Applications – real time/non-real time

Outline

- Traditional applications
  - Electronic mail (SMTP, MIME)
  - File transfers (FTP)
  - World wide web (HTTP)
- Multimedia applications
  - Application requirements
  - Interactive multimedia
  - Streaming multimedia

Traditional applications and protocols

- Traditional = elastic data traffic, without timeliness requirements
  - real time traffic treated later...
- SMTP: Simple Mail Transfer Protocol
  - exchange of electronic mail between mail servers
- FTP: File Transfer Protocol
  - for downloading files between client/server
- HTTP: HyperText Transport Protocol
  - communication between Web browsers and Web servers
  - HTML specifies the form or the Web pages

Outline

- Traditional applications
  - Electronic mail (SMTP, MIME)
  - File transfers (FTP)
  - World wide web (HTTP)
- Multimedia applications
  - Application requirements
  - Interactive multimedia
  - Streaming multimedia

Electronic mail (SMTP, MIME)

- Mail service consists of
  - a mail reader,
  - a message transfer protocol (SMTP) and
    - SMTP = Simple Message Transfer Protocol
  - companion protocols RFC 822 & MIME

- Mail access protocol: retrieval from server
  - reader programs: Netscape Messenger, Outlook, Eudora, Mozzilla
  - POP3: Post Office Protocol (RFC 1939)
    - authorization (agent ⇔ server) and download
    - downloads mails to your own local host
  - IMAP: Internet Mail Access Protocol (RFC 1730)
    - more features (more complex)
    - manipulation of inbox and stored messages on server
  - HTTP: Hotmail , Yahoo! Mail, etc.

Electronic mail (cont)

- Message format:
  - RFC 822: message has two parts - a header and a body
    - both in ASCII text
  - MIME: extends RFC 822 so that message can contain all sorts of data
    - data still presented as ASCII text
  - ASCII format ⇒ human can pretend to be an smtp client

- Message header:
  - series of <CRLF>-terminated lines (carriage-return + line-feed)
  - separated from message body by blank line
  - each header line contains a Type and a Value separated by a colon
    - To: student@hut.fi
    - Subject: lecture notes

MIME

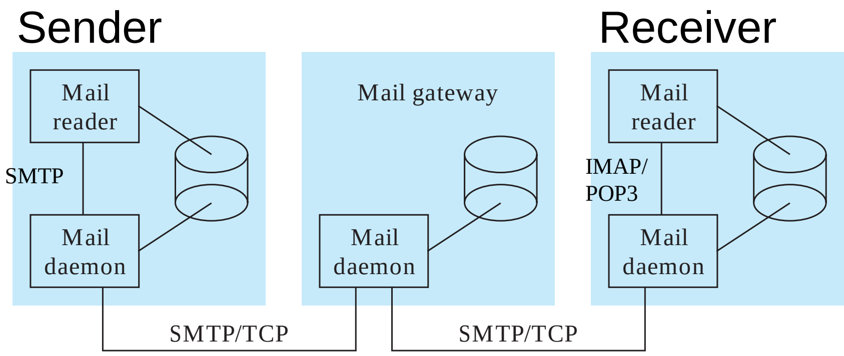
- Extends RFC 822 to allow email messages to carry audio, video, images, Word documents etc.
- Consists of 3 basic pieces
  - collection of header lines
    - extend the original set defined in RFC 822
    - ex. MIME-version, Content-Description, Content-Type, Content-Transfer-Encoding..
  - definitions for a set of content types
    - ex. image/gif, image/jpeg, text/plain, text/richtext, application/postscript, application/msword
  - a way to encode various data types so that they can be shipped in an ASCII mail message
    - base64 coding of binary data into ASCII: map every 3 bytes of the original binary data into 4 ASCII characters

MIME example (text + attached file)

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="XXXXboundary text"
From: John Doe <jdoe@microsoft.com>
To: Jane Doe <janedoe@nowhere.edu>
Subject: ...
Date: Tue, 04 Feb 2003 20:15:00 -0200

This is a multipart message in MIME format.

--XXXXboundary text
Content-Type: text/plain

this is the body text

--XXXXboundary text
Content-Type: text/plain; filename="test.txt"
Content-Disposition: attachment;

this is the attachment text

--XXXXboundary text--
```

Message transfer (SMTP)

- E-mail delivery
  - mail reader ⇒ message to mail daemon ⇒ daemon uses SMTP running over TCP to get message to a daemon in another machine ⇒ this daemon puts the message into user's mailbox
  - SMTP uses TCP on port 25
- Mail traverses many mail gateways that store and forward email msgs
  - mail gateway vs. IP router? IP routers store datagrams in memory and send them for a short period of time (fraction of seconds), mail gateways buffer messages on disk and try (re)sending for days or so



SMTP Example

- Exchange between sending host cs.princeton.edu and receiving host cisco.com (responses in italics)
  - you can be an SMTP client by first starting a TCP connection at a mail server in port 25, "telnet <servername> 25", then use the commands below

