protocol* (RTCP) [RFC 1889, 1996]. Este último permite o *feedback* ao emissor de parâmetros como *jitter*, atraso e perda de pacotes;
- Sinalização: é uma das características retiradas da telefonia tradicional, sendo responsável pelo estabelecimento, controlo e finalização de chamadas multimédia;
- Aplicações: são responsáveis por implementar características da IPtel como a

sinalização e pelas aplicações dos utilizadores. A sinalização é feita recorrendo aos protocolos SIP e H.323. Um exemplo é a aplicação desenvolvida para esta dissertação que utiliza o protocolo SIP. Algumas das características disponibilizadas pelas aplicações são: chamadas em espera, encaminhamento, transferência de chamada, conferência e outras;

- Descoberta de recursos: é responsável pela descoberta de servidores na rede, tais como *gateways*, servidores de sinalização e terminais. Este componente recorre a protocolos de descoberta como o *Domain Name System* (DNS) [RFC 1034, 1987], [RFC 1035, 1987], o *X.500*, o *whois++* [RFC 1835, 1995], [RFC 1913, 1996] e o TRIP – *Telephony Routing over IP* [Rosenberg, 2000], [RFC 2871, 2000].

2.4 Vantagens da IPtel

Um dos principais interesses e provavelmente o mais significativo de transportar áudio e dados sobre a mesma rede foi a redução de custos, nomeadamente nas corporações com grandes redes de dados. Seguidamente serão enunciadas algumas das vantagens do IPtel para a criação de novos serviços:

- Simplicidade: a IPtel é composta por uma infra-estrutura integrada, que tem como suporte o protocolo IP para todas as formas de comunicação, tornando-se eficiente e normalizada, reduzindo a complexidade e promovendo mais flexibilidade do que a PSTN;
- Eficiência: a integração das várias formas de comunicação permite um melhor aproveitamento da largura de banda;
- Arquitectura aberta: a IPtel assenta numa arquitectura aberta e normalizada. Esta arquitectura permite que novos serviços sejam criados mais facilmente e de uma forma modular. O utilizador tem à sua escolha uma maior oferta de mercado, não dependendo apenas de um fabricante;
- Novos serviços: o aparecimento de novos serviços integrando a *Web*, *e-mail*, presença, *chat* e mensagens instantâneas com a inclusão de voz e vídeo, que não podem ser facilmente implementados em redes de comutação de circuitos, permite uma maior interoperabilidade entre diferentes plataformas, tornando mais flexível e global a forma de comunicar. A característica de nas redes IP a "inteligência"

estar descentralizada ou distribuída normalmente nos terminais permite que a substituição possa ser mais frequente e independente de outros elementos da rede. Conclusões das primeiras experiências indicam que é mais fácil e mais rápido desenvolver novos serviços avançados para uma rede comutada de pacotes do que para uma rede PSTN [Low, 1996], [Low 1997];

- Segurança: o problema da segurança na Internet tem sido abordado de uma forma prioritária nos últimos anos. Vários serviços de segurança são actualmente utilizados na Internet, como a cifra, autenticação e autorização, garantindo a segurança da informação. O protocolo SIP permite a cifra e autenticação de mensagens e o RTP suporta a cifra dos meios; a utilização destes dois protocolos permite comunicações cifradas e seguras [RFC 3261, 2002];
- Multimédia: oferece a possibilidade de transporte de informação para além do áudio, como por exemplo o vídeo, mensagens instantâneas de texto e partilha de aplicações;
- Compressão e supressão de silêncio: A IPtel possibilita a utilização de mecanismos de compressão e supressão de silêncio nos terminais. Estes mecanismos permitem a redução do consumo de largura de banda e por serem usados em terminais implicam um custo menor. Numa conversa telefónica, verifica-se que em cerca de 50% da sessão não existe tráfego de voz [Kulathumani, 2000]. Recorrendo-se a mecanismos de detecção e supressão de silêncio, é possível obter um aproveitamento considerável da largura de banda;
- Escolha da qualidade da sessão dos meios: a IPtel permite controlar a qualidade da troca dos meios entre terminais. Esse controlo é feito nos terminais, permitindo escolher diversos métodos de compressão de áudio e vídeo;
- Identificação do utilizador: a utilização do protocolo RTP permite a identificação ou informação adicional do utilizador, isto é, no caso de estar estabelecida uma sessão de voz permite identificar quem fala;
- Privacidade: nos telefones tradicionais não existe a possibilidade de filtrar as chamadas, a única privacidade é o número não aparecer na lista telefónica. A IPtel permite facilmente a autenticação de quem faz a chamada, através de uma palavra-chave e certificados criptográficos;
- Anonimato: a IPtel oferece um modo de endereçamento independente da

localização. Um endereço IPv4 pode ser facilmente localizado a nível geográfico, embora seja relativamente fácil através de serviços de anonimato esconder essa informação. Na rede PSTN qualquer número de telefone, excepto os números 800, revela uma localização geográfica [Lennox, 2000];

- Programação de serviços: a IPtel introduz um novo modelo de configuração de serviços, a configuração nos terminais [Schulzrinne, 2000]. Enquanto que na telefonia tradicional o utilizador apenas dispõe de um conjunto de opções limitado para alterações de parâmetros nos serviços que lhe são disponibilizados, como por exemplo toque diferenciado consoante quem faz a chamada, a IPtel substitui o conceito de configuração de serviços pela programação de serviços. Um utilizador pode programar o procedimento de receber chamadas com base na disponibilidade indicada no calendário electrónico ou na identidade de quem faz a chamada, verificando se esta se encontra ou não no livro de endereços electrónico;
- Custos: não existe nenhum estudo autenticado comparando em observações gerais a rede PSTN e a IPtel em termos de custos de operação, manutenção e escalabilidade, mas é convincente que a IPtel é a opção mais barata [ITU/BDT, 2002].

2.5 Obstáculos à IPtel

Apesar da IPtel prometer na globalidade uma melhor prestação que a telefonia tradicional, ainda é necessário muito trabalho para se consolidar no mercado das comunicações. Alguns dos pontos fracos e obstáculos a serem ultrapassados são enunciados em seguida.

- Qualidade de Serviço: como a Internet foi projectada para transportar dados, oferece um serviço do tipo "melhor esforço" e trata todo o tipo de informação de igual modo. Em serviços como o transporte de meios em tempo real, a congestão da rede provoca atrasos e significativas perdas de pacotes, traduzindo-se numa redução da qualidade de serviço. Para que seja possível manter o padrão de qualidade exigido, desenvolveram-se vários mecanismos na área da QoS, alguns dos quais abordados no ponto 2.7;
- Segurança: embora provavelmente seja mais fácil quebrar a segurança num dispositivo da rede convencional do que num dispositivo da rede IP, a verdade é

que a Internet tem ainda a reputação de ser insegura. A IPtel oferece mecanismos para garantir um bom nível de segurança, alguns deles já referidos anteriormente;

- Utilização: a fiabilidade e disponibilidade elevadas que a rede telefónica tradicional oferece são uma forte oposição à entrada de uma nova tecnologia;
- Interoperabilidade: Embora a IPtel traga grandes vantagens, não implica o desaparecimento da telefonia tradicional. Assim, devem existir sistemas que forneçam a interoperabilidade necessária para consolidar o funcionamento de serviços de redes IP e de redes não IP (ex. PSTN, móveis, PBX, *Frame Relay* e ATM);
- Custo dos terminais: uma das grandes desvantagens da telefonia IP é o custo de um terminal. O terminal telefónico convencional tem um custo que varia entre os €20 e os €250, além de não necessitar de uma fonte de alimentação externa. O equipamento para o uso da IPtel, que não seja uma aplicação de um computador pessoal, tem preços que variam entre €300 e €1000 e necessita de uma fonte de alimentação externa. Estes factores são fortemente limitativos mas futuramente resolvidos para a expansão da IPtel em larga escala;
- Problemas conceptuais: com a natureza conceptual da Internet, aparecem um vasto número de novas características no ambiente da telefonia IP em relação à telefonia tradicional. Um desses exemplos é o caso dos pacotes dos meios serem trocados apenas pelos terminais, sem passarem por servidores intermediários, não podendo estes explorar qualquer tipo de serviço transparente aos terminais. Outro problema reside no facto dos terminais terem controlo absoluto da sinalização: embora esta característica traga vantagens também levanta o problema da rede querer contrariar decisões dos terminais;
- Modelo de confiança: actualmente, na telefonia tradicional, o utilizador assume que pode confiar na sua operadora do serviço, em questões como a provisão de informação correcta da chamada e na confidencialidade de informação que esta possui acerca dele. Este modelo de confiança baseia-se em que quando uma operadora recebe um sinal de chamada de um subscritor este é de facto desse subscritor e quando recebe um sinal de outra operadora de telefonia esta é digna de confiança. No entanto este modelo deixa de ter sentido quando as chamadas são feitas ponto a ponto e em casos em que o fornecedor de serviços (ISP) pode ser

qualquer um [Lennox, 2000].

2.6 Arquitectura protocolar

Uma das características mais relevante da tecnologia IPtel é a separação de funcionalidades, a qual está a cargo de um conjunto de protocolos. A Figura 2.3 [Schulzrinne, 2000] representa a pilha dos principais protocolos usados na IPtel, permitindo uma identificação de correspondência com o modelo OSI (*Open Systems Interconnection*). Pode-se também identificar alguns dos protocolos usados nos componentes de um serviço IPtel.

Ao nível da aplicação são implementados diversos serviços: a codificação de meios que é feita recorrendo a codificadores de áudio e vídeo, a sinalização em que são utilizados os protocolos como o H.323 e SIP, a qualidade de serviço que procura ser assegurada recorrendo aos protocolos *Resource ReServation Protocol* (RSVP) [RFC 2205, 1997] e RTCP, e o transporte de meios através do RTP. A um nível inferior aparecem as camadas mais baixas do modelo OSI que não são abordadas nesta dissertação.

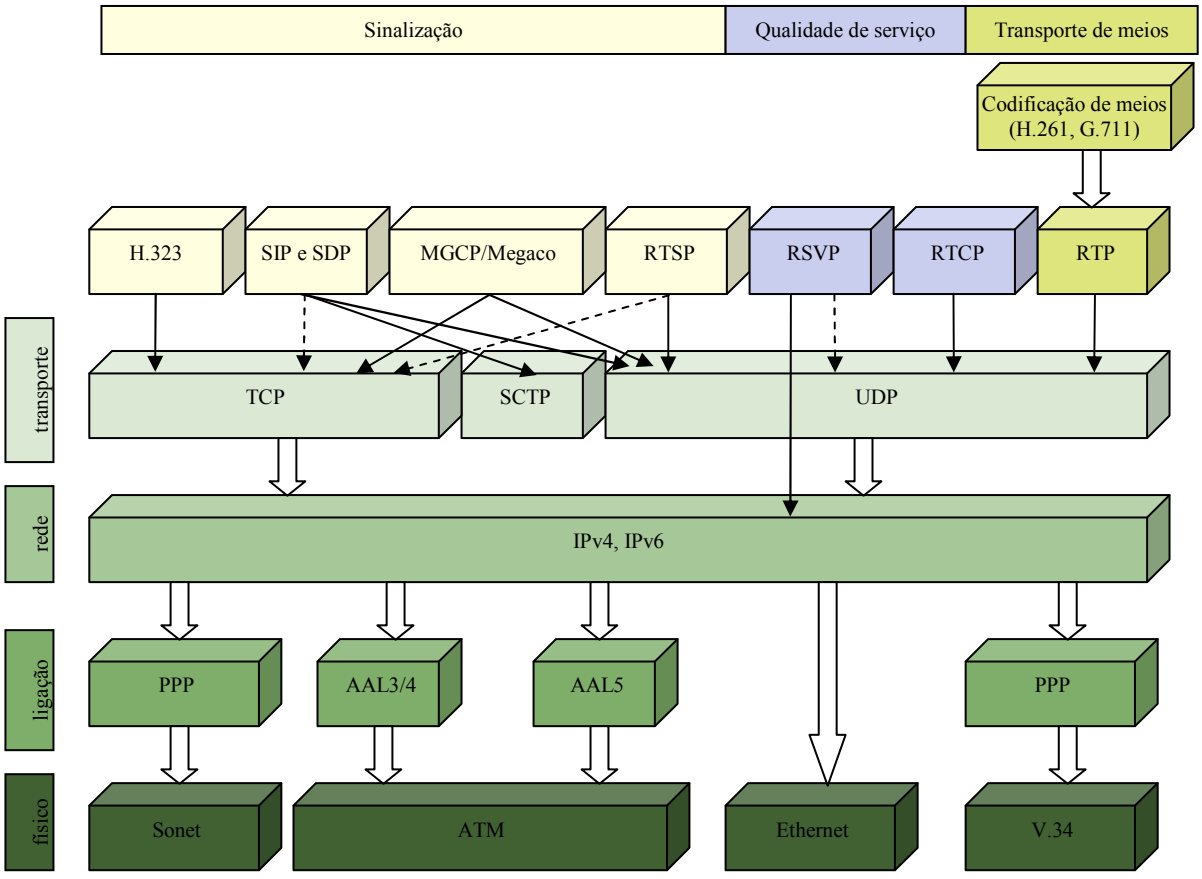


Figura 2.3 – Arquitectura protocolar da IPtel

2.6.1 Sinalização

Uma das características mais relevantes da IPtel é a capacidade de uma entidade enviar mensagens de sinalização para um ou mais participantes, procedendo à iniciação de uma chamada. Actualmente dois protocolos competem pelo domínio da sinalização da IPtel sendo eles o SIP e o H.323, abordados em seguida.

2.6.1.1 SIP

Desenvolvido pelo grupo MMUSIC do IETF, o *Session Initiation Protocol* (SIP), foi inicialmente publicado na RFC 2543 em 1996 e agora obsoleta com a publicação da RFC 3261 em Junho de 2002. É um protocolo de controlo (sinalização) ao nível da aplicação para a criação, alteração e finalização de sessões entre um ou mais intervenientes. Estas sessões incluem chamadas de Telefonia sobre IP, distribuição e conferência multimédia. Após o grande sucesso do SIP, o IETF decidiu criar o *SIP Working Group*, um grupo independente para o desenvolvimento deste protocolo iniciado pelo MMUSIC.

A Figura 2.4 [Schulzrinne, 2001] mostra alguns dos protocolos do IETF utilizados para a construção de uma arquitectura multimédia, na qual se verifica que o SIP não é um sistema integrado de comunicações para implementar um serviço multimédia, como é o caso do H.323, descrito na secção seguinte. Normalmente essas arquitecturas incluem um número de protocolos diferentes. O RTP para assegurar o transporte dos meios e o RTCP para fornecer informação útil ao nível de QoS. O *Session Description Protocol* (SDP) [RFC 2327, 1998] para descrever a sessão multimédia. O *Real-Time Streaming Protocol* (RSTP) [RFC 2326, 1998] para o controlo da entrega de *streams*. O *Media Gateway Control* (MEGACO) [RFC 3015, 2000] para o controlo das *gateways* que fazem o interface com a redes PSTN. O DNS para a determinação do destinatário dos pedidos. O protocolo *Lightweight Directory Access Protocol* (LDAP) [RFC 1777, 1995] para o acesso directo à base de dados de um servidor de localização. O TRIP [RFC 3219, 2002] para troca de informação de encaminhamento entre domínios administrativos de telefonia. Finalmente o RSVP para estabelecer a reserva de recursos.

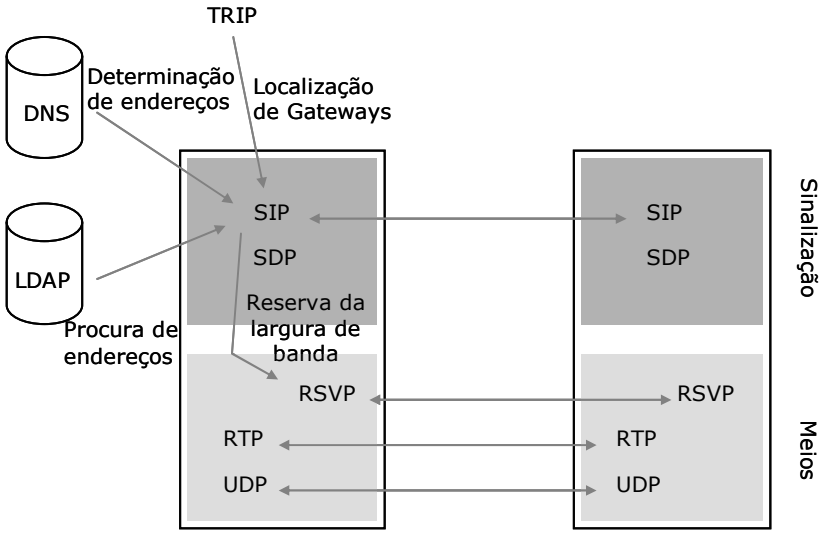


Figura 2.4 – Interacção do SIP com outros protocolos

O SIP é um protocolo baseado no *Simple Mail Transfer Protocol* (SMTP) [RFC 2327, 1998] usado como protocolo base do serviço de *e-mail* e também no *HyperText Transfer Protocol* (HTTP) [RFC 821, 1982], o protocolo base da *Web*. O SIP é um protocolo de texto, que reutiliza várias propriedades do HTTP e baseia-se no modelo cliente/servidor: o cliente faz pedidos e o servidor retorna respostas aos pedidos do cliente. Utiliza uma semântica e sintaxe semelhante, como se pode verificar nos campos dos cabeçalhos das mensagens, e recorre também aos métodos de autenticação do HTTP. Embora possa correr sobre o *Transmission Control Protocol* (TCP) e o *Stream Control Transmission Protocol* (SCTP), o SIP é mais utilizado sobre o protocolo *User Datagram Protocol* (UDP) [RFC 768, 1980], disponibilizando para isso os seus próprios mecanismos de recuperação de erros e permitindo o envio de mensagens *multicast*.

Ao nível dos serviços, o SIP inclui na sua recomendação inicial os seguintes serviços:

- Localização do utilizador: responsável pela localização do terminal para estabelecer a comunicação;
- Disponibilidade do utilizador: responsável pela determinação da vontade do utilizador em estabelecer uma sessão de comunicação;
- Recursos do utilizador: responsável pela determinação dos meios a utilizar e dos seus parâmetros;
- Características de negociação: responsável pela negociação e por chegar a acordo relativamente aos recursos disponíveis, reconhecendo que nem todas as partes

apresentam o mesmo nível de recursos;

- Gestão da sessão: possibilidade de transferir, colocar em espera ou terminar sessões, assim como de modificar parâmetros das mesmas e de invocar serviços;
- Alteração das características da sessão: possibilidade de alterar as características da sessão no decurso da mesma.

Todo o *software* SIP está no terminal que interage com o utilizador designado por *User Agent* (UA). O UA pode funcionar como *software* cliente num PC, num dispositivo móvel, ou como *firmware* num telefone IP. Um UA tem dois componentes: o *User Agent Client* (UAC) e o *User Agent Server* (UAS). O UAC é o responsável pela iniciação da chamada, enviando pedidos, os quais o UAS processa e aos quais responde enviando respostas. Este protocolo será analisado de uma forma mais pormenorizada no capítulo 3.2.

2.6.1.2 H.323

A recomendação H.323 foi criada pela *International Telecommunication Union* (ITU), responsável pelo estudo técnico, operacional e de questões de tarifas, que publicou recomendações sobre estes, com vista a normalizar as telecomunicações numa perspectiva mundial. O sector de Telecomunicações do *International Telecommunication Union* (ITU-T), é responsável pelo desenvolvimento de um conjunto de normas para conferências multimédia sobre redes de pacotes e inter-conexão com redes de comutação de circuitos. Este conjunto de normas está sob competência da Recomendação H.323 [H.323, 1996].

O H.323 não é um protocolo individual, mas antes um conjunto completo de protocolos integrados verticalmente, que definem terminais, equipamento e serviços para comunicações multimédia sobre redes de dados como a Internet. A recomendação H.323 está limitada à definição da sinalização, controlo de fluxo, formato de pacotes e normas de compressão dos meios. As especificações sobre captação dos meios, como os formatos de captação de áudio e vídeo, ou aplicações de dados estão fora do âmbito da recomendação H.323.

A primeira versão da Recomendação H.323 foi publicada pelo ITU-T em 1996. Tinha como objectivo inicial os serviços de comunicação de áudio e vídeo sobre LANs sem garantia de qualidade de serviço (QoS), foi no entanto criticada pelo seu baixo desempenho e problemas de compatibilidade entre diferentes fabricantes, levando estes a adicionar as suas próprias extensões de forma proprietária. A segunda versão, publicada em Janeiro de 1998, dispunha de mecanismos de conexão rápida rectificando o problema na demora do estabelecimento da

chamada, novos recursos que eliminaram a necessidade de extensões proprietárias, assim como novos protocolos [Packetizer, 2002a]. A terceira versão, que foi aprovada em Setembro de 1999, usufruiu de melhoramentos modestos em relação à Recomendação H.323v2, introduzindo poucas alterações no documento base. No entanto a Recomendação H.323 evoluiu substancialmente através da adição de novos anexos ao H.323 e ao H.225.0, melhorando a arquitectura H.323 [Packetizer, 2002b]. A última versão (H.323v4), aprovada em Novembro de 2000, inclui melhorias em várias áreas importantes como sejam escalabilidade, flexibilidade e segurança. Novas características foram adicionadas nas *Gateways* e *Multipoint Control Unit* (MCU), de modo a facilitar a inclusão de novas soluções adequadas às necessidades crescentes do mercado [Packetizer, 2002c]. A próxima versão, H.323v5, está prevista para 2003, não estando ainda disponível nenhum documento [Packetizer, 2002d].

A Recomendação H.323 define um conjunto de identidades num sistema H.323 (Figura 2.5), estando incluídas o terminal H.323, *Gatekeeper*, *Multipoint Controller* (MC), *Multipoint Processor* (MP), *Multipoint Control Unit* (MCU) e *Gateway*. Todas estas entidades são descritas em seguida e têm uma operacionalidade diferente no funcionamento da rede, podendo pertencer a uma única rede ou estar distribuídas por várias redes com diferentes infra-estruturas.

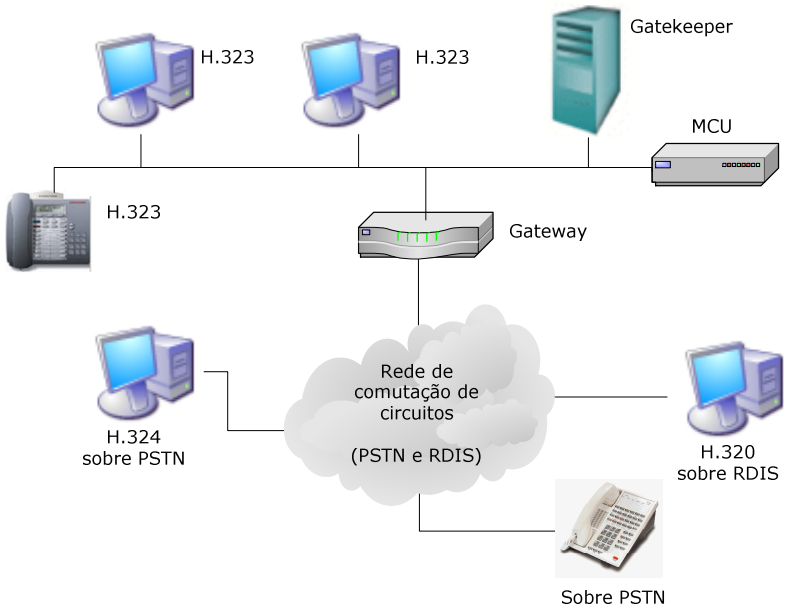


Figura 2.5 – Exemplo de uma rede H.323

- Terminal H.323: é um terminal numa rede, que permite a interface com o utilizador e a comunicação bidireccional em tempo real com outro terminal H.323,

Gateway ou *Multipoint Control Unit* (MCU). Esta comunicação consiste na troca de áudio, vídeo e/ou dados em qualquer combinação entre dois terminais. Um terminal H.323 pode ser um telefone IP ou um PC com microfone, altifalantes e câmara de vídeo;

- *Gatekeeper*: é um elemento opcional numa conferência H.323 que permite serviços de controlo de chamadas, tradução de endereços, controlo de admissão à rede a terminais H.323, *Gateways* e MCUs. Disponibiliza também outros serviços a estes terminais, destacando-se a reserva da largura de banda e a localização de *Gateways*;
- *Multipoint Control Unit*: é um terminal numa rede H.323 que permite que três ou mais terminais e/ou *Gateways* participem numa conferência multiponto. Dois terminais podem, no entanto, iniciar uma conferência ponto a ponto e mais tarde evoluir para uma conferência multiponto. Um MCU consiste em duas partes, uma obrigatória, o *Multipoint Controller* (MC), e uma opcional, o *Multipoint Processor* (MP);
    - *Multipoint Controller*: é um controlador para conferência multiponto, que tem a capacidade de negociação com todos os terminais de modo a obter níveis comuns de comunicação. Pode também controlar recursos numa conferência como por exemplo saber de quem é uma emissão de vídeo *multicast*;
    - *Multipoint Processor*: permite a mistura, comutação e outro tipo de processamento de *streams* sob o controlo de um MC. Permite também o processamento centralizado de *streams* dependendo do tipo de conferência suportada.
- *Gateway*: faz a conversão necessária entre diferentes tipos de terminais, o que permite a interoperabilidade de sistemas H.323 com outros sistemas de conferência multimédia integrados em diferentes tipos de redes, como RDIS, PSTN e ATM, entre outras. Disponibiliza também serviços como compressão e empacotamento.

A Recomendação H.323 utiliza nas suas diversas funcionalidades um conjunto de recomendações ITU-T, algumas delas representadas na Figura 2.6. A ITU-T define também outros procedimentos e protocolos ligados à Internet, como a recomendação H.245 para o controlo, a H.225 para o estabelecimento das conexões, a H.332 para conferências de maiores

dimensões, a H.450.x para serviços suplementares, a H.235 para a segurança e cifra, a H.246 para interoperabilidade com serviços em redes de comutação de circuitos, a H.324 para videoconferência sobre conexões de baixa capacidade como a PSTN e H.320 para videoconferência sobre RDIS.

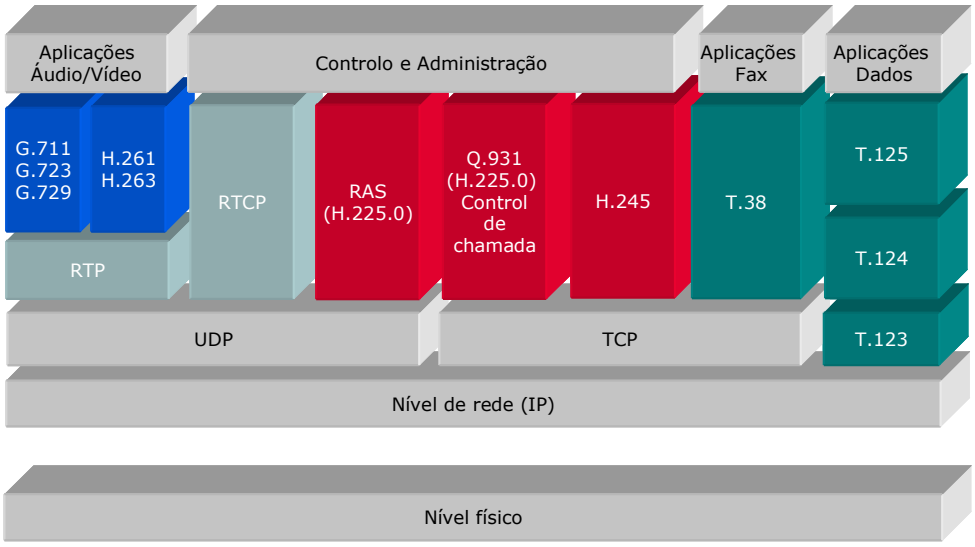


Figura 2.6 – Arquitectura protocolar do H.323

No estabelecimento de uma sessão básica o H.323 utiliza três protocolos de controlo, o *Registration Admission and Status* (RAS), o H.225.0/Q931 e o H.245. A Figura 2.7 [Michaely, 2000] descreve a troca de mensagens entre as entidades, numa rede H.323.

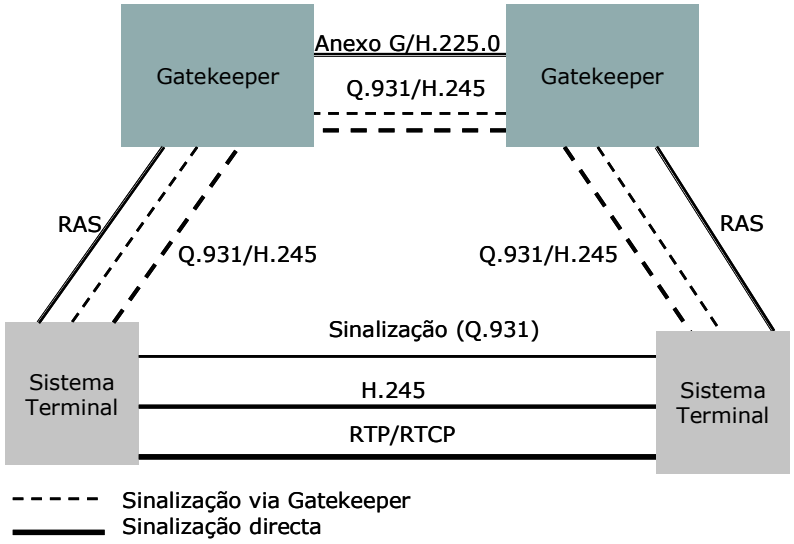


Figura 2.7 – Troca de mensagens entre entidades H.323

As mensagens RAS definidas na recomendação H.225.0 permitem a sinalização para as funções de registo, admissão, alteração da largura de banda e procedimentos de libertação de

recursos entre terminais e *Gatekeepers*.

A Recomendação H.225.0 [H.225.0, 1998] descreve o protocolo de sinalização de chamada usado para o controlo de admissão e estabelecimento de conexões entre dois ou mais terminais. Esta recomendação define também um procedimento (Q.931) usado pelos terminais H.323 para a sinalização de chamadas.

A Recomendação H.245 [H.245, 1998] define o protocolo H.245, o qual permite a troca de informação sobre capacidades, negociação de canais e comutação entre diferentes tipos de meios. Especifica a sintaxe e a semântica das mensagens, assim como procedimentos para o uso delas na negociação de canais no início e durante a comunicação.

Uma vez estabelecida a chamada, a transmissão dos meios é iniciada. O transporte de áudio e vídeo é realizado recorrendo ao protocolo RTP, não esquecendo o RTCP para o controlo e monitorização da entrega de dados do RTP. No entanto também aqui a Recomendação H.225.0 define procedimentos para a formatação e transporte dos pacotes de áudio e vídeo. Por fim a codificação e descodificação de áudio e vídeo, segundo a Recomendação H.323, são feitas utilizando codificadores definidos nas Recomendações G.711, G.722, G.728, G.729 e G.723.1 para áudio e H.261 e H.263 para vídeo.

2.6.2 Codificadores

Os codificadores e descodificadores, comummente denominados *codecs*, são dispositivos que permitem reduzir a largura de banda para a transmissão de dados utilizando técnicas de compressão. Estas técnicas de compressão devem para isso operar em tempo real, devido a características do próprio serviço, como a comunicação interactiva. A compressão de sinais é baseada em técnicas de processamento que eliminam informação redundante, ou mesmo desnecessária. Na compressão pode haver ou não perda de informação dependendo principalmente do método utilizado.

