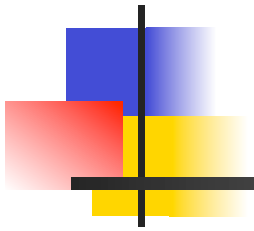


Cap. 2 – O nível aplicação

(3ª parte)

Nota prévia

A estrutura da apresentação é semelhante e utiliza algumas das figuras, textos e outros materiais do livro de base do curso

James F. Kurose and Keith W. Ross, "Computer Networking - A Top-Down Approach Featuring the Internet," Addison Wesley Longman, Inc., 3rd Edition, 2005

Leitura prévia

Nesta 3ª parte do capítulo 2 do curso, a leitura prévia corresponde às secções 7.1 a 7.3 do capítulo 7 do livro de suporte ao mesmo

James F. Kurose and Keith W. Ross, "Computer Networking - A Top-Down Approach Featuring the Internet," Addison Wesley Longman, Inc., 3rd Edition, 2005

Organização do capítulo

- Aplicações em rede ou distribuídas
- Conceitos de base, paradigmas e tipos de transportes
- Protocolo HTTP (Web)
- O DNS ("Domain Name System")
- Protocolo SMTP — Correio electrónico
- Transferência de ficheiros - Protocolo FTP e sistemas P2P
- **Os protocolos RTP e SIP**

Aplicações Multimédia sobre a Internet

Classes destas aplicações:

1) Streaming stored audio and video (video on-demand)

2) Streaming live audio and video (IP TV, IP Radio, ...)

3) Real-time interactive audio and video (IP Phone, Video conference, ...)

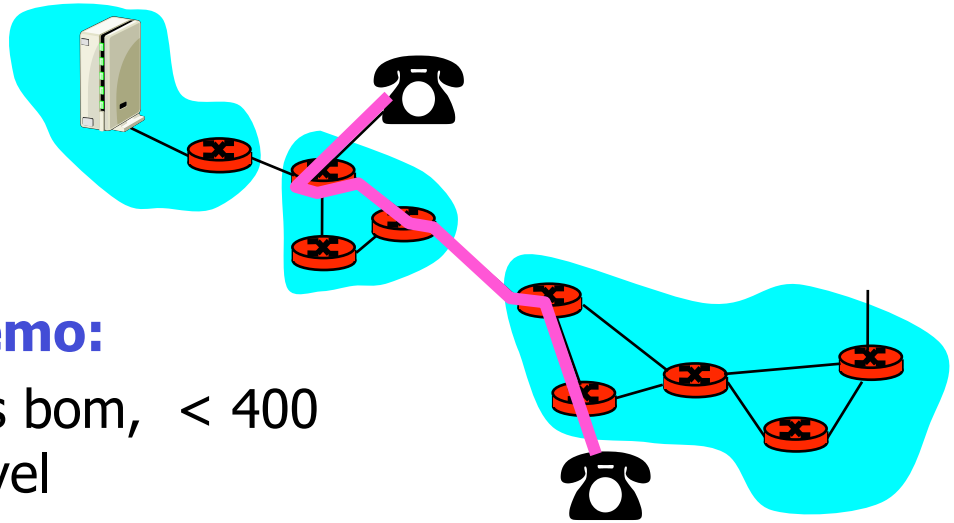
Jitter é a variação do tempo de trânsito (*delay*) dos pacotes do mesmo fluxo

Características fundamentais:

- Tipicamente **sensíveis ao atraso**
  - Atraso extremo a extremo
  - *jitter*
- Mas **toleram perca de pacotes**: pequenas percas provocam defeitos toleráveis no som ou na imagem
- Antítese das aplicações "elásticas" que toleram variação da capacidade mas não toleram perca de dados

IP Phone, Video Conference, ...

- **Aplicações interactivas em tempo real**



- **Requisitos extremo a extremo:**
  - áudio: < 150 mili segundos bom, < 400 mili segundos ainda aceitável
    - Inclui o tempo para digitalizar, empacotar, trânsito pela rede, ...
    - Atrasos superiores impedem a interactividade
- **Início da sessão:**
  - Como é que se sabe os endereços e formatos usados pelos interlocutores ?