```
HELLO cs.princeton.edu
250 Hello daemon@mail.cs.princeton.edu [128.12.169.24]

MAIL FROM:<Bob@cs.princeton.edu>
250 OK

RCPT TO:<Alice@cisco.com>
250 OK

RCPT TO:<Tom@cisco.com>
550 No such user here

DATA
354 Start mail input; end with <CTRL>.<CTRL>
Blah blah blah …
<CTRL>.<CTRL>
250 OK

QUIT
221 Closing connection
```

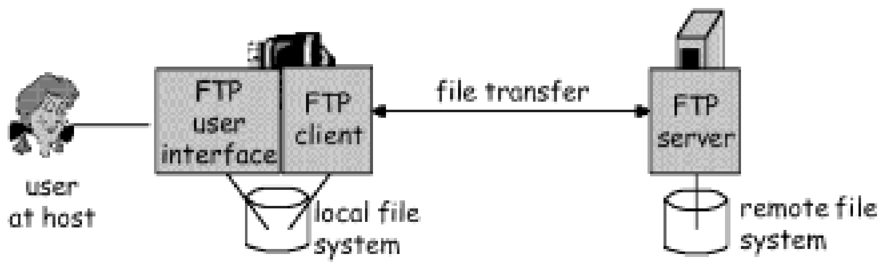
Outline

- Traditional applications
  - Electronic mail (SMTP, MIME)
  - File transfers (FTP)
  - World wide web (HTTP)
- Multimedia applications
  - Application requirements
  - Interactive multimedia
  - Streaming multimedia

FTP – File Transfer Protocol

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/

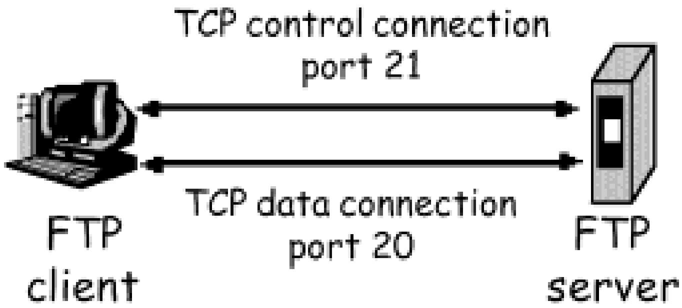
- Transfer file to/from remote host
  - RFC 959
- Client/server model
  - client: side that initiates transfer (either to/from remote)
  - server: remote host
- ftp server: port 21



FTP – separate control/data connections

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/

- ftp client contacts ftp server
  - at port 21, specifying TCP as transport protocol
- Two parallel TCP connections opened:
  - control: exchange commands, responses between client and server
    - "out of band control", port 21
  - data: file data to/from server
    - port 20
- ftp server maintains "state":
  - current directory
  - earlier authentication



Outline

- Traditional applications
  - Electronic mail (SMTP, MIME)
  - File transfers (FTP)
  - World wide web (HTTP)
- Multimedia applications
  - Application requirements
  - Interactive multimedia
  - Streaming multimedia

World Wide Web (HTTP)

- Web is a collection of cooperating clients and servers
  - everyone uses same protocol, HTTP
  - web browser used to open web pages
    - URL (Uniform Resource Locator) specifies location of object on the web (e.g., http://www.hut.fi/index.html)
  - opening a URL makes the browser open a TCP connection to port 80 to the given location, e.g., www.hut.fi, and the file index.html would be downloaded to your machine using HTTP over TCP
  - like SMTP, HTTP is a text oriented protocol

- Each HTTP message has the general form

```
START_LINE <CRLF>
MESSAGE_HEADER <CRLF>
<CRLF>
MESSAGE_BODY<CRLF>
```


HTTP, request message

- First line of HTTP message:
  - operation, Web page operation should be performed on, version of HTTP
  - for example, getting our laboratory's homepage manually