Existem várias entidades responsáveis por normalizar codificadores de áudio e vídeo, tais como a *International Telecommunication Union* (ITU), *Telecommunication Industries Association* (TIA) e *United States Federal Standards* (USFS). Para codificar o sinal áudio em tempo real alguns dos codificadores mais conhecidos são ITU-T G.711, ITU-T G.722, ITU-T G.726, ITU-T G.723, ITU-T G.728, ITU-T G.729, CELP, GSM e MPEG-Audio e para a codificação de vídeo em tempo real os codificadores H.261, H.262, H.263 e JPEG.

2.6.3 Transporte

O transporte de meios difere substancialmente do transporte de dados como a transferência de ficheiros, em que propriedades como a congestão e perda de pacotes, *jitter* e atraso de pacotes são fundamentais para a comunicação entre utilizadores. Protocolos como o HTTP e o *File Transfer Protocol* (FTP) são utilizados sobre o protocolo TCP que tem a particularidade de ser um protocolo com confirmação de entrega de dados, permitindo, quando um pacote é perdido ou corrompido, a sua retransmissão. No entanto o atraso introduzido pela funcionalidade de reenvio não é favorável à transmissão do *stream* e é facilmente dispensada a confirmação, sendo necessário o recurso a outros protocolos. O protocolo UDP é uma das soluções mais utilizadas pois não utiliza mecanismos de retransmissão mas não garante a entrega de pacotes nem que estes vão chegar ao destino pela ordem de saída.

Para a entrega de meios em tempo real é usado o *Real-Time Transport Protocol* (RTP) definido no RFC 1889. Este protocolo é normalmente utilizado sobre o UDP e permite serviços de entrega ponto a ponto para a transmissão de dados em tempo real.

2.6.3.1 Real-Time Transport Protocol

O *Real-Time Transport Protocol* (RTP) [RFC 1889, 1996] é utilizado para o suporte de serviços de transporte em aplicações de tempo real, como por exemplo *streaming* a pedido e serviços interactivos como sejam a videoconferência e o IPtel. O RTP permite funções de transporte ponto a ponto na rede e é apropriado para aplicações que transmitem dados em tempo real como áudio, vídeo, sobre serviços de redes *unicast* ou *multicast* [RFC 1889, 1996]. Inclui também serviços de reconstrução de meios com informação temporal, detecção de perdas de pacotes, segurança, monitorização da entrega e identificação de conteúdo; no entanto a utilização individual deste protocolo não resolve o problema da reserva de recursos da rede nem garante a QoS no transporte de dados em tempo real. Para controlar e monitorizar a entrega de dados em tempo real feita pelo RTP, é utilizado o protocolo RTCP. Este protocolo disponibiliza ainda *feedback* da qualidade dos dados transportados e permite incluir um identificador ao nível do transporte para a sincronização de áudio e vídeo. Normalmente as aplicações correm o RTP sobre UDP fazendo uso dos serviços de multiplexagem e *checksum*. Contudo o RTP é um protocolo independente da camada de transporte subjacente podendo ser usado sobre o protocolo *Connection Less Network Protocol* (CLNP) ou o *Internetwork Packet Exchange* (IPX), entre outros. Foi também experimentado directamente sobre AAL5 utilizando serviços nativos de ATM [Schulzrinne, 2000]. A utilização do RTP

sobre UDP ou IP é não fiável, já que estes também não são. Pode-se no entanto recorrer à fiabilidade providenciada por camadas inferiores como o caso das camadas ATM AAL3/4 ou AAL5.

Em seguida são descritas algumas das funcionalidades que são disponibilizadas pelo RTP [Schulzrinne, 1999]:

- Ordenação: caso cheguem pacotes desordenados estes podem ser reordenados pelo destinatário em tempo real. Permite ainda detectar, caso exista, a perda de pacotes e compensá-la sem retransmissões;
- Sincronização intra-meios: deve ser transmitida a informação sobre o intervalo de tempo entre os instantes em que pacotes sucessivos devem ser descodificados. Por exemplo, quando se utiliza mecanismos de detecção de silêncio não são enviados pacotes durante esse período, no entanto a duração desse silêncio deve ser reconstruída apropriadamente;
- Sincronização inter-meios: disponibiliza mecanismos que permitem a sincronização de diferentes tipos de meios. Por exemplo no caso de uma videoconferência, o áudio pode ser reproduzido devidamente sincronizado com o vídeo;
- Identificação do *Payload*: permite identificar o tipo de dados que estão a ser transmitidos. Esta propriedade é necessária quando se pretende modificar o codificador do meio transmito devido à variação de parâmetros, como sejam a capacidade ou a qualidade do *stream* pretendido;
- Identificação de *frames*: o vídeo e áudio são enviados em unidades lógicas chamadas frames. É necessário indicar ao destinatário onde começa e acaba esse frame, de modo a auxiliar a entrega sincronizada a camadas superiores;
- *Multicast* simples: o RTP e RTCP foram construídos para *multicast*, tanto para pequenos grupos (ex. videoconferência com três participantes), como para a difusão de eventos (ex. emissões de rádio pela Internet);
- Serviços para meios genéricos de tempo real: podem ser usados outros codificadores de meios, sendo a informação sobre esses codificadores definida em especificações próprias;
- Misturadores e Tradutores: os Misturadores são dispositivos que recebem meios de

vários utilizadores, misturando-os num único *stream* e possibilitando a alteração do formato do *stream*, que é enviado posteriormente. São úteis para reduzir exigências de largura de banda de um *stream* antes de ser enviado para uma ligação com menor largura de banda, sem que para isso seja necessário diminuir a taxa da transferência da fonte do meio;

- *Feedback* relativo ao QoS: o RTCP permite aos destinatários fornecer *feedback* com informação da qualidade de recepção. As fontes de RTP podem usar esta informação para ajustar a taxa de transferência de dados, enquanto os destinatários podem por exemplo determinar se os problemas de qualidade de serviço são da rede local ou de toda a rede;
- Liberdade no controlo da sessão: com o RTCP os participantes podem trocar informação de identificação como o nome, *e-mail*, número de telefone e mensagens curtas;
- Cifra: de modo a garantir privacidade numa sessão, *streams* de RTP podem ser cifrados usando chaves que são trocadas por algum método não RTP, por exemplo pelos protocolos SIP ou SDP.

2.6.3.2 Real Time Control Protocol

A principal função do protocolo RTCP [RFC 1889, 1996] é fornecer *feedback* da qualidade dos dados distribuídos. É baseado na transmissão periódica de pacotes de controlo a todos os participantes na sessão, usando o mesmo mecanismo de distribuição que o de pacotes de dados. Todos os participantes numa sessão RTP enviam pacotes RTCP. Os remetentes dos meios (fontes) e os destinatários enviam periodicamente pacotes RTCP para o mesmo grupo *multicast*. Os pacotes RTCP podem conter informação sobre a qualidade do serviço para os participantes da sessão, informação sobre a fonte do *stream* que está a ser transmitido, ou estatísticas sobre os dados que já foram transmitidos até ao momento.

O protocolo RTCP fornece quatro funções:

- A principal função é fornecer *feedback* da qualidade dos dados distribuídos, desempenhada através de relatórios RTCP;
- O RTCP transporta um identificador global ao nível da camada de transporte designado *Canonical Name* (CNAME), que permite particularizar a fonte RTP. Este identificador é associado aos diferentes *streams* gerados pelo mesmo

participante, possibilitando por exemplo sincronizar a reprodução de áudio e vídeo;

- As primeiras duas funções requerem que todos os participantes enviem pacotes RTCP. Normalmente estes relatórios consomem cerca de 5% da largura de banda ocupada pelo *stream* [Campbell, 1997]. De notar que é necessário controlar a taxa de envio de pacotes RTCP em relação aos pacotes RTP num cenário de participantes em larga escala a fim de evitar situações de congestão da rede;
- A última função é opcional e permite transportar o mínimo de informações de controlo, como por exemplo o transporte da identificação do participante. Esta função é mais adequada em sessões pouco controladas nas quais os utilizadores entram e saem sem autenticação ou a negociação de parâmetros [RFC 1889, 1996].

Um participante ou uma fonte RTP que gera um *stream* envia juntamente pacotes de controlo designados por *Sender Reports* (SR). Os participantes que recebem o *stream* numa sessão RTP entregam periodicamente relatórios designados por *Receivers Reports* (RR) a todas as fontes RTP pertencentes à sessão. Os *Sender Reports* descrevem a quantidade de dados enviados até ao momento, bem como a relação entre o instante de amostragem e o tempo absoluto de modo a permitir a sincronização de diferentes tipos de meios. Os *Receiver Reports* contêm informação instantânea e cumulativa sobre a taxa de pacotes perdidos e *jitter* de uma fonte. Indicam também o maior número da sequência recebida e o tempo de envio (*timestamp*), podendo este ser usado para estimar o atraso *round-trip* entre o remetente e o destinatário.

2.6.3.3 Formatos *Payload*

Já foi referido que o protocolo RTP é utilizado para o suporte de serviços de transporte genérico de áudio, vídeo e dados em tempo real. De modo a ser flexível, este protocolo permite a utilização de codificadores particulares, utilizando a definição de formatos de dados específicos para esses codificadores. Estes formatos descrevem a sintaxe e a semântica dos dados RTP. A associação entre codificadores e formatos, é através do registo de nomes/*Payload Type* (PT) no *Internet Assigned Numbers Authority* (IANA).

No RFC 1890 [1996], é definido um conjunto de codificadores normalizados e os seus nomes quando são usados dentro do RTP. As especificações de codificação são definidas independentemente do mecanismo de transporte usado.

Alguns formatos de *payload* foram já definidos para RTP, como o caso dos codificadores de vídeo H.261 [RFC 2032, 1996], H.263 [RFC 2190, 1997], MPEG [RFC 2250, 1998] e codificadores de áudio G.722.1 [RFC 3047, 2001]. Existem também outros formatos de *payload* criados para permitir serviços genéricos, como por exemplo o RFC 2198 [1997] que define um formato *payload* RTP para a transmissão de dados codificados de uma maneira redundante que permite recuperar informação perdida.

2.7 Qualidade de serviço

Uma das características conhecidas nas redes IP é o seu procedimento baseado na ideia de igualdade de acessos e sem tratamento particular para qualquer nó ou serviço. Ainda recentemente estas redes apenas forneciam um tipo de serviço, o de melhor esforço (*best effort*), não garantindo a QoS.

As redes de melhor esforço apresentam várias desvantagens:

- Não oferecem garantias na perda de pacotes, atrasos e *jitter*, comprometendo o desempenho de aplicações como a videoconferência em redes cuja capacidade possa diminuir em qualquer momento devido a um aumento do tráfego;
- Possibilidade de congestão na rede. Esta acontece quando não se reduz o envio de pacotes e a congestão força os elementos da rede a descartar pacotes. O bom funcionamento destas redes está pois dependente do modo de procedimento que as aplicações têm para evitar a congestão;
- Estes procedimentos para evitar a congestão podem no entanto levar a situações de injustiça caso não sejam bem implementados. É o caso de duas aplicações que partilham o mesmo recurso e uma delas tem políticas para evitar a congestão e a outra não. Esta situação pode levar a que uma das aplicações responda às indicações de congestão, por exemplo reduzindo o envio de pacotes, enquanto a outra aplicação, por não ter procedimentos nesse sentido, pode consumir essa parte do recurso libertado.

As desvantagens destas redes têm sido alvo de algumas iniciativas de estudo de forma a garantir a QoS ou pelo menos optimizar alguns parâmetros. Existem várias abordagens para a obtenção de melhores soluções na garantia de QoS. A primeira é melhorar o desempenho da

camada de rede, assegurando a reserva de recursos. A segunda abordagem tem como objectivo fornecer a diferenciação de serviços em redes IP. A terceira aproximação baseia-se na negociação de serviços, para a utilização de múltiplos serviços de rede. A quarta abordagem tenta reduzir as perdas e os atrasos de pacotes, através do uso de mecanismos adaptativos utilizados ponto a ponto. O quinto método baseia-se no envio de correcções de erro, fornecendo mecanismos de redundância para ultrapassar a perda de pacotes em *streams* multimédia. Por último, a QoS também pode ser beneficiada com características introduzidas dos equipamentos de rede. São exemplos, as filas de espera, modulação da forma do tráfego e filtros que conseguem implementar prioridades de tráfego e controlar a congestão ponto a ponto.

A primeira abordagem foi apresentada pelo grupo de trabalho *Integrated Services* do IETF, que propôs uma arquitectura de integração de serviços, chamada *Internet Integrated Services* (IIS). O esquema de reserva de recursos proposto possibilita que as aplicações possam reservar uma ligação virtual que lhes oferece uma largura de banda fixa ponto a ponto, permitido um fluxo de baixo atraso para os pacotes. Permite ainda que as aplicações tenham condições para atribuir prioridades aos seus pacotes. Esta reserva é feita nos elementos da rede, tais como *routers* e *gateways*, utilizando um protocolo específico. Vários protocolos podem ser usados, como o *Scable Reservation Protocol* (SRP) [Almesberger, 1998], o *YEt another Sender Session Internet Reservations* (YESSIR) [Pan, 1998] e o *Resource ReSerVation Protocol* (RSVP) [RFC 2205, 1997 e RFC 2209, 1997], sendo este último o mais utilizado para a reserva de recursos na camada de rede. No entanto, problemas de escalabilidade encontrados nesta arquitectura levaram o IETF e considerar soluções mais simples e eficientes para o estabelecimento da QoS na Internet.

A segunda abordagem baseia-se numa arquitectura de diferenciação de serviços na Internet. Desenvolvida pelo grupo *Differentiated Services* (DiffServ) do IETF, esta arquitectura permite hierarquizar agregando o tráfego através de uma marcação no pacote ao nível da camada IP, usando o campo DS. Estes serviços podem ser implementados ponto a ponto ou intra-domínio, podem também satisfazer requisitos de desempenhos quantitativos (ex. reserva de largura de banda), assim como desempenhos relativos (ex. diferenciação de classes) [RFC 2475, 1998]. Esta arquitectura é possível substituindo o campo *Type of Service* (ToS) do cabeçalho IP, pelo campo DS, e a partir da análise deste fornecer diferentes tipos de tratamentos aos pacotes de cada fluxo, transportados pela rede [RFC 2474, 1998].

A terceira aproximação baseia-se na negociação de serviços de rede, isto é, a selecção e utilização de um serviço específico envolve a negociação entre o utilizador e a rede, acordando as especificações necessárias ou pretendidas para tal serviço. Uma das soluções recorre ao protocolo *Resource Negotiation and Pricing Protocol* (RNAP) [Wang, 1999]. O RNAP permite a negociação de serviços entre a aplicação e a rede, assim como entre domínios adjacentes. Possibilita também a informação do preço e despesa para os serviços requisitados e ainda o suporte para utilizadores com capacidades limitadas de negociação [Wang, 1999].

A quarta abordagem baseada na utilização de mecanismos adaptativos, permite reduzir o atraso e a perda de pacotes. Neste caso os terminais analisam as condições da rede ponto a ponto, ajustando atributos dos *stream* de modo a adaptar as aplicações às condições da rede. Estes mecanismos não asseguram um suporte explícito além do transporte normal de pacotes da rede, sendo por isso apenas considerados mecanismos ponto a ponto. Estas soluções adaptativas são também utilizadas em aplicações *multicast*, analisando o *feedback* do transporte dos meios. Os mecanismos adaptativos que actuam no ajustamento dinâmico de QoS para aplicações multimédia apareceram pouco tempo depois dos protocolos de transporte de meios. Busse e Deffner [Busse, 1995] propuseram um controlo dinâmico de QoS em aplicações que utilizam os relatórios RTP para saber a perda de pacotes e o *jitter*. Essa informação permitia que uma aplicação de videoconferência (*vic*) ajustasse o tráfego gerado à largura de banda disponível. Outro mecanismo, mas neste caso para áudio, foi proposto por Ramjee e Schulzrinne [Ramjee, 1994]. Esta solução continha vários algoritmos que actuavam sobre os *buffers* permitindo adaptar a reprodução de áudio de modo a compensar o atraso variável da rede.

A última abordagem, denominada por correcção antecipada (*Forward Error Correction* – FEC), lida apenas com problemas de perdas de pacotes. Este problema é mais significativo na transmissão de áudio. Na existência de pequenas perdas de pacotes, aparecem ruídos desagradáveis durante a reprodução do áudio sendo mesmo ininteligível quando as perdas atingem maiores proporções. Como não é possível a retransmissão selectiva de pacotes baseada em tempos de espera para a detecção de pacotes perdidos em áudio e vídeo, o envio de informação para a correcção de erros é um modo de fornecer redundância em troca de maiores requisitos de largura de banda e de tempos de latência mais elevados. Existem vários mecanismos que são usados para a compensação da perda de pacotes na rede, alguns deles podem ser encontrados na RFC 2354 [RFC 2354, 1998]. Algumas das aproximações incluem:

- Reparação local: não depende de informação adicional enviada pelo emissor e, quando um pacote é perdido, compensa-se a falta de informação utilizando algum tipo de interpolação. No entanto alguns estudos mostram que estes mecanismos não funcionam em algumas situações resultando em perda de sincronismo do codificador e descodificador [Rosenberg, 2001];
- Codificadores redundantes: consiste em transmitir dois níveis de codificação de pacotes, um com a qualidade pretendida pelo utilizador e outro de menor qualidade. O primeiro é utilizado para a reprodução de som e o segundo é utilizado para fornecer a redundância. Caso o *stream* seja transmitido com sucesso, então é reproduzido. Se existir perda de pacotes provocando um vazio, então esse será preenchido com os pacotes do *stream* de menor qualidade, pois a redução da qualidade causada pela perda de um pacote é normalmente insignificante [Rosenberg, 2001];
- Pacote FEC: neste caso os códigos tradicionais (ex. paridade e códigos *Reed-Solomon*) são aplicados sobre os pacotes. Assim os pacotes adicionais FEC podem ser usados para recuperar pacotes perdidos. Esta aproximação tem a vantagem de recuperar informação perdida independente do tipo de meios, seja áudio, vídeo ou dados [Rosenberg, 2001].

Capítulo 3

Especificação do Serviço sIPtel

A oferta de mais serviços e de com melhores funcionalidades é um dos principais factores para que a IPtel possa vingar no sector das comunicações e substituir a rede PSTN. Lennox [Lennox, 1999] descreve diversas características que são encontradas na rede PSTN e enuncia como podem ser implementadas na IPtel seguindo as descrições da Recomendação Q.1211 do ITU-T e utilizando o protocolo SIP e as suas extensões.

Neste capítulo são abordados os requisitos básicos de um serviço IPtel com maior destaque para a característica de sinalização. Posteriormente é explicada a arquitectura e o funcionamento do protocolo de sinalização usado para a implementação do serviço. Finalmente são enunciadas características de alguns dos codificadores mais utilizados para a compressão de áudio e vídeo.

3.1 Requisitos

Na implementação de um serviço IPtel devem ser consideradas diversas propriedades ao nível da sinalização e da troca de meios. Deste modo a sinalização deve oferecer um conjunto de características como:

- Tradução de nomes e localização de utilizadores: esta propriedade pode ser implementada pelo terminal IPtel, recorrendo por exemplo ao serviço DNS. Esta opção pode ser dispensada quando por omissão o terminal IPtel envia pedido para um servidor SIP que implementa esse serviço;
- Capacidade de negociação: permite aos utilizadores que pretendam estabelecer uma sessão de comunicação acordarem o tipo de meios utilizados na sessão e

parâmetros da mesma;

- Controlo de chamadas: após o estabelecimento de uma chamada entre utilizadores, os participantes devem por exemplo ter a possibilidade de colocar utilizadores em espera e retomar novamente a sessão quando bem entenderem;
- Mudanças de características da sessão: durante uma chamada deve ser possível alterar características da sessão, por motivos opcionais dos utilizadores ou por necessidade dos recursos envolvidos na sessão.

Ao nível do fluxo de meios, parâmetros de digitalização de áudio e vídeo e algoritmos de compressão são fundamentais para a viabilização da transmissão multimédia através da rede. Para isso devem ser disponibilizadas várias opções de configuração de parâmetros relacionados com a digitalização de áudio e vídeo e deve ser permitida a escolha entre vários codificadores de áudio e vídeo possibilitando a variação de características como qualidade dos meios, atrasos e largura de banda.

3.2 Sinalização utilizando o SIP

A sinalização na IPtel é realizada pela troca de mensagens entre os vários componentes da rede, sendo o formato dessas mensagens definido pelos protocolos de sinalização. É durante a sinalização que o chamador e o chamado definem parâmetros para o estabelecimento da chamada e troca de dados, como os endereços de transporte para o envio de meios, o tipo de meios transmitidos, mecanismos de codificação de meios, autorização para iniciação e aceitação da chamada, requisitos de largura de banda, e mecanismos de segurança, entre outros.

O SIP tem como função principal o estabelecimento, a modificação e a finalização de sessões entre um ou mais participantes. A aplicação dessas sessões inclui telefonia sobre IP, distribuição e conferências multimédia. O SIP ou extensões do SIP podem ainda ser usados para mensagens instantâneas, notificação de presença, e jogos distribuídos.

3.2.1 Componentes SIP

O SIP suporta cinco funcionalidades para o estabelecimento e finalização de sessões multimédia: localização, disponibilidade e recursos do utilizador, e características de

negociação descritas anteriormente no capítulo 2.6.1.1.

Para implementar estas funcionalidades, existem vários componentes distintos no SIP. São eles o *User Agent*, o *Proxy Server*, o *Registrar Server* e o *Redirect Server*.

- User Agent (UA): consiste em duas partes distintas, o *User Agent Client* (UAC) e o *User Agent Server* (UAS). O UAC é uma entidade lógica que gera pedidos SIP e recebe respostas a esses pedidos. O UAS é uma entidade lógica que gera respostas aos pedidos SIP. O UA permite normalmente a interface com o utilizador, mas pode também ser um sistema automático que não envolva interacção como um sistema de voice mail ou um sistema de redireccionamento de chamadas. A Figura 3.1 (Sessão SIP A) ilustra um exemplo de uma sessão entre dois UAs;
- *Proxy Server*: é uma entidade intermediária que actua como cliente e servidor para o propósito de estabelecer chamadas entre os utilizadores. Com uma funcionalidade semelhante à de um *Proxy HTTP*, o *Proxy Server* tem a tarefa de encaminhar os pedidos que recebe para outras entidades mais "próximas" do destinatário. A Figura 3.1 (Sessão SIP B) ilustra um exemplo de um utilizador (Utilizador A) que recorre a um *Proxy Server* para contactar outro utilizador (Utilizador B). Existem dois tipos de *Proxy Servers*: o *Stateful Proxy* e o *Stateless Proxy*;
  - *Stateful Proxy*: mantém o estado das transacções durante o processamento dos pedidos. Permite dividir um pedido em vários (*fork*), na tentativa de encontrar em paralelo múltiplas localizações do chamado e apenas enviar as melhores respostas ao utilizador que fez a chamada;
  - *Stateless Proxy*: não mantém o estado das transacções durante o processamento dos pedidos. São mais adequados quando existem requisitos de velocidade como numa *backbone* de uma infra-estrutura SIP;
  - *Outbound Proxy*: é um *proxy* que recebe pedidos de um utilizador, mesmo que não seja ele o destinatário do pedido. Esta configuração é muito utilizada e adequada quando existem *firewalls*, em que o UA é configurado para enviar pedidos e receber pedidos através de um *Outbound Proxy*. É típico nestes *proxys* a integração de um serviço de controlo sobre as *firewalls* que analisa as mensagens SDP de modo a alterar as configurações da *firewall* e até modificar as mensagens SDP do UA que representa;

- *Registrar*: é um servidor que aceita pedidos de registo de utilizadores e guarda a informação desses pedidos para fornecer um serviço de localização e tradução de endereços no domínio que controla;
- *Redirect Server*: É um UAS que gera respostas de redireccionamento aos pedidos que recebe, que devem ser consideradas para completar o pedido iniciado. Este servidor não reencaminha os pedidos para o próximo servidor, mas responde com uma mensagem de redireccionamento (3xx) que contém o endereço do próximo servidor a ser contactado, ao cliente que fez o pedido. A Figura 3.1 (Sessão SIP C), mostra um exemplo de um UA (Utilizador A) que utiliza um *Redirect Server* para obter uma lista de localizações alternativas e formular um novo pedido SIP para a localização ou localizações obtidas (neste caso foi directamente para o Utilizador B).

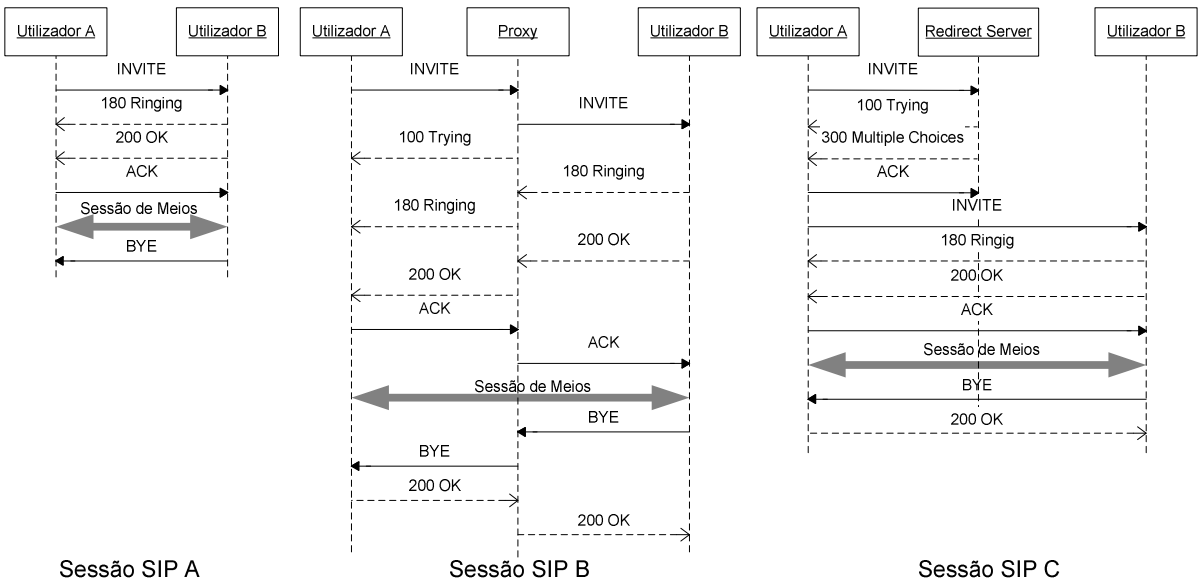


Figura 3.1 – Exemplos de sessões SIP

3.2.2 Mensagens SIP

O SIP é um protocolo de texto com uma semântica semelhante à do protocolo HTTP. Esta propriedade permitiu a reutilização de código e uma integração mais simples dos servidores SIP com servidores de *Web* e de *e-mail*. Tal como o HTTP, o SIP define a comunicação através de dois tipos de mensagens: os pedidos e as respostas. Os UACs fazem os pedidos e os UASs retornam respostas aos pedidos dos clientes. Uma mensagem SIP consiste numa linha inicial seguida de um ou mais cabeçalhos (*headers*), uma linha vazia que indica o fim

dos cabeçalhos e, por fim, o corpo da mensagem que é opcional. Os cabeçalhos são usados para transportar informação necessária às entidades SIP, de modo a processarem os pedidos ou respostas. Caso exista o corpo da mensagem, este é usado para descrever a sessão, contendo normalmente o protocolo *Session Description Protocol* (SDP); no entanto pode ter outro tipo de conteúdo como ASCII ou HTML.

Os pedidos SIP são caracterizados pela linha inicial da mensagem, chamada *Request-Line*, que contém o nome do método, o *Request-URI* (identifica o destinatário do pedido no próximo servidor) e a versão do protocolo SIP. Existem seis métodos diferentes na actual versão do SIP (versão 2) que descrevem o pedido desejado pelo cliente e que são explicados em seguida:

- REGISTER: regista a informação de contacto;
- INVITE: permite convidar um utilizador ou serviço para participar numa sessão ou para modificar parâmetros numa sessão já existente;
- ACK: confirma o estabelecimento de uma sessão para se dar a troca de meios.
- BYE: é usado para terminar uma sessão;
- CANCEL: é usado para terminar um pedido pendente;
- OPTIONS: é usado para solicitar informação sobre as capacidades do chamado.

Um exemplo de um INVITE básico sem corpo de mensagem é ilustrado na Figura 3.2.

```
INVITE sip:utilizadorB@utad.pt SIP/2.0
Via: SIP/2.0/UDP pc1.ipb.pt;branch=z9hG4bK776as555
Max-Forwards: 50
To: Utilizador B <sip:utilizadorB@utad.pt>
From: Utilizador A <sip:utilizadorA@ipb.pt>;tag=1233434545
Call-ID: a456cd3455@pc1.ipb.pt
CSeq: 1 INVITE
Contact: <sip:utilizadorA@pc1.ipb.pt>
```

Figura 3.2 – Exemplo de um INVITE

De salientar no entanto que podem ser utilizadas extensões SIP publicadas em RFCs que definem métodos adicionais aos apresentados.

As respostas SIP são semelhantes aos pedidos, diferindo destes na linha inicial, chamada *Status-Line*, que contém o código da resposta e uma frase descritiva. O código da resposta é composto por três dígitos que permitem classificar os diferentes tipos existentes (Tabela 3.1). O primeiro dígito define a classe da resposta e os últimos dois dígitos não têm qualquer regra

de classificação. Por essa razão qualquer resposta compreendida entre 100 e 199 é referida como uma "resposta 1xx". As respostas 1xx são conhecidas como provisórias, pois contêm informação do progresso da chamada; as respostas que têm os códigos desde 200 até 699, são conhecidas como respostas finais e como próprio nome indica, finalizam as transacções SIP.

Código	Classe	Categoria	Exemplo
1xx	Informal	Provisória	180 Ringing
2xx	Sucesso	Final	200 OK
3xx	Redireccional	Final	302 Moved Temporarily
4xx	Erro no Cliente	Final	404 Not Found
5xx	Erro no Servidor	Final	501 Not Implemented
6xx	Falha Global	Final	600 Busy Everywhere

Tabela 3.1 – Classe de respostas SIP

Ao pedido INVITE apresentado anteriormente, o utilizador pode, por exemplo, obter uma resposta de sucesso (200 OK) como mostra a Figura 3.3.