Digitalização do som

- O sinal analógico é amostrado (*sampled*) a um ritmo constante
  - telefone: 8,000 amostras / s
  - CD: 44.100 amostras / s
- Cada amostra é quantificada (aproximada por um valor)
  - Por exemplo 2^8=256 valores possíveis
- Cada valor é representado em bits
  - 8 bits permitem representar 256 valores possíveis
- Exemplo: 8.000 amostras / s, com 256 valores possíveis cada, implica uma velocidade de transmissão de 64,000 bps ou 64 Kbps
- O receptor volta a realizar a conversão para sinal analógico
  - Implica necessariamente alguma perda de informação

Requisitos de capacidade

As aplicações multimédia destinam-se, em última análise, a transmitir informação multimédia. Um bom ponto de partida é analisar os requisitos de capacidade, necessários para transmitir este tipo de informação.

Exemplos com som codificado PCM – "*pulse code modulation*":
Voz "telefónica" codificada em 8 bits (2^8 = 256 valores distintos): com 8 K Hz de frequência de amostragem, o som exige 8 x 8.000 = 64 K bps por canal
Som do CD codificado em 16 bits: 44,1 K Hz de frequência de amostragem; os dois canais estéreo exigem em conjunto cerca de 1,411 Mbps

Exemplos com som codificado com compressão:
Telefone e IP Radio: 8 Kbps (G.729), 14 Kbps (GSM), 30, 48, 96, ... Kbps (som estéreo de boa qualidade de uma emissão rádio)
Internet telephony (VoIP): 5.3 - 13 Kbps
MP3: 93, 128, 180, ... Kbps (MPEG layer 3 ou MP3 – som estéreo de qualidade quase comparável à do CD)

CODECS

- **Um CODEC é um dispositivo hardware com software, ou um circuito VLSI, que realiza a transformação do sinal analógico para uma codificação digital ou vice-versa (code / decode).**

- **Há CODECS simples como os CODECS PCM dos telefones digitais ou das centrais telefónicas. Estes dispositivos apenas transformam o som codificado de forma analógica em digital e vice versa através de uma amostragem de 8 bits com uma frequência de amostragem de 8 KHz.**

- **Há CODECS muito complexos como os CODECS MPEG-2 existentes nos DVDs, em placas para PC ou em software. Estes CODECS codificam / separam e comprimem / descomprimem vários canais de vídeo e de voz**

- **Há CODECS públicos, isto é, normalizados, e CODECS proprietários, isto é patenteados e de utilização sujeita a pagamento.**

IP Phone: desafios e soluções possíveis

- Os protocolos de transporte TCP e UDP providenciam um serviço sem garantias no que toca a capacidade, atraso ou *jitter*

Soluções:

- Usar UDP para evitar os atrasos suplementares introduzidos pelo TCP quando há perca de pacotes

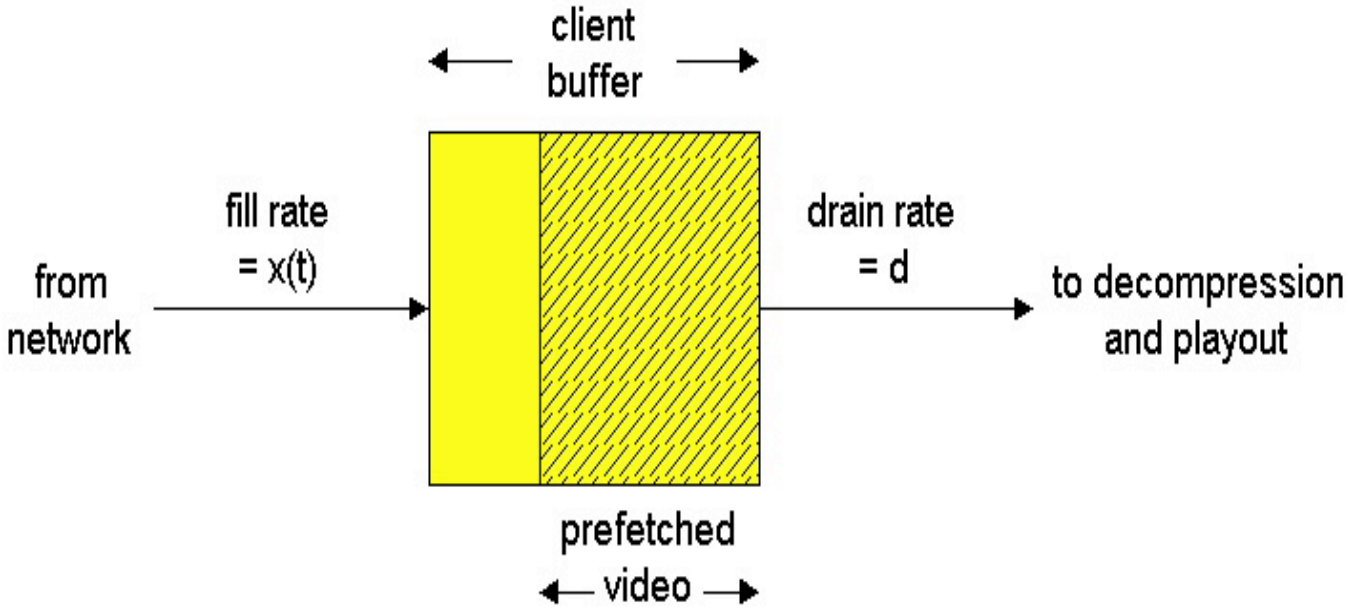
- Colocar em filas de espera (*bufferizar*) os dados antes de os começar a reproduzir para o utilizador, para acomodar, tanto quanto possível, as variações do ritmo de chegada (*jitter*)