```
> telnet www.netlab.hut.fi 80

GET /index.html HTTP/1.1
Host: www.netlab.hut.fi
```

- Operations
  - OPTIONS request information about available options
  - GET retrieve document identified in URL
  - HEAD retrieve metainformation about document in URL
  - POST give information to server
  - PUT store document under specified URL
  - DELETE delete specified URL
  - TRACE loopback request message
  - CONNECT for use by proxies

HTTP, response message

- START_LINE: version of HTTP, 3-digit response code, text string giving reason for response
  – HTTP/1.1 202 Accepted
  – HTTP/1.1 404 Not Found

- Response message contains one or more MESSAGE_HEADER lines (additional information) and the requested page (HTML document, nontextual data encoded using MIME)

Code	Type	Example Reason
1xx	Info	request received, continuing process
2xx	Success	action successfully received, understood and accepted
3xx	Redirection	further action must be taken to complete the request
4xx	Client Error	request contains bad syntax or cannot be fulfilled
5xx	Server Error	server failed to fulfill an apparently valid request

HTTP and TCP connections

- HTTP version 1.0 made a separate TCP connection for each data item
  – waste of resources, especially when most items are small sized

- HTTP version 1.1 allows persistent connections: client and server can exchange multiple request/response messages over the same TCP connection
  – good:
    - eliminates the connection setup overhead
    - client can send multiple request messages -> TCP’s congestion window mechanism operates more efficiently (not necessary to do slow start for each request)
  – bad:
    - neither the client nor server knows how long to keep a particular TCP connection open (problem for servers with thousands of connections)
    - client and server must watch if the other side has elected to close the connection (recall, both sides need to close the TCP connection)

Caching

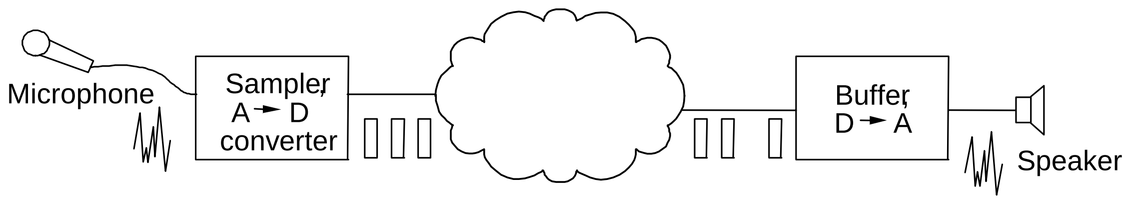
- WWW cache = web proxy
- Benefits:
  - pages from nearby cache can be displayed quickly
  - can reduce servers' load
- Implementation at several (hierarchical) layers:
  - in user's browser
  - user's site can support a single sitewide cache (takes advantage of pages previously downloaded by other users)
  - ISPs may have their own caches
- Cache needs to make sure it is not responding with an out-of-date version of the page
  - server may assign an expiration date (Expires header field) to each page
  - HTTP conditional requests by using "If-Modified-Since"-message header

Outline

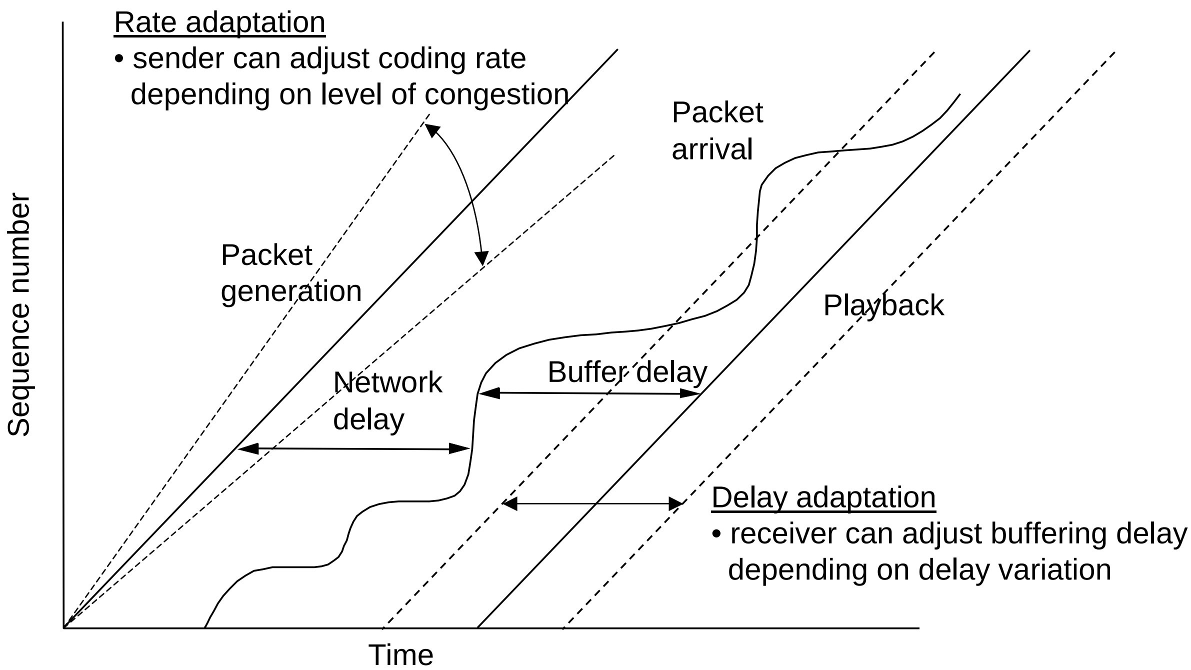
- Traditional applications
  - Electronic mail (SMTP, MIME)
  - File transfers (FTP)
  - World wide web (HTTP)
- Multimedia applications
  - Application requirements
  - Interactive multimedia
  - Streaming multimedia

Real time application

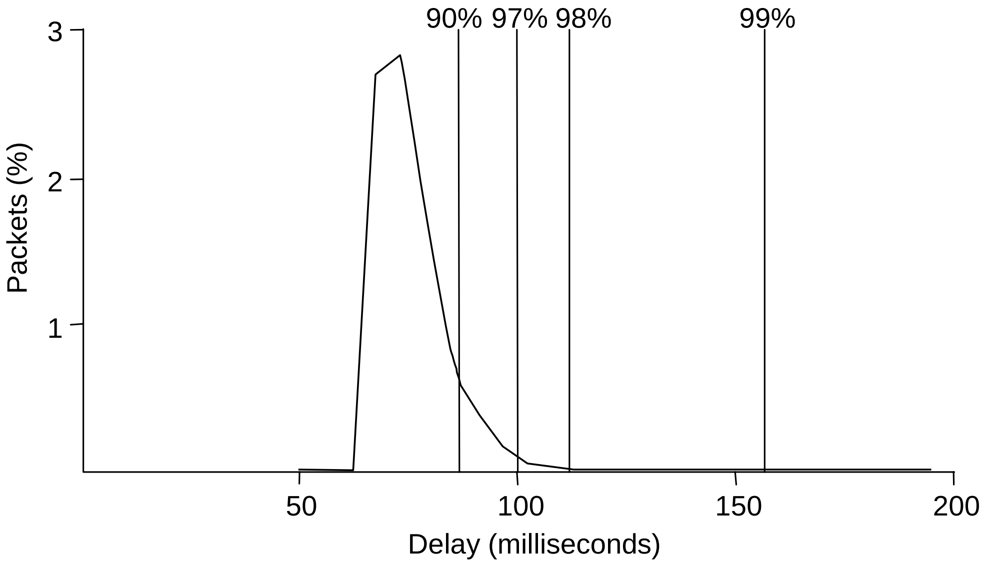
- Example application (audio)
  - sample voice once every 125us
  - each sample has a playback time
  - packets experience variable delay in network
