A comprehensive multimedia control architecture for the Internet

Henning Schulzrinne
Dept. of Computer Science
Columbia University
schulzrinne@cs.columbia.edu

NOSSDAV'97
St. Louis, MO

May 19, 1997

Overview

- Internet multimedia architecture
- signaling for telephony and control: SIP
- stream control: RTSP
- integration

Functionality

Have standards for data (RTP, RSVP, IPv6, . . .), need control ➠

- phone call: ≥ 2 parties
- invitation to multi-party conference (announced via SAP/SDP)
- near media-on-demand via time-staggered multicast
- media-on-demand
- virtual presentations
- distributed digital editing
- caching, proxies and firewalls
- combining stored, live, interactive multimedia
- web integration ➠ start conferences, playback from web page

lots of demos, but no *coherent system*

Protocol Design

Internet binary: RTP, lower-layers ➠

- efficient if aligned, fewer ambiguities; nesting, options tedious
- simple applications are hard

ASN.1: SNMP, H.323 ➠

- not self-describing
- inefficient (BER) or parser needed (PER)

SMTP, HTTPish: query-response, headers, error codes, ldots ➠

- binary data not important
- extensions: PEP, JEPI, PICS, ...
- easy to parse & generate for Tcl, Perl, Python, ...
- overhead (space, time)? unidirectional?
- but ≠ HTTP: not object retrieval, state (RTSP), ...

Corba, DCOM: ➠ *potentially* replace *all* upper-layer Internet protocols

- cost of entry, maturity (security, extensions, multicast, ...)

Protocol Design

SIP and RTSP are not HTTP ➠

support UDP: no data stream, low latency desired

multicast: group signaling, user location

avoid HTTP mistakes: e.g.,

- relative request paths
- no extension mechanism
- 8859.1 coding

SIP: Session Initiation

Signaling Roles

- communications mode (phone, fax, mobile, pager, ...) $\surd$
- negotiation of media and their parameters within mode $\surd$
- resource reservation ➠ parameters from SIP
- human accessibility $\surd$

Ordering?

H.323

- H.323 = Q.931 (ISDN signaling) + H.245 (capabilities) over TCP
- very complex (200+ pages)
- no multicast signaling
- limited multicast conferences (➠ MCUs)
- no redirection
- but: capability negotiation (H.245)
- limited media instances (one video, audio stream)
- no media servers
- several TCP connections ➠ very long latency
- but: industry support…

Session Initiation Protocol (SIP)

- don't *need* to know IP address du jour of user's host
- invite to new and on-going conferences
- allow for manual and automatic forwarding
- *personal mobility* (complements data link/IP mobility) ➠ change of terminal (PC, digital cordless, palmtop), location
- call center: reach first (load distribution) or reach all (department conference)
- possibly change permissible encodings during conference
- typical latency: 1 RTT (with UDP)

Universal Communication Identifier

SIP uses email-like addresses:

email address: RFC 822 ➠ president@whitehouse.gov or Bill Clinton <president@whitehouse.gov> ➠ needs to be translated