- Tentar compensar ao nível aplicação a eventual perca de pacotes

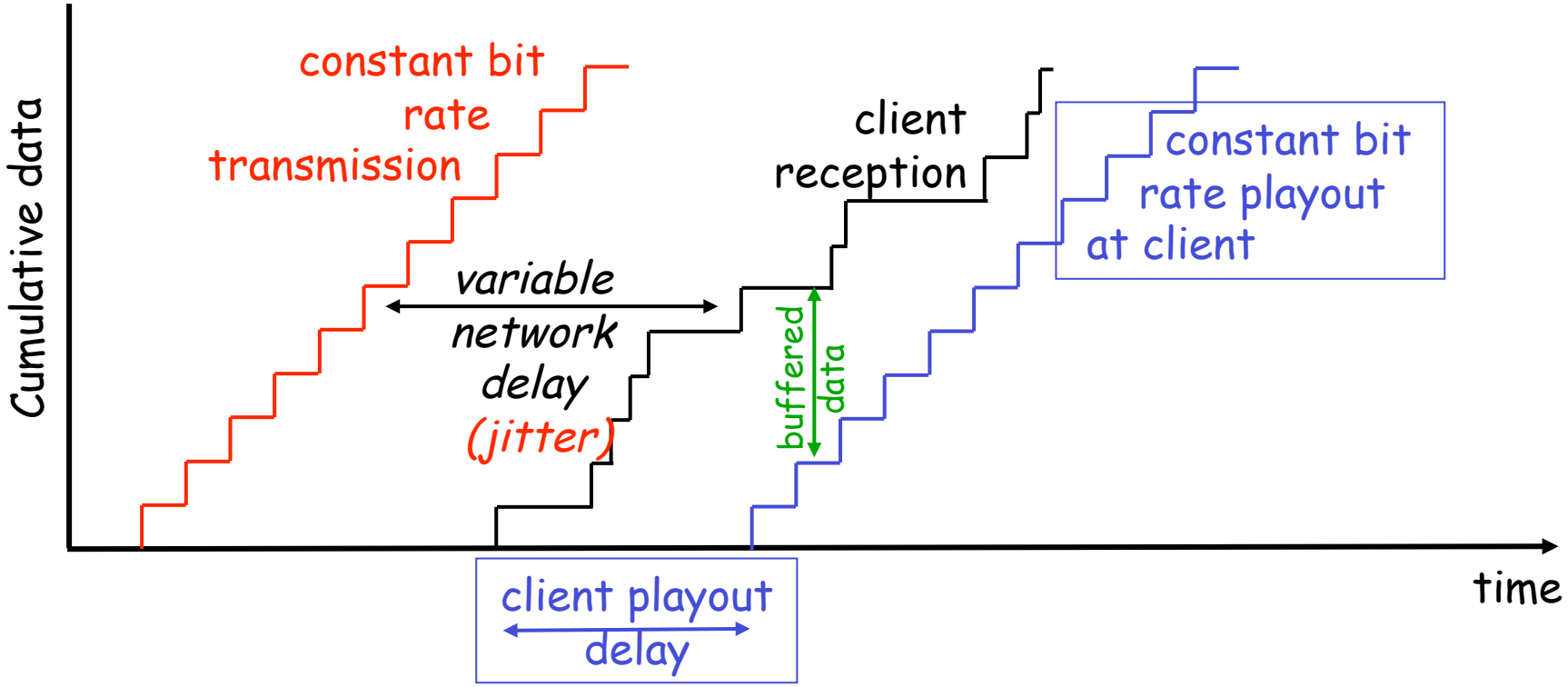
- Adaptar o tipo de compressão e de resolução, isto é, o CODEC, à capacidade disponível