  - playback time = point in time at which data is needed in the receiving host
    - data arriving after playback time is useless
  - playback point = constant offset added to playback time
    - delays vary in time, each packet can come with different delay
    - absorb variations by using a playback buffer
    - ok, as long as playback buffer does not drain



Playback buffer



Example distribution of delays



- One way delay of a path across Internet measured over a whole day
- If playback buffer delay (play back point) = 100 ms ⇒ 3% packets are lost
- Play back point of almost 200 ms needed to have all data arrive on time

Application performance issues

- Issues affecting real-time application performance
  - large delay prohibits conversation
  - variations in delay
    - can be smoothened by using buffers, but overall delay increases
    - resource reservation may help (note problems with reservations)
  - echo
    - needs to be removed when large delays
  - lost packets
    - replace by silence, extrapolation, or previous data
  - observing silence
    - can compress data more
  - address conversion: phone number ⇔ IP address
  - need control messages end-to-end
  - security
    - fire wall may add difficulties

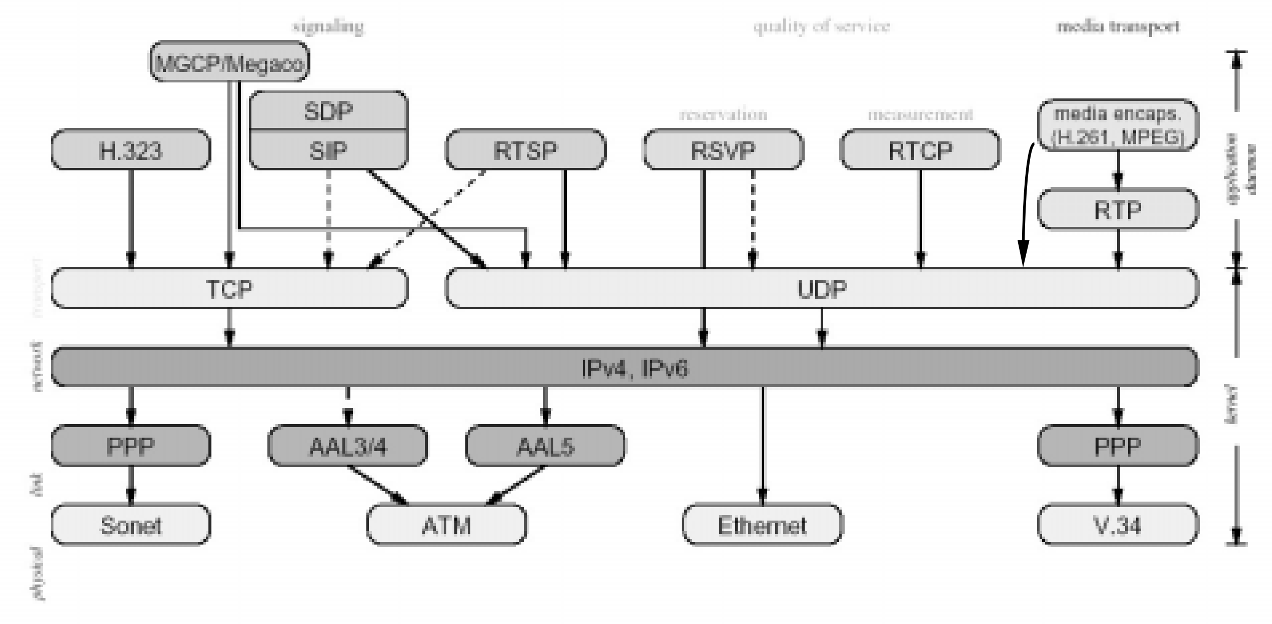
Interactive vs. streaming

- Requirement: deliver data in “timely” manner
  – reliability: 100% reliability not always required
- Interactive multimedia: short end-end delay
  – e.g., IP telephony, tele/videoconferencing, Internet games, virtual worlds
  – excessive delay impairs human interaction
  – end-end delay requirements:
    - video: < 150 msec acceptable
    - audio: < 150 msec good, < 400 msec OK
    - includes application-level (packetization) and network delays
- Streaming (non-interactive) multimedia: sensitive to delay variation
  – data must arrive in time for “smooth” playout (e.g., RealPlayer, WindowsMediaPlayer)
  – late arriving data introduces gaps in rendered audio/video
    - media stored at source
    - transmitted to client
    - streaming: client playout begins before all data has arrived
    - timing constraint for still-to-be transmitted data: in time for playout

Different protocols

- For streaming multimedia, need protocols for
  – data transport: RTP, application specific transport over UDP (or TCP => overkill, unnecessary retransmits)
  – stream control: RTSP
- For interactive multimedia, need protocols for
  – data transport: RTP, application specific over UDP
  – connection control: RTCP
  – session and call control: H.323 (SDP, SIP,…)

Protocol roadmap for Internet multimedia



H. Schulzrinne and J. Rosenberg, "Internet Telephony: Architecture and Protocols and IETF Perspective", available from http://www.cs.columbia.edu/~hgs/

S-38.188 - Computer Networks - Spring 2004

Outline

- Traditional applications
  - Electronic mail (SMTP, MIME)
  - File transfers (FTP)
  - World wide web (HTTP)
- Multimedia applications
  - Application requirements
  - Interactive multimedia
  - Streaming multimedia

Real-time Transport Protocol (RTP)