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP pc1.ipb.pt;branch=z9hG4bKnas555;received=192.0.1.1
To: Utilizador B <sip:utilizadorB@utad.pt >;tag=a4554d
From: Utilizador A <sip:utilizadorA@ipb.pt>;tag=1233434545
Call-ID: a456cd3455
CSeq: 1 INVITE
Contact: <sip:utilizadorB@192.0.1.1>
```

Figura 3.3 – Exemplo de uma resposta 200 OK

3.2.3 Cabeçalhos SIP

Os cabeçalhos SIP são similares aos cabeçalhos HTTP tanto na semântica como na sintaxe. Alguns desses cabeçalhos são usados em todas as mensagens enquanto outros só fazem sentido em pedidos ou em respostas. Quando um cabeçalho aparecer numa mensagem e não fizer parte da categoria dessa mensagem deve ser simplesmente ignorado. O SIP disponibiliza um mecanismo para transmitir os cabeçalhos mais comuns de uma forma abreviada, útil para situações em que por qualquer motivo as mensagens se tornem mais extensas. A Tabela 3.2 [RFC 3261, 2002] mostra alguns dos cabeçalhos mais comuns nas mensagens SIP.

Via	Indica o transporte a ser usado para a transacção e identifica a localização para onde a resposta deve ser enviada.
From	Indica a entidade lógica do iniciador do pedido.
To	Indica a entidade lógica do destinatário do pedido.
Call-ID	Contém um identificador único global que identifica uma chamada. Deve ser igual para todas as mensagens dentro de uma transação.
CSeq	Permite identificar e ordenar mensagens dentro das transacções.
Contact	Contém uma ou mais localizações que podem ser usadas para contactar o utilizador.
Content-Length	Tamanho da mensagem em bytes.
Content-Type	Indica o tipo de meio do corpo da mensagem.
Max-Forwards	Serve para limitar o número de reencaminhamentos até ao seu destino.

Tabela 3.2 – Exemplos de cabeçalhos SIP

3.2.4 Endereços SIP

O SIP identifica o utilizador através de um tipo de *Universal Resource Identifier* (URI) chamado SIP URI [RFC 3261, 2002]. O SIP URI utiliza a forma mais comum de endereçamento de utilizadores na Internet, o formato do endereço de *e-mail*, como por exemplo: *sip:utilizador@dominio*, *sip:utilizador@host*, *sip:utilizador@IP-address* ou *sip:numero-telefone@gateway*. O SIP permite ainda recorrer a identificadores para utilizadores associados a comunicações seguras, denominados SIPS URIs. Este identificador especifica que o recurso a ser contactado é seguro (ex. *sips:utilizador@dominio*). A primeira parte do SIP ou SIPS URI está associada ao utilizador, serviço ou número de telefone. Quando se pretende especificar um utilizador num terminal específico, a segunda parte é normalmente um endereço IP ou o nome do computador no domínio (ex. *sip:jpaulo@sip-multimedia-host.ipb.pt*). Quando é o endereço é independente da localização, é normalmente especificado o nome de um domínio (ex. *sip:jpaulo@ipb.pt*).

A solução de identificação do SIP é também baseada em entidades existentes na rede IP, como o DNS. Recentemente foi publicada a RFC 3263 [RFC 3263, 2002] que descreve os procedimentos DNS utilizados pelos clientes para traduzir o SIP URI num endereço IP, porta e protocolo de transporte, ou pelos servidores para retornar uma resposta ao cliente caso o pedido falhe. Esta característica permite, como acontece com todos os URIs utilizados na Internet, que os SIP e SIPS URIs possam ser colocados em páginas *Web*, mensagens de *e-mail* ou outras aplicações. O SIP segue desta forma um paradigma muito simples que lhe permite a utilização de diversos serviços com uma metodologia de "um só endereço". A Figura 3.4

mostra alguns exemplos de SIP e SIPS URIs.

```
sip:jpaulo@ipb.pt
sip:pc1@ipb.pt
sip:123456789@gateway-ipb.pt
sips:jpaulo@ipb.pt
```

Figura 3.4 – Exemplos de endereços SIP

3.2.5 Criação e finalização de chamadas

A Figura 3.5 ilustra um exemplo do estabelecimento de uma chamada entre dois utilizadores utilizando dois *SIP Proxys*. O *utilizador1* pretende através de um *software* IPtel instalado no seu PC e identificado como *utilizador1@ipb.pt*, fazer uma chamada para o *utilizador2* dentro do domínio *utad.pt*, que está registado no *SIP Proxy* como *utilizador2@utad.pt*. Os dois *SIP Proxys* ilustrados na Figura 3.5 têm a função de controlar cada um dos domínios e o registo dos utilizadores que pertencem a esse domínio. Têm como objectivo facilitar o estabelecimento das sessões e podem ter também a função de permitir a comunicação com o exterior, no caso, por exemplo, da existência de uma *firewall* que protege cada um dos domínios.

Para iniciar a chamada o *utilizador1* envia ao *utilizador2* um INVITE, que transporta no corpo da mensagem os parâmetros da sessão que pretende estabelecer, utilizando o protocolo SDP. Porque o *software* IPtel não conhece a localização do *utilizador2*, ou tem na sua configuração o *proxy ipb.pt* como *Outbound Proxy*, o INVITE é enviado para o *proxy ipb.pt* que controla o domínio *ipb.pt* e que terá como função reencaminhar o pedido para um servidor mais próximo do *utilizador2*. No INVITE é incluído o cabeçalho *Route* com o endereço do *proxy ipb.pt* de modo a garantir que todas as mensagens trocadas entre o *utilizador1* e o *utilizador2* passem por este *proxy*.

O *Proxy Server* ao receber a mensagem terá que tomar uma decisão de encaminhamento do pedido recebido. No decorrer desta decisão o *proxy* envia uma resposta *100 Trying* ao *utilizador1* a indicar que recebeu o INVITE e está a processar o pedido. O *proxy ipb.pt* através de um serviço de localização como uma consulta a uma base de dados conclui que não conhece o *utilizador2*. Utilizando então uma procura DNS descobre o *proxy utad.pt* que controla o domínio *utad.pt* e encaminha o INVITE para esse servidor. Antes de reencaminhar o pedido, o *proxy ipb.pt* adiciona à mensagem um cabeçalho *Via* e um cabeçalho *Record-Route* que contém o seu próprio endereço. O cabeçalho *Via* é utilizado na devolução das

respostas e o *Record-Route* é para garantir que todas as trocas entre os dois utilizadores de mensagens passam por ele.

O *proxy utad.pt* recebe o pedido e responde ao *proxy ipb.pt* com uma resposta *100 Trying*. Como o *utilizador2* está registado no seu domínio, o *proxy* encaminha o INVITE para o *software* IPtel deste utilizador. Também aqui é adicionado um cabeçalho *Via* e um cabeçalho *Record-Route* que contém o endereço do *proxy utad.pt*.

A aplicação, ao receber o pedido, alerta o *utilizador2* que tem uma nova chamada e envia uma resposta *180 Ringing* para o *utilizador1* através dos servidores, mas agora em sentido contrário. A aplicação do *utilizador1*, ao receber a mensagem *180 Ringing*, alerta o utilizador indicando que está a "chamar". Os *proxys* utilizam a informação contida nos cabeçalhos *Via* para devolver a resposta, removendo o cabeçalho *Via* do topo da mensagem que continha o seu endereço. Enquanto no encaminhamento do pedido INVITE foi necessário recorrer ao DNS e a serviços de localização, as respostas (ex. *180 Ringing*) podem ser retornadas sem invocar os serviços de localização ou mesmo sem manter o estado da transacção nos *proxys*.

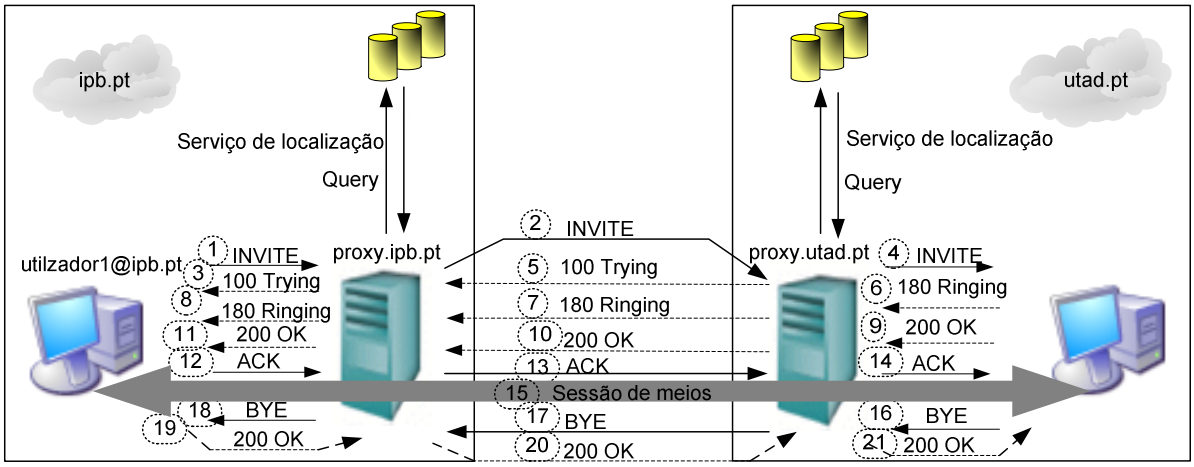


Figura 3.5 – Exemplo de uma chamada entre dois utilizadores

Como no exemplo o *utilizador2* aceita a chamada, a sua aplicação envia ao *utilizador1* uma resposta final 200 OK, que transporta no corpo da mensagem os parâmetros da sessão dos meios que pretende estabelecer, utilizando o protocolo SDP, completando a negociação básica de capacidades disponibilizada pelo SIP. A aplicação do *utilizador1* ao receber a resposta 200 OK envia uma mensagem ACK para confirmar que recebeu a resposta. Quando recebe a resposta 200 OK é criada uma relação SIP ponto a ponto baseada na combinação dos parâmetros *From tag*, *To tag* e *Call-ID* referida como *diálogo* (*Dialog*). Em seguida é iniciada a troca de meios utilizando os formatos acordados durante o estabelecimento da chamada

recorrendo ao protocolo SDP. O SIP não exerce qualquer controlo sobre o encaminhamento dos pacotes dos meios entre os utilizadores, podendo estes percorrer caminhos diferentes das mensagens de sinalização. Para terminar a sessão o *utilizador2* envia um BYE ao *utilizador1*. A aplicação ao receber este pedido, termina o envio dos meios e retorna uma resposta 200 OK, concluindo a chamada e o *diálogo* entre os dois utilizadores.

3.2.6 Alteração de uma sessão

Após o estabelecimento de uma sessão através do envio de um INVITE, o chamador e o chamado podem modificar os parâmetros da sessão, como por exemplo, adicionar ou remover um dos meios, alterar endereços e portas, alterar a codificação dos meios, sem que para isso seja necessário finalizar a chamada. A Figura 3.6 ilustra um exemplo da criação de uma sessão contendo um ou mais meios e da alteração dos parâmetros dessa mesma sessão. Para isso, o *utilizador1* envia um INVITE que transporta na mensagem SDP os novos parâmetros da sessão, informando o *utilizador2* da sua intenção de alterar a sessão. O *utilizador2* ao receber o re-INVITE, e caso aceite a nova sessão, retorna uma resposta 200 OK e o requerente envia uma confirmação ACK. Estes pedidos são tratados de modo semelhante ao inicial, excepto que neste caso o UAS não retorna respostas provisórias. Caso não sejam aceites as alterações a chamada permanece activa mas com as características anteriores.

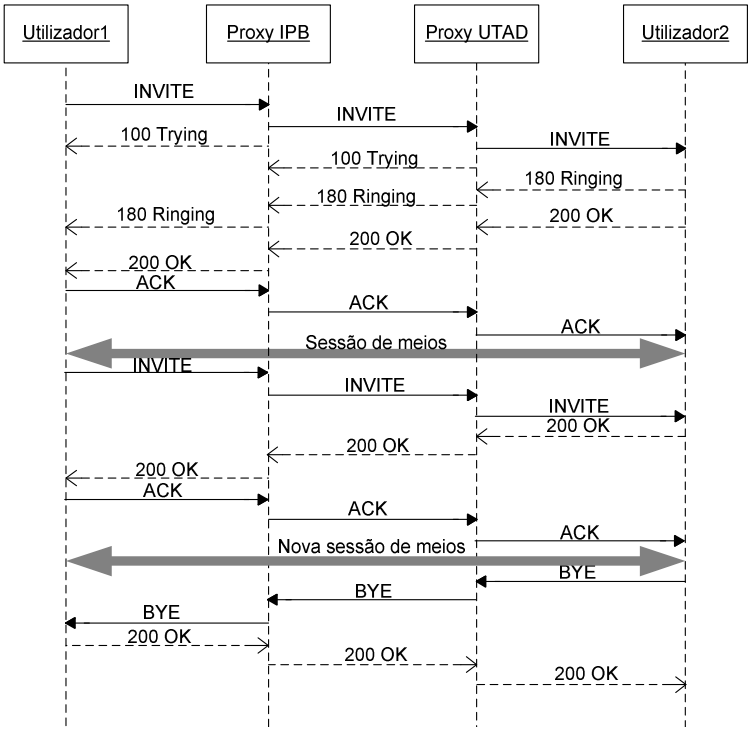


Figura 3.6 – Exemplo da alteração de parâmetros na sessão dos meios

3.2.7 Registar um utilizador

O serviço de registar um utilizador é utilizado para fornecer informação para a sua localização, permitindo ao mecanismo de localização do SIP descobrir em que terminal é que o utilizador pretende receber os pedidos. O registo de um utilizador é feito através de um pedido REGISTER ao servidor *Registrar*, como ilustra a Figura 3.7. Este pedido contém o endereço do utilizador no cabeçalho *To* que é indexado a um ou mais endereços que o cabeçalho *Contact* transporta.

É usual no controlo de domínios o mesmo servidor SIP fornecer as funcionalidades de *Registrar* e *Proxy Server*, possibilitando ao *Proxy Server* receber um pedido e encaminhá-lo segundo políticas definidas, ou determinar o próximo endereço através de um serviço de localização que fornece a tradução de endereços para um determinado domínio (ex. procurar o *Request-URI* na base de dados do *Registrar*). O registo tem também a função de criar ligações de endereçamento para um domínio particular. Essas ligações de endereços indexam os SIP ou SIPS URI (ex. *sip:utilizador1@ipb.pt*) a um ou mais endereços que estão de algum modo mais "próximos" do utilizador desejado (ex. *sip:utilizador1.sip-group.ipb.pt*). O registo do utilizador pode também servir para disponibilizar informação sobre o estado do utilizador. Quando um pedido REGISTER é enviado ao servidor, transporta no cabeçalho *Expires* ou no parâmetro *expires*, contido no cabeçalho *Contact*, o tempo de vida do contacto, indicando que durante esse tempo o utilizador está disponível para receber chamadas nesse URI. Se posteriormente não activar o registo, tal facto pode ser compreendido como "não está disponível para receber chamadas".

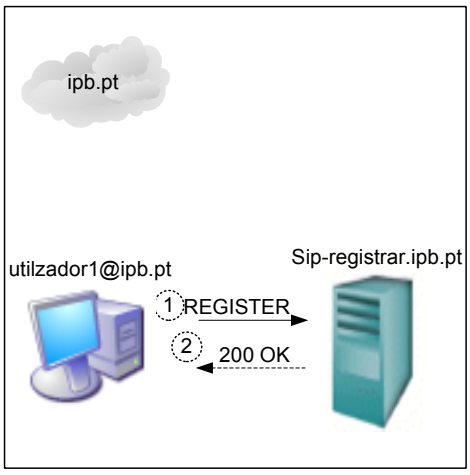


Figura 3.7 – Exemplo de registo de um utilizador

O SIP permite a actualização dos seus contactos ou da duração dos registos de uma forma

muito simples, bastando para isso enviar um novo pedido REGISTER com a nova informação. Do mesmo modo, é feito o cancelamento do registo de um utilizador num servidor *Registrar*. Este procedimento difere apenas no cabeçalho *Expires* ou no campo '*expires'* que contém o valor de zero para o endereço de contacto transportado no pedido. Neste caso, para anular todos os contactos indexados ao utilizador, o cabeçalho *Contact* deve conter o valor '*' e o cabeçalho *Expires* o valor de zero.

3.2.8 Utilização do protocolo SDP

O SIP utiliza o protocolo *Session Description Protocol* (SDP) [RFC 2327, 1998] para descrever as sessões multimédia. O SDP expressa uma lista de capacidades para áudio e vídeo e indica para onde os meios devem ser enviados, sendo estas informações transportadas como uma parte de dados que é adicionada à mensagem SIP. Permite também programar as sessões dos meios e repetições dessas sessões no futuro. Mais que um protocolo, o SDP na verdade é uma formatação que contém informação sobre o meio a usar e seu destino, nome da sessão e propósito, duração da sessão, informação de contacto e de largura de banda a ocupar pela sessão multimédia. O SIP utiliza um modelo de oferta e resposta, recorrendo ao SDP para acordar a delineação da sessão entre utilizadores. Isto é, um participante oferece ao outro uma descrição da sessão desejada e o outro utilizador responde com uma descrição da sessão que ele pretende. Normalmente a oferta é transportada no INVITE e a resposta é recebida no 200 OK. Contudo pode acontecer que a oferta seja feita pelo chamado através do 200 OK e a resposta seja dada pelo chamador através do ACK.

A Figura 3.8 mostra uma mensagem SIP que utiliza uma mensagem SDP para descrever o tipo de sessão pretendida e a informação necessária. Uma mensagem SDP é composta por uma série de linhas separadas pelo *carriage-return* e cada linha é iniciada com uma letra que representa um parâmetro seguindo-se o sinal de igual que separa o valor desse parâmetro.

Uma mensagem SDP consiste numa secção que descreve a sessão, seguida por zero ou mais secções que descrevem cada um dos *streams*. A descrição da sessão é iniciada pelo parâmetro '*v*' e finalizada pela primeira secção que descreve um *stream*. Esta secção por sua vez é iniciada pelo parâmetro '*m*' e finalizada pela próxima secção que descreve novamente um *stream*. Em geral os valores os valores assumidos para os streams são por omissão, os definidos na secção que descreve a sessão, a menos que esses valores sejam definido posteriormente na respectiva secção de cada stream.

```
INVITE sip:utilizadorB@utad.pt SIP/2.0
Via: SIP/2.0/UDP pc1.ipb.pt;branch=z9hG4bK776as555
Max-Forwards: 50
To: Utilizador B <sip:utilizadorB@utad.pt>
From: Joao Paulo <sip:jpaulo@ipb.pt>;tag=1233434545
Call-ID: a456cd3455@pc1.ipb.pt
CSeq: 1 INVITE
Contact: <sip:jpaulo@pc1.ipb.pt>
Content-Type: aplication/sdp
Content-Length: 251
Content-Length: 223

v=0
o=jpaulo 6629869483951460685 6629869483951460685 IN IP4 193.136.194.29
s=sIPtel Call
c=IN IP4 193.136.194.29
t=0 0
m=audio 2002/1 RTP/AVP 4
m=video 1000/1 RTP/AVP 34
a=rtpmap:4 G723 RTP/8000/1
a=rtpmap:34 H263 RTP/16000/1
```

Figura 3.8 – Exemplo da utilização do SDP numa mensagem SIP

Nas Tabela 3.3, Tabela 3.4 e Tabela 3.5, são ilustrados os tipos de campos permitidos pelo protocolo SDP. Algumas linhas em cada uma das tabelas são necessárias e outras são opcionais, devendo aparecer obrigatoriamente pela ordem descrita nas tabelas. As linhas opcionais estão assinaladas com um '*'.

Tipo	Descrição
v	Versão do protocolo
o	Criador e identificador da sessão
s	Nome da sessão
i*	Informação da sessão
u*	Descrição do URI
p*	Número de telefone
c*	Informação da conexão
b*	Informação da largura de banda
z*	Ajuste do fuso horário
k*	Chave para a cifra
a*	Zero, uma ou mais linhas de atributos da sessão

Tabela 3.3 – Descrição da sessão

Tipo	Descrição
t	Tempo em que a sessão está activa
r*	Zero ou mais vezes de repetição

Tabela 3.4 – Descrição temporal

Tipo	Descrição
m	Nome do meio e endereço do transporte
i*	Título do meio
c*	Informação da conexão – opcional se incluída no nível de sessão
b*	Informação da largura de banda
k*	Chave de encriptação
a*	Zero, uma ou mais linhas de atributos da sessão

Tabela 3.5 – Descrição do meio

As funções dos vários elementos da mensagem SDP ilustrada na Figura 3.9, são apresentadas seguidamente. A primeira linha define a versão do protocolo usado. O parâmetro '*o*' contém parâmetros como o proprietário e o identificador global da sessão, e o endereço IP onde esta foi criada a sessão. O parâmetro '*s*' descreve a sessão pretendida; neste caso trata-se de uma "sIPtel Call". O parâmetro '*c*' contém o endereço de conexão e indica o endereço IP em que se pretende receber o *stream*. Por exemplo, se este campo contiver o valor '0.0.0.0', indica que o meio não deve ser enviado; este método é usado quando se pretende colocar um utilizador em espera. O parâmetro '*t*' permite definir o tempo durante o qual a sessão está activa; o valor que é atribuído ao parâmetro neste exemplo indica que a sessão é ilimitada. O parâmetro '*m*' descreve um *stream* da sessão, isto é, indica o tipo do *stream* seguido pelo número da porta para onde deve ser enviado.

Seguidamente surge o indicador do transporte usado, que normalmente é o RTP/AVP indicando a utilização do protocolo RTP, o perfil de áudio/vídeo (*Audio/Video Profile – AVP*) [RFC 1890, 1996] e os formatos de dados usados na sessão. O parâmetro '*a*' é o método mais utilizado para expandir o SDP. O atributo '*rtpmap'* é utilizado para suportar os tipos de dados dinâmicos (dados que não são definidos no RFC 1890), transportando o identificador do formato de dados (*payload type*) juntamente com a informação de codificação para alocar dinamicamente os formatos de dados. Nesta mensagem é usado um identificador do formato *payload* RTP/AVP que não está atribuído (34) para o *stream* vídeo codificado em H.263. Também recorrendo ao parâmetro '*a*', é descrito o tipo de fluxo do meio que é pretendido. Cada *stream* pode ser unidireccional (só receber ou só enviar) ou bidireccional (enviar e receber). No exemplo são mostrados os casos unidireccionais; caso o valor do atributo seja '*sendrecv'* ou nada seja especificado, trata-se do valor por omissão que é bidireccional. Neste exemplo o utilizador apenas pretende receber um *stream* áudio e enviar um *stream* vídeo.