Atraso da reprodução para compensar o *jitter*

- Antes de reproduzir o som recebido, a aplicação pode manter os dados numa fila de espera (*buffer*); desta forma existe a possibilidade de compensar, até certo nível, a variação ou *jitter* do tempo de trânsito

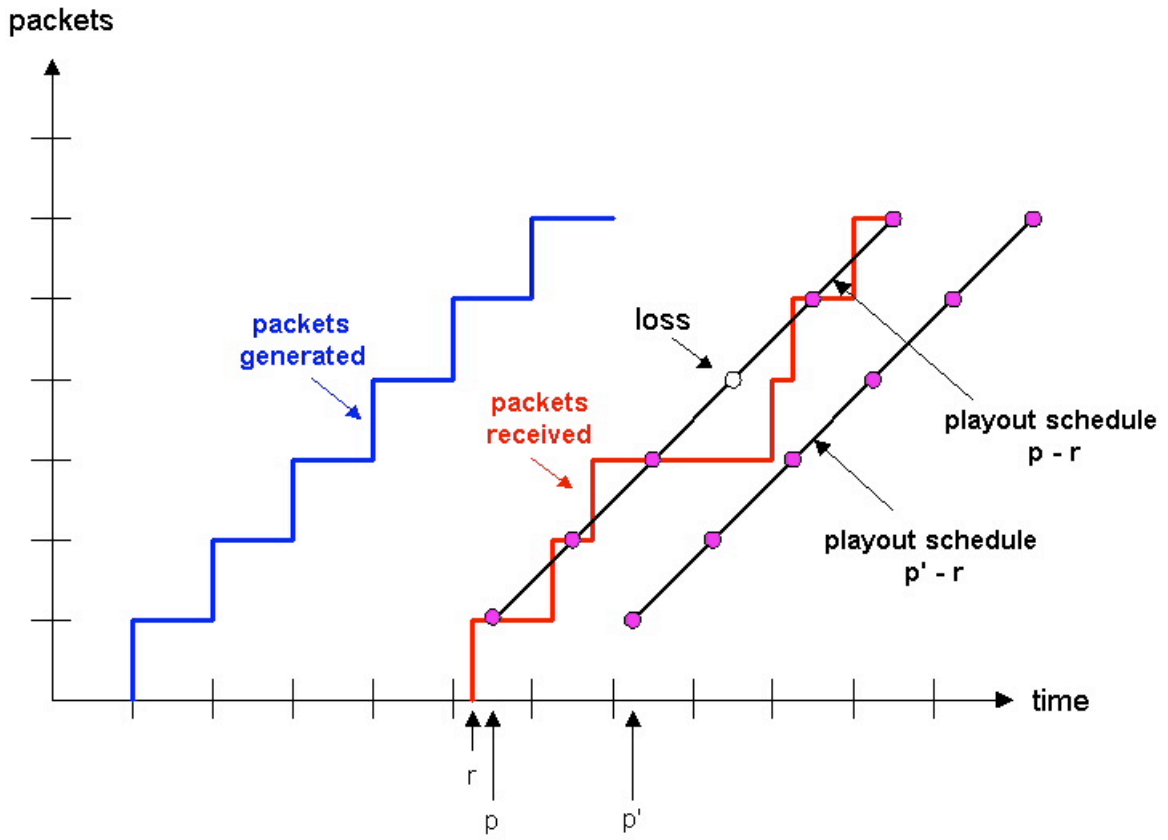


Como funciona



- Os pacotes são emitidos, por exemplo, de 20 em 20 mili segundos, mas a diferença de tempo entre cada dois pacotes recebidos pode ser maior ou menor que 20 mili segundos

Que valor para o *playout delay* ?



Playout delay

- ***Playout Delay.*** **Quanto maior for o *playout delay* maior é a capacidade de acomodar um maior *jitter* e maior é a probabilidade de não perder pacotes.**

- **Com efeito, os pacotes que chegam depois de esgotado o *playout delay* são ignorados e é como se se tivessem perdido, mas os que chegarem antes ainda são aproveitados.**

- **O problema é que um *playout delay* junta-se ao atraso de extremo a extremo para tornar a situação cada vez pior para o ouvinte.**

- **Em resumo. Se o atraso médio é *Am* e se se usa um *playout delay* de *D*, então o atraso de extremo a extremo passa a ser em média *Am+D*.**

- **Outra alternativa mais sofisticada é adoptar um *playout delay* variável e adaptável à situação da rede em função do tempo de trânsito médio.**