- RTP contains functionality that is specific to multimedia
  - development of RTP based on earlier work on “vat” conference application for MBone (Multicast backbone)
  - much of work on multimedia protocols for Internet based on MBone development
- Runs on top of transport-layer protocol (usually UDP)
  - called transport protocol as provides common end-to-end functions (difficulty in fitting to the strict protocol layer model)
  - pays attention to interaction between different applications (e.g., synchronization of audio and video streams)

RTP properties

- RTP allows similar applications to interoperate with each other
  - possible to communicate/negotiate encoding and compressing schemes
- Timing mechanism
  - RTP provides time stamping to enable the recipient of data stream to determine the timing relationship in data $\Rightarrow$ data can be played back correctly
- Synchronization
  - RTP supports synchronization of multiple media in a conference
- Congestion control
  - RTP gives indications of packet losses (some multimedia applications can adjust to congestion, for example change coding)
  - up to application to react to these
- Frame boundary indication
  - frame boundaries can signify, e.g., talk/silence periods, application can use these to its advantage
- “User friendly” identification of users (user@domain.com)
- Supports multicast

RTP and RTCP

- RTP DOES NOT
  - reserve resources
  - guarantee QoS
  - guarantee timely delivery
  - guarantee reliable delivery (if needed)
  - $\Rightarrow$ relies on lower-layer services to do so

Application
RTP
UDP
IP
Subnet

- RTP standard has too parts
  - RTP to exchange multimedia data (packet format)
    - fields for frame indications, sequence numbers, timestamps …
  - RTP Control Protocol (RTCP) to monitor QoS and convey information about the participants in an ongoing session

- RTP and RTCP do not address session control

RTP Control Protocol - RTCP

- Based on periodic transmission of control packets to all participants in the session, using the same distribution mechanism as the data packets

- Performs 4 functions
  - provides feedback on the quality of the data distribution
    - related to flow and congestion control functions of other transport protocols
    - rate must be controlled in order to scale up to large number of participants
  - carries CNAME, persistent transport-level identifier for an RTP source
    - keeps track on each participant
  - corrolation and synchronization of media streams
    - different streams may have different clocks
  - optional, minimal session control information

- RTCP is an outband control protocol
  - if RTP connection in UDP port N $\Rightarrow$ RTCP in port N+1

RTCP (cont)

- Different packet types
  - sender reports (active senders report transmission and reception statistics)
  - receiver reports (as above, used by those who don't send)
  - source descriptions (carry CNAME and other info)
  - application-specific control packets

- If a large session, control traffic may consume a lot of bandwidth