- may use SMTP EXPN or VRFY, `.forward`
- send email as last resort

user@host: user at physical host

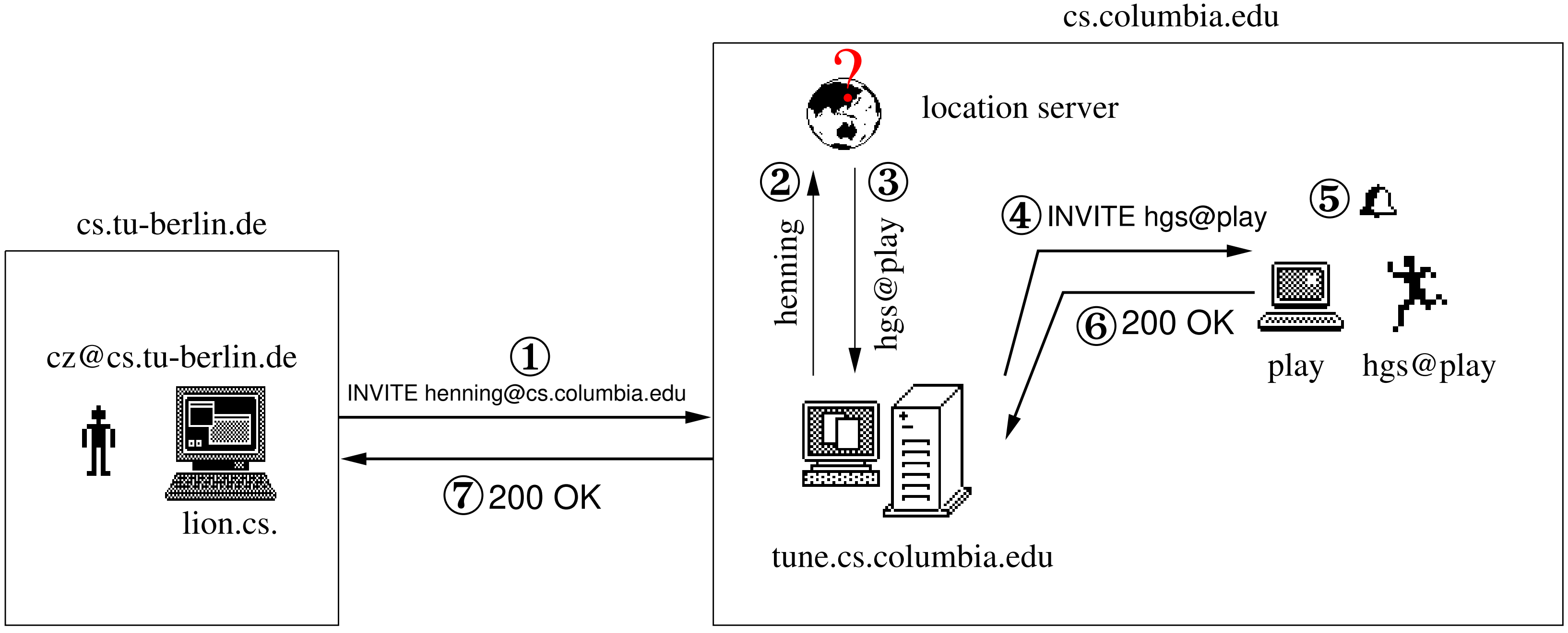
telephone number: E.164 ➠ +1.202.The.Pres or phone://+1.212.555.1212@gateway.net2phone.com or phone://+1.212.555.1212