Perca de pacotes

- **A perca de pacotes numa aplicação de tempo real é tomada num sentido lato: pacotes que nunca chegam ou que chegam depois de um limite de tempo que os tornam inúteis.**

- **As técnicas de retransmissão, como na transferência de dados, não se aplicam porque as mesmas necessariamente introduzem aumento do tempo de transferência e da respectiva variância (*jitter*) e obrigariam a um *playout delay* demasiado grande para a aplicação.**

- **Os diferentes parâmetros (perca de pacotes, tempo de trânsito e *jitter*) são críticos e numa rede de pacotes é bastante normal termos RTT significativos e variáveis, mesmo sem retransmissões.**

- **Assim, as técnicas que se usam para compensar a perca de pacotes baseiam-se, geralmente, não em retransmitir, mas em tentar compensar as percas através de algum tipo de informação redundante (*FEC- Forward Error Correction*) ou de *interleaving* (que é uma outra técnica usada para distribuir o impacto das percas).**

Como compensar os pacotes perdidos

- É sempre possível tentar **compensar os pacotes perdidos no receptor**. Se o receptor detecta um pacote perdido pode repetir o último ou interpolar. Repetir é o mais fácil do ponto de vista computacional. Interpolar é mais pesado. Estas técnicas conseguem compensar as percas quando estas se mantêm dentro de limites aceitáveis (por exemplo inferiores a 1%).

- Em alternativa podem usar-se **técnicas especiais na emissão**.

- Uma delas consiste em introduzir **FEC (*Forward Error Correction*).** A ideia é enviar após cada N pacotes, **um pacote redundante calculado como sendo o XOR dos anteriores**. Desta forma é possível compensar até 1 pacote perdido de N em N pacotes.

- A outra forma consiste em introduzir um ***stream* redundante de qualidade inferior (*piggybacking*)**.

- Finalmente, é também possível **distribuir as repercussões das percas** através de uma técnica designada ***interleaving*.**

Resumo: soluções a nível aplicacional para problemas do nível rede

- **usar UDP** evitando assim os atrasos impostos pelo TCP devido à sua fiabilidade
- Cliente: ***playout delay* adaptativo**: para compensar o *jitter*
- Servidor: **adaptar a capacidade exigida pelo *stream*** à disponível entre o cliente e o servidor
- Compensar os erros (por cima do UDP, ao nível aplicacional)
  - FEC, *interleaving*, ou compensar os erros através de um *stream* redundante

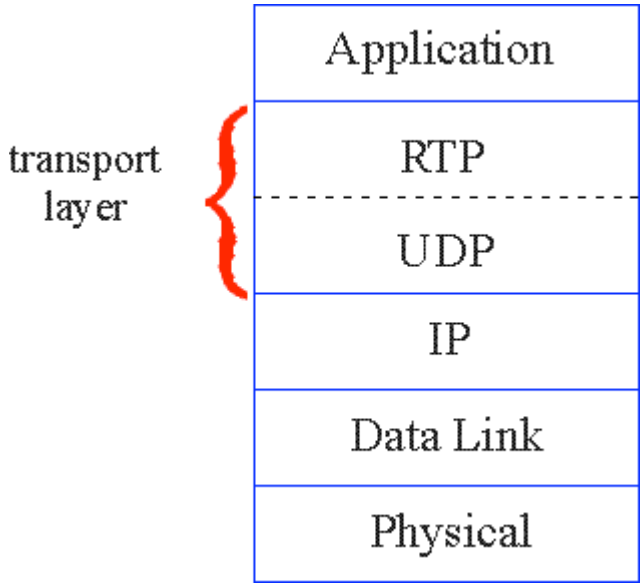
Real-Time Protocol (RTP)

- O RTP especifica a estrutura dos pacotes que contém áudio e vídeo
- RFC 1889.
- Providencia
  - Identificação do tipo do conteúdo
  - Números de sequência
  - Etiquetas temporais (*timestamps*)
- O RTP é um protocolo aplicacional só conhecido dos sistemas finais.
- Os pacotes ou datagramas RTP são transportados em segmentos UDP
- Interoperação: se duas aplicações de IP Phone distintas usam RTP, devem poder funcionar em conjunto

RTP funciona sobre o UDP

As *RTP libraries* providenciam uma interface que estende o UDP com:

- *payload type identification* (CODEC)
- *packet sequence numbering* (números de sequência)
- *time-stamping* (etiquetas temporais)



Cabeçalho RTP

RTP Header

Payload Type (7 bits): Indica o tipo de codificação (CODEC) usado. Se o emissor resolver alterá-lo, indica-o ao receptor mudando este campo.