  - RTCP provides a method which tries to limit control traffic to 5%

Session Control and Call Control (H.323)

- Session control: how to make information available for holding a videoconference?
  - need to convay, for example, Time, Encoding, multicast IP address, sending data using RTP over UDP port number 4000
  - need protocols for session and call control
  - ITU-T umbrella standard for session and call control: H.323

- Protocols developed in IETF for this purpose
  - SDP (Session Description Protocol)
  - SAP (Session Announcement Protocol)
  - SIP (Session Initiation Protocol)
  - SCCP (Simple Conference Control Protocol)

- Compare:
  - want to announce a conference session (use SDP or SAP)
    - need to send information to a well-known address
  - make an internet phone call (use SDP and SIP)
    - locate users, announce desire to talk, negotiate encoding etc.

Session Description Protocol (SDP)

- For describing multimedia sessions to do session announcements, session invitations, other session initiations
- General purpose: does not support negotiation of session content or media encodings
- Contains: name and purpose of session, session time, media and address, required bandwidth, connection information
- Usage:
  - VoIP phone calls
  - multicast session description (session content)
    - SAP used for multicast announcements

Session Initiation Protocol (SIP)

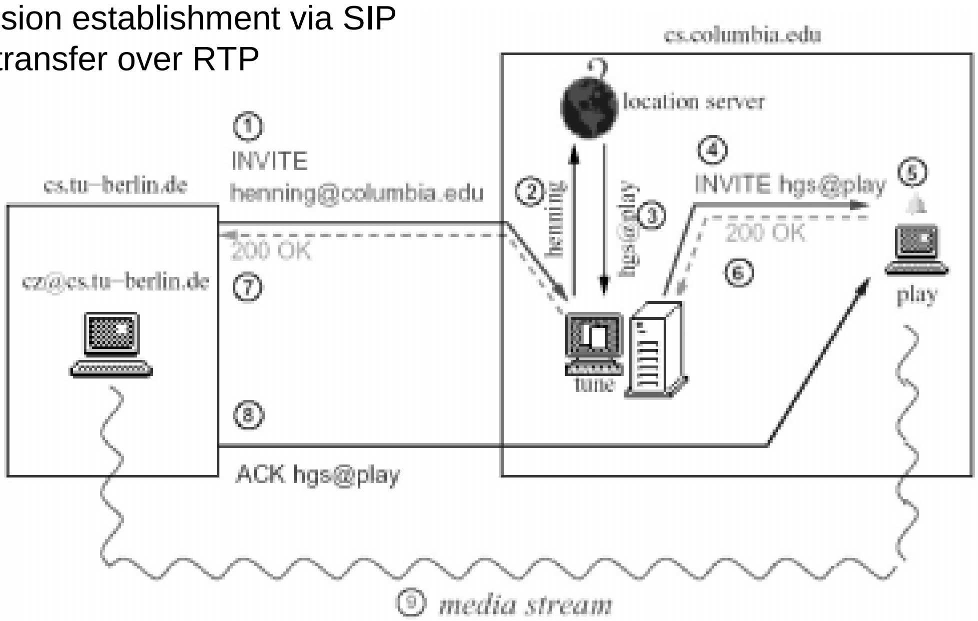
- Application-layer protocol that can establish, modify and terminate multimedia sessions (conferences) or Internet telephony calls
  - also media encodings are negotiated
  - Request-response protocol, requests sent by clients and received by servers
  - Can invite participants to unicast or multicast sessions, media and participants can be added to existing sessions
  - Supports name mapping and redirection services, enable personal mobility (personal number)
  - ASCII protocol, transport onn top of UDP
- Designed as a part of overall IETF multimedia data and control architecture (RSVP, RTP, RTSP, SAP, SDP)
  - functionality or operation does not depend on any these protocols
  - for example, any session description format can be used
  - in practice, SDP is used exclusively (SDP lists media types and the supported encodings)
- SIP does not
  - offer conference control services
  - allocate multicast addresses
  - reserve network resources (RSVP)

SIP methods

- INVITE : invites a user to a conference
- BYE : terminates a connection between two users in a conference
- OPTIONS : requests information about capability
- ACK : used for reliable message exchange
- REGISTER : conveys location info to a SIP server

VoIP example

1-8: session establishment via SIP
9: data transfer over RTP



H. Schulzrinne and J. Rosenberg, "Internet Telephony: Architecture and Protocols and IETF Perspective", available from http://www.cs.columbia.edu/~hgs/

VoIP example (cont)

- SIP INVITE message with embedded SDP session presentation
  - Alice calls Bob
  - Bob accepts audio but not video