Basic Operation

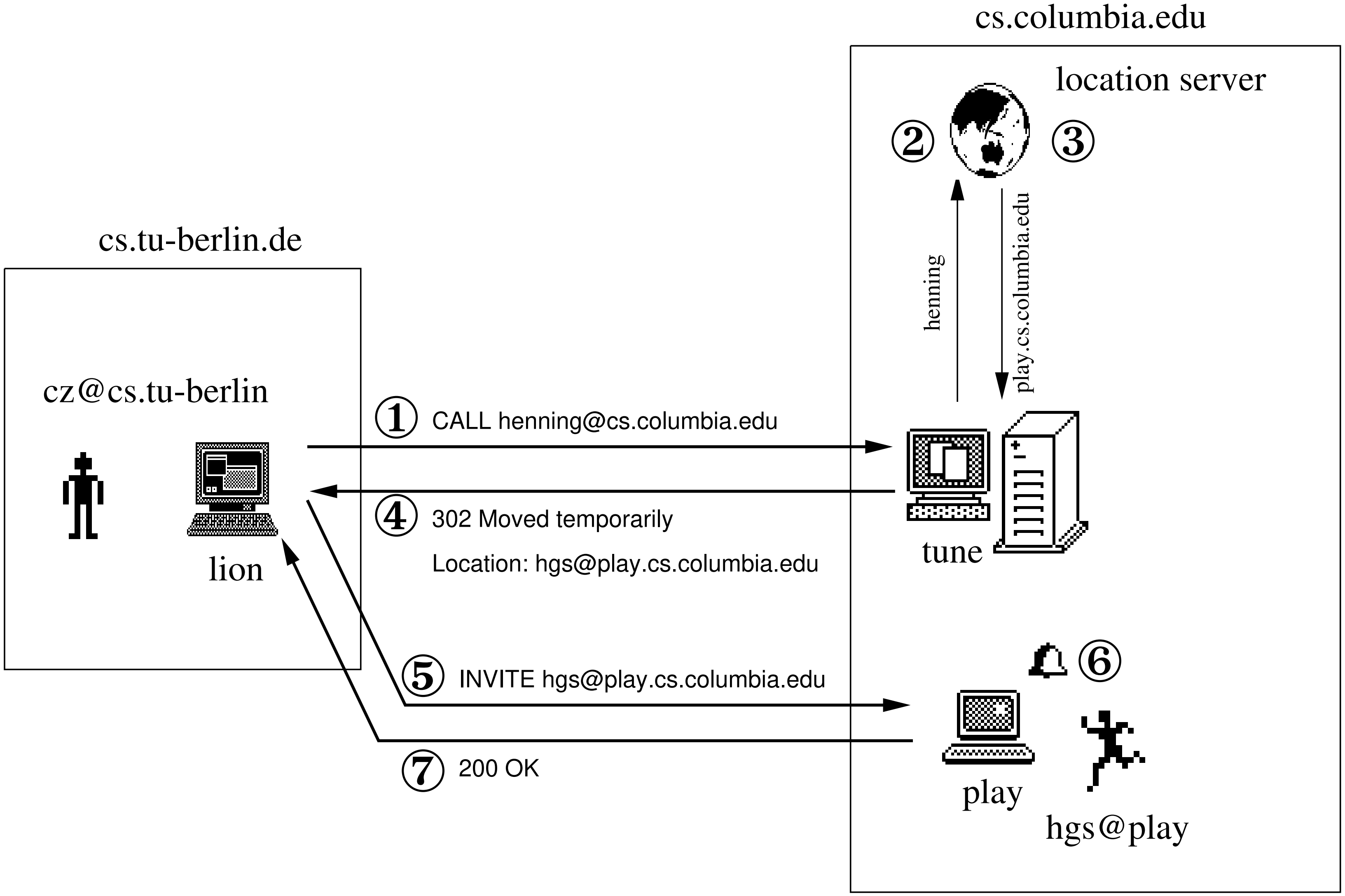
Example: call `schulzrinne@cs.columbia.edu`

- may use directory or user location service (e.g., LDAP)
- caller: `cs.columbia.edu` $\longrightarrow$ actual host (DNS SRV, MX)
- send protocol (SIP) request to server
- may ask for capabilities first, without "ringing"
- may request "variants": pager, fax, phone, ...
- server either ...
    1. accepts/rejects/forwards call, maybe after asking user
    2. bridges to PSTN
    3. locates user ➠ forwards call

Operation in Proxy Mode



Operation in Redirect Mode



Protocol Request

```
INVITE schulzrinne@cs.columbia.edu SIP/2.0
From: Christian Zahl <cz@cs.tu-berlin.de>
To: Henning Schulzrinne <schulzrinne@cs.columbia.edu>
Via: SIP/2.0/UDP 131.215.131.131, SIP/2.0 foo.com
Session: 199612061103.AA1528@cloud9.cs.tu-berlin.de
Content-Type: application/sdp
Content-Length: 187
Subject: New error codes
```

session description

Response

- accept or reject call, manually or automatically
- redirect temporarily or permanently