Payload type 0: PCM mu-law, 64 kbps
Payload type 3, GSM, 13 kbps
Payload type 7, LPC, 2.4 kbps
Payload type 26, Motion JPEG
Payload type 31. H.261
Payload type 33, MPEG2 video

Sequence Number (16 bits): incrementado em cada pacote enviado o que permite detectar a perca ou troca dos pacotes.

Continuação

- **Timestamp field (32 bits).** Reflecte o momento em que os dados contidos no pacote foram gerados em termos do relógio usado para realizar a amostragem para a digitalização:
  - Com som, o *timestamp clock* é tipicamente incrementado de 1 por cada período de amostragem (em cada 125 micro segundos quando se faz amostragem a 8 KHz ou 8.000 vezes por segundo)
  - Se cada pacote contiver 160 amostras, o valor deste campo é incrementado de 160 em cada pacote. O valor deste relógio continua a ser incrementado mesmo que a fonte esteja inactiva e não emita pacotes.
- **SSRC field (32 bits).** Identifica a fonte do *stream*. Uma sessão RTP pode ter vários *streams* e cada um deve ter um valor de SSRC diferente.

SIP — Session Initiation Protocol

Visão de longo prazo

- Um dia todas as chamas telefónicas, os chats, os jogos e as vídeo conferências serão realizadas sobre a Internet

- Os interlocutores terão nomes ou endereços de correio electrónico ao invés de números de telefone

- É possível chegar à pessoa chamada mesmo que esta esteja fora da rede habitual, independentemente do dispositivo (terminal) que esteja a usar e do endereço IP do mesmo.

SIP - Session Initiation Protocol

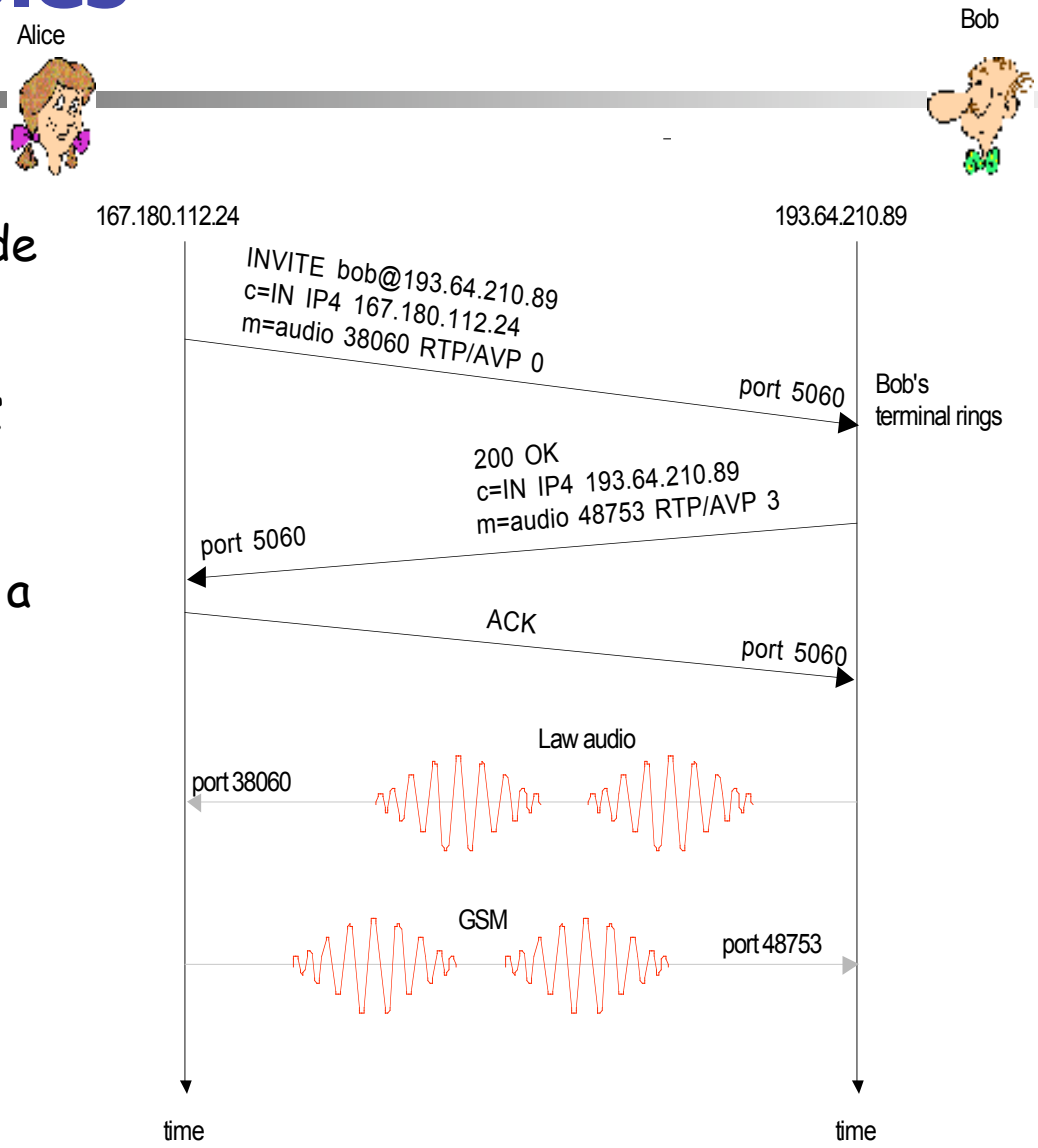
- **Este protocolo providencia mecanismos para um utilizador estabelecer uma sessão interactiva com outro ou outros utilizadores, negociar CODECS, terminar a sessão, etc.**
- **Uma sessão interactiva pode ser uma chamada telefónica, uma videoconferência ou uma simples sessão de "chat".**
- **Providencia mecanismos para determinar o endereço IP de cada participante. Este endereço pode mudar em função da localização e do dispositivo usado pelo participante.**
- **Os participantes são identificados por endereços SIP que são semelhantes a endereços de correio electrónico.**
- **Providencia mecanismos para a gestão da sessão nomeadamente, mudança dos fluxos usados e respectiva codificação, entrada de novos participantes, transferência das chamadas, colocação da chamada em espera, etc.**