```
INVITE sip:bob@macrosoft.com SIP/2.0
From: sip:alice@wonderland.com
To: sip:bob@macrosoft.com
Call-ID: 31415@wonderland.com
CSeq: 42 INVITE
Content-Type: application/sdp

v=0
o=user1 536 2337 IN IP4 h3.wonderland.com
c=IN IP4 h3.wonderland.com
m=audio 3456 RTP/AVP 0 1
m=video 4000 RTP/AVP 38 39
```

```
SIP/2.0 200 OK
From: sip:alice@wonderland.com
To: sip:bob@macrosoft.com
Call-ID: 31415@wonderland.com
CSeq: 42 INVITE
Content-Type: application/sdp

v=0
o=user1 535 687637 IN IP4 m.macrosoft.com
c=IN IP4 m.macrosoft.com
m=audio 1200 RTP/AVP 1
m=video 0 RTP/AVP
```

type
port nr.
transport
available encodings

embedded SDP descriptions

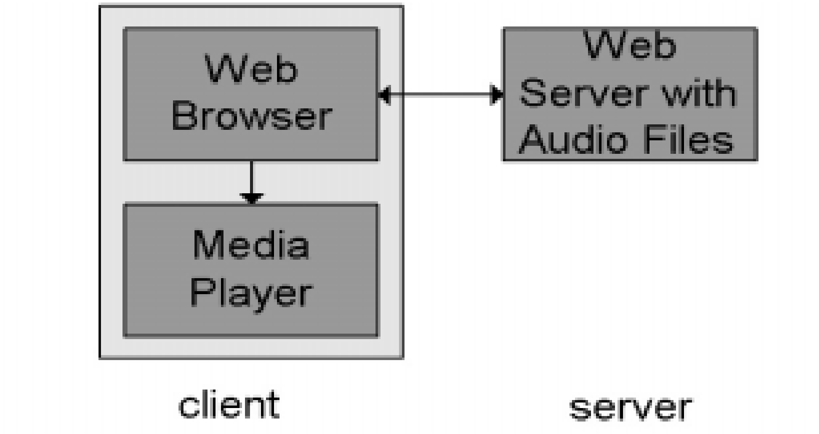
Outline

- Traditional applications
  - Electronic mail (SMTP, MIME)
  - File transfers (FTP)
  - World wide web (HTTP)
- Multimedia applications
  - Application requirements
  - Interactive multimedia
  - Streaming multimedia

Internet multimedia: simplest approach

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/

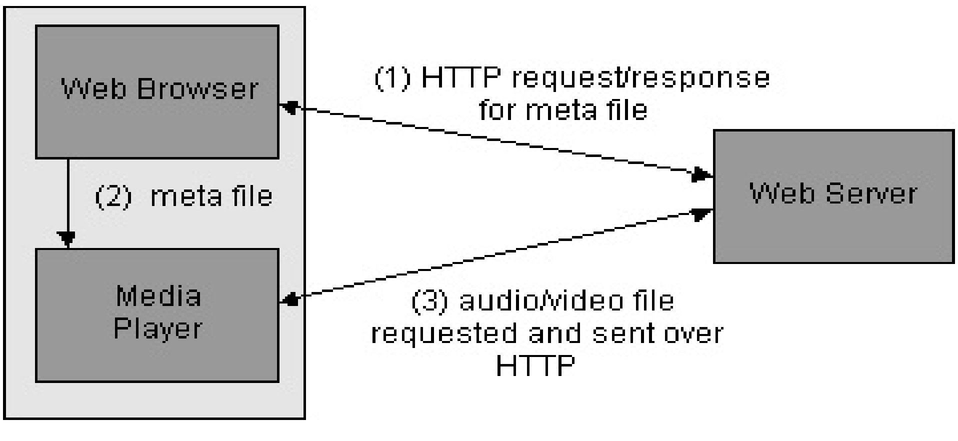
- Audio or video stored in file
- Files transferred as HTTP object
  - received in entirety at client
  - then passed to player
- Audio, video not streamed:
  - no “pipelining”, long delays until playout!



Internet multimedia: streaming approach

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/

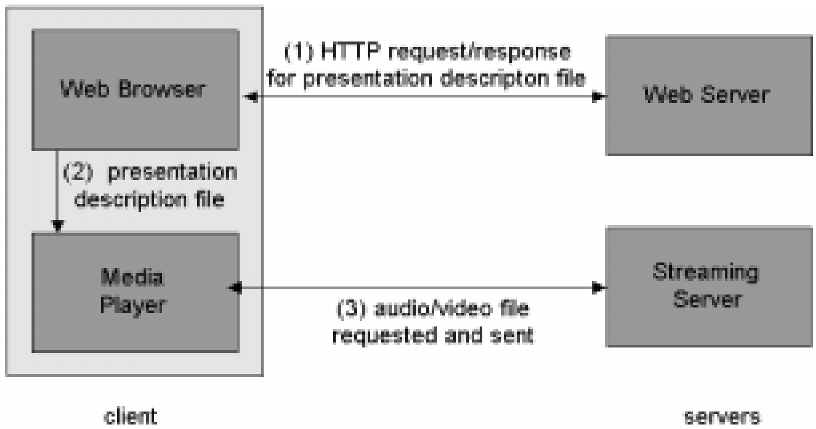
- Browser GETs **metafile**
  - metafile tells type of media (application) and URL of the actual media file
- Browser launches player, passing metafile
- Player contacts server
- Server streams audio/video to player
  - streaming by using HTTP over TCP ⇒ can not send stream control commands to server (rewind, stop, pause, …)



Streaming from a streaming server

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/