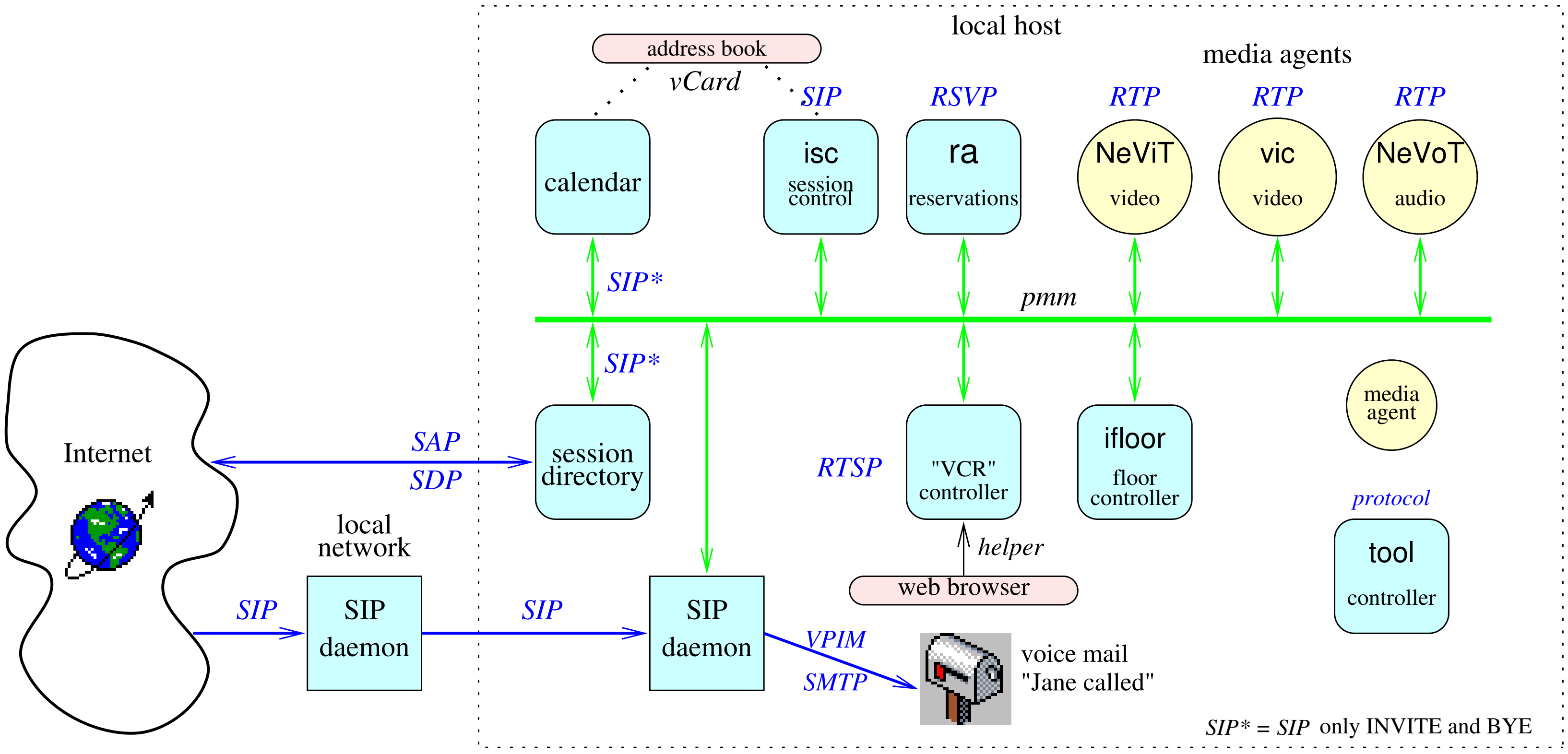
Redirection ➠ indicate several possible locations

```
SIP/2.0 100 Trying to find user

SIP/2.0 150 Ringing

SIP/2.0 302 Callee has moved temporarily
Location: sip://jones@salt.lab3.company.com
Location: sip://jones@pepper.lab3.company.com
Location: phone://1.212.939.7042
```