```
v=0
o=jpaulo 6629869483951460685 6629869483951460685 IN IP4 193.136.194.29
s= sIPtel Call
c=IN IP4 193.137.96.154
t=0 0
m=audio 2002/1 RTP/AVP 4
m=video 1000/1 RTP/AVP 34
a=rtpmap:4 G723 RTP/8000/1
a=recvonly
a=rtpmap:34 H263 RTP/16000/1
a=sendonly
```

Figura 3.9 – Mensagem SDP

3.2.9 Segurança

O SIP utiliza diversos mecanismos de segurança adequados a diferentes aspectos e aplicações, como por exemplo a preservação da confidencialidade e integridade das mensagens, prevenção de ataques que permitam os desvios de mensagens ou provoquem a indisponibilidade do serviço ou que proporcionem a autenticação de utilizadores e a privacidade dos participantes numa sessão. A cifra de todas as mensagens é o melhor mecanismo de segurança para a sinalização, garantindo a confidencialidade e integridade das mensagens. O SIP não permite a cifra das mensagens ponto a ponto, devido à possibilidade de estas poderem percorrer várias entidades intermediárias da rede (ex. *Proxy Server*), que têm de analisar os pedidos e repostas para os poderem encaminhar correctamente e que podem também adicionar ou remover informação como os cabeçalhos *Via*. Para obter este grau de segurança são preferencialmente recomendados mecanismos de segurança a um nível mais baixo, em que as mensagens são cifradas entre entidades SIP e permitem aos terminais verificar a identificação dos servidores para quem são enviadas as mensagens de forma segura, utilizando sistemas de autenticação criptográfica. A solução para este mecanismo de segurança passa pela utilização dos protocolos *Transport Layer Security* (TLS) [RFC 2246, 1999] e IPSEC [RFC 2401, 1998], que fornecem segurança ao nível da camada de transporte e ao nível da camada de rede respectivamente, permitindo a confidencialidade e integridade das mensagens. O SIP define URI seguros chamados SIPS URI, que permitem o estabelecimento de sessões seguras, garantindo que é utilizado transporte criptográfico (TLS) para entregar as mensagens.

Para a autenticação da identidade dos utilizadores, o SIP define [RFC 3261, 2002] o método de autenticação *Digest* que se baseia no esquema de autenticação HTTP *Digest*, utilizado pelo protocolo HTTP. Este método permite aos utilizadores identificarem-se perante uma entidade

através do nome do utilizador e de uma palavra-chave cifrada, utilizando a informação que lhe é fornecida pelas respostas 401 ou 407. Por exemplo, quando um utilizador se pretende registar num *Registrar* ou enviar um INVITE através de um *SIP Proxy*, o servidor responde com uma resposta 401 ou 407 indicando que é necessária a sua autenticação e transportando as suas credenciais. Quando o utilizador recebe a resposta formula um novo pedido, que desta vez transporta a informação necessária para confirmar a sua identidade. Este mecanismo de segurança permite evitar ataques em que utilizadores mal intencionados assumem a identidade não autorizada de outros utilizadores; no entanto não garante a confidencialidade e integridade das mensagens.

3.2.10 Mobilidade do SIP

O SIP permite ao terminal e ao utilizador terem a propriedade da mobilidade, isto é, a sinalização SIP está habilitada a encaminhar pedidos de estabelecimento de chamadas para vários locais, onde os utilizadores indicaram poderem ser contactados. O encaminhamento dos pedidos poderá ser feito de uma forma paralela e ao mesmo tempo (*forking*) ou de uma forma sequencial.

Numa procura paralela (Figura 3.10) o *proxy* distribui pedidos pelos vários endereços onde é possível o utilizador responder a esse pedido. Cada um desses pedidos pode gerar respostas que são retornadas através dos vários *proxys* por onde esses pedidos passaram. O SIP possibilita neste processo a definição de regras para o tratamento e retorno dessas respostas ao cliente. Este mecanismo dinâmico disponibilizado de forma básica pelo SIP permite ao utilizador decidir quando, onde e quais os recursos pretende utilizar para responder ao pedido.

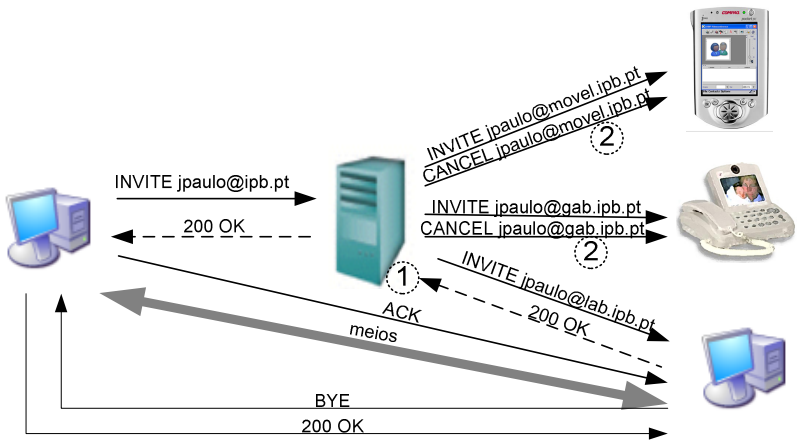


Figura 3.10 – Exemplo de uma procura paralela

Numa procura sequencial (Figura 3.11), o servidor *proxy* tenta contactar cada endereço de contacto sequencialmente, fazendo a próxima tentativa apenas depois da tentativa anterior resultar numa resposta final. A procura acaba quando for gerada uma resposta pertencente à classe sucesso (2xx) ou falha global (6xx) [RFC 3261, 2002].

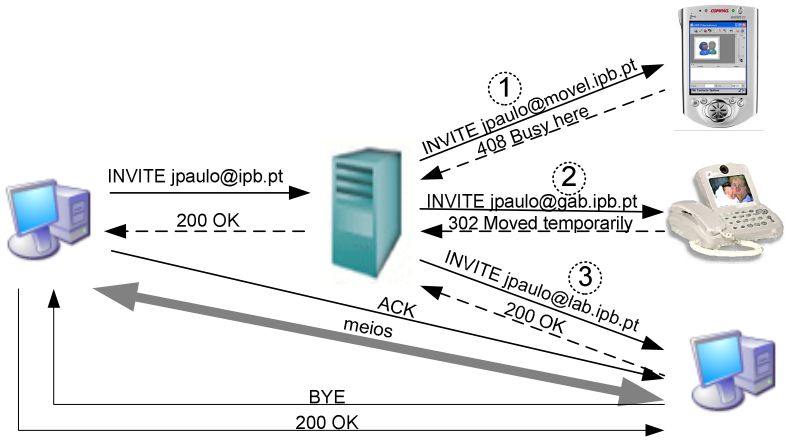


Figura 3.11 – Exemplo de uma procura sequencial

3.3 Codificação dos meios

A utilização de codificação tem como objectivo principal reduzir as exigências de transmissão multimédia particularmente da informação áudio e vídeo, recorrendo a algoritmos de compressão e tendo como consequência a redução da qualidade de informação.

Existem normalmente dois tipos de codificação: *sem perda de informação* (*Lossless Compression*) que garante que o processo de compressão e descompressão não alteram a informação original e *com perda de informação* (*Lossy Compression*) em que após a compressão e descompressão a informação não é exactamente igual à inicial.

Existem várias abordagens à compressão de dados, tendo duas delas suscitado maior interesse. A primeira abordagem está relacionada com a *Codificação da Forma de Onda* e recorre a técnicas de codificação e compressão que não utilizam a informação natural do sinal para ser comprimido e permite apenas a compressão sem perdas de informação. Esta codificação deve ser utilizada quando é exigido um padrão de qualidade mais elevado. A segunda abordagem é a *Codificação da Fonte*, que consiste na modelação do processo de produção de informação, isto é, optimiza a compressão de acordo com a semântica da informação inicial; por exemplo durante uma conversa a supressão de silêncio é uma

característica que depende da semântica do sinal de áudio [Fluckiger, 1995]. Este tipo de codificação é utilizada quando é necessária uma taxa de transmissão baixa devido a limitações de largura de banda e suporta a compressão com ou sem perda de informação.

Em seguida serão descritos sumariamente alguns codificadores de voz normalizados, divididos em duas categorias, *Codificação da Forma de Onda* e *Codificação da Fonte*.

Codificação da forma de onda

- ITU-T G.711: o sinal de voz é amostrado a uma frequência de 8 KHz e codificado em 8 bits com as codificações logarítmicas A-law ou μ-law. Este codificador ocupa uma largura de banda de 64 Kbps e provoca atrasos constantes. Não tem supressão de silêncio.

Codificadores da fonte

- ITU-T G.723.1: é uma combinação dos codificadores G.721 e G.723. Este codificador produz níveis de compressão de voz de 20:1 e 24:1. Conforme o algoritmo esteja configurado para gerar taxas de 6,3 ou 5,3 Kbps. A única diferença entre as duas taxas de transmissão é ao nível do processamento exigido para a compressão da informação. Devido à sua baixa taxa de transferência é ideal para fazer a inter conexão entre a tecnologia IPtel e a telefonia tradicional. Tem supressão de silêncio.
- IMA/DVI: esta norma utiliza o algoritmo de compressão *Adaptive Differential Pulse Code Modulation* (ADPCM), que comprime os 16 bits de dados de cada amostra em 4 bits e permite um débito de 32 Kbps.
- GSM: norma para a rede móvel de segunda geração usada particularmente na Europa. Este codificador utiliza os algoritmos compressão *Regular Pulse Excitation* (RPE) e *Long-Term Predictor* (LTP). A versão 6.1 permite uma taxa de 13 Kbps e tem uma qualidade de som inferior à de codificadores como o G.711 e G.722.
- MPEG-Audio: esta norma é utilizada para compressão de sinais não só de voz mas também de áudio com alta qualidade. O MPEG-Audio não é apenas um codificador mas um conjunto de três codificadores e esquemas de compressão denominados por MPEG-Audio Layer-1, Layer-2 e Layer-3. As taxas de amostragem podem ser de 32, 44.1 ou 48 KHz, produzindo taxas de 192 ou 256

Kbps por canal no Layer-1, 96 ou 128 Kbps por canal no Layer-2 e 64 Kbps no Layer-3; este último é o nível mais eficiente da série e permite obter uma qualidade de som próxima da qualidade de CD [Fluckiger, 1995].

O método *Mean Opinion Score* (MOS) permite classificar a qualidade dos sistemas de transmissão de voz. A escala do método MOS varia de um a cinco, representando os níveis de qualidade dos sistemas de transmissão de voz, como mostra a Figura 3.12.

Figura 3.12 – Escala do MOS

No primeiro nível, o utilizador simplesmente não entende a mensagem descodificada. No segundo nível, o sinal possui interrupções devido às degradações sendo difícil a percepção da mensagem. No terceiro nível o sinal está degradado mas não tem interrupções e a qualidade da voz é má. No quarto nível, as degradações são mínimas e a voz de boa qualidade. No quinto nível, o utilizador não consegue distinguir a mensagem enviada da mensagem original.

A Tabela 3.6 exibe as características de alguns codificadores de áudio [Richey, 2002].

Codificadores	MOS (aprox.)	Taxa (Kbps)
G.711	4,4	64
G.723.1	3,9	5,3
	3,9	6,3
GSM	3,5	13
MPEG-AUDIO	4,1	64
IMA/DVI	3	32

Tabela 3.6 – Tabela de análise dos codificadores.

Para a codificação de vídeo em tempo real são explicados em seguida três codificadores de vídeo: H.261 e H.263 da ITU e JPEG também designado por *Motion-JPEG* do ISO/IEC.

A norma H.261 foi publicada em 1990, pela ITU. Este codificador foi projectado para taxas de transmissão múltiplas de 64 Kbps, motivo pelo qual também é designado "p×64", em que *p* varia entre 1 e 30. Estas taxas de transmissão são apropriadas para linhas ISDN para as quais o H.261 foi construído.

O H.261 suporta dois formatos de imagem (Tabela 3.7), *Common Interchange Format* (CIF) e o *Quarter Common Interchange format* (QCIF). O CIF tem resoluções de 352 *pixels* por linha e 288 linhas por imagem, ambas codificadas com uma componente de luminância e duas de crominância, enquanto que o QCIF tem resoluções de 176 *pixels* por linha e 144 linhas por imagem. A razão da apresentação das imagens 4:3 é proporcional a uma imagem de televisão convencional.

Este algoritmo utiliza uma combinação de predição inter-imagem, de codificação por transformada e compensação de movimento. A predição inter-imagem remove a redundância temporal e a redundância espacial é removida pela codificação por transformada. Os vectores de movimento são usados para auxiliar o codificador de compensação de movimento. Por fim, para remover qualquer redundância transmitida no *stream*, são utilizadas a codificação *run-length* e a codificação de comprimento variável. As taxas geradas por este codificador podem variar entre 64 Kbps e 2 Mbps aproximadamente [Ghanbari, 1999].

O H.263, publicado em 1995 pelo ITU, foi inicialmente projectado para a codificação de vídeo utilizando apenas baixas taxas de transmissão para aplicações em redes móveis, PSTN e e RDIS de banda estreita. Apenas podia ser usado para codificar imagens pequenas como SQCIF (Sub-QCIF) e QCIF e com uma taxa baixa de *frames*. Posteriormente essa limitação foi removida, sendo possível codificar imagens de alta resolução. O algoritmo de codificação é similar ao usado pelo H.261, no entanto tem algumas melhorias e mudanças com o intuito de aumentar o seu desempenho e a recuperação de erro. O H.263 utiliza para a compensação de movimento meio *pixel* de precisão ao contrário do H.261 que utiliza um *pixel* de precisão. Pode também ser configurado para gerar menores taxas ou uma melhor recuperação de erro[Ghanbari, 1999].

O H.263 suporta cinco formatos de imagem, podendo alguns parâmetros ser vistos na Tabela 3.7 [Cherriman, 1996]. Além dos QCIF e CIF já suportados pelo H.261, suporta ainda os novos formatos SQCIF, 4CIF e 16CIF. O formato SQCIF tem aproximadamente metade da resolução do CIF, enquanto que o 4CIF e 16CIF têm respectivamente a resolução 4 e 16 vezes maiores que a resolução do CIF. Com estes dois últimos formatos o H.263 consegue competir com codificadores de elevadas taxas de resolução de vídeo, como a norma MPEG [Cherriman, 2002].

Formato de Imagem	*Pixels* por linha	Linhas por imagem	H.261	H.263	Taxa do sinal não codificado (Mbps)			
					10 *frame*/s		30 *frame*/s	
					Tons de Cinzento	Cor	Tons de Cinzento	Cor
SQCIF	128	96	Não	Sim	1.0	1.5	3.0	4.4
QCIF	176	144	Sim	Sim	2.0	3.0	6.1	9.1
CIF	353	288	Opcional	Opcional	8.1	12.2	24.3	36.5
4CIF	704	576	Não	Opcional	32.4	48.7	97.3	146.0
16CIF	1408	1152	Não	Opcional	129.8	194.6	389.3	583.9

Tabela 3.7 – Formatos de imagem suportados.

O codificador JPEG teve origem no *Joint Photographic Expert Group* do ISO/IEC tendo sido desenvolvido em colaboração com a ITU [Fluckiger, 1995] e foi inicialmente proposto para a codificação de imagens.

Este codificador permite a codificação de imagens em tons de cinzento ou a cores, utiliza a combinação da transformada discreta do co-seno, quantificação, *run-length* e técnicas de codificação de *Huffman* [Fluckiger, 1995]. Para abranger uma vasta gama de classes de qualidade de imagem e de compressão, o JPEG funciona em quatro modos de operação [Fluckiger, 1995]:

- Codificação Sequencial: é o modo utilizado por omissão, existindo um varrimento da esquerda para a direita e de cima para baixo;
- Codificação Progressiva: esta compressão permite reconstruir a imagem em múltiplas camadas de qualidade crescente, sendo a codificação feita com múltiplos varrimentos;
- Codificação sem Perdas: o processo de compressão é reversível;
- Codificação Hierárquica: esta codificação abrange diversos níveis de resolução que podem ser descodificados separadamente.

Nas imagens em movimento existem dois tipos de redundâncias: a redundância espacial e a redundância temporal. O JPEG apenas faz uso da eliminação da redundância espacial já que não tira partido da inter-dependência de informação entre *frames*. No entanto esta propriedade traz diversas vantagens: durante a transmissão a perda de uma frame não causa qualquer impacto nas frames seguintes; quando existe uma pausa na recepção do vídeo a última imagem recebida contém a informação toda; o atraso de compressão é menor pois a fonte não necessita de armazenar várias frames antes da transmissão [Fluckiger, 1995].

Capítulo 4

Implementação do Serviço sIPtel

Um dos aspectos mais importantes num serviço de comunicações é a interoperabilidade, já que esta elimina o risco dos clientes dependerem de um tipo de comunicação proprietária e possibilita às empresas verem os seus produtos comunicarem entre si. Na implementação do sIPtel houve uma constante preocupação com a interoperabilidade, tanto na parte da sinalização como na parte do transporte. Deste modo, utilizaram-se protocolos normalizados do IETF abertos à competição do poderoso mercado das comunicações e disponíveis através da Internet no seu site oficial.

Com o propósito de garantir o máximo de interoperabilidade com outros serviços da mesma classe, procurou-se também cumprir todas as regras definidas pelos protocolos utilizados, principalmente no protocolo de sinalização. A preocupação com este último justificou-se pelo facto de no desenvolvimento do serviço se implementarem características de funcionamento do protocolo SIP, o que já não aconteceu com os codificadores de meios ou com o protocolo RTP.

Este capítulo tem como objectivo descrever primeiramente as funcionalidades oferecidas pelo serviço implementado, apresentar e explicar seguidamente a sua arquitectura de construção e finalmente as partes que o compõem.

4.1 Funcionalidade do sIPtel

O sIPtel oferece a maior parte das funcionalidades de um serviço IPtel, que são disponibilizadas através da interface gráfica ilustrada na Figura 4.1. Existem ainda outras componentes da interface gráfica que podem ser vistas no Anexo A. Esta interface principal está dividida em quatro partes:

- Uma área de controlo que inclui uma barra de ferramentas que disponibiliza o acesso rápido aos comandos, uma barra deslizante que permite controlar o volume de som e um botão para ligar ou desligar o som;
- Uma área de visualização do vídeo local (à esquerda) e do stream de vídeo remoto (à direita);
- Uma área de visualização dos estados e de selecção das chamadas em curso.
- Uma zona que contém duas barras de endereços para inserção do SIP URI do chamador e do chamado.

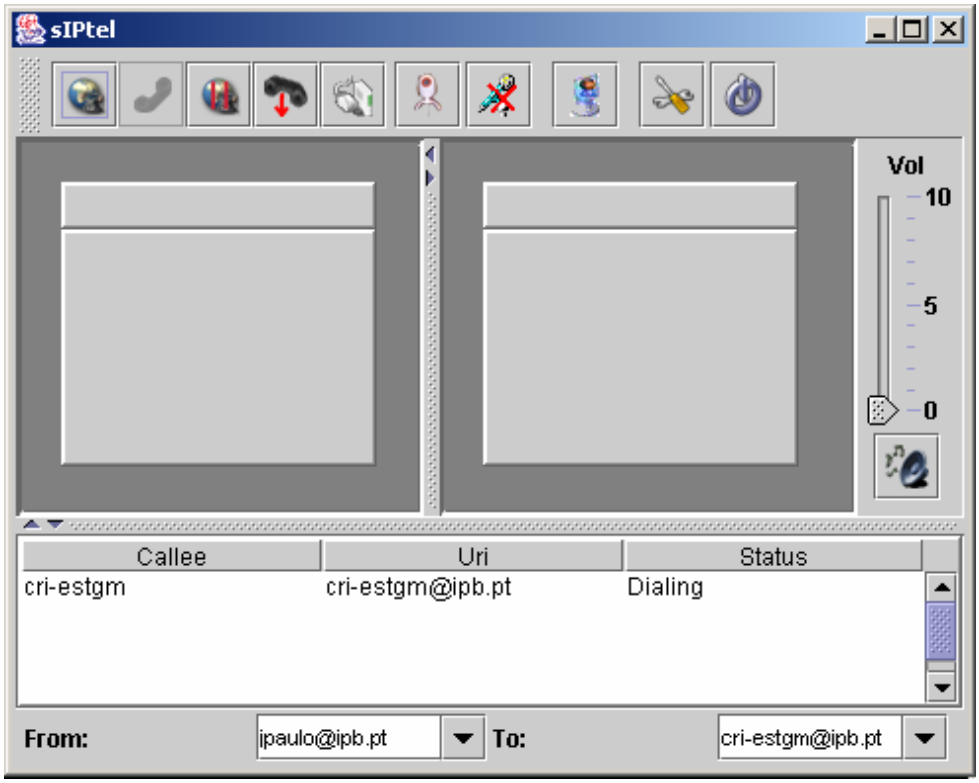


Figura 4.1 – Interface gráfica principal do sIPtel

É através desta interface gráfica que são disponibilizadas as funcionalidades do sIPtel representadas na Figura 4.2. Algumas por serem intuitivas e outras por serem inerentes ao funcionamento do protocolo SIP e já explicadas no capítulo 3.2, as referidas funcionalidades não motivo de qualquer referencia adicional.

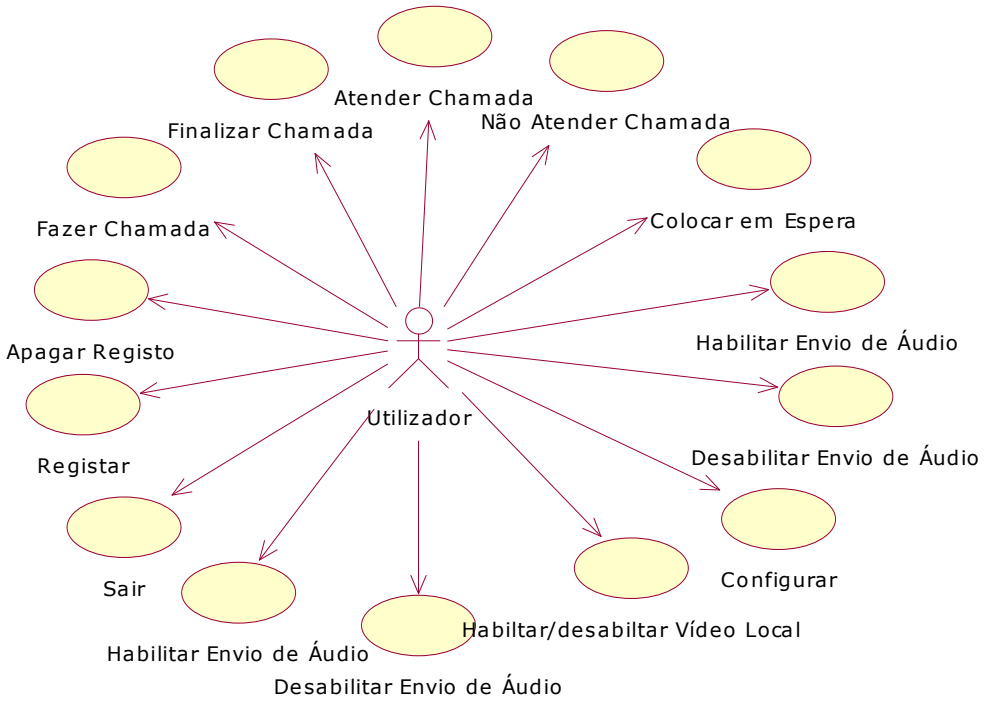


Figura 4.2 – Funcionalidades do serviço sIPtel

A expansão da opção de configuração está ilustrada na Figura 4.3, sendo possível a escolha dos codificadores e outros parâmetros de áudio e vídeo.

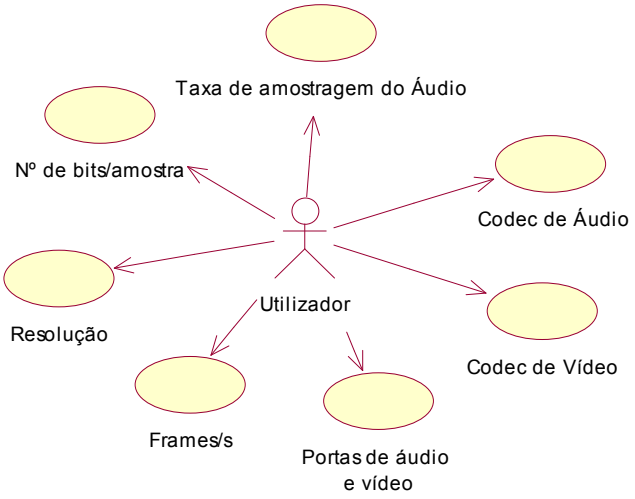


Figura 4.3 – Opções de configuração

4.1.1 Sinalização no sIPtel

No desenvolvimento da aplicação procurou-se que esta tivesse as funcionalidades mais utilizadas por um serviço típico IPtel e também opções encontradas em alguns *softwares* disponíveis na Internet. Em seguida são especificadas as características do serviço sIPtel ao

nível da sinalização:

- Suporte para uma chamada activa em qualquer momento. Entenda-se por chamada activa a troca de meios entre dois utilizadores;
- Suporte de múltiplos pedidos para a colocação de chamadas em espera;
- Suporte para múltiplas chamadas em espera;
- Suporte de dois estados de espera, isto é, o utilizador A foi colocado em espera pelo utilizador B e posteriormente devido por exemplo ao convite por parte de um utilizador C, o utilizador A colocou o utilizador B em espera de modo a que este, quando retomar a chamada, não perturbe a comunicação entre o utilizador A e o utilizador C. Posteriormente é possível a retoma da chamada por parte do utilizador A e a conclusão da chamada por parte de qualquer um dos utilizadores em qualquer momento;
- O utilizador apenas pode atender uma chamada se não tiver nenhuma chamada activa;
- O utilizador apenas pode fazer chamadas se não tiver nenhuma chamada activa;
- Possibilidade de especificar se pretende receber e/ou enviar cada um dos dois tipos de meios;
- Possibilidade de registar utilizadores com e sem autenticação;
- Permissão para a autenticação de utilizadores. Garante que o utilizador não seja incomodado por pessoas que não estejam na sua lista de utilizadores. Esta funcionalidade deve ser configurada antes de executar a aplicação.

Para o desenvolvimento do mecanismo de sinalização foi utilizada a NIST-SIP *Application Programming Interface* (API) desenvolvida pelo *National Institute of Standards and Technology* (NIST), e apresentada posteriormente no ponto 4.3.1.

4.1.2 Codificadores suportados

O sIPtel suporta cinco codificadores de áudio e dois codificadores de vídeo. Estes codificadores e todo o mecanismo de tratamento dos meios é feito recorrendo às classes da API *Java Media Framework* (JMF), disponível no site oficial da SUN. Os codificadores utilizados são os disponibilizados pelo JMF, para o áudio G.723.1, IMA/DVI, ULAW, GSM e

MPEG-Audio e para o vídeo H.263 e JPEG, não implementando nenhum *plug-in* para o suporte adicional de outros codificadores.

O sIPtel permite a alteração de codificadores durante uma chamada, bastando para isso colocar a chamada em modo de espera, proceder à alteração dos codificadores e ou parâmetros destes e retomar novamente a chamada. Todos os parâmetros das configurações dos codificadores são guardados num ficheiro, que são restaurados na próxima execução do serviço.

4.2 Arquitectura do sIPtel

Ao nível da implementação, o sIPtel está dividido em cinco partes, cada uma correspondendo a um pacote (*package*). Na Figura 4.4 são apresentadas as dependências entre os diversos pacotes que são enunciados em seguida:

- ipb.iptel.gui: contém as classes que proporcionam a interface com o utilizador e a apresentação dos meios;
- ipb.iptel.sip: contém as classes que permitem implementar a parte de sinalização de chamadas;
- ipb.iptel.rtp: fazem parte deste pacote as duas classes responsáveis pelo envio e recepção de meios;
- ipb.iptel.util: contém classes utilitárias utilizadas por várias classes contidas nos outros pacotes;
- ipb.iptel.test: contém dois utilitários que permitem testar a reprodução de áudio e vídeo no sistema.

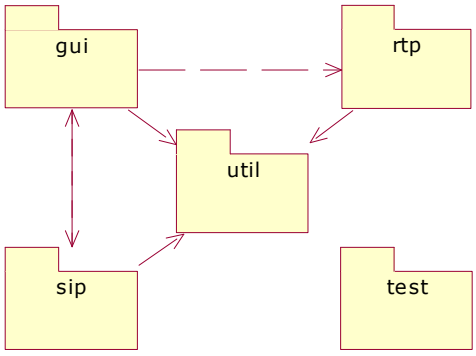


Figura 4.4 – Dependências entre os pacotes que compõem o sIPtel

No desenvolvimento da parte de sinalização do sIPtel, foi necessário ter em consideração a estrutura do protocolo SIP e as regras definidas em cada um dos seus níveis, assim como o modo de operação deste protocolo e algumas definições e mensagens utilizadas por este, anteriormente abordadas no capítulo 3.2. Em seguida são explicados os níveis do protocolo, dando-se maior importância ao nível de transacção pelo facto de ter sido totalmente implementado seguindo as especificações definidas na RFC 3261.

4.2.1 Nível de transacção

O protocolo SIP está estruturado como um protocolo de camadas, em que o nível mais baixo é a análise da sintaxe das mensagens e a sua codificação. A segunda camada é o nível de transporte, que define como um cliente envia pedidos e recebe respostas e como um servidor recebe pedidos e envia respostas através da rede. A terceira camada é o nível transacção, que trata das retransmissões ao nível aplicacional, associa as respostas aos pedidos e lida com os *timeouts*. A camada superior ao nível de transacção é chamada *Transaction User* (TU) e será analisada posteriormente. Em seguida são explicadas as funcionalidades definidas pelo nível de transacção e pelo TU, isto porque o nível de sintaxe e codificação, e o nível de transporte são disponibilizados pela API NIST-SIP, utilizada para a implementação da parte de sinalização.

Uma transacção SIP ocorre entre um cliente e um servidor e engloba todas as mensagens desde o primeiro pedido enviado pelo cliente ao servidor, até receber do servidor uma resposta do tipo final (não-1xx). Se o pedido for um INVITE e a resposta final não for do tipo 2xx, a transacção inclui o envio de um ACK à resposta. O envio de um ACK a uma resposta 2xx de um INVITE, é uma transacção separada [RFC 3261, 2002].

Todas as transacções têm um lado cliente e outro lado servidor. O lado cliente é conhecido como uma transacção cliente e o lado servidor como uma transacção servidor. Uma transacção cliente é responsável por receber os pedidos de um TU, entregar esses pedidos a uma transacção servidor, e finalmente receber e processar as respostas dessa transacção. Um TU é um nível do processamento do protocolo que reside acima do nível de transacção e pode ser um UA ou um *Stateful Proxy*. O objectivo de uma transacção servidor é receber pedidos do nível de transporte e entregá-los ao TU, aceitar as respostas deste e entregá-las novamente ao nível de transporte para as transmitir para a rede. Estes dois tipos de transacções são representados no nível de transacção por máquinas de estado finitas, criadas para processar

cada um dos pedidos ou respostas. Cada transacção, segundo a RFC 3261, é identificada pelo parâmetro '*branch'* do cabeçalho *Via*. Se este parâmetro não existir ou não contiver um conjunto de caracteres que identifica que o nível de transacção é definido segundo a RFC 3261, devem ser seguidas as regras de identificação de uma transacção segundo a RFC 2543.

Existem dois tipos de máquinas de estados para uma transacção cliente [RFC 3261, 2002]:

- *Invite Client Transaction*: processa as transacções cliente para pedidos INVITE;
- *Non Invite Client Transaction*: trata de todas as transacções do tipo cliente para todos os pedidos excepto o INVITE e o ACK.

De igual modo existem dois tipos de máquinas de estado para uma transacção servidor [RFC 3261, 2002]:

- *Invite Server Transaction*: se o pedido recebido for um INVITE;
- *Non Invite Server Transaction*: se for um pedido qualquer excepto o INVITE e o ACK.