- This architecture allows for non-HTTP protocol between server and media player
  – can have a richer control protocol for media streams
- Can also use UDP instead of TCP for data transport



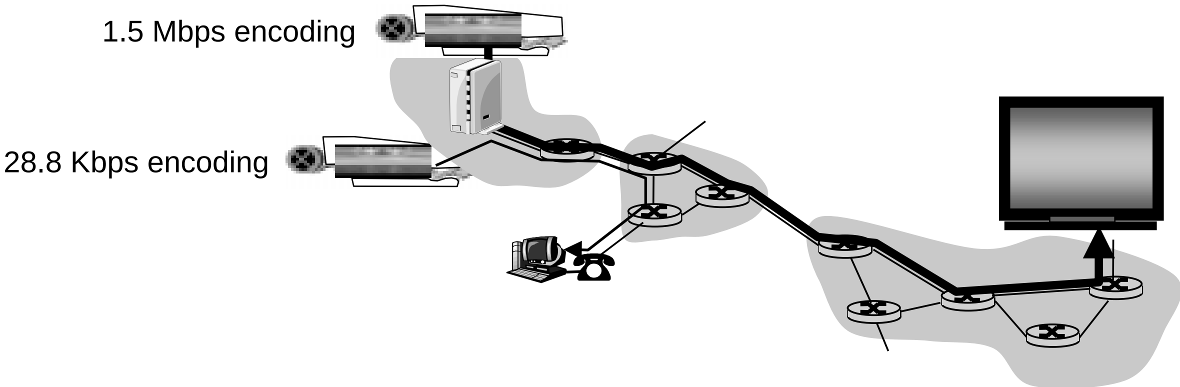
Streaming multimedia: UDP or TCP?

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/

- UDP
  – server sends at rate appropriate for client
    - with no regard to network congestion
  – short playout delay (2-5 seconds) to compensate for network delay jitter
  – error recover: time permitting
- TCP
  – send at maximum possible rate under TCP
  – congestion loss: retransmission, rate reductions
    - depending on playout delay, retransmissions may even be “too late” and thus unnecessary
  – needs larger playout delay to eliminate effects of sending rate fluctuations
- Experimental solution proposal: DCCP
  – Datagram Congestion Control Protocol (formerly DCP!)
  – TCP friendly congestion control without retransmissions

Streaming multimedia: client rates

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/



- Question: How to handle different client receive rate capabilities?
  - 28.8 Kbps dialup
  - 100Mbps Ethernet

- Answer: server stores, transmits multiple copies of video, encoded at different rates

User control of streaming multimedia

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/

- Real Time Streaming Protocol (RTSP): RFC 2326
  - user control: rewind, FF, pause, resume, etc…
  - out-of-band protocol:
    - one port (544) for control msgs
    - one port for media stream
  - TCP or UDP for control msg connection

- Scenario:
  - metafile communicated to web browser
  - browser launches player
  - player sets up an RTSP control connection, data connection to server

RTSP operation

[1]Jim Kurose, lecture notes for the course MPSCI 591E Computer Networking, http://www-net.cs.umass.edu/cmpsci_591_453/