In-Call Operation

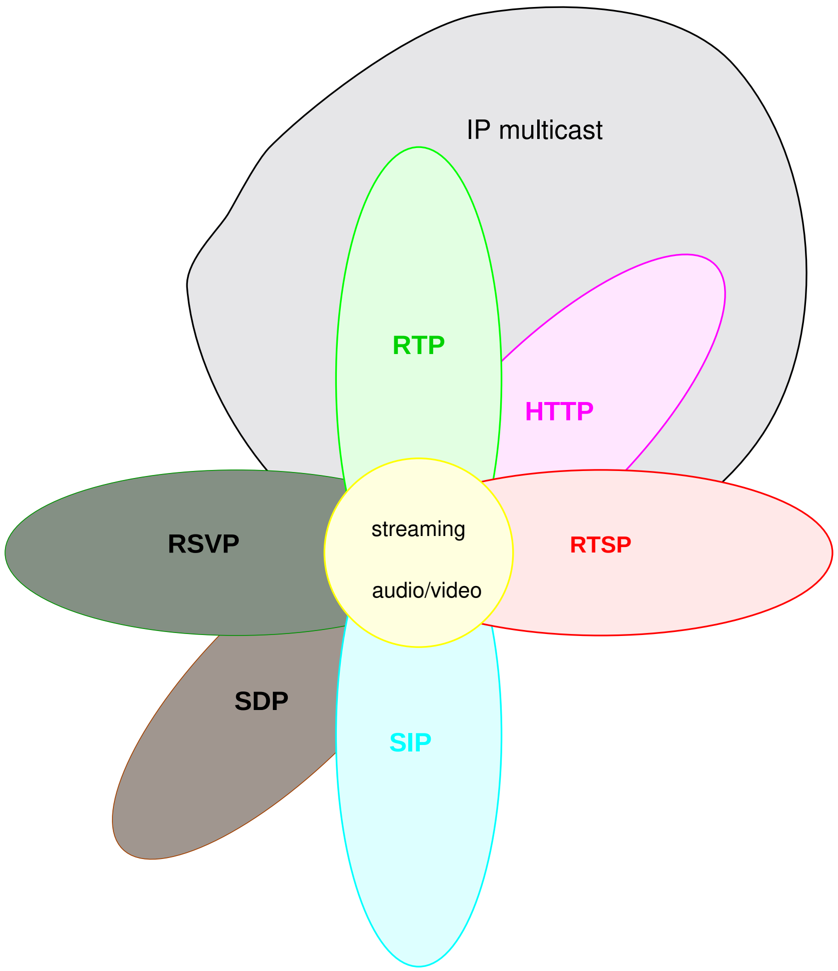
- easy for two-party calls: BYE, BYE with redirect ➠ all basic PBX functionality
- doesn't quite work in multi-party call ➠ may need conference control protocol?
- also missing: floor control, voting, application sharing, ...