Exemplo simples

- A Alice envia a mensagem de estabelecimento da sessão (INVITE) e indica o seu endereço IP, porta e CODEC (PCM).

- A mensagem do Bob aceita a sessão (200 OK) e indica o seu endereço IP, a porta e o CODEC (GSM)

- O protocolo SIP pode usar TCP ou UDP. A porta por defeito é 5060.



Outras hipóteses

- Negociação do CODEC:
  - Supondo que o Bob não tem o CODEC PCM pode responder com:
  - *606 Not Acceptable Reply and list of acceptable encoders.*
  - A Alice pode então repetir a mensagem INVITE com o CODEC adequado.
- Rejeição da chamada:
  - Códigos possíveis: *"busy," "gone," "payment required," "forbidden".*
- O conteúdo multimédia pode ser enviado por RTP sobre UDP ou por qualquer outro protocolo.

Outro exemplo mais completo

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
Content-Type: application/sdp
Content-Length: 885

c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

Notas:

- **Sintaxe das mensagens SMTP**
- **sdp = session description protocol**
- **Call-ID é único para cada chamada tal como as mensagens SMTP**

- Neste caso o endereço IP do Bob é desconhecido pelo que serão necessários servidores suplementares

- A Alice envia e recebe as mensagens a partir da porta SIP por defeito: 506

- A Alice especifica no cabeçalho via a opção "Via" que está a usar SIP 2.0 sobre UDP e que o seu endereço é 167.180.112.24

Localização dos utilizadores

- O utilizador que inicia a chamada só tem o nome ou o endereço e-mail do interlocutor.
- Necessita do endereço IP do terminal do interlocutor:
  - Este move-se
  - Este tem diferentes terminais com endereços diferentes (PC, PDA, telefone móvel pessoal, telefone do carro, telefone de casa, do escritório, ...)
- O resultado pode depender da:
  - hora
  - de quem chama (não quer receber chamadas do trabalho em casa)
  - do estado (disponibilidade para atender a chamada ou não)

Os servidores SIP:

- SIP *registrar server*
- SIP *proxy server*

Servidor *SIP Registrar*

- Quando Bob liga o seu terminal, este envia a mensagem SIP REGISTER para o seu servidor SIP (trata-se de uma função semelhante à do servidor de Instant Messaging).
- O SIP Registrar é semelhante ao servidor de DNS do domínio do utilizador.