Para facilitar a sequência de mensagens e o encaminhamento apropriado de pedidos entre os UAs, existem os *diálogos* (*Dialog*) a um nível superior. Um *diálogo* é uma relação ponto a ponto entre dois UAs que persiste por um determinado tempo e é estabelecido quando existe uma resposta 2xx a um INVITE. Um *diálogo* é identificado por um *ID*, que consiste no campo *Call-ID*, no parâmetro *tag* do cabeçalho *From* e no parâmetro *tag* do cabeçalho *To*. Um *diálogo* tem influência no modo de funcionamento do protocolo porque um UA não pode, por exemplo, enviar um pedido BYE para terminar a chamada fora de um *diálogo*.

Em seguida é descrito o funcionamento dos diagramas de estado *Invite Client Transaction* e *Invite Server Transaction*, como auxílio à explicação dos processos de envio e recepção de um INVITE por parte do UAC e UAS, respectivamente. Devido ao funcionamento dos diagramas de estado *Non Invite Client Transaction* e *Non Invite Server Transaction* ser similar aos explicados nos dois pontos seguintes, estes são apenas apresentados no Anexo B.

4.2.1.1 Diagrama de estados para uma transacção cliente no envio de um INVITE

Uma transacção INVITE consiste num *three-way handshake*. A transacção cliente envia um INVITE, a transacção servidor uma resposta final e por fim a transacção cliente envia um ACK. Na Figura 4.5 é ilustrado o diagrama de estados para uma *Invite Client Transaction* e em seguida é explicada uma transacção cliente para o envio de um INVITE. Esta explicação é

baseada na RFC 3261, pelo que deverá ser consultada para um maior esclarecimento.

Quando o TU ordena o envio de um INVITE, este é passado ao nível de transporte para ser transmitido e a máquina de estados passa ao estado *Calling*. Se estiver a ser utilizado um protocolo sem confirmação (ex. UDP) para o envio de mensagens SIP é retransmitido um INVITE cada vez que o *Timer A* dispara. Após a entrada no estado *Calling* é activado o *Timer B* para qualquer tipo de protocolo de transporte que funciona como um *timeout* para a recepção de uma resposta. Caso esse *timeout* seja atingido ou ocorra um erro de transporte no envio do INVITE, a máquina de estados passa ao *Terminated* e o TU é avisado do acontecimento. Se a transacção cliente receber uma resposta provisória (1xx) enquanto está no estado *Calling*, transita para o estado *Proceeding* e informa o utilizador. No estado *Proceeding* a transacção cliente não deve transmitir mais pedidos e todas as repostas recebidas são transmitidas ao utilizador.

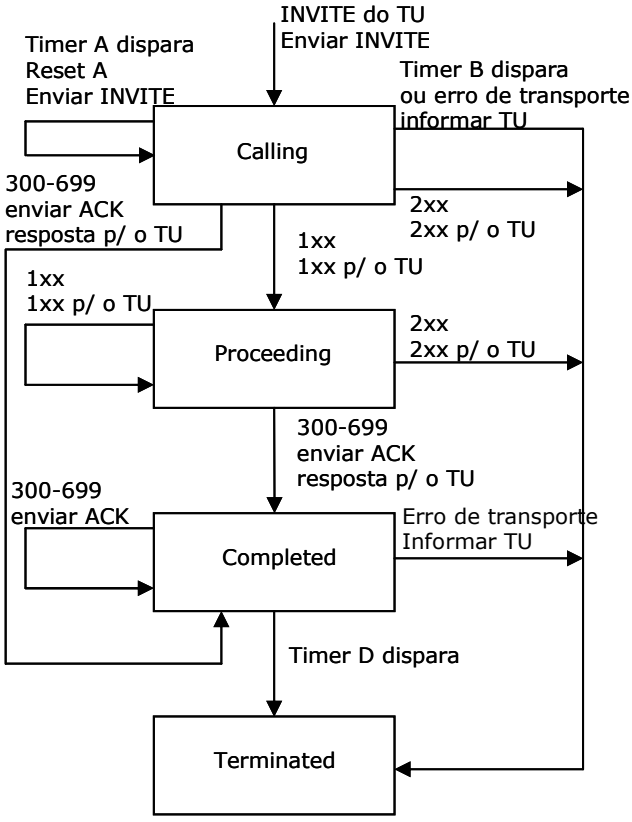


Figura 4.5 – Diagrama de estados Invite Client Transaction

Se durante o estado *Calling* ou *Proceeding* chegar uma resposta entre 300 e 699 a máquina de estados comuta para o estado *Completed*, o TU é informado e é enviado um ACK para o mesmo endereço e porta que o primeiro pedido, mesmo no caso de serem utilizados protocolos com confirmação. Quando a transacção cliente entra no estado *Completed* é

iniciado o *Timer D* com o valor de pelo menos 32 segundos para protocolos sem confirmação e zero segundos para protocolos com confirmação. Qualquer retransmissão da resposta final recebida enquanto a máquina de estados se encontre no estado *Completed* deve causar o envio do ACK.

A transacção cliente deve passar para o estado *Terminated* em qualquer dos seguintes casos: quando o *Timer D* disparar; se ocorrer algum erro de transporte e nesse caso o *Timer D* é cancelado e o TU informado; na recepção de uma reposta 2xx quando a máquina de estados se encontra no estado *Calling* ou *Proceeding*, sendo neste caso o TU informado da resposta recebida e estabelecido um *diálogo* entre o UAC e o UAS.

Quando a transacção cliente entra no estado *Terminated* deve ser destruída. Para um UAC a recepção de uma resposta final gera ainda o envio de um ACK que é passado ao nível de transporte e não ocorre qualquer processamento ao nível da transacção [RFC 3261, 2002]. Após o envio do ACK a aplicação inicia a sessão. O ACK é enviado para o mesmo endereço e porta, através do mesmo protocolo de transporte que foi usado para o primeiro pedido que foi enviado.

4.2.1.2 Diagrama de estados para uma transacção servidor na recepção de um INVITE

Para processar um INVITE recebido é criada uma transacção servidor que opera segundo o diagrama de estados ilustrado na Figura 4.6.

Após a chegada do INVITE a máquina de estados entra no estado *Proceeding*, o TU deve ser informado da nova chamada e a transacção servidor tem que transmitir uma resposta *100 Trying*, se o UAS não gerar uma resposta final dentro de 200 milissegundos; contudo é possível enviar uma resposta *100 Trying* antes desse tempo. Enquanto a transacção servidor permanecer no estado *Proceeding*, todas as respostas recebidas do TU devem ser transmitidas pelo nível do transporte. Se for recebida uma retransmissão de um INVITE durante este estado deve ser transmitida a última resposta provisória passada pelo TU para retransmissão.

Se durante o estado *Proceeding* o TU passar uma reposta 2xx à transacção servidor esta tem que passar a resposta ao nível de transporte para ser transmitida e a transacção servidor comuta para o estado *Terminated*. Ainda no estado *Proceeding*, se o TU passar uma resposta entre 300 e 699 à transacção servidor, esta tem de passar a resposta ao nível do transporte para ser transmitida e a transacção servidor comuta para o estado *Completed*.

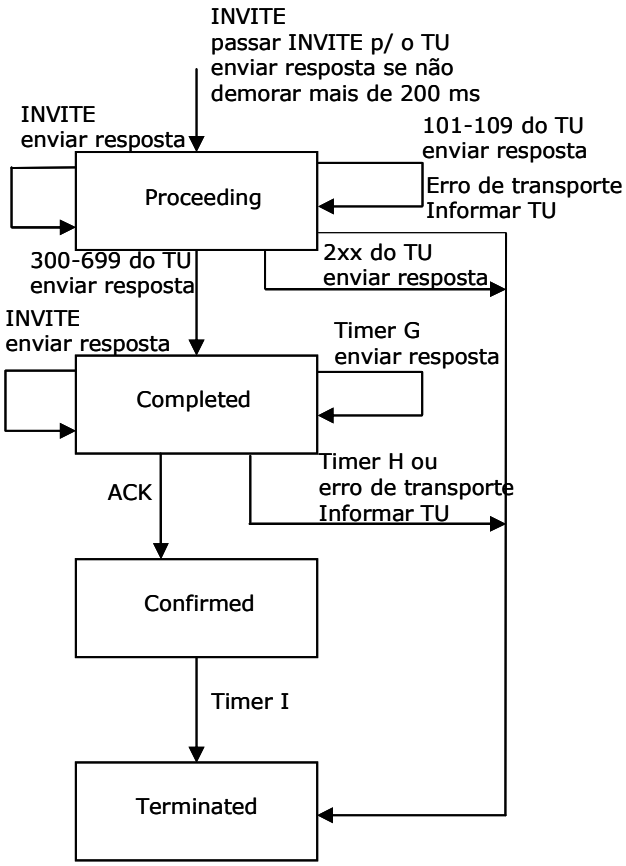


Figura 4.6 – Diagrama de estados *Invite Server Transaction*

Após ter entrado no estado *Completed* o *Timer H* é activado independentemente do tipo de transporte. Este *timer* determina o tempo dado para a retransmissão de respostas e tem o mesmo valor que o *Timer B*. Cada vez que o *Timer G* dispara, a resposta é retransmitida e em seguida o *Timer G* é iniciado com o *min(2*T1, T2)*, isto é, com o valor 2*T1 se este não exceder o valor de T2, senão é iniciado com o valor T2. Se durante o estado *Completed* é recebido um ACK a transacção servidor tem que passar para o estado *Confirmed* e cancelar os *timers* activos. Se ainda durante o estado *Completed* disparar o *Timer H* ou houver um erro de transporte, o utilizador tem de ser informado do tipo de acontecimento ocorrido e a transacção servidor passa ao estado *Terminated*.

Uma vez no estado *Confirmed*, se forem usados protocolos sem confirmação o *Timer I* deve ser configurado com o valor de T4 e com o valor de zero segundos se forem usados protocolos com confirmação. O tempo T4 representa a totalidade de tempo que a rede leva para apagar mensagens entre transacções clientes e servidores [RFC 3261, 2002]. Quando o *Timer I* dispara, a transacção servidor passa ao estado *Terminated* e é destruída em seguida.

4.3 Descrição das APIs

Para o desenvolvimento da parte de sinalização e para o suporte de meios no sIPtel, foram utilizadas respectivamente as NIST-SIP API e a JMF API, com distribuição gratuita e descritas em seguida.

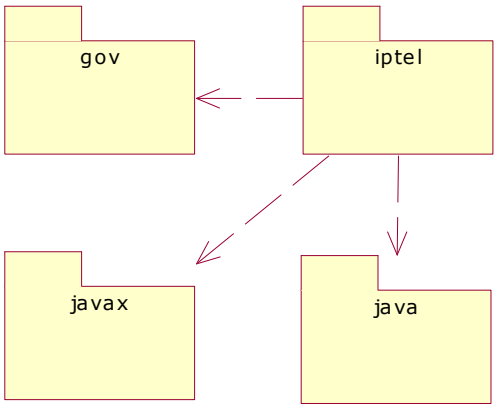


Figura 4.7 – Dependências entre API's utilizadas

4.3.1 API NIST-SIP

A distribuição NIST-SIP [NIST-SIP, 2002] é o resultado de um trabalho incluído no projecto "*Internet Telephony/VOIP*" em desenvolvimento por um grupo de investigação pertencente ao *Advanced Networking Technologies Division* (ANTD) do *National Institute of Standards and Technology* (NIST), que se dedica à investigação, normalização e testes na área das redes. Esta distribuição tem como objectivo simplificar a construção de componentes SIP como *User Agents* (UA) e Servidores SIP. A NIST-SIP API contém um conjunto de *parsers* para analisar as mensagens e cabeçalhos/campos para o SIP e SDP, uma SIP *stack* e uma implementação JAIN-SIP certificada e baseada nos *parsers* e na *stack* NIST-SIP. Os pacotes usados para a implementação do serviço são descritos em seguida ao nível do seu conteúdo e ou funcionalidade.

4.3.1.1 Pacote gov.nist.sip

Contém os objectos de base para os cabeçalhos, campos e mensagens dos pacotes *nist.sip.net*, *nist.sip.sipheaders*, *nist.sip.sdpfields* e *nist.sip.msgparser*.

4.3.1.2 Pacote gov.nist.sip.msgparser

Contém as classes que suportam as mensagens SIP segundo a RFC 2543 e SDP, e *parsers* que têm como função analisar mensagens SIP e SDP e endereços URL. Disponibiliza interfaces para fazer a análise das mensagens SIP e SDP.

4.3.1.3 Pacote gov.nist.sip.net

Contém as classes comuns ao SIP e SDP relacionadas com a rede.

4.3.1.4 Pacote gov.nist.sip.sdpfields

Contém as classes dos vários campos SDP definidos na RFC 2327.

4.3.1.5 Pacote gov.nist.sip.sipheaders

Contém as classes dos cabeçalhos SIP definidos segundo a RFC 2543 bis 02 e cabeçalhos HTTP.

4.3.1.6 Pacote gov.nist.sip.stack

Este pacote contém as classes para a implementação da *stack* SIP. A NIST-SIP *stack* é essencialmente um nível de mensagens que recorre ao *parser* NIST-SIP para analisar as mensagens e define abstracções para o processamento de mensagens e tratamento de Entrada/Saída de mensagens SIP. Quando chega uma mensagem à *stack*, esta é processada pelo analisador de mensagens NIST-SIP e em seguida é chamada a implementação da interface *SIPStackMessageFactory*, para criar um objecto *SIPServerRequest* ou *SIPServerResponse*, dependendo se a mensagem é um pedido ou uma resposta.

A *stack* NIST-SIP disponibiliza as seguintes funções:

- Entrada/Saída: recebe e envia mensagens através de *sockets* TCP e UDP, permitindo a total abstracção de funções de I/O ao nível da aplicação;
- Formatação de mensagens: gera mensagens;
- Segurança: disponibiliza um nível de abstracção para os métodos de autenticação definidos no pacote *gov.nist.sip.stack.security*;
- Registo: disponibiliza funcionalidades de registo das mensagens em ficheiros de texto;
- Transacção: disponibiliza um nível de transacção de acordo com o parágrafo 17 da

RFC 2543 Bis 09 e é implementado através das classes contidas no pacote *gov.nist.sip.stack.transaction*. No entanto não foi utilizada esta funcionalidade, porque foi adicionada posteriormente à realização do sIPtel.

4.3.1.7 Pacote gov.nist.sip.stack.security

Este pacote, permite o suporte de segurança para a *stack* NIST-SIP e contém a implementação dos métodos de autenticação *Basic* e *Digest*.

4.3.2 API JMF

A Java Media Framework (JMF) é uma API que permite incorporar dados como áudio e vídeo em aplicações Java e *applets*. Esta API pode ser dividida em dois níveis: um nível superior chamado JMF *Presentation and Processing* API, que controla a captação, processamento e apresentação de meios, e um nível inferior chamado JMF Plug-in API, que permite a configuração e extensão. A API JMF suporta ainda a captura e reposição de dados, o desenvolvimento de aplicações em Java com suporte para *streams* e o desenvolvimento proprietário de codificadores, multiplexadores, desmultiplexadores e processadores de efeitos, entre outros.

Algumas classes e interfaces pertencentes à API JMF 2.0 utilizadas para o suporte de meios são explicadas em seguida:

- *CaptureDeviceManager*: mantém o registo dos dispositivos de captura dos meios disponíveis. É utilizado para saber que dispositivos estão disponíveis e para obter informação sobre eles;
- *DataSource*: é utilizada para controlar a entrega de dados;
- *MediaLocator*: esta classe permite registar informação sobre parâmetros utilizados na sessão, como o endereço, portas, tipo de dados, TTL e SSRC. Tem um formato similar a um URL, "*rtp://endereço:porta[:SSRC]/tipo-de-dados/[TTL]*". Os parâmetros entre parêntesis rectos são opcionais;
- *SessionAddress*: esta classe é utilizada para representar dois endereços e as portas respectivas. Um dos pares *endereço:porta* é utilizado para as sessões RTP e outro para as sessões RTCP. Segundo a RFC 1889 a sessão RTP utiliza apenas as portas pares e a sessão RTCP utiliza as portas ímpares;

- *Manager*: permite o acesso a mecanismos para a criação de *Players*, *Processors*, *DataSources* e *DataSinks*;
- *SessionManager*: é usada para coordenar uma sessão RTP. Mantém o estado da sessão do ponto de vista do participante, definindo métodos para iniciar, participar e encerrar sessões, adicionar e remover *streams*;
- *Player*: é um objecto utilizado para processar a entrada de um *stream* e reproduzi-lo num dispositivo físico em tempo adequado. Permite o acesso a um componente de controlo e a um componente visual onde o meio é exibido. No caso de ser áudio a componente visual não existe e então o seu valor é *null*;
- *Processor*: é um objecto semelhante ao *Player* utilizado para processar e entregar dados;
- *SendStream*: os *streams* dentro de uma sessão RTP são representados pelos objectos *ReceiveStream* e *SendStream*. O *SendStream* representa o *stream* de dados que chega do *Processor* e vai ser enviado através da rede para o participante remoto;
- *ReceiveStream*: representa o *stream* que está a ser recebido do participante remoto;
- *DataSink*: é um objecto utilizado para ler dados do *DataSource* e escrever esses dados noutro destino, geralmente outros dispositivos que não sejam de apresentação. Por exemplo o *DataSink* pode escrever dados num ficheiro ou enviá-los para a rede;
- *Manager*: esta classe permite a construção de *Player*, *Processors*, *DataSources* e *DataSinks*.

No envio e na recepção de meios, os *streams* de áudio e de vídeo são tratados de forma independente, não partilhando qualquer uma das entidades *DataSource*, *Processor* ou *DataSink*. A Figura 4.8 mostra o fluxo de dados entre as diversas entidades utilizadas no sIPtel para a transmissão e recepção de meios.

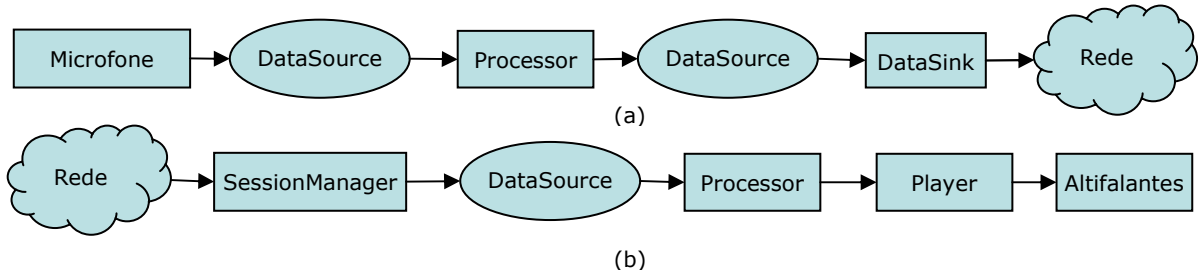


Figura 4.8 – Conexões JMF a alto nível para: (a) o envio e (b) recepção dos meios

No esquema (a) da Figura 4.8, a voz é capturada através de um dispositivo físico, no caso particular um microfone, e é entregue a um *DataSource*. Em seguida o *Processor* recebe e codifica o *stream*, criando um novo *DataSource* que é passado ao *DataSink* que tem a função de enviar o *stream* através da rede. O esquema (b) da Figura 4.8 ilustra o cenário do utilizador que recebe o *stream* de áudio através da rede. É criado um *SessionManager* que constrói e mantém o objecto que representa o *stream* recebido, criando em seguida um *DataSource* que é entregue ao *Processor*, que processa o *stream* (ex. descodifica o *stream*) e entrega-o ao *Player* que neste caso faz a reprodução da voz nos altifalantes.

4.4 Código do sIPtel

Algumas das partes que estão incluídas na implementação deste serviço foram já abordadas anteriormente ao nível do funcionamento: da parte de sinalização, o nível de transporte e o nível de transacção, e da parte de meios, os componentes utilizados no processamento dos dados áudio e vídeo.

Os diagramas de classes ilustrados neste capítulo têm como objectivo descrever a arquitectura e a estrutura estática do sistema. Os diagramas de sequência pretendem representar o fluxo de processamento do sistema.

4.4.1 Classes

Neste ponto são explicadas as classes relacionadas com a interface gráfica do utilizador, a sinalização, a implementação do nível de transacção, e o envio, recepção e apresentação dos meios.

4.4.1.1 Classe sIPtel

A classe sIPtel implementa a interface gráfica do utilizador (*Graphical User Interface – GUI*). Esta classe admite como parâmetro na linha de comandos o nome de um ficheiro com a configuração necessária para iniciar a *stack* NIST-SIP. Este ficheiro (*configuration.properties*) contém informação para:

- Habilitar/desabilitar a autenticação de utilizadores;
- Especificar o protocolo de transporte para a sinalização e a porta. Por omissão o protocolo de transporte para o SIP é o UDP e a porta 5060;
- O nome da *stack*. Pode ter o formato *nome.dominio*;
- O endereço IP onde a *stack* é iniciada. Nesta implementação é um endereço IP de um computador;
- O nome do ficheiro no qual são gravadas as mensagens SIP recebidas e enviadas;
- O endereço do *Outbound Proxy*. Este endereço é obrigatório para o correcto funcionamento do serviço.

Através da interface gráfica desta classe, são também disponibilizadas as funcionalidades enunciadas no ponto 4.1, com a excepção da configuração dos parâmetros da *stack*.

A descrição das classes ilustradas na Figura 4.9 é apresentada em seguida. A classe sIPtel deriva da classe *JFrame* e permite criar uma janela com um título e uma borda. A classe *TableScrollPane* implementa uma tabela que informa o utilizador sobre o estado das chamadas e possibilita a selecção da chamada na qual o utilizador pretende intervir. A classe *VideoScrollPane* permite criar um (ou mais) *VideoPanel* onde o vídeo local ou remoto é apresentado. Foram criados dois objectos *VideoScrollPane*, um para suportar um objecto *VideoPanel*, onde é apresentado o vídeo local, e outro para suportar vários *VideoPanels* dos *streams* de vídeo remotos. A classe *Sounds* reproduz ficheiros de áudio em formato *wave* e é usada para sinalizar quando o utilizador recebe uma chamada. A interface *Player* é utilizada para apresentar e controlar o vídeo local e o áudio remoto. A classe *Processor* pertence à API JMF e permite processar cada um dos meios locais (áudio e vídeo) antes de serem transmitidos. A classe *ServerMain* inicia a *stack* e faz a ligação entre a parte gráfica e as classes responsáveis pela implementação da parte de sinalização.

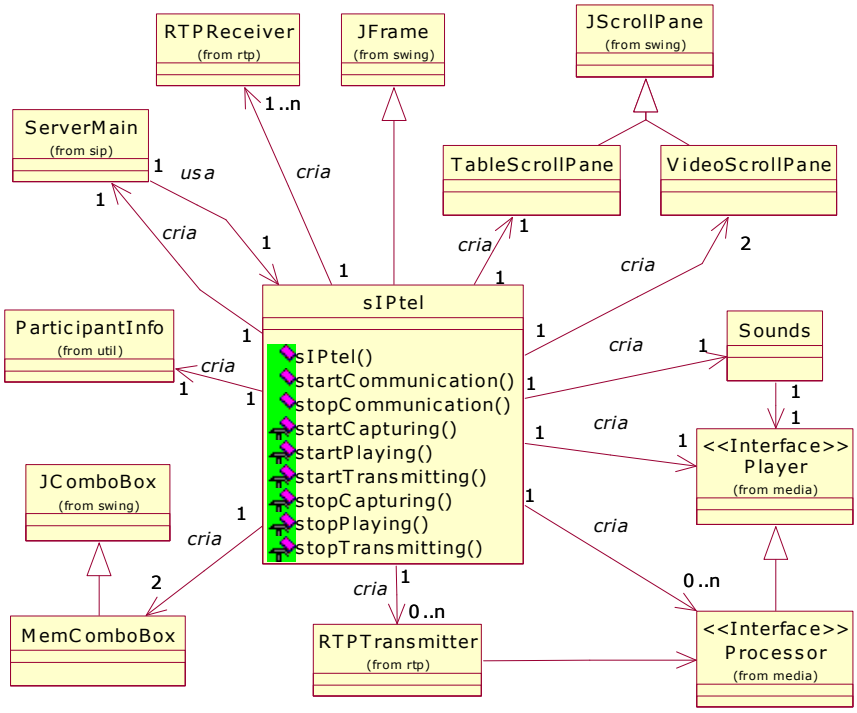


Figura 4.9 – Componente principal: diagrama de classes

A classe *MemComboBox* deriva da classe *JComboBox* e permite ler e armazenar num ficheiro os contactos introduzidos durante a utilização do serviço.

4.4.1.2 ServerMain

A classe *ServerMain* implementa o ponto de referência principal para todas as classes da parte de sinalização. É nesta classe que é criada a *stack* NIST-SIP configurada com os parâmetros do ficheiro *configuration.properties* explicado no ponto 4.4.1.1. Esta *stack* utiliza o *parser* NIST-SIP que analisa as mensagens recebidas e invoca a classe *SIPStackMessageFactoryImpl*. Esta última cria um *SIPServerRequest* ou *SIPServerResponse* dependendo da mensagem ser um pedido ou uma resposta e chama o método *processRequest* de um destes objectos. A criação e envio dos pedidos SIP são feitos através da classe *HandlerRequests*.

A classe *ServerMain* permite ainda registar o estado de todas as transacções de sinalização de modo a implementar as quatro máquinas de estado do nível de transacção referidas anteriormente no ponto 4.2.1. O nível de transacção inexistente durante o desenvolvimento deste serviço na API NIST-SIP, foi implementado segundo a RFC 3261 e é responsável pela retransmissão e associação das respostas aos pedidos e pelos *timeouts*. O estado das transacções é partilhado por várias classes sIPtel: *ServerMain*, *HandlerRequests*, *SipServerResponse*, *SIPServerRequest* e *Timers*. A classe *Timers*, por exemplo, permite criar

os diversos *timers* necessários para a retransmissão de mensagens e *timeouts* exigidos pelo nível de transacção, consultando os estados das transacções para verificar se deve ou não alterar o seu estado.

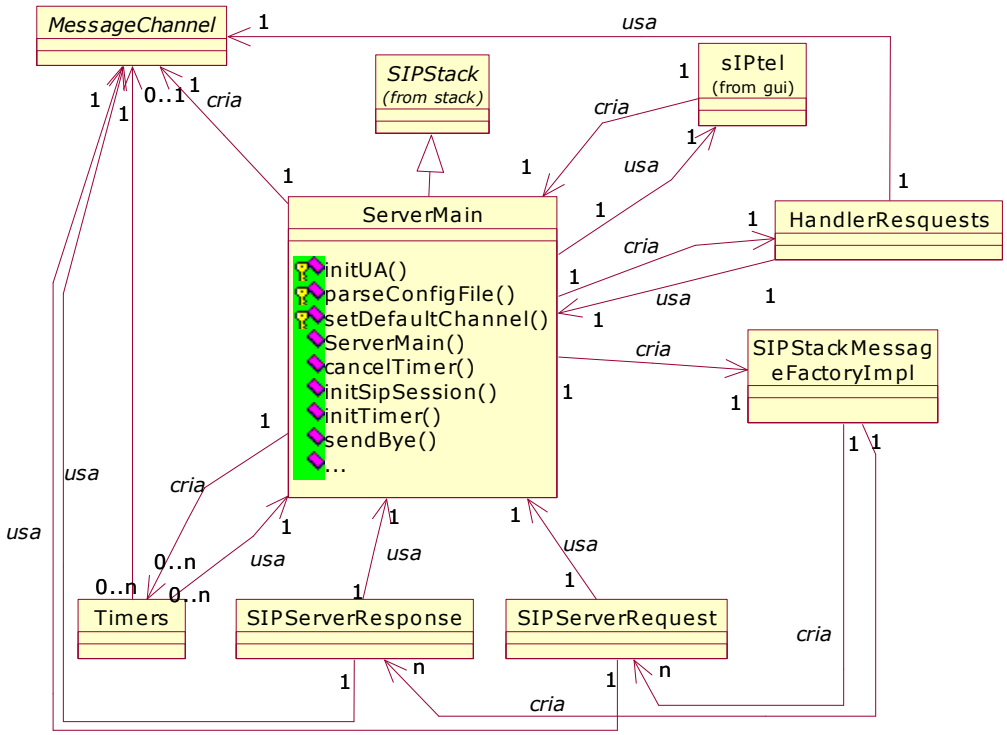


Figura 4.10 – Componentes de sinalização: diagrama de classes

4.4.1.3 HandlerRequests

Esta classe tem a responsabilidade de construir e enviar os pedidos REGISTER, INVITE, CANCEL e BYE, registar as transacções do tipo cliente criadas e actualizar a tabela de estados das chamadas em curso. É também nesta classe que é tomada a decisão do envio de pedidos que podem variar com o estado da transacção à qual pertence esse pedido, como o caso do envio de um BYE ou CANCEL para finalizar uma chamada.

A classe *HandlerRequests* implementa as interfaces *sIPtelConstants* e *SIPKeywords*. A primeira interface define uma lista de diversas constantes utilizadas pelo sIPtel e a segunda, que é disponibilizada pela API NIST-SIP, define uma lista de constantes usadas pelos analisadores e classes que implementam os cabeçalhos e as mensagens SIP.

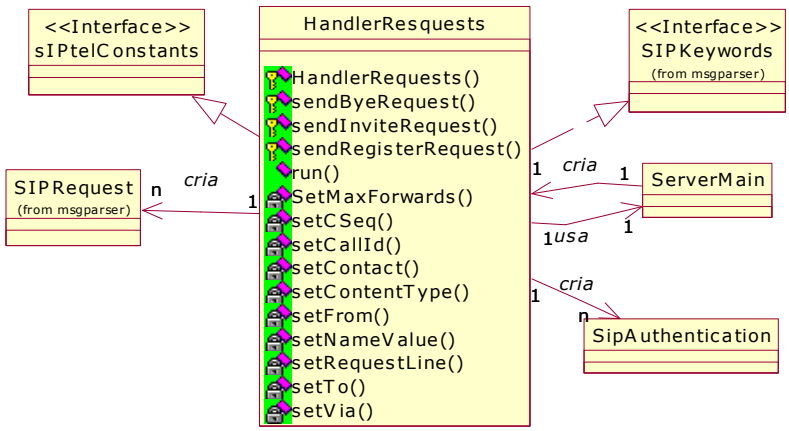


Figura 4.11 – Envio de pedidos: diagrama de classes

Quando é seleccionado o envio de um pedido REGISTER com um cabeçalho de autenticação para autenticar o pedido no servidor *Registrar* é utilizada a classe *SipAuthentication*. Esta classe, com base em informação introduzida pelo utilizador (nome do utilizador, *realm* e *password*), gera informação cifrada para os campos *nonce* e *response* do cabeçalho *Authentication*.

4.4.1.4 SIPServerRequest

Esta classe implementa a interface *SIPServerRequestInterfacen* que permite, cada vez que chegue uma mensagem SIP, criar um objecto *SIPServerRequest* ou *SIPServerResponse* pelo método *newSIPServerRequest* ou *newSIPServerResponse* da classe *SIPStackMessageFactoryImpl*, dependendo se é um pedido ou uma resposta. O objecto *SIPServerRequest* é utilizado para processar os pedidos e o *SIPServerResponse* para processar as respostas. As restantes interfaces implementadas por esta classe permitem a utilização de constantes por elas definidas.

Quando é criado um objecto *SIPServerRequest* são passados dois objectos, o *SIPRequest* e o *MessageChannel* e em seguida é invocado o método *SIPServerRequest.processRequest* que identifica o tipo de pedido, executando em seguida um bloco de código que permite responder a esse pedido. Este processamento do pedido inclui a actualização da tabela de estado das transacções, o envio de repostas, a actuação sobre os meios e a actualização da tabela que informa o utilizador sobre os estados das chamadas.

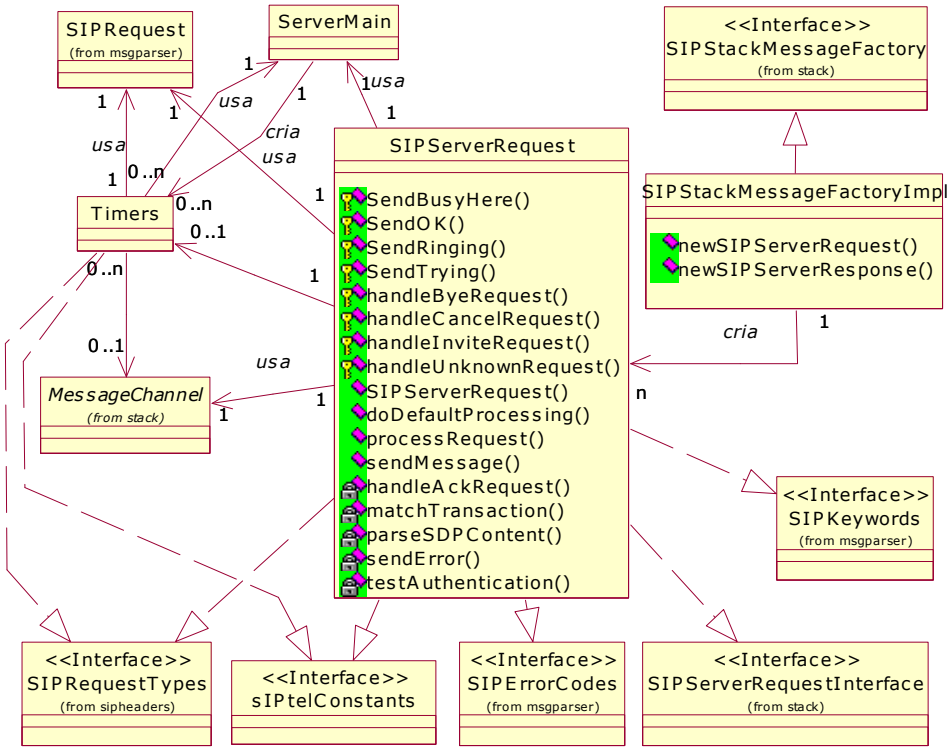


Figura 4.12 – Recepção de pedidos: diagrama de classes

4.4.1.5 SIPServerResponse

Esta classe implementa a interface *SIPServerResponseInterface* que permite criar um objecto *SIPServerResponse* cada vez que a *stack* recebe uma resposta e invocar o método *SIPServerResponse.processResponse* para a processar. Também aqui é identificado o tipo de resposta e a transacção a que esta pertence, e é feita a actualização do estado da transacção e da tabela de estados da interface gráfica, entre outras acções.

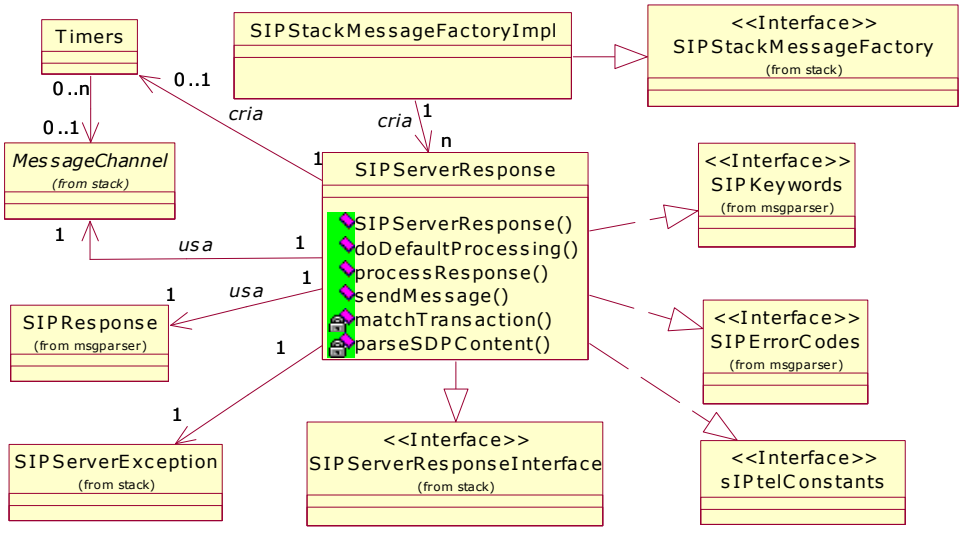


Figura 4.13 – Recepção de respostas: diagrama de classes

Tal como a classe *SIPServerRequest* a classe *SIPServerResponse* implementa as interfaces *SIPKeywords*, *SIPErrorCodes* e *sIPtelConstants* para aceder a diferentes tipos de constantes.

4.4.1.6 Classe RTPReceiver

A classe *RTPReceiver* está contida no pacote *ipb.iptel.rtp* e é responsável por detectar, processar e apresentar os *streams* que chegam. Esta classe implementa três interfaces:

- *ControllerListener*: através desta interface é possível detectar eventos gerados pelos objectos *Controller*. Nesta classe o *Player* é registado como *ControllerListener* o que permite determinar qual o estado do *Player* e responder aos eventos gerados por este. Nesta implementação a interface é explorada para detectar problemas no funcionamento do *Player*;
- *ReceiveStreamListener*: recebe notificações das mudanças de estado do *stream* RTP que está a ser recebido. Com a implementação desta interface a classe *RTPReceiver* fica habilitada a receber notificações quando: novos *streams* são criados, a transferência de dados é iniciada ou finalizada, a transferência de dados excedeu o tempo previsto, é feita a atribuição de um participante a um *ReceiveStream* anteriormente órfão, acontece a recepção de um pacote RTCP APP, os dados do *stream* ou o seu formato foram alterados. Nesta implementação foi utilizado o evento *NewReceiveStreamEvent* que é gerado quando é detectada a recepção de um novo *stream*;
- *RemoteListener*: permite receber notificações de eventos ou mensagens de controlo vindas de participantes remotos. Esta interface é utilizada com o objectivo de monitorizar alguns parâmetros da sessão, não tendo no entanto qualquer função de decisão no envio de pacotes subsequentes.

A classe *InetAddress* permite encapsular um endereço IP para o utilizador local e utilizador remoto. Esta classe é utilizada para criar dois objectos *SessionAddress*, o primeiro contém informação para preparar a sessão do *SessionManager* e o segundo para iniciar a sessão, aos quais são passados como parâmetros dois objectos *InetAddress*.

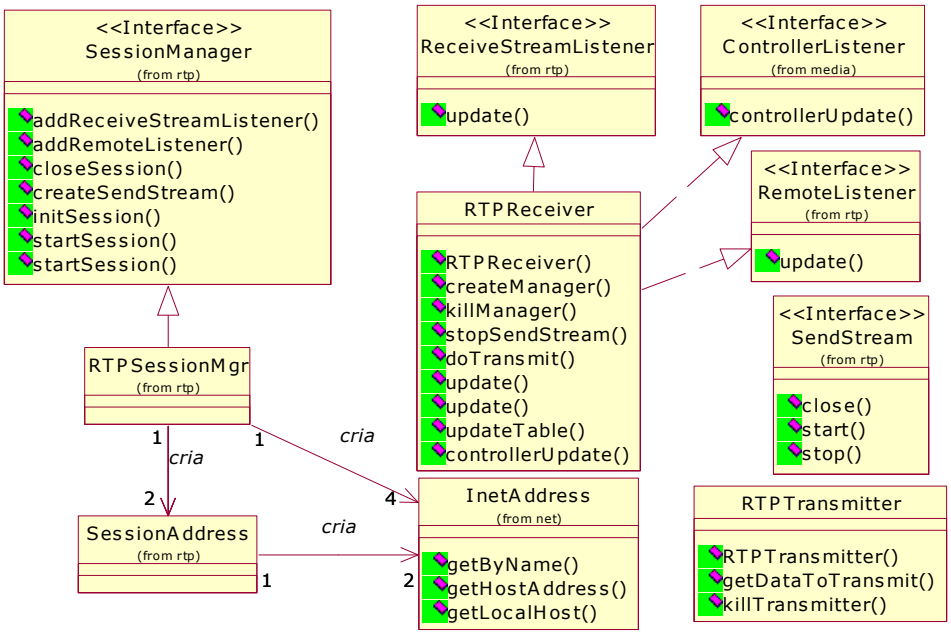


Figura 4.14 – Componente da recepção do *stream*: diagrama de classes