Implementation



RTSP – Real-Time Stream Protocol

Streaming Multimedia

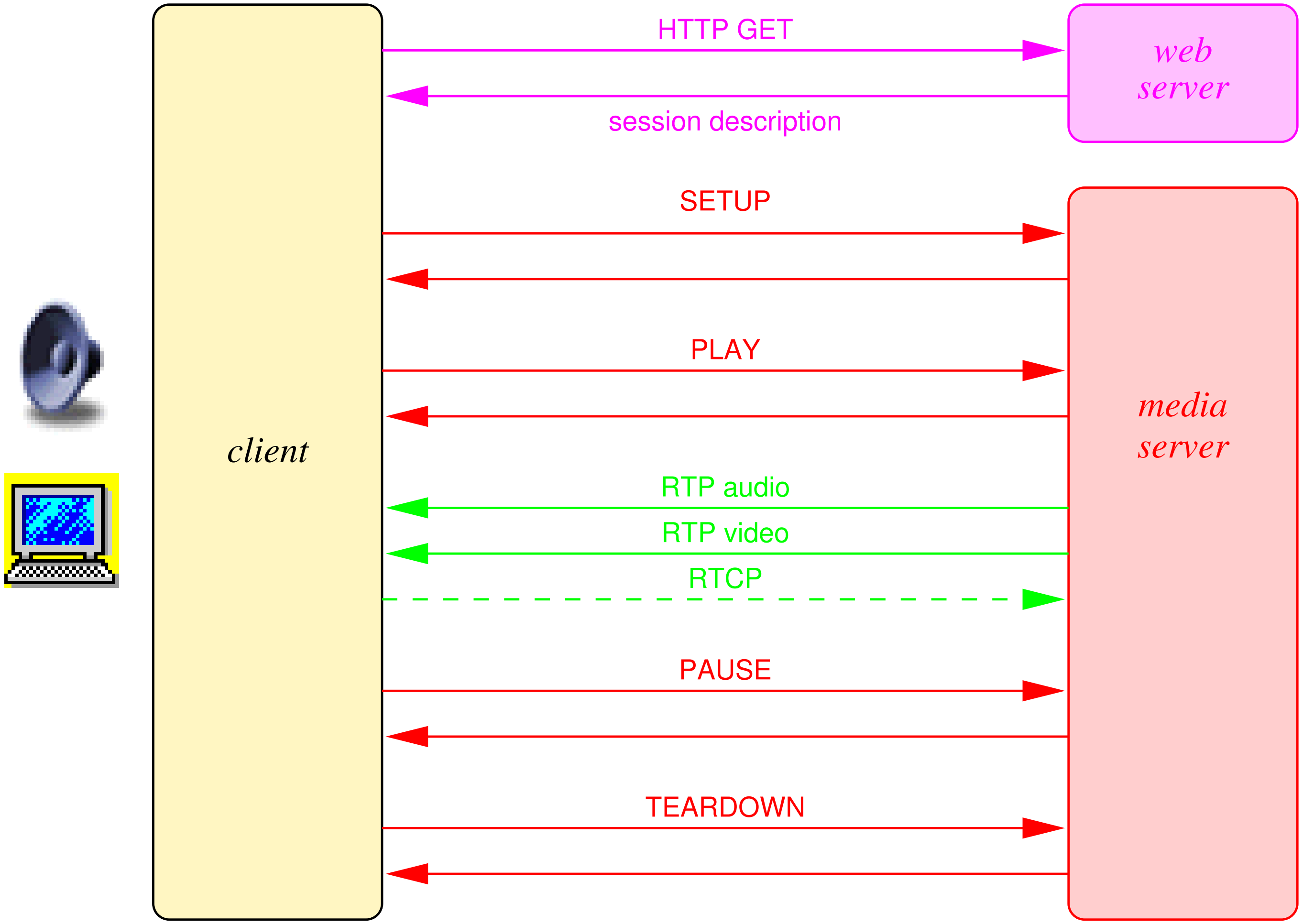


(similar functionality, much simpler than DSM-CC)