Register Message:

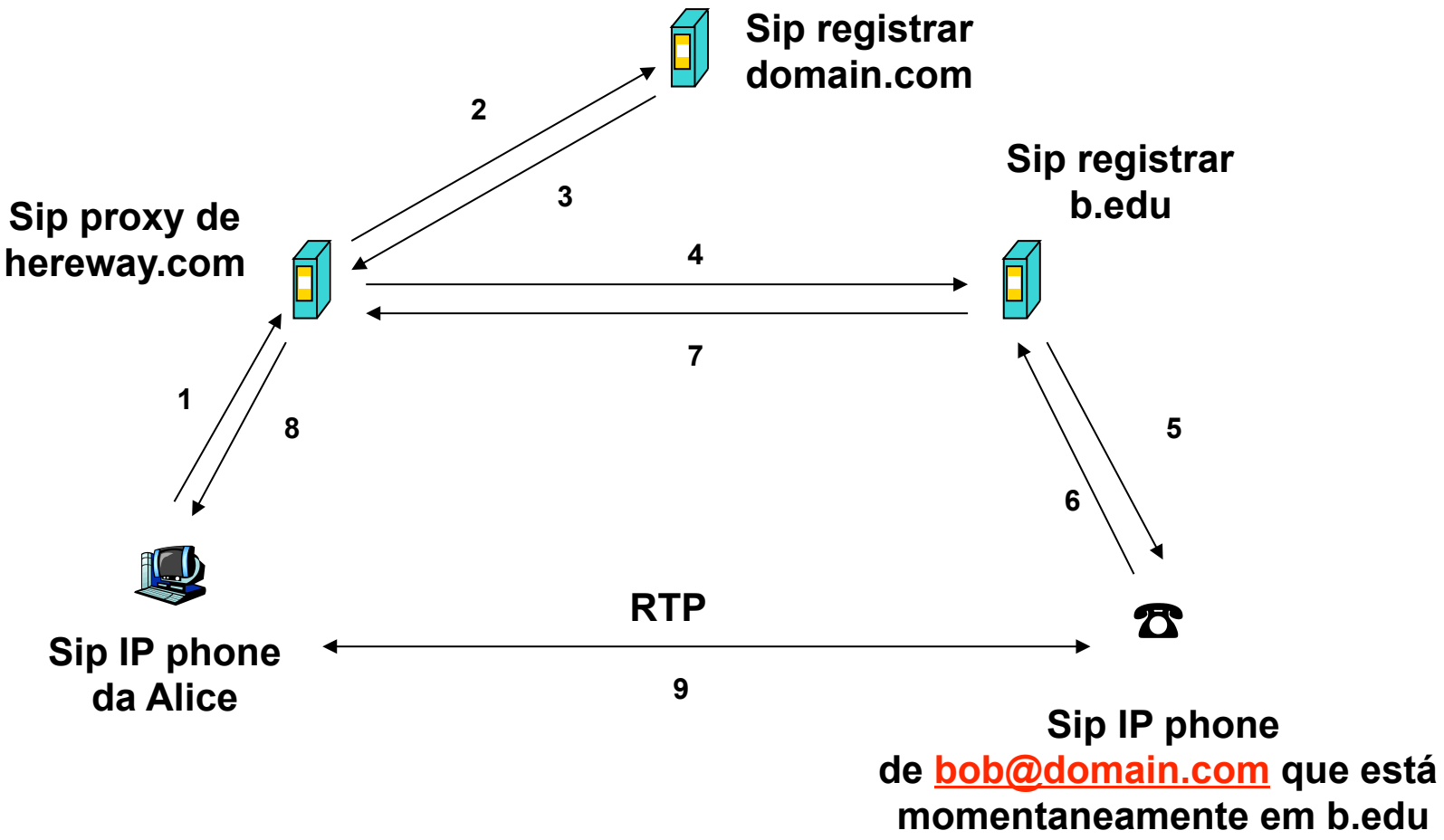
```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

SIP Proxy

- A Alice envia a mensagem INVITE para o seu servidor proxy
  - Contendo o endereço sip:bob@domain.com
- O Proxy é responsável por encaminhar as mensagens SIP para o utilizador chamado
  - Provavelmente através de diferentes proxies
- O chamado responde através da mesma cadeia de proxies
- O Proxy devolve a mensagem SIP de resposta à Alice
  - Contendo o endereço IP de Bob

- Nota: o proxy é análogo ao servidor local de DNS

SIP em acção


Sip registrar
domain.com
2
3
Sip registrar
b.edu
Sip proxy de
hereway.com
4
7
1
8
5
6
RTP
9
Sip IP phone
da Alice
Sip IP phone
de bob@domain.com que está
momentaneamente em b.edu