A Interface *SendStream* é utilizada no caso de ser invocado o método *RTPReceiver.doTransmit*. Existe uma opção de configuração inicial em que permite ao utilizador optar por não enviar áudio ou vídeo numa chamada; se mais tarde desejar habilitar o envio destes meios é invocado este método, que cria um objecto *RTPTransmitter*, passando-lhe como parâmetros um *Processor* previamente configurado e o tipo de meio a enviar. Através do método *RTPTransmitter.getDataToTransmit* é retornado o *DataSource* para transmitir. O método *SessionManager.createSendStream* cria um *SendStream* para o envio do meio através do *SessionManager*, dado este conhecer o endereço remoto para onde o meio será enviado. Para parar o envio do *stream* é utilizado o método *stopSendStream*.

4.4.1.7 Classe RTPTransmitter

A classe *RTPTransmitter* está contida no pacote *ipb.iptel.rtp* e é responsável pela codificação e envio de *streams* multimédia. O objecto *DataSource* é utilizado para entregar o *stream* codificado ao *DataSink,* que o transmite para a rede.

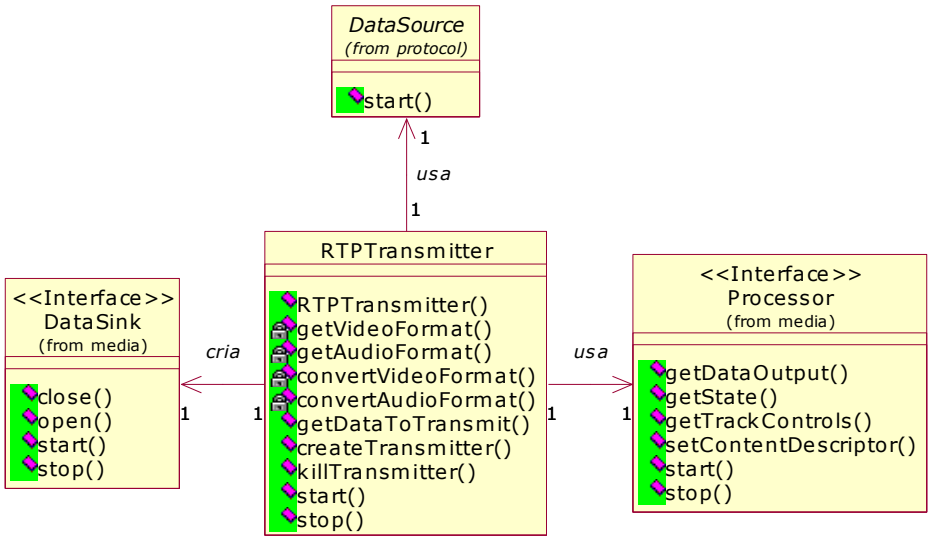


Figura 4.15 – Componente de transmissão do *stream*: diagrama de classes

Os métodos *getAudioFormat* e *getVideoFormat* são privados (apenas são vistos pela própria classe) e retornam os codificadores seleccionados através da opção de configuração. Estes métodos são utilizados para configurar o *Processor* através dos métodos *convertAudioFormat* e *convertVideoFormat*.

Para permitir o controlo da transmissão de cada um dos *streams*, foram implementados os métodos *start* e *stop*. A destruição do objecto que envia o *stream* (*DataSink*) é da responsabilidade do método *killTransmitter*.

4.4.2 Envio de um INVITE

O diagrama de sequência da Figura 4.16 mostra a operação básica do envio de um INVITE. Após o envio do INVITE, o processo ao nível da transacção segue o diagrama de estado *Invite Client Transaction*, referenciado no ponto 4.2.1.1.

O envio de um INVITE (criação de uma chamada) é iniciado através do botão *Dial* da *Toolbar* do sIPtel após o preenchimento das duas *ComboBox From* e *To* nas quais se colocam respectivamente os endereços SIP URI do remetente e do destinatário da chamada. Através do método *ServerMain.sendInvite* é dada a ordem para enviar um INVITE à parte da sinalização. Este último método invoca, por sua vez, o método *HandlerRequests.sendInviteRequest* que tem como função criar o pedido INVITE instanciando o objecto *SIPRequest* e configurar esse mesmo pedido através do método *SIPRequest.setHeader*, ao qual é passada informação previamente guardada na classe *InviteInfo*. Ainda no método

HandlerRequests.sendInviteRequest é criado um objecto *GenerateMessageSDP* que permite criar de um modo mais simples o conteúdo da mensagem SDP através dos parâmetros actuais seleccionados no painel *Settings*.

Depois de construir o pedido INVITE é necessário passá-lo ao nível de transacção, que por sua vez o passa ao nível de transporte. Para implementar os dois *timers* (*Timer A* e *Timer B*), definidos na transacção *Invite Cliente Transaction*, é instanciada a classe *Timers* que permite criar o *Timer A* para a retransmissão dos pedidos INVITE e o *Timer B* como *timeout* para receber uma resposta. Cada vez que o *Timer A* dispara é enviado o pedido INVITE através do método *MessageChannel.sendMessage* que tem como parâmetros, entre outros, o objecto *SIPRequest* previamente configurado. A classe *MessageChannel* disponibiliza métodos que permitem às aplicações enviar as mensagens SIP. O método desta classe utilizado é o *MessageChannel.sendMessage(SIPMessage)*, que tem como parâmetro a mensagem SIP a enviar, que neste caso é um pedido. Por fim é actualizada a tabela da interface gráfica com a mensagem *Calling* e o nome do utilizador que está ser chamado.

Durante todo este processo são armazenados e consultados os estados de cada transacção que estão em tabelas de estado implementadas através das classes *Hashtable* e *HashMap*.

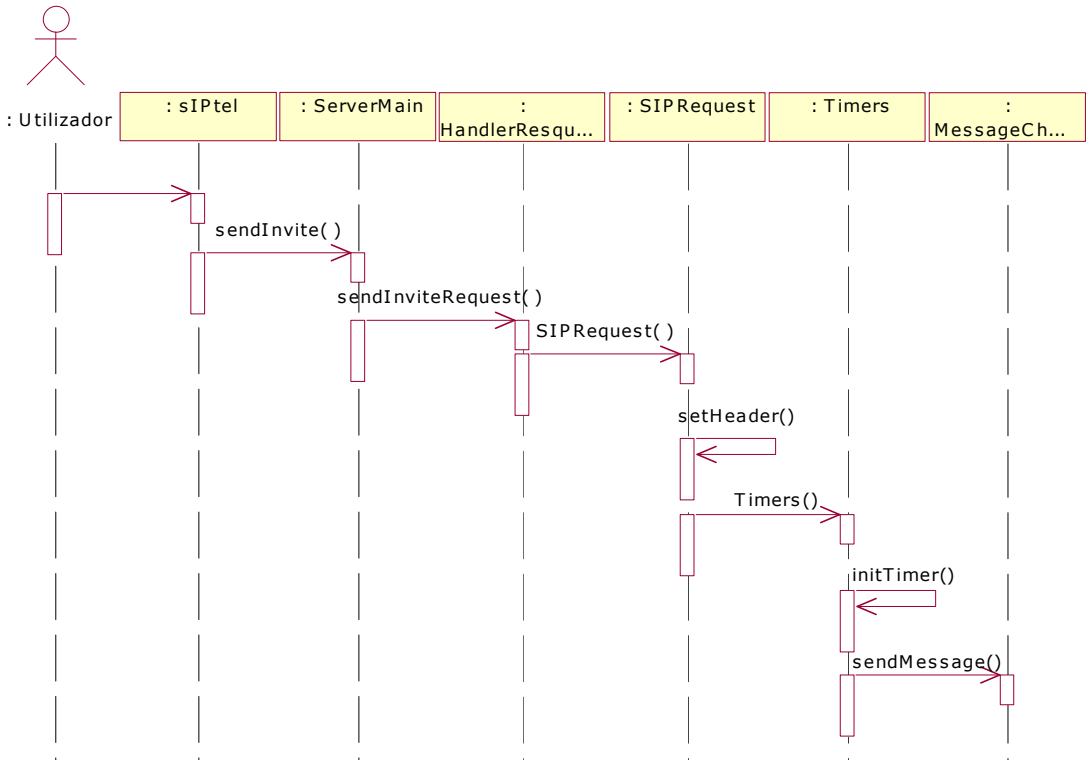


Figura 4.16 – Exemplo de um processo do envio de um INVITE: diagrama sequencial

4.4.3 Recepção de um INVITE

Na Figura 4.17 é apresentado o diagrama de sequência de um caso particular para o tratamento de um INVITE quando a *Invite Server Transaction* está no estado *Proceeding* e transmite uma resposta *100 Trying*.

Quando um INVITE chega à *stack*, esta chama o analisador de mensagens NIST-SIP que processa o pedido e invoca a classe *SIPStackMessageFactoryImpl*. Esta classe por sua vez instancia a classe *SIPServerRequest* e chama o método *SIPServerRequest.processRequest* que passa como parâmetros o *SIPRequest* e o *MessageChannel*, dando início a uma transacção *Invite Server Transaction*.

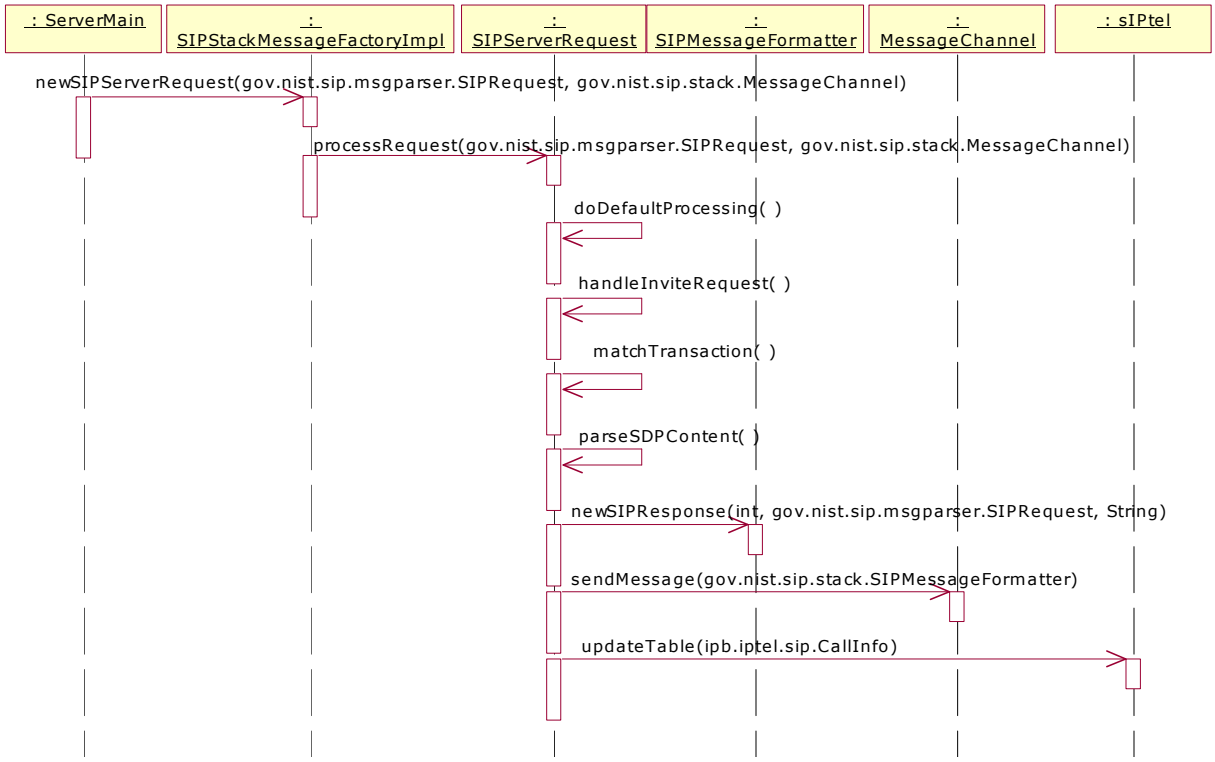


Figura 4.17 – Exemplo de um processo da recepção de um INVITE: diagrama sequencial

O método *SIPServerRequest.doDefaultProcessing* identifica qual o método do pedido chegado e, neste caso, chama o método *SIPServerRequest.handleInviteRequest* para o tratamento da mensagem INVITE. Dentro deste último método é identificada a transacção e analisado o conteúdo da mensagem SDP através dos métodos *SIPServerRequest.matchTransaction* e *parseSDPContent*, respectivamente. Neste caso supõe-se que este é o primeiro INVITE de uma transacção, que as mensagens SIP e SDP não têm erros e que o TU demora mais de 200 milissegundos para dar uma resposta provisória ou final, sendo obrigatório enviar uma resposta *100 Trying*. É então criada uma resposta *100*

Trying através do método *SIPMessageFormatter.newSIPResponse* passando como parâmetros o código de estado desta resposta e o pedido que a originou. Por fim, é actualizada a tabela de estados na interface gráfica do sIPtel com a mensagem *Alerting*, indicando que chegou uma chamada nova. No entanto como foi referido no ponto 4.2.1, a transacção não está concluída até ser recebida do servidor uma resposta do tipo final (não-1xx).

4.4.4 Envio do *stream* RTP

Quando é estabelecida uma chamada e existe a intenção de enviar um *stream* para o outro utilizador é invocado pelo *SIPServerRequest* ou *SIPServerResponse* o método *sIPtel.startCommunication*, que para além de outras funcionalidades, chama o método *sIPtel.startTransmiting*, que por sua vez dá início ao processo representado na Figura 4.18.

No JMF cada dispositivo físico de captação de vídeo ou áudio, como uma câmara ou um microfone, é representado por um *CaptureDeviceInfo*. Este objecto é obtido invocando o método *CaptureDeviceManager.getDeviceList(Format)*, sendo passado como parâmetro o objecto *Format* que descreve o formato do meio. O objecto *CaptureDeviceManager* tem a função de procurar dispositivos físicos no sistema que suportem o formato desejado retornando objectos *CaptureDeviceInfo* que contêm informação sobre cada dispositivo de captura. Para capturar os meios através de um dispositivo é necessário obter o *MediaLocator* do mesmo, através do método *CaptureDeviceInfo.getLocator*. O *MediaLocator* é depois usado para a criação do *DataSource* e com este último é criado o *Processor* através do método *Manager.createProcessor*(*DataSource*).

Posteriormente é criado o objecto *RTPTransmitter* responsável pela configuração e controlo do *Processor*, sendo este utilizado para permitir o controlo da codificação do meio fornecido pelo *DataSource*. Antes de utilizar o *Processor* é necessário configurá-lo. A sua configuração é feita através do método *configureProcessor* da classe *ProcessorUtil*, colocando o *Processor* num estado de configuração que reúne a informação necessária para construir um *TrackControl* para cada canal de dados, chamados pistas. O ou os *TrackControl* são obtidos e guardados num vector através do método *Processor.getTrackControls*, já que um *stream* pode conter várias pistas. Um *stream* pode conter, por exemplo, uma pista de áudio e outra de vídeo.

O método *Processor.setContentDescriptor* é utilizado para especificar o formato dos dados de saída do *Processor*. Neste caso especifica o tipo de conteúdo da saída do *Processor* destinado

ao *DataSource* num formato não comprimido, através do método *contentDescriptor.RAW*. No entanto esta invocação é utilizada para evitar chamar o método *Processor.getTrackControls* com o *Processor* num estado de *Configuring*, o que causa o lançamento de um erro ou excepção.

Como o *Processor* gera dados codificados, cada pista a ser transmitida é configurada com um formato RTP de modo a converter os dados de um formato para outro, neste caso de um formato não codificado (*raw*) para um formato codificado (*rtp_Format*). Para isso é utilizado o método *setFormat* da interface *TrackControl* que permite especificar as conversões do formato para cada uma das pistas (*tracks*). O formato RTP é seleccionado especificando uma *string* de codificação na classe *AudioFormat* para o áudio e *VideoFormat* para o vídeo (ex. *AudioFormat.GSM_RTP* para áudio e *VideoFormat.H263_RTP* para vídeo).

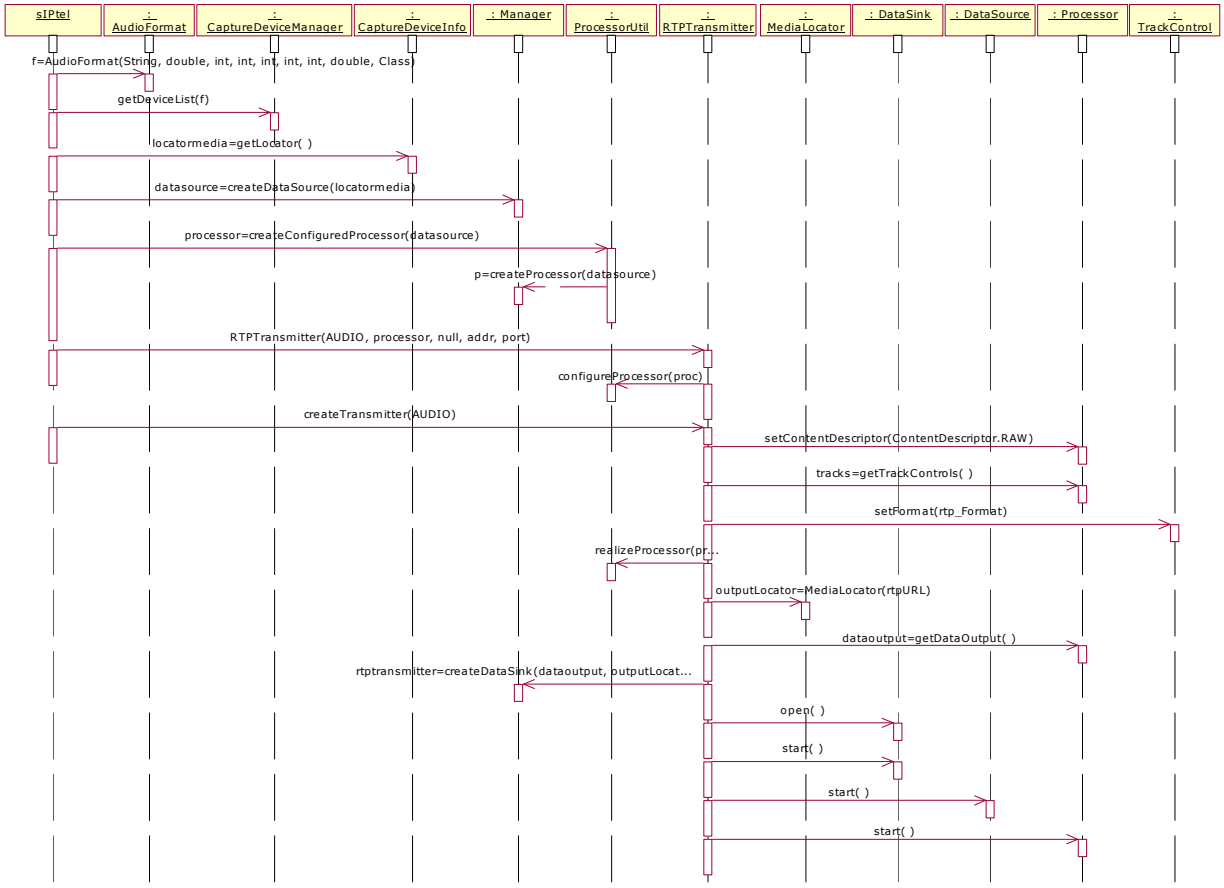


Figura 4.18 – Processo de captação e envio de um *stream* de áudio: diagrama sequencial

Após a configuração do *Processor* e seleccionadas as opções de processamento das pistas, é alterado o estado do *Processor* que passa do estado *Configured* ao estado *Realized*. Neste estado qualquer alteração nas opções de processamento pode implicar o mau funcionamento do *Processor*, sendo na maior parte dos casos, lançada uma excepção do tipo

FormatChangeException.

A próxima tarefa é obter o *DataSource* do *Processor* e construir o *MediaLocator* para especificar para onde deve ser enviado o *stream* de dados. O primeiro passo é feito utilizando o método *Processor.getDataOutput*, e o segundo é conseguido com o endereço IP, porta do destino e o tipo de meio a ser enviado, criando uma *string* com um URL num formato de *rtp://endereço:porta/tipo-de-dados*.

Passando como parâmetros o *DataSource* que representa os meios a transmitir, e o *MediaLocator* que indica a sessão RTP para onde o *DataSource* vai ser transmitido, é criado o objecto *DataSink* com o método *createDataSink* da classe *Manager*. O *DataSink* tem a função de ler os dados do *DataSource* e enviá-los para o endereço remoto indicado no *MediaLocator*. Para controlar a transmissão são utilizados os métodos *start* e *stop* do *DataSink*.

4.4.5 Apresentação dos *streams* recebidos

Após o estabelecimento da chamada é invocado o método *sIPtel.startCommunication* que cria um objecto *RTPReceiver* para cada tipo de meio áudio e/ou vídeo da sessão RTP. Cada um dos *RTPReceiver* tem a função de escutar a chegada de um *stream* e invocar as entidades necessárias para apresentar esses dados. A Figura 4.19 ilustra um diagrama sequencial que representa os pontos mais importantes no processo de configuração, recepção e apresentação de um *stream*.

Foi já referido que cada objecto *RTPReceiver* escuta a chegada de novos *streams*; este mecanismo de escuta é possível porque a classe *RTPReceiver* através da implementação da interface *ReceiveStreamListener* fica habilitada a receber notificações do tipo *ReceiveStreamEvent*, que informam a classe de todos os eventos recebidos num *ReceiveStream* particular. Para que a classe fique registada como *ReceiveStreamListener* é necessário:

- Criar um *SessionManager* recorrendo ao *com.sun.media.rtp.RTPSessionMgr*, uma implementação do JMF;
- Chamar o método *RTPSessionMgr.addReceiveStreamListener(this)* para registar a classe *RTPReceiver* como *listener*;
- Iniciar a sessão através do *RTPSessionMgr.initSession* e começar essa sessão com o método *RTPSessionMgr.startSession*, ficando o objecto *RTPReceiver* apto para

receber *streams* de dados.

Neste ponto a aplicação está pronta a receber *streams* de dados. Quando a aplicação recebe um novo *stream* o *SessionManager* gera um evento *NewReceiveStreamEvent* o qual é capturado pelo método *RTPReceiver.update*. Neste método é retirado o novo *ReceiveStream* desse evento através do método *getReceiveStream,* e do *ReceiveStream* é retirado o *DataSource* com o método *ReceiveStream.getDataSource*. Em seguida é criado um novo *Player* para o *ReceiveStream* com o método *Manager.createPlayer(DataSource)*, sendo-lhe passado o *DataSource*.

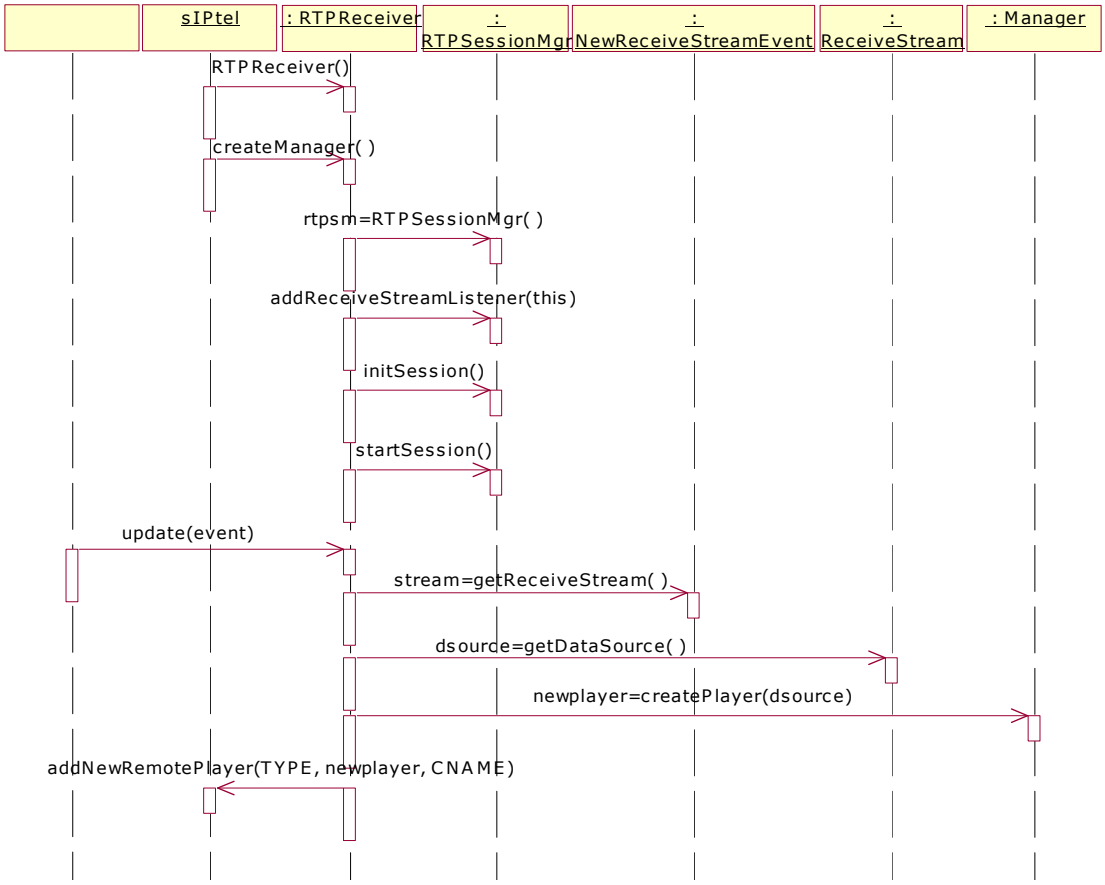


Figura 4.19 – Processo de recepção de um *stream* de áudio: diagrama sequencial

Após a criação do *Player*, este lança um evento do tipo *RealizeCompleteEvent*. Estes eventos são detectados pela interface *ControllerListener* que responde aos eventos gerados pelos *Controllers*, neste caso o *Player*, e define o método *controllerUpdate* para responder a esses eventos. A classe responsável pela implementação da interface *ControllerListener* é a classe *VideoPlayer*. Esta classe fica registada como *ControllerListener* para ser actualizada quando o estado do *Player* é alterado. A apresentação do vídeo é feita quando é captada a ocorrência de um evento *RealizeCompleteEvent* pelo método *controllerUpdate*, sendo em seguida anexada a

componente visual do *Player* à classe *VideoPanel* que deriva da classe *JPanel*.

Outro processo executado pelo método *controllerUpdate* ocorre quando o *Player* detecta que existem alterações no *Format* do seu meio e lança o evento *FormatChangeEvent*. Este é capturado e o método *controllerUpdate*, verifica se para este evento houve alteração na componente visual do *Player*. Caso tenha havido, remove a componente visual do *Player* do painel de visualização (*VideoPanel*) através do método *VideoPanel.deattachVideo* e anexa ao mesmo a nova componente visual através do método *VideoPanel.attachVideo(Player.getVisualComponent)*.

Capítulo 5

Testes

Após a implementação do sIPtel procedeu-se à realização de testes com o intuito de avaliar o nível de interoperabilidade do serviço. Foram considerados dois tipos de testes: o primeiro consistiu em avaliar o nível de interoperabilidade através de critérios de avaliação definidos pelo *Technical Program Committee* (*TPC*), e o segundo consistiu em verificar a interoperabilidade do sIPtel com outros *softwares* IPtel com utilizassem o protocolo SIP para a sinalização de chamadas.

5.1 Classificação do nível de interoperabilidade

O *Technical Program Committee* é responsável por organizar a parte técnica dos eventos de teste de interoperabilidade SIP. Estes eventos são encontros de grupos de programadores SIP que se reúnem de quatro em quatro meses, com o objectivo de testarem a interoperabilidade das suas implementações com as de outros grupos.

Este comité definiu uma classificação de interoperabilidade das implementações que consiste em classificar as entidades SIP (UAs e *Proxys*) em implementações básicas, intermédias e avançadas, de acordo com o cumprimento de uma lista de características necessárias para cada um dos níveis. Uma implementação satisfaz um determinado nível se cumprir 80% dos critérios desse nível. Nas Tabela 5.1, 5.2 e 5.3 [Schulzrinne, 2002b] podem-se ver os critérios para os vários níveis e o resultado da aplicação dos testes ao sIPtel.

Nível	Característica	Suporte
Básico	Envia INVITE sobre UDP	Sim
	Envia INVITE sobre TCP	Sim
	Gera ACK apropriadamente	Sim
	Aceita e rejeita chamadas	Sim
	SDP com uma única linha m= e c= e um único codificador	Sim
	Os cabeçalhos *To*, *From*, *Call-ID*, *CSeq*, *Via*, *Content-Lenght*, *Content-Type* tratados convenientemente	Sim
	Gera *tags* no campo *To*	Sim
	Terminas chamadas com BYE sobre UDP	Sim
	Recebe BYE sobre UDP	Sim
	Suporta cabeçalhos de forma compacta	Sim
	Rejeita pedidos desconhecidos com a reposta 501	Sim
	Envia/recebe *streams*, com a possibilidade de não enviar pacotes RTCP	Sim

Tabela 5.1 – Critérios de avaliação da interoperabilidade SIP – nível básico

Nível	Característica	Suporte
Intermédio	Suporte do TCP para todas as mensagens	Sim
	Suporte dos cabeçalhos *Require* e *Proxy-Require*	Não
	Retransmite pacotes perdidos para INVITE e BYE	Sim
	Tem em consideração o cabeçalho *Contact* no INVITE e a resposta 2xx a um INVITE, retransmitindo os pedidos seguintes ponto a ponto	Não
	Processa o CANCEL para o INVITE	Sim
	Autenticação para o Registo utilizando o método *Basic*	(a)
	Autenticação para o Registo utilizando o método *Digest*	Sim
	Permite o redireccionamento para páginas *Web* ou *e-mail*	Não
	Recebe texto ou HTML em respostas 3xx ou 4xx	Não
	Suporta o cabeçalho *Accept* sem SDP	Não
	Suporta DNS SRV *records*	Não
	Suporta o registo com períodos de refrescamento para endereços *unicast*, tendo em consideração o cabeçalho *Expires* da resposta REGISTER	Sim
	Suporta o redireccionamento	Não
	Suporta múltiplos codificadores na linha '*m*'	Sim
	Múltiplas linhas m= são processadas correctamente	Sim
	Tipos de meios desconhecidos e encontrados na linha '*c*' são tratados convenientemente. (i.e. são rejeitados com a porta 0)	Sim
	Tanto o nome do domínio como o endereço IP encontrados no campo '*c*' são aceites	não
	Gera pacotes RTCP	Não
	Responde ao pedido OPTIONS	Sim
	Suporta URLs não-SIP no REGISTER	Não
	Copia a *Record-Route* da resposta para a *Route* do pedido que é encaminhado correctamente	Sim
	Verifica a igualdade do parâmetro *action* no REGISTER (deixou de existir)	(b)
	Consegue obter registos actuais	Não
	Consegue apagar registos com o *Contact*:* e *Expires*	Sim

Tabela 5.2 - Critérios de avaliação da interoperabilidade SIP – nível intermédio

Nível	Característica	Suporte
	Tenta automaticamente vários redireccionamentos	Não
	Gera pedidos REGISTER *multicast*	Não
	Suporta re-INVITE: suspende um *stream*	Sim
	Suporta re-INVITE: retoma um *stream*	Sim
	Suporta re-INVITE: desactiva um único *stream*	Sim
	Suporta re-INVITE: altera codificadores	Sim
Avançado	Suporta re-INVITE: adiciona um *stream*	Sim
	Suporta re-INVITE: altera o endereço do meio para um endereço ou porta diferente (mobilidade)	Não
	Suporta o cabeçalho *Expires* do INVITE	Não
	Registo de um terceiro elemento	Não
	Gera pedidos com URLs do tipo tel: e entrega-os a um servidor próprio	Não
	Processa múltiplas respostas MIME	Não

Tabela 5.3 – Critérios de avaliação da interoperabilidade SIP – nível avançado

Esta tabela teve a sua última actualização em Abril de 2001, isto é, antes do lançamento da RFC 3261, baseando-se na RFC 2543 e talvez em alguns dos *Internet-drafts* lançados posteriormente. Deste modo alguns dos critérios de avaliação não são verificados pelo sIPtel visto que a sua implementação teve em consideração a RFC 3261.

Ao analisar a tabela na coluna "Suporte" aparecem duas excepções às respostas afirmativas e negativas. A primeira excepção, assinalada com (a), verifica-se pelo facto da RFC 3261 definir que o método de autenticação *Basic* foi abandonado, utilizando-se apenas o método *Digest*. O mesmo se passa em relação ao segundo caso assinalado, já que a RFC 3261 também anuncia o abandono do parâmetro *action*.

Utilizando este critério de avaliação verifica-se que o sIPtel satisfaz em 100% os critérios básicos de interoperabilidade. Quanto ao nível intermédio obedece a 50% dos critérios, não atingindo os 80% necessários para aprovação. O mesmo acontece para o nível avançado, que tem uma taxa menor que no nível intermédio, 42%, que não chegam para satisfazer o nível avançado.

5.2 Interoperabilidade com outras aplicações

O segundo conjunto de testes consistiu em confrontar o sIPtel com alguns *softwares* SIP, disponibilizados na Internet gratuitamente ou em versões de demonstração. Não se optou por analisar o nível de interoperabilidade de cada um deles, devido a em alguns casos, não ser possível ao nível da aplicação verificar a satisfação das características da Tabela 5.1, 5.2 e 5.3

e se essa funcionalidade está implementada da maneira correcta.

5.2.1 Características das aplicações

Em seguida são apresentadas, algumas das características dos *User Agents* e dos Servidores utilizados na realização dos testes.

5.2.1.1 SCS-Client

É uma das ferramentas mais completas do mercado e foi desenvolvido pela *Software House Siemens Switzerland Ltd*. Pode ser utilizado nos sistemas operativos Windows NT/2000/XP. Disponibiliza um vasto número de serviços, dos quais se destacam:

- Suporte de áudio (G.711 aLaw, G.711 uLaw, G.723.1 e GSM), vídeo (H.263) e presença;
- Suporte de mensagens instantâneas, lista de amigos, *web*, *chat* e transferência de ficheiros com ligação segura.
- Autenticação de utilizadores e autenticação nos servidores;
- Configuração de vários codificadores de áudio;
- Conferência de áudio e chat;
- Suporte de chamadas em espera e reencaminhamento de chamadas.

A Figura 5.1 ilustra o estabelecimento de uma chamada entre o sIPtel e o *SCS-Client* através do NIST-SIP Proxy. Este cenário decorre apenas num computador e pretende apenas mostrar os dois serviços em comunicação. O mesmo caso se passa para os cenários representados nas Figura 5.2, Figura 5.3 e Figura 5.4. Mais à frente serão abordados os testes efectuados entre o sIPtel e os restantes softwares abordadas neste ponto.

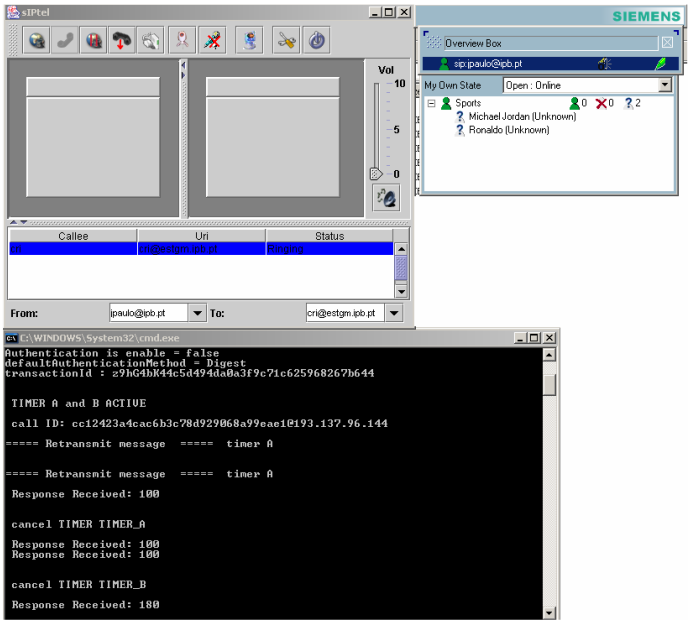


Figura 5.1 – Estabelecimento de uma sessão entre o sIPtel e o *SCS-Client*

5.2.1.2 Instant xpressa

É um produto da *Pingtel Corporation*, que segue a RFC 3261 na implementação da sinalização. Pode ser executado em Windows 98/2000/NT/XP. Com uma interface semelhante à de um telefone, este produto comercial oferece um conjunto de funcionalidades que podem ser encontradas nos telefones digitais de gama alta. Algumas das características são enunciadas em seguida:

- Suporte de áudio (G.729 Anexo B – supressão de silêncio);
- Suporte de chamadas em espera, redireccionamento e reencaminhamento de chamadas;
- Conferência de áudio;
- Integração com um portal *web* (*MyPingtel*).

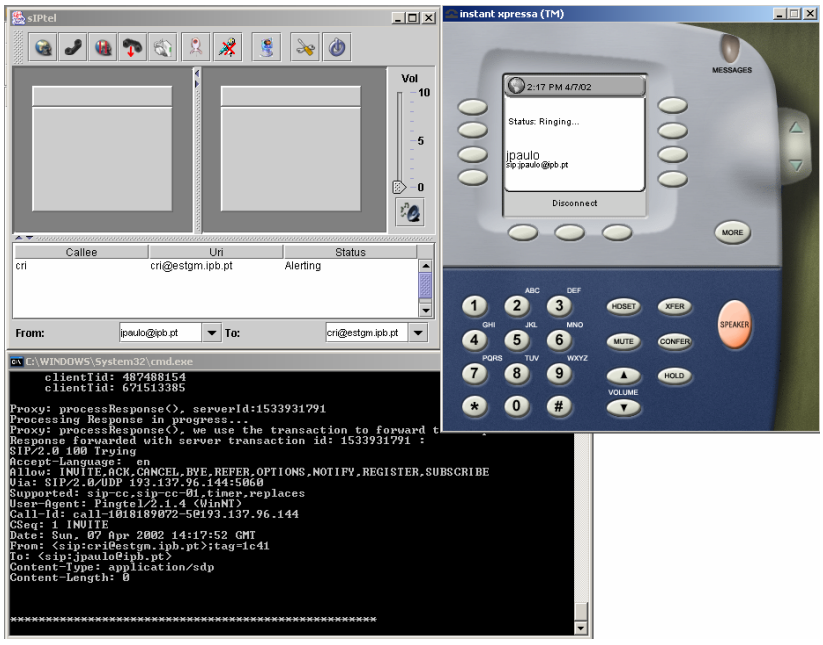


Figura 5.2 – Estabelecimento de uma sessão entre o xpressa e o sIPtel

5.2.1.3 eStara SoftPhone

Desenvolvido pela *eStara* pode ser executado no Windows 95/98/NT4/2000/XP e tem as seguintes características:

- Suporte de áudio (G.721 e G.723) e presença;
- Chamadas em espera, reencaminhamento e transferência de chamadas;
- Autenticação em *Proxys*.

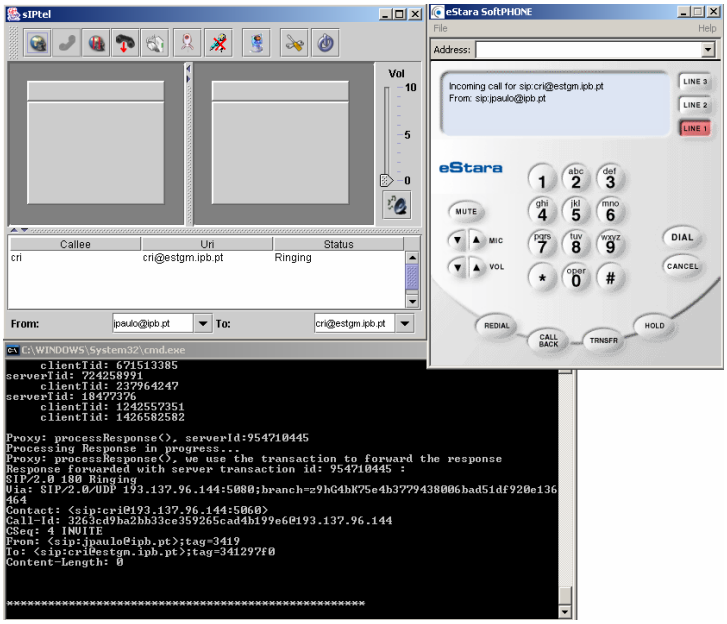


Figura 5.3 - Estabelecimento de uma sessão entre o sIPtel e o eStara.

5.2.1.4 Ubiquity's User Agent

É propriedade da *Ubiquity Software Corporation* e o SIP é definido segundo a RFC 2543. De todos os softwares com que o sIPtel foi testado este é o que oferece menos funcionalidade. Algumas delas são referidas em seguida:

- Suporte de áudio (G.711);
- Suporte de chamadas em espera.

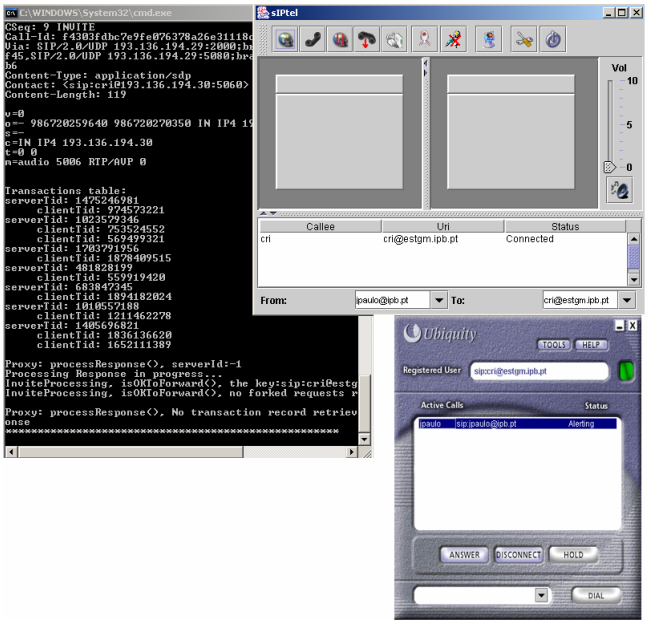


Figura 5.4 – Estabelecimento de uma sessão entre o sIPtel e o *Ubiquity's UA*

5.2.1.5 Servidores SIP

Os SIP Proxys representados na Tabela 5.4 foram utilizados para fazer o registo de utilizador com e sem autenticação:

Nome	Fabricante	Características
NIST-SIP Proxy Server	NIST	*Stateful Proxy* *Registrar Server* *Presence Server* Suporte de autenticação
Proxy Server	Siemens – mySIP	*Registrar Server* *Stateful Proxy*

Tabela 5.4 – Software servidor utilizado na realização dos testes

5.2.2 Testes e resultados

Dos UAs referidos anteriormente, o *SCS-Client* é o único que suporta vídeo e é também o mais completo dos quatro na oferta de serviços. Para a realização de testes com *software* que não suportava vídeo, desactivaram-se no sIPtel as opções de envio e recepção de vídeo no painel de configuração. Todos os testes realizados tiveram a intervenção de um *Outbound Proxy* também com função de servidor *Registrar*. Nos testes estiveram envolvidos três computadores (Figura 5.5), dois que executavam os UAs e outro em que era executado o servidor *Proxy* e *Registrar*. No entanto, nada impede que as três entidades estejam distribuídas apenas por um ou dois computadores.

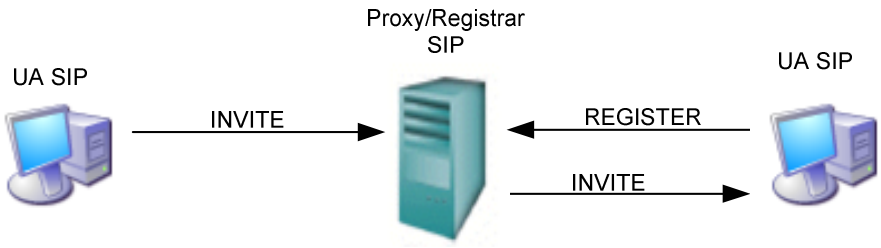


Figura 5.5 – Topologia do cenário de testes

O registo de utilizadores sem autenticação no servidor NIST-SIP *Proxy* e no *Proxy* da *Siemens* foi efectuado com sucesso, assim como o registo com autenticação utilizando o método *Digest* no NIST-SIP *Proxy*.

Foi ainda considerado o exemplo básico de uma chamada entre um UAC e um UAS através de um *proxy*. Este cenário, denominado *SIP Trapezoid*, representa as funções básicas de localização de um terminal, sinalização da intenção de comunicar, negociação de parâmetros da sessão e finalização da chamada estabelecida. Todos os testes executados neste cenário, envolvendo o sIPtel e os quatro UAs, foram efectuados com sucesso.