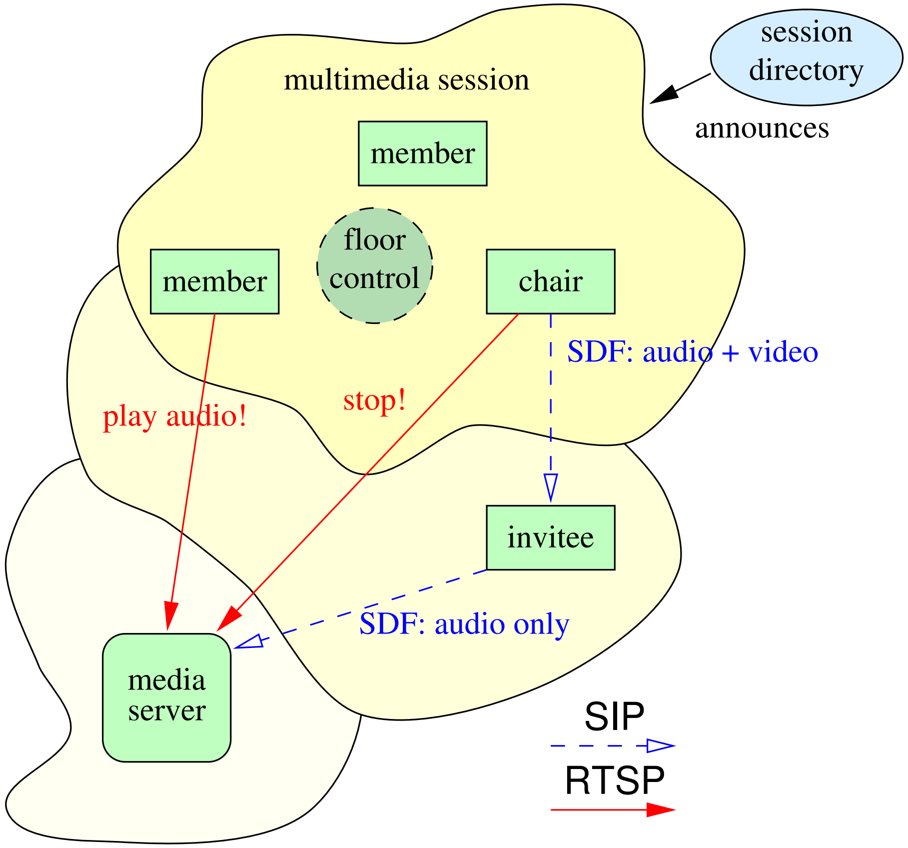
RTSP

- similar design as SIP (TCP + UDP, HTTP, . . .)
- "passing around the remote" ➠
  - TCP connection ≠ RTSP association
  - connection/stream, connection/command, UDP
  - session by identifier
- virtual presentations = synchronized playback from several servers ➠ RTP SRs + command timing
- load balancing using redirection at connect, during stream
- supports any session description
- remote digital editing ➠ queued play lists, recording
- device control ➠ camera pan, zoom, tilt
- caching: similar to HTTP, except "cut-through"

RTSP Operation



SIP and RTSP Integration



- provide transport parameters to RTSP explicitly
- H.323 needs introductions ➠ conference identifier

Summary

- SIP, RTSP fill in missing details of Internet conferencing architecture
- RTSP: widespread deployment (Netscape, RealAudio) before end of 1997
- SIP: Mbone tools, H.323 introducer?

Components

- applications are easy, embedded tools and meta-tools hard
- A/V interfaces ➠ every OS *release* for each hardware differs:
  - mixing
  - out-of-band control (volume, ports, VU meter, ...)
  - read/write on same descriptor
  - full/half duplex (timing!)
  - sampling rates, formats for read/write
  - encoding hardware support
  - blocking read through event handler
  - ➠ libraries (AF, NCD, MS, etc.) generally a disaster
  - X-model doesn't work ➠ local $\neq$ remote
- threads (NT) vs. Unix event model
- interconnecting applications ➠ vat, PMM, DCOM

"Foundation Classes"

needed for almost every application:

- different event models (Tk, Motif, daemon, ...) ➠ abstract event library
- multi-timer library
- encoding conversion
- misc. RTP protocol items (time-outs, user list management, collision detection, ...) with call-backs

What can be done?

- simple API for encoders, decoders on Unix
- uniform *simple* audio API: file descriptors, blocking or non-blocking, mixing by OS, flexible block size
- need small pieces, not whole framework
- module library: RTP, SIP, RTSP, audio/video functions
- licensing for G.723.1, G.729, GSM, any interesting A/V codec
- Java: by fiat