Foi também possível estabelecer uma chamada entre o sIPtel e o *SCS-Client*, a qual envolveu uma sessão com áudio e vídeo.

Durante o decorrer da sessão foi ainda testada a colocação de chamadas em espera (*Hold*), operação que foi realizada com sucesso nas quatro aplicações.

Capítulo 6

Conclusões

Alguns investigadores acreditam que o futuro das comunicações tende apenas para a tecnologia de comutação de pacotes, que permitirá o transporte de voz, vídeo e dados numa única conexão. Esta tecnologia fornece os mesmos serviços que a rede pública de telefones e disponibiliza ainda outros serviços que dificilmente poderão ser implementados neste tipo de tecnologia, tornando a rede pública de telefones, em teoria, obsoleta. Contudo, esta tecnologia é ainda a grande adversária da IPtel, isto porque a rede pública de telefones assume na actualidade uma disponibilidade ubíqua e uma fiabilidade elevada; além disso, a IPtel aparenta ser apenas uma maneira diferente de comunicar, oferecendo por esse motivo pouco interesse à substituição da telefonia tradicional pela IPtel. Todavia, estudos indicam que o número de telefones IP está a crescer significativamente e diversas empresas têm mostrado que a IPtel pode funcionar bastante bem e reduzir custos de equipamento, das comunicações e da operação da rede.

A recente actualização do protocolo SIP (RFC 3261), utilizado para implementar a sinalização do sIPtel, veio esclarecer alguns detalhes menos claros em relação à versão inicial (RFC 2543) e melhorar o desempenho ao nível da segurança. Segundo diversos especialistas com estas alterações e com a sua crescente divulgação, este protocolo está a ter grande aceitação por parte de um elevado número de empresas.

O SIP foi também escolhido pelo *3rd Generation Partnership Project* (3GPP), para estabelecer sessões multimédia na rede UMTS (R5), e por grandes operadoras, como a WorldCom e AOL, assumindo-se como a principal alternativa à recomendação H.323.

6.1 Trabalho desenvolvido

Esta dissertação teve como objectivo desenvolver um serviço IPtel que permitisse a comunicação entre dois participantes em tempo real com áudio e vídeo, utilizando o protocolo de sinalização SIP para o estabelecimento das sessões. Para tal analisaram-se as características e a arquitectura protocolar típica do serviço IPtel. Estudaram-se os protocolos de sinalização e transporte mais utilizados nesta tecnologia, com destaque para os protocolos SIP e RTP. Estudou-se ainda a arquitectura do protocolo SIP e serviços incluídos na sua recomendação inicial, devido à necessidade da implementação da parte de sinalização.

Embora a qualidade de serviço seja um aspecto importante, e por isso foi abordado nesta dissertação, convém porém referir que não foram implementadas soluções no sIPtel que garantam a qualidade de serviço associadas à transmissão de áudio e vídeo em tempo real sobre redes IP.

Foi feita também uma pesquisa de *software* auxiliar para o suporte da sinalização SIP, mais propriamente bibliotecas em Java, que ajudassem a construir aplicações SIP e aplicações com suporte multimédia. Foi também necessário procurar aplicações servidor, especificamente *Proxy Server* e *Registrar Server*, para testar o desenvolvimento do serviço. Em seguida foram especificados os requisitos de sinalização e de meios que o sIPtel deveria suportar.

Posteriormente iniciou-se o processo de implementação do serviço anteriormente definido. Deve-se destacar, devido à sua dificuldade, o nível de transacção definido na RFC 3261. Entre os vários problemas inerentes ao funcionamento desta aplicação, salientam-se a utilização de uma API ainda em fase de desenvolvimento, assim como a implementação do serviço coincidir com a alteração do documento que especifica o protocolo SIP, isto é, a actualização da RFC 2543 para a RFC 3261. Também no decorrer da implementação do sIPtel foram detectados erros na API que impediam o seu correcto funcionamento. Alguns destes erros foram corrigidos após o envio do código fonte alterado à lista de *e-mail* do NIST-SIP e outros rapidamente solucionados após discussão com outros membros da referida lista.

Finalmente, foram realizados testes de modo a avaliar o nível de interoperabilidade através de critérios de avaliação definidos pelo *Technical Program Committee* e testes de interoperabilidade com diversos *software*s IPtel comerciais disponíveis na Internet.

Numa retrospectiva do trabalho realizado é de referir que se pretendia integrar o sIPtel no

projecto VESPER (*Virtual Home Environment for Service PErsonalization and Roaming*), um projecto da União Europeia que envolve universidades, laboratórios de pesquisa, empresas e operadores de telecomunicações com o objectivo de validar o conceito de *Virtual Home Environment* (VHE). Devido à interrupção deste projecto, por motivo ao qual este trabalho está totalmente alheio, essa integração não se veio a verificar. Todavia fica mais uma contribuição para os trabalhos realizados dentro da IPtel utilizando o protocolo SIP, que na iniciação desta dissertação eram claramente escassos e durante este espaço temporal surgiram em abundância, revelando a aceitação deste novo protocolo e a forte divulgação deste serviço.

Com certeza que a curiosidade e o interesse pelo tema motivariam a implementação de outras funcionalidades neste serviço, porém a limitação temporal foi decisiva no término do desenvolvimento do sIPtel. As principais contribuições feitas neste trabalho, além de uma ampla revisão bibliográfica, consistem em ter sido possível concluir sobre o estado actual desta tecnologia e a sua importância dentro do sector das telecomunicações, bem como um serviço capaz de comunicar com outras aplicações SIP com a vantagem de poder ser executado em múltiplas plataformas.

6.2 Perspectivas de evolução futura

Futuramente o sistema poderá ser melhorado em duas vertentes: por um lado a integração de novos serviços e por outro lado o fornecimento de qualidade de serviço.

Na fase de implementação do sIPtel a API NIST-SIP encontrava-se em fase de desenvolvimento, não suportando ainda a criação de serviços de presença e mensagens instantâneas. Deste modo, estes serviços com grande sucesso actualmente na Internet e oferecidos por fornecedores de serviços como o AOL, Yhaoo e MSN, podem ser facilmente adicionados ao sIPtel. A integração destes dois serviços não será de grande complexidade, já que a actual versão (v.1.1) da API NIST-SIP fornece as funcionalidades do nível de transacção, não sendo necessário implementar nenhuma máquina de estados para o processamento dessas transacções.

Ao contrário das redes de comutação de circuitos que oferecem características como baixa latência, baixas perdas e não introduzem *jitter*, as redes de comutação de pacotes debatem-se ainda com problemas como atrasos variáveis e elevados, *jitter*, congestionamento e perdas de pacotes. Dado que no desenvolvimento sIPtel não foram implementadas soluções que

garantam qualidade de serviço associada à transmissão de áudio e vídeo em tempo real e devido à relevância deste parâmetro nesta tecnologia, seria útil implementar alguns dos métodos anteriormente abordados no ponto 2.7.

Na indústria da informática e comunicações tudo evolui rapidamente. O aumento da largura de banda e a utilização do novo protocolo IP versão 6 que suporta a implementação da reserva de largura de banda e níveis de qualidade de serviço serão, num futuro próximo, aspectos preponderantes para a implantação da IPtel no mercado das comunicações. Existe também uma grande expectativa relativamente a esta tecnologia, com o aparecimento da terceira geração de telemóveis (3G), prevendo-se nos próximos anos a oferta de serviços de voz e vídeo em pacotes, e o aparecimento de um grande número de sistemas terminais ligados à rede IP.

Referências

[Almesberger, 1998] W. Almesberger, T. Ferrari, and J.-Y. Le Boudec, “SRP: a scalable resource reservation protocol for the internet”, In Proceedings of the 6 th International Workshop on Quality of Service (IWQoS'98), pp. 107-16, Napa, California, Maio de 1998. URL: http://citeseer.nj.nec.com, Setembro 2002.

[Arrowood, 1996] Arrowood, A. - “CU-SeeMe Communications in an emergent technology” - LCC/IDT, OIT/NS GRA, Georgia Institute of Technology, Fevereiro 1996. URL: http://citeseer.nj.nec.com/, Junho 2002.

[Busse, 1995] I. Busse, B. Deffner, H. Schulzrinne, “Dynamic QoS control of multimedia applications base on RTP”, in First International Workshop on High Speed Networks and Open Distributed Platforms, (St. Petersburg, Russia), Junho 1995. URL:http://www.cs.columbia.edu/~hgs/papers/Buss9601_Dynamic.ps.gz, Setembro 2002.

[Campbell, 1997] Andrew T. Campbell, Aurel A. Lazar, H. Schulzrinne e R. Stadler, “Building open programmable multimedia networks”, IEEE Multimedia 4, No. 1, pag. 77-82 (January-March 1997). URL: http://comet.ctr.columbia.edu/~campbell/papers/comet.pdf, Julho 2002.

[Casner, 1992] Stephen Casner, Stephen Deering, “First IETF Internet audiocast”, ACM SIGCOMM Computer Communications Review, pag. 92-97, Julho 1992, URL: http://citeseer.nj.nec.com/, Junho 2002.

[Cherriman, 1996] P. Cherriman, L. Hanzo, “H261 and H263-based programable video transceivers”, in Proceedings of ICCS'96/ISPAC'96, pag. 1369-1373, 1996. URL: http://www-mobile.ecs.soton.ac.uk/peter/publications/iccs96-web.pdf, Novembro 2002.

[Cherriman, 2002] Peter Cherriman, “H.263 video coding”, http://www-

mobile.ecs.soton.ac.uk/peter/h263/h263.html, Agosto 2002.

[Richey, 2002] Rodger Richey, “Training embedded apps to process speech”, http://www.circuitcellar.com/pastissues/articles/richey110/text.htm, Setembro de 2002.

[Dorcey, 1995] Dorcey, T. - “CU-SeeMe desktop videoconferencing software”, Connexions, vol. 9, no. 3, Março 1995. URL: http://physics.hallym.ac.kr/resource/CU-SeeMe/msattler/, Junho 2002.

[Fluckiger, 1995] François Fluckiger, “Networked multimedia applications and technology”, Prentice Hall Europe, 1995.

[Ghanbari, 1999] Mohammed Ghanbari, “Video coding, an introduction to standard codecs”, The Institution of Electrical Engineers, 1999.

[H.225.0, 1998] ITU-T Recommendation H.225.0, "Call signaling protocols and media stream packetization for packet based multimedia Communications Systems", 1998.

[H.245, 1998] ITU-T Recommendation H.245, "Control protocol for multimedia communication", 1998.

[H.323, 1996] International Telecommunication Union, “Visual telephone systems and equipment for local area networks which provide a non-guaranteed quality of service,” Recommendation H.323, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, Maio1996.

[iptel.org, 2002] Grupo independente do Instituto Fraunhofer FOKUS, IP Telephony Overview, http://www.iptel.org/info/overview.html, Setembro 2002.

[ITU/BDT, 2002] International Telecommunication Union, “IP Telephony report by the Group of Experts on Internet Protocol (IP) Telephony / ITU-D”, World Telecommunication Development Conference, Março 2002. URL: http://citeseer.nj.nec.com/, Junho 2002.

[Kulathumani, 2000] Vinodkrishnan, “Voice over IP: products, services and issues”, 2000. URL: ftp://ftp.netlab.ohio-state.edu/pub/jain/courses/cis788-99/voip_products.pdf, Julho 2002.

[Lennox, 1999] Jonathan Lennox, Henning Schulzrinne e Thomas F. La Porta, “Implementing intelligent network services with the Session Initiation Protocol”, Columbia University Technical Report CUCS-002-99, New York, NY, Janeiro 1999. URL: http://citeseer.nj.nec.com/cs, Junho 2002:

[Lennox, 2000] Jonathan Lennox e Henning Schulzrinne, “Feature interaction in Internet Telephony”, Proceedings Feature Interaction in Telecommunications and Software Systems VI, Glasgow, United Kingdom, Maio 2000. URL: http://www.cs.columbia.edu/~hgs/papers/Lenn0005_Feature.pdf, Julho 2002.

[Low, 1996] C. Low, “The Internet Telephony red herring”, in Proceedings of Global Internet'96, London, England, Novembro 1996. URL: http://citeseer.nj.nec.com/, Junho 2002.

[McCane, 1995] S. McCane, V. Jacobson - “vic: a flexible framework for packet video”, Proceedings of ACM Multimedia, pp. 511-522, San Francisco, CA, Novembro 1995. URL: http://citeseer.nj.nec.com/, Junho 2002.

[Michaely, 2000] Boaz Michaely, “In depth H.323 overview”, Novembro 2000. URL: http://www.securitytechnet.com/resource/hot-topic/voip/indepth_H323.ppt, Junho 2002.

[Nee, 1997] Peter A. Nee, “Experimental evaluation of two-dimensional media scaling techniques for Internet Videoconferencing”, Tese de Mestrado, Universidade da Carolina do Norte, 1997. URL: http://www.cs.unc.edu/~jeffay/students/nee-97/nee-97.pdf, Setembro 2002.

[NIST-SIP, 2002] M. Ranganathan, “Internet Telephony/VOIP”,

http://dns.antd.nist.gov/proj/iptel/, Novembro 2002.

[Packetizer, 2002a] Packetizer, “H.323 version 2 - overview”, http://www.packetizer.com/iptel/h323/whatsnew_v2.html, Outubro 2002.

[Packetizer, 2002b] Packetizer, “H.323 version 3 - overview”, http://www.packetizer.com/iptel/h323/whatsnew_v3.html, Outubro 2002.

[Packetizer, 2002c] Packetizer, “H.323 version 4 - overview”, http://www.packetizer.com/iptel/h323/whatsnew_v4.html, Outubro 2002.

[Packetizer, 2002d] Packetizer, “Current H.323 series documents in progress”, http://www.packetizer.com/iptel/h323/doc_status.html, Outubro 2002.

[Pan, 1998] P. Pan and H. Schulzrinne, “YESSIR: A simple Reservation Mechanism for the Internet”, in Proceedings of NOSSDAV'98, Cambridge, UK, Junho 1998. URL: http://www.cs.columbia.edu/~pingpan/papers/yessir.pdf, Julho 2002.

[Ramjee, 1995] R. Ramjee, J. Kurose, D. Towsley, H. Schulzrinne, “Adaptive playout mechanisms for packetized audio applications in wide-area networks”, in Proceedings of the Conference on Computer Communications (IEEE Infocom), Toronto, Canada, 1994. URL: http://citeseer.nj.nec.com, Setembro 2002.

[RFC 741, 1977] D. Cohen, “Specification for Network Voice Protocol (NVP)”, Network Working Group, Request For Comments 741, Internet Engineering Task Force, Novembro 1977. URL: http://www.ietf.org, Julho 2002.

[RFC 768, 1980] J. Postel, “User Datagram Protocol”, Request for Comments 768, Internet Engineering Task Force, Agosto de 1980. URL: http://www.ietf.org, Junho 2002.

[RFC 1034, 1987] P. V. Mockapetris, “Domain names - concepts and facilities”, Request for Comments 1034, Internet Engineering Task Force, Novembro 1987. URL: http://www.ietf.org, Junho 2002.

[RFC 1035, 1987] P. V. Mockapetris, “Domain names - implementation and specification,” Request for Comments 1035, Internet Engineering Task Force, Novembro 1987. URL: http://www.ietf.org, Junho 2002.

[RFC 1777, 1995] W. Yeong, T. Howes, S. Kille, “Lightweight Directory Access Protocol”, Request for Comments 1777, Internet Engineering Task Force, Março 1995. URL: http://www.ietf.org, Agosto 2002.

[RFC 1835, 1995] P. Deutsch, R. Schoultz, P. Faltstrom e C. Weider, “Architecture of the whois++ service”, Request for Comments 1835, Internet Engineering Task Force, Agosto 1995. URL: http://www.ietf.org, Julho 2002.

[RFC 1889, 1996] Schulzrinne, H., Casner, S., Frederick, R. E V. Jacobson, "RTP: a transport protocol for real-time applications", Request For Comments 1889, Internet Engineering Task Force, Janeiro 1996. URL: http://www.ietf.org, Junho 2002.

[RFC 1890, 1996] H. Schulzrinne , “RTP profile for audio and video conferences with minimal control”, Request for Comments 1890, Internet Engineering Task Force, Janeiro 1996. URL: http://www.ietf.org, Julho 2002.

[RFC 1913, 1996] C. Weider, J. Fullton e S. Spero, “Architecture of the whois++ index service”, Request for Comments 1913, Internet Engineering Task Force, Fevereiro 1996. URL: http://www.ietf.org, Julho 2002.

[RFC 2032, 1996] T. Turletti e C. Huitema, “RTP payload format for H.261 video streams”, Request for Comments 2032, Internet Engineering Task Force, Outubro 1996. URL: http://www.ietf.org, Julho 2002.

[RFC 2190, 1997] C. Zhu, “RTP payload format for H.263 video streams”, Request for Comments 2190, Internet Engineering Task Force,

Setembro1997. URL: http://www.ietf.org, Agosto 2002.

[RFC 2198, 1997] C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J.C. Bolot, A. Vega-Garcia, S. Fosse-Parisis, “RTP payload for redundant audio data”, Request For Comments 2198, Internet Engineering Task Force, Setembro 1997. URL: http://www.ietf.org, Agosto 2002.

[RFC 2205, 1997] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, “Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification”, Request for Comments 2205, Internet Engineering Task Force, Setembro 1997. URL: http://www.ietf.org, Julho 2002.

[RFC 2205, 1997] R. Braden, L. Zhang, S. Berson, S. Herzog e S. Jamin, “Resource ReSerVation protocol (RSVP) – version 1 functional specification”, Request for Comments 2205, Internet Engineering Task Force, Setembro 1997. URL: http://www.ietf.org, Julho 2002.

[RFC 2209, 1997] R. Braden, L. Zhang, “Resource ReSerVation Protocol (RSVP) – Version 1 message processing rules”, Request for Comments 2209, Internet Engineering Task Force, Setembro 1997. URL: http://www.ietf.org, Agosto 2002.

[RFC 2246, 1999] Dierks, T. e C. Allen, "The TLS protocol version 1.0", Request for Comments 2246, Internet Engineering Task Force, Janeiro 1999. URL: http://www.ietf.org, Setembro 2002.

[RFC 2250, 1998] D. Hoffman, G. Fernando, V. Goyal e M. Civanlar, “RTP payload format for MPEG1/MPEG2 video”, Request for Comments 2250, Internet Engineering Task Force, Janeiro1998. URL: http://www.ietf.org, Agosto 2002.

[RFC 2326, 1998] H. Schulzrinne, A. Rao e R. Lanphier, "Real Time Streaming Protocol (RTSP)", Request For Comments 2326, Abril 1998. URL: http://www.ietf.org, Agosto 2002.

[RFC 2327, 1998] M. Handley e V. Jacobson, "SDP: Session Description Protocol", Request For Comments (RFC 2327), Internet Engineering Task

Force, Abril 1998. URL: http://www.ietf.org, Julho 2002.

[RFC 2354, 1998] C. Perkins e O. Hodson, “Options for repair of streaming media,” Request for Comments 2354, Internet Engineering Task Force, Junho 1998. URL: http://www.ietf.org, Agosto 2002.

[RFC 2401, 1998] Kent, S. e R. Atkinson, "Security architecture for the Internet protocol", Request for Comments 2401, Internet Engineering Task Force, Novembro 1998. URL: http://www.ietf.org, Setembro 2002.

[RFC 2474, 1998] K. Nichols, S. Blake, F. Baker, D. Black, “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 headers”, Request for Comments 2474, Internet Engineering Task Force, Dezembro 1998. URL: http://www.ietf.org, Agosto 2002.

[RFC 2475, 1998] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 headers”, Request for Comments 2475, Internet Engineering Task Force, Dezembro 1998. URL: http://www.ietf.org, Agosto 2002.

[RFC 2871, 2000] J. Rosenberg e H. Schulzrinne, “A framework for telephony routing over IP”, Request for Comments 2871, Internet Engineering Task Force, Junho 2000. URL: http://www.ietf.org, Julho 2002.

[RFC 3015, 2000] F. Cuervo, N. Greene, A. Rayhan, C. Huitema, B. Rosen e J. Segers, "Megaco protocol version 1.0", Request For Comments 3015, Internet Engineering Task Force, Novembro 2000. URL: http://www.ietf.org, Julho 2002.

[RFC 3047, 2001] P. Luthi, “RTP payload format for ITU-T recommendation G.722.1”, Request for Comments 3047, Internet Engineering Task Force, Janeiro 2001. URL: http://www.ietf.org, Setembro 2002.

[RFC 3219, 2002] J. Rosenberg, H. Salama, M. Squire, “Telephony Routing over IP (TRIP)”, Request For Comments 3219, Internet Engineering Task Force, Janeiro 2002. URL: http://www.ietf.org, Julho 2002.

[RFC 3261, 2002] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J.

Peterson, R. Sparks, M. Handley, E. Schooler, " SIP: Session Initiation Protocol ", Request For Comments 3261, Internet Engineering Task Force, Junho 2002 (RFC 2543 obsoleta). URL: http://www.ietf.org, Julho 2002.

[RFC 3263, 2002] J. Rosenberg, H. Schulzrinne, "Session Initiation Protocol (SIP): locating SIP servers", Request for Comments 3263, Internet Engineering Task Force, Junho 2002 (RFC 2543 obsoleta). URL: http://www.ietf.org, Julho 2002.

[RFC 821, 1982] J. Postel, "Simple Mail Transfer Protocol", Request for Comments 821, Internet Engineering Task Force, Agosto 1982. URL: http://www.ietf.org, Agosto 2002.

[Rosenberg, 1998] J. Rosenberg e H. Shulzrinne, "Timer reconsideration for enhanced RTP scalability", Proceedings of Conference on Computer Communications (IEEE Infocom), San Franscico, California, Março/Abril 1998. URL: http://www.cs.columbia.edu/~hgs/papers/Rose9803_Timer.pdf, Julho 2002.

[Rosenberg, 2000] J. Rosenberg, H. Salama e M. Squire, "Telephony Routing over IP (TRIP)", Internet Draft, Internet Engineering Task Force, Novembro 2000. Work in progress. URL: http://www.ietf.org, Julho 2002.

[Rosenberg, 2001] J. Rosenberg, "Distributed algorithms and protocols for scalable Internet Telephony", PhD thesis, Columbia University, 2001. URL: http://www.jdrosen.net/papers/abstract.pdf, Julho 2002

[Schooler, 1989] Eve M. Schooler e Stephen L. Casner, "A packet-switched multimedia conferencing system", ACM Special Interest Group on Office Information Systems (SIGOIS) Bulletin, Vol. 10, pag. 12-22, Janeiro 1989. URL: http://citeseer.nj.nec.com/, Junho 2002.

[Schulzrinne, 1998] H. Schulzrinne e J. Rosenberg, "Signaling for Internet Telephony", in Proceedings of 6th IEEE International Conference on Network Protocols (ICNP), Austin, Texas, Outubro 1998. URL:

http://www.cs.columbia.edu/~hgs/papers/Schu9810_Signaling.pdf, Julho 2002.

[Schulzrinne, 1999] H. Schulzrinne e J. Rosenberg, "Internet Telephony: architecture and protocols an IETF perspective", Computer Networks and ISDN Systems, vol. 31/3 pag. 237-255, Fevereiro 1999. URL: www.cs.columbia.edu/~hgs/papers/Schu9902_Internet.pdf, Julho 2002.

[Schulzrinne, 2000] H. Schulzrinne e J. Rosenberg, "Internet Telephony", Fevereiro 2000. URL: http://citeseer.nj.nec.com/, Junho 2002.

[Schulzrinne, 2001] H. Schulzrinne, "The Session Initiation Protocol, tuturial", Maio 2001. URL: http://www.cs.columbia.edu/~hgs/, Abril 2002.

[Schulzrinne, 2002a] H. Schulzrinne, "Historical notes", http://www.cs.columbia.edu/~hgs/rtp/history.html, Julho 2002.

[Schulzrinne, 2002b] H. Schulzrinne, "Classification for SIP interoperability test event", http://www.cs.columbia.edu/~hgs/sip/sipit/classification.html

[Umair, 2002] Umair, "ECTE997 group project: real-time streaming of digital media files", http://edt.uow.edu.au/edtlab/ecte997/ecte997-group-projects/group03/conferencing.html, Agosto 2002.

[Wang, 1999] Xin Wang, H. Schulzrinne, "RNAP: A Resource Negotiation and Pricing Protocol", In Proceedings International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'99), Basking Ridge, New Jersey, pag. 77-93, Junho de 1999. URL: http://www.cs.bu.edu/pub/imic/talks/schulzrinne.pdf, Setembro 2002.

Anexo A

Interfaces gráficas do sIPtel

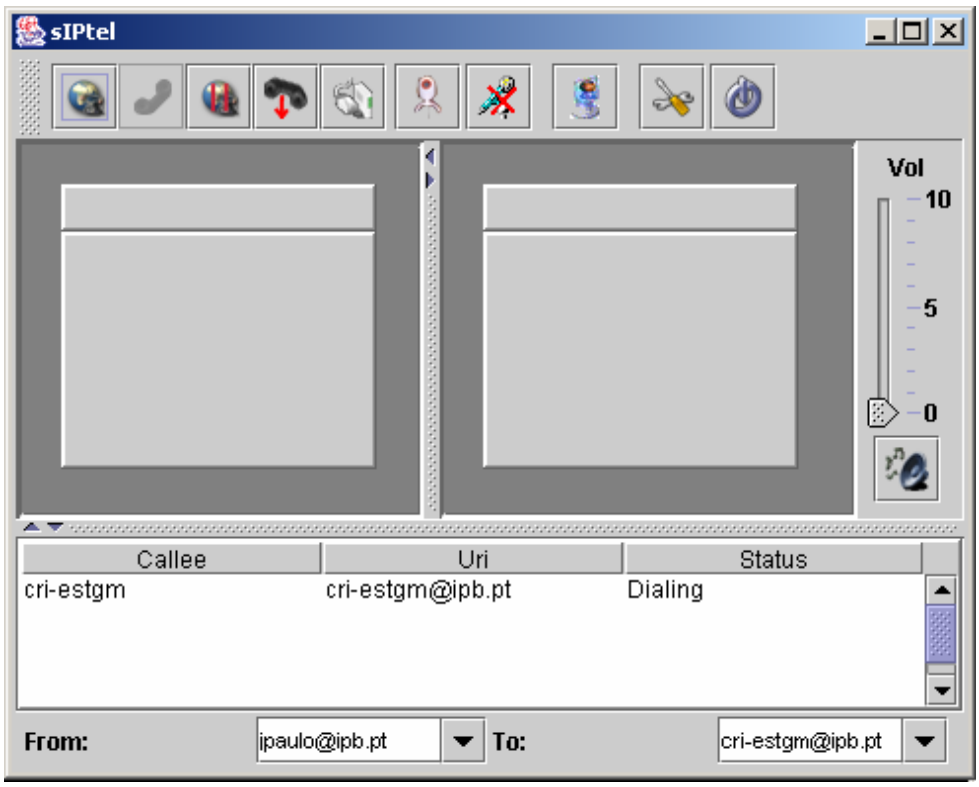


Figura A.1 – Interface principal do sIPtel.

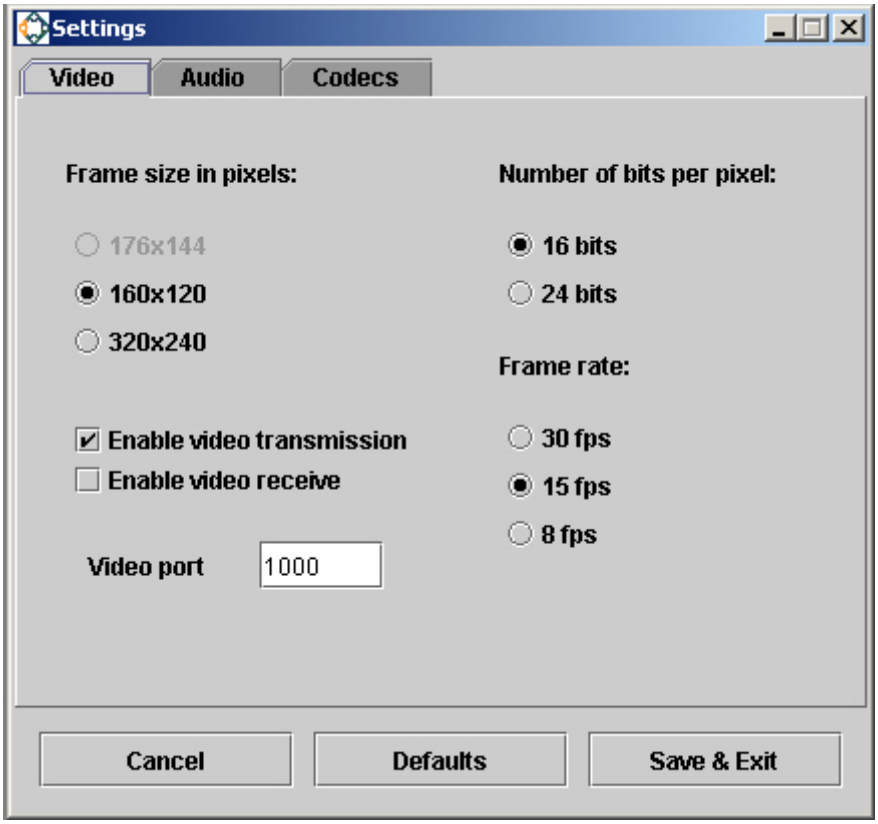


Figura A.2 – Janela de configuração de parâmetros de vídeo

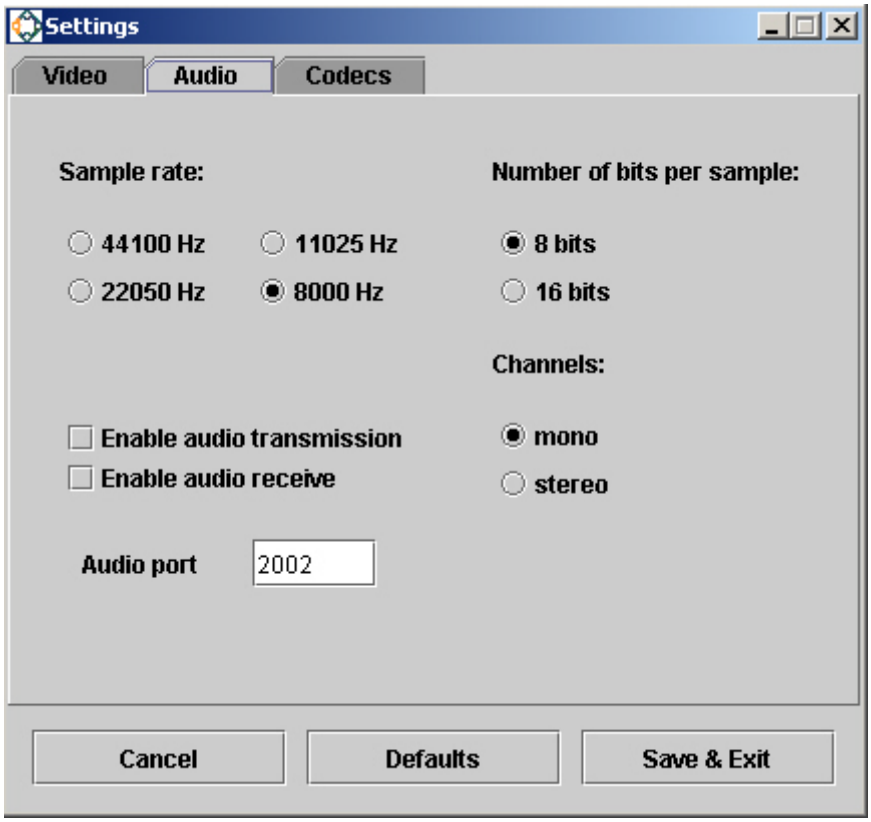


Figura A.3 – Janela de configuração de parâmetros de áudio

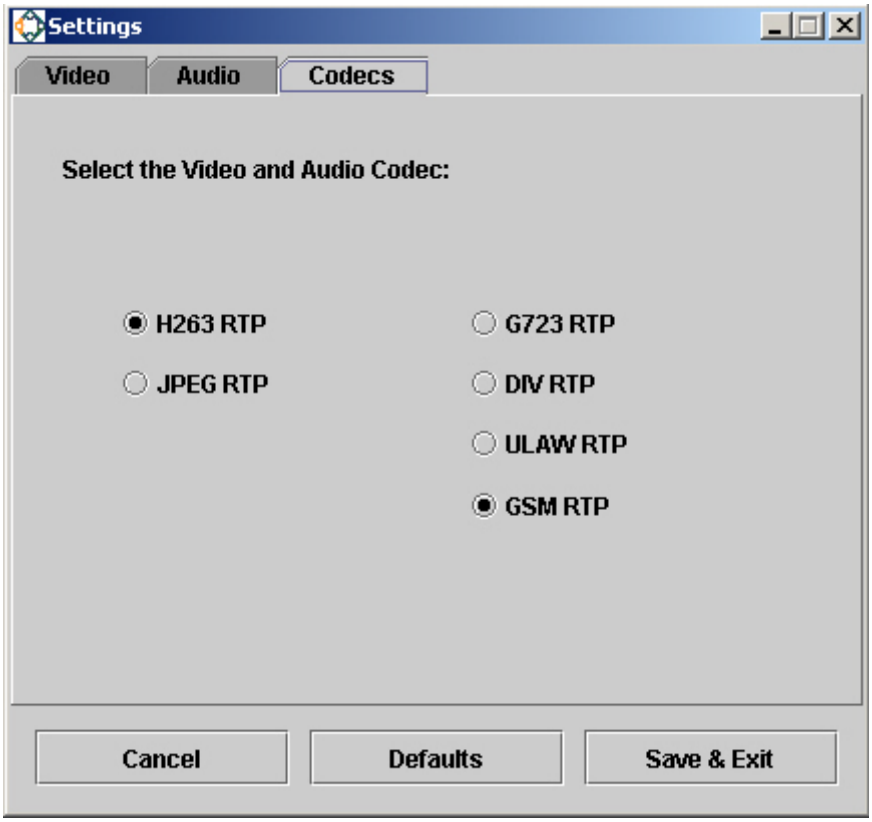


Figura A.4 – Janela de configuração de codificadores

Anexo B

Diagramas de estado

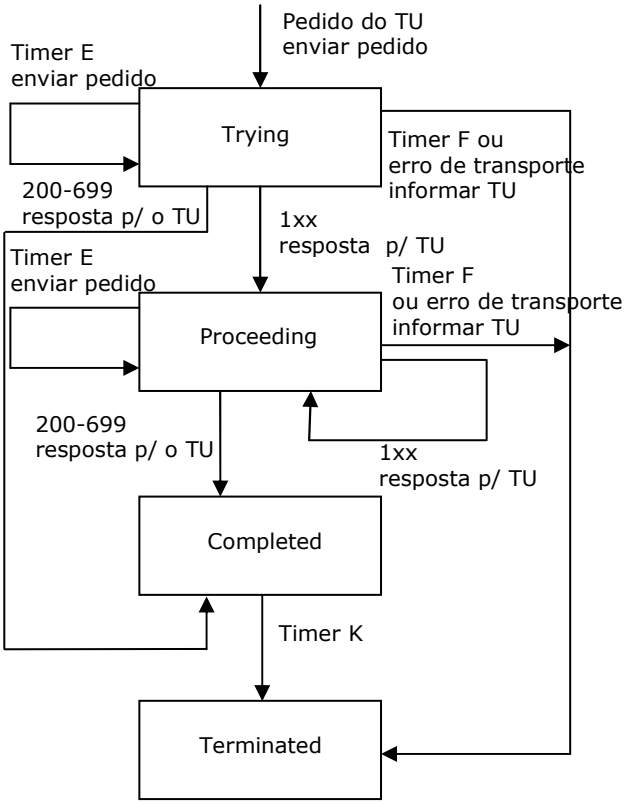


Figura B.1 – Diagrama de estados *Non-Invite Client Transaction*

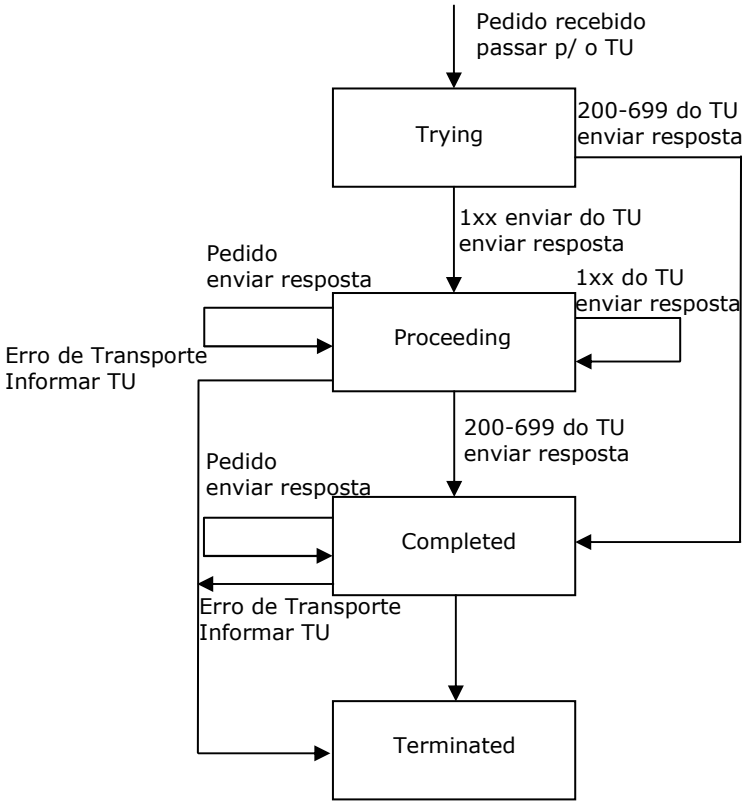


Figura B.2 – Diagrama de estados *Non-Invite Server Transaction*

Anexo C

Cenários de chamadas

Este cenário representa um exemplo típico da troca de mensagens entre dois utilizadores também denominado *SIP Trapezoid*, para o estabelecimento, negociação de parâmetros dos meios da sessão e a finalização da sessão estabelecida. O utilizador *sipadmin* inicia uma chamada enviando um INVITE através do sIPTel para o utilizador *jpaulo*. Entre os dois utilizadores existe um servidor *SIP Proxy* que facilita o estabelecimento da chamada. Após o estabelecimento da chamada e da troca dos meios, o *sipadmin* finaliza a chamada através do envio de um BYE para o *jpaulo*, que confirma a recepção do BYE com o envio de um 200 OK.

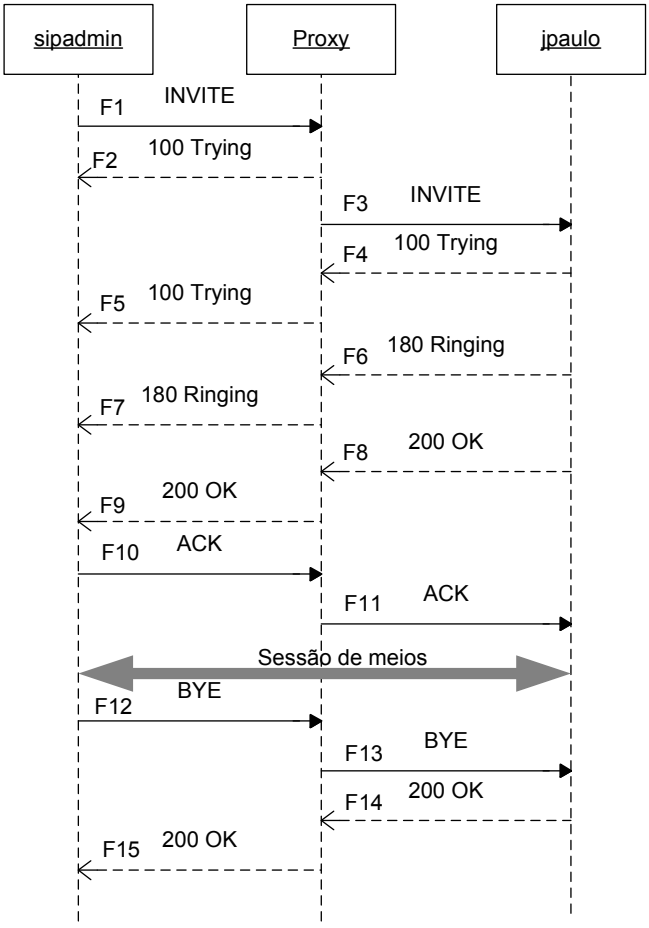


Figura C.1 – Exemplo típico de troca de mensagens entre dois utilizadores

Mensagens SIP:

```
------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.154:5060
to = 193.137.96.144:2000
F1 – INVITE sipadmin -> Proxy

INVITE sip:jpaulo@ipb.pt SIP/2.0
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
Contact: <sip:sipadmins@193.137.96.154:5060>
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>
CSeq: 1 INVITE
Max-Forwards: 70
Content-Type: aplication/sdp
Content-Length: 173

v=0
o=sipadmins 3512760627430083382 3512760627430083382 IN IP4 193.137.96.154
s=VoIP
c=IN IP4 193.137.96.154
t=0 0
m=audio 2002/1 RTP/AVP 3
a=rtpmap:3 GSM RTP/8000/1


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.154:5060
F2 – 100 Trying Proxy -> sipadmin

SIP/2.0 100 Trying
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>
CSeq: 1 INVITE
Content-Type: aplication/sdp
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.144:5060
F3 – INVITE Proxy -> jpaulo

INVITE sip:jpaulo@193.137.96.144:5060 SIP/2.0
Contact: <sip:sipadmins@193.137.96.154:5060>
Max-Forwards: 70
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 INVITE
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>
Content-Type: aplication/sdp
Via: SIP/2.0/UDP
```

```
193.137.96.144:2000;branch=z9hG4bKb5050691aa23d1e22f3140374c2c004c,SIP/2.0/
UDP 193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Content-Length:0020173

v=0
o=sipadmins 3512760627430083382 3512760627430083382 IN IP4 193.137.96.154
s=VoIP
c=IN IP4 193.137.96.154
t=0 0
m=audio 2002/1 RTP/AVP 3
a=rtpmap:3 GSM RTP/8000/1


-------------------------------------------------------------------------
-------------------------------------------------------------------------
from = 193.137.96.144:5060
to = 193.137.96.144:2000
F4 - 100 Trying jpaulo -> Proxy

SIP/2.0 100 Trying
Via: SIP/2.0/UDP
193.137.96.144:2000;branch=z9hG4bKb5050691aa23d1e22f3140374c2c004c,SIP/2.0/
UDP 193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 INVITE
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Content-Length: 0


-------------------------------------------------------------------------
-------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.154:5060
F5 - 100 Trying Proxy -> sipadmin

SIP/2.0 100 Trying
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>
CSeq: 1 INVITE
Content-Type: aplication/sdp
Content-Length: 0


-------------------------------------------------------------------------
-------------------------------------------------------------------------
from = 193.137.96.144:5060
to = 193.137.96.144:2000
F6 - 180 Ringing jpaulo -> Proxy

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP
193.137.96.144:2000;branch=z9hG4bKb5050691aa23d1e22f3140374c2c004c,SIP/2.0/
UDP 193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>
```

```
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 INVITE
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Contact: <sip:jpaulo@193.137.96.144:5060>
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.154:5060
F7 - 180 Ringing Proxy -> sipadmin

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 INVITE
Contact: <sip:jpaulo@193.137.96.144:5060>
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:5060
to = 193.137.96.144:2000
F8 - 200 OK jpaulo -> Proxy

SIP/2.0 200 OK
Via: SIP/2.0/UDP
193.137.96.144:2000;branch=z9hG4bKb5050691aa23d1e22f3140374c2c004c,SIP/2.0/
UDP 193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>;tag=1663
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 INVITE
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Contact: <sip:jpaulo@193.137.96.144:5060>
Content-Type: aplication/sdp
Content-Length: 239

v=0
o=jpaulo 3512760627430083383 3512760627430083383 IN IP4 193.137.96.144
s=VideoConference
c=IN IP4 193.137.96.144
t=0 0
m=audio 2002/1 RTP/AVP 4
m=video 1000/1 RTP/AVP 34
a=rtpmap:4 G723 RTP/8000/1
a=rtpmap:34 H263 RTP/16000/1


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.154:5060
F9 - 200 OK Proxy -> sipadmin

SIP/2.0 200 OK
```

```
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Contact: <sip:jpaulo@193.137.96.144:5060>
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 INVITE
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>;tag=1663
Content-Type: aplication/sdp
Content-Length: 239

v=0
o=jpaulo 3512760627430083383 3512760627430083383 IN IP4 193.137.96.144
s=VideoConference
c=IN IP4 193.137.96.144
t=0 0
m=audio 2002/1 RTP/AVP 4
m=video 1000/1 RTP/AVP 34
a=rtpmap:4 G723 RTP/8000/1
a=rtpmap:34 H263 RTP/16000/1


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.154:5060
to = 193.137.96.144:2000
F10 – ACK sipadmin -> Proxy

ACK sip:jpaulo@ipb.pt SIP/2.0
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 ACK
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>;tag=1663
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.144:5060
F11 – ACK Proxy -> jpaulo

ACK sip:jpaulo@193.137.96.144:5060 SIP/2.0
Via: SIP/2.0/UDP
193.137.96.144:2000;branch=z9hG4bKb5050691aa23d1e22f3140374c2c004c
CSeq: 1 ACK
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>;tag=1663
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
Max-Forwards: 32
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.154:5060
to = 193.137.96.144:2000
F12 – BYE sipadmin -> Proxy
```

```
BYE sip:jpaulo@ipb.pt SIP/2.0
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 BYE
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>;tag=1663
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.144:5060
F13 – BYE Proxy -> jpaulo

BYE sip:jpaulo@193.137.96.144:5060 SIP/2.0
Max-Forwards: 70
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 BYE
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>;tag=1663
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=udp>
Via: SIP/2.0/UDP
193.137.96.144:2000;branch=z9hG4bKd65da64ad1837421fca4e185938f4a7b
Route: <sip:jpaulo@193.137.96.144:5060>
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:5060
to = 193.137.96.144:2000
F12 – 200 OK jpaulo -> Proxy

SIP/2.0 200 OK
Via: SIP/2.0/UDP
193.137.96.144:2000;branch=z9hG4bKd65da64ad1837421fca4e185938f4a7b
From: <sip:sipadmins@ipb.pt>;tag=2581
To: <sip:jpaulo@ipb.pt>;tag=1663
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 BYE
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Content-Length: 0


------------------------------------------------------------------------
------------------------------------------------------------------------
from = 193.137.96.144:2000
to = 193.137.96.154:5060
F12 – 200 OK Proxy -> sipadmin

SIP/2.0 200 OK
Via: SIP/2.0/UDP
193.137.96.154:5060;branch=z9hG4bK4923ed7ac5826486d20181cc2922e923
Record-Route: <sip:193.137.96.144:2000;maddr=193.137.96.144;transport=UDP>
Call-Id: 0e21b41109e658a3e80a729aa80b27f6@193.137.96.154
CSeq: 1 BYE
From: <sip:sipadmins@ipb.pt>;tag=2581
```

```
To: <sip:jpaulo@ipb.pt>;tag=1663
Contact: <sip:jpaulo@193.137.96.144:2000;transport=udp>
Content-Length: 0
```

Neste cenário o utilizador *sipadmin* estabelece uma chamada com o utilizador *jpaulo* através das mensagens F1 até F11, seguindo-se a troca dos meios especificados na mensagem SDP. Enviando um novo INVITE (F12) o *sipadmin* coloca o *jpaulo* em espera. Isto acontece porque o INVITE transporta na mensagem SDP o parâmetro '*c*' que contém o endereço 0.0.0.0. Finalmente o utilizador *sipadmin* retoma a chamada no INVITE F19 com os novos parâmetros da sessão.

Devido às mensagens trocadas serem semelhantes às do cenário anterior, optou-se por não as enunciar. No entanto, diversos cenários podem ser encontrados em *Internet Drafts* do IETF.

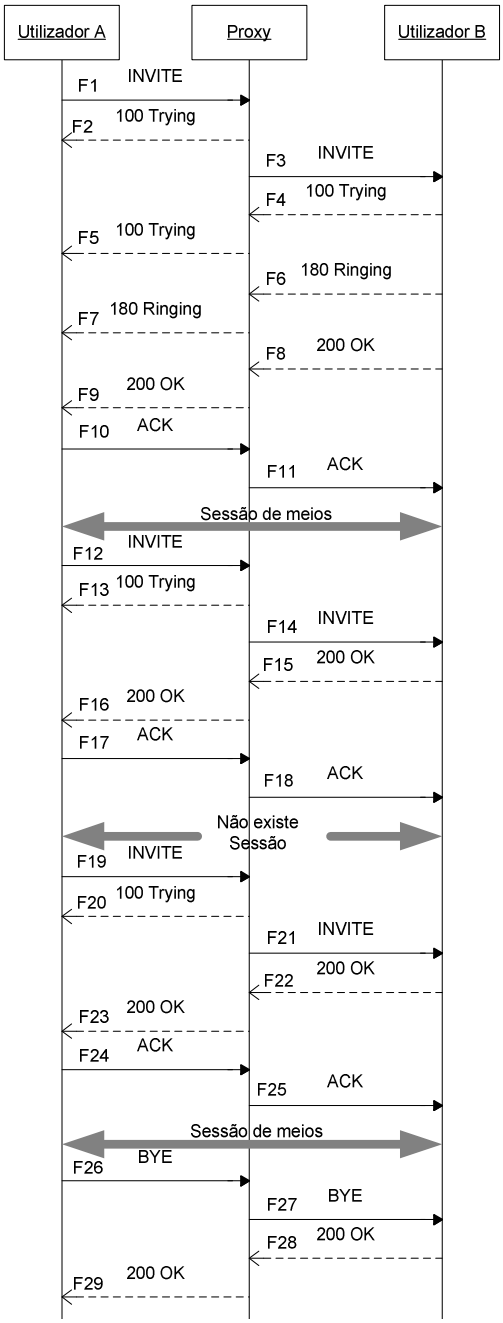
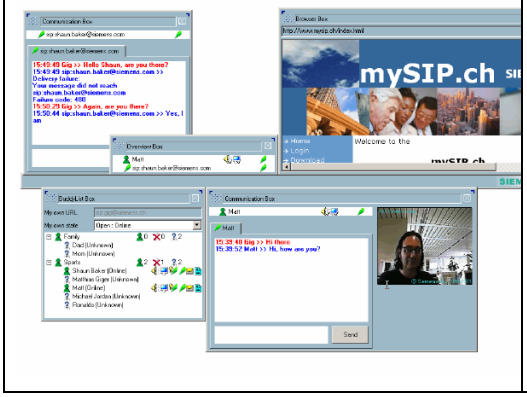


Figura C.2 – Estabelecimento de chamada e colocação desta em espera

Anexo D

Exemplos de Telefones IP com suporte para vídeo

	Telefone IP com suporte para vídeo, DV324 da 8x8 Inc. Inclui uma câmara e um *display* LCD a cores. Utiliza a recomendação H.324 para a transmissão de áudio e vídeo sobre a rede analógica PSTN. É também compatível com outros sistemas H.324 incluindo pacotes de *software* utilizados nos PCs.
	Telefone IP com suporte para vídeo, DV325 da 8x8 Inc. Inclui uma câmara e um *display* LCD a cores. Contém cancelamento de eco e suporta vídeo até 30 frames por segundo. É baseado no SIP e é compatível com outros terminais SIP com suporte para vídeo ou apenas áudio, incluindo o *Microsoft XP Messenger*.
	Telefone IP com suporte para vídeo, CV-300 da Citron. Inclui uma câmara e um *display* LCD a cores. Utiliza o protocolo H.323, suporta conferência e é compatível com o *Microsoft NetMeeting*.

	Software IPtel, SCS Client da Siemens. Suporta vídeo, mensagens instantâneas, notificação de presença, *web browser*, transferência de ficheiros, conferência de áudio e vídeo, autenticação sobre SSL e utiliza o protocolo SIP. É compatível como o *Microsoft XP Messenger* e maior parte dos produtos que utilizam a tecnologia SIP.