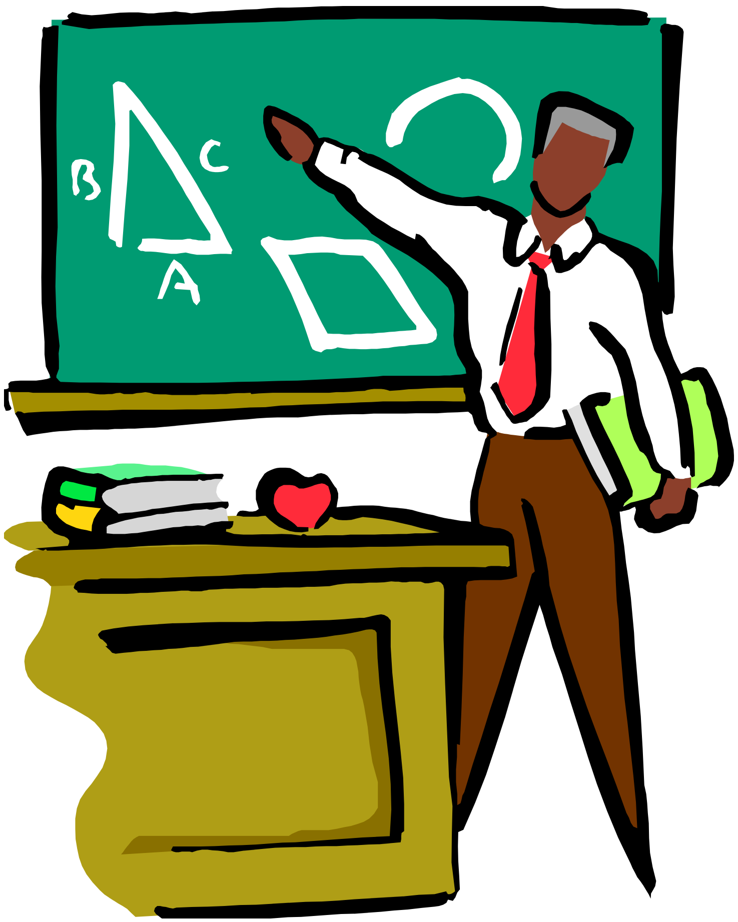
Chapter V III
Media Handling and Interworking Technologies

Hechmi KHLIFI/ Roch H. Glitho

Media handling …



1. Introduction
2. Media generation and transportation
3. The programmer’s viewpoint

Introduction …

Signaling vs Media handling

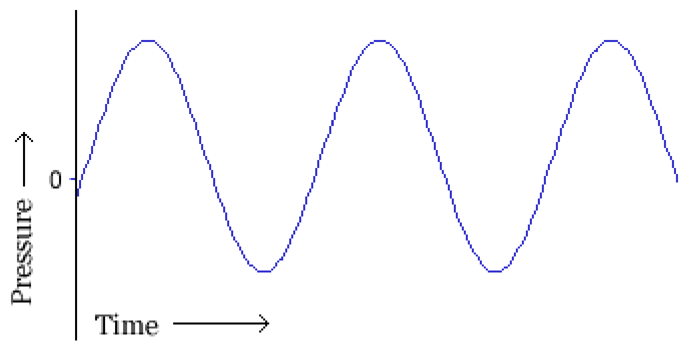
- Session establishment, tear down .. Vs actual media

Media handling …

- Encoding / Decoding
- Transportation …
- Multiplexing
- Mixing

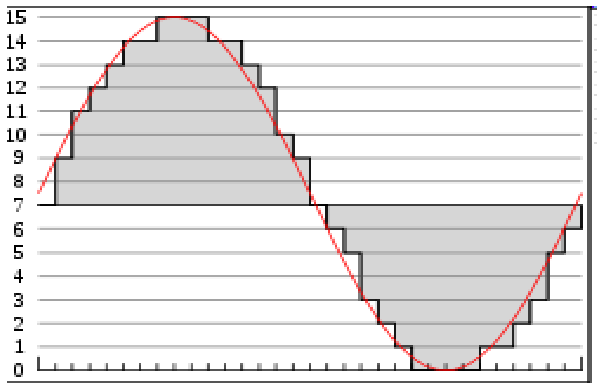
Sound

- **What is sound ?**
  - **Sounds are pressure waves of air.**
  - **We hear sounds because our ears are sensitive to these pressure waves.**
- How does microphone works ?
  - A microphone consists of a small membrane that is free to vibrate, along with a mechanism that translates movements of the membrane into electrical signals.
  - Acoustical waves are translated into electrical waves by the microphone. Typically, higher pressure corresponds to higher voltage, and vice versa.



From Analog to Digital

- **Digital:**
  - Sequence of 0 and 1.
  - Easy to transport, manipulate.
  - Good quality. Easy to reconstruct.
- Sampling and quantization



- Nyquist Sampling theorem

  The sampling frequency should be at least twice the highest frequency contained in the signal.

From Analog to Digital

- **Application of Nyquist theorem:**
  - Human voice frequency 300 - 3400 Hz
  - Sampling frequency 8000 Hz
  - One sample each 1/8000 sec =125 microsecond.
  - 8000 samples / sec
- If we take a sample of human voice each 125 microseconds and transmit it to the other side we will be able to reconstruct human voice.

Codec/ decodec

- **Multiple codec/ decodec**
- Waveforms codec.
- Predictive codecs.
- G.711 most popular.
  - Each sample is encoded on 1 byte (8 bits)
  - Bandwidth = 8000 *8 = 64 kbit/s
-

Media transportation …

Requirements:

- **Small delay: One way trip should not exceed 150 ms.**
- **Packets should be ordered to reconstruct stream.**

UDP or TCP?

- **TCP reliable : sequence, retransmission,… ➔ delay**
- **UDP no sequence, no retransmission**

Solution:

- **RTP: UDP + sequence information**
- **Feedback reporting:** RTP Control Protocol (RTCP)

Actual transportation:

Real-time Transport Protocol (RTP)

- Control of transportation:
  RTP Control Protocol (RTCP)

RTP concepts …

Session

- Logical association between parties communicating with RTP
  - Identified for each participant by:
    - IP address (may be common for all participants)
    - RTP port
    - RTCP port

End system

- Application that generates the content to be sent and/or
- receive the content to be consumed
- Examples: IP phones, PCs, microphones …

RTP concepts …

Mixers / translators

- Intermediate systems
- Connect 2 or more transport level clouds
  - End systems
  - Mixers / translators
- Use cases
  - Centralized conference bridges
  - Heterogeneous conferences
    - Low speed connection
    - High speed connection
    - Different encoding schemes
  - Some participants behind firewalls

RTP concepts …

Synchronization source (SSRC)

- Grouping of data sources for playing back purpose (e.g. voice vs. video)
- An end system can act as several synchronization sources (e.g. IP phone with video capabilities)
- Translators forward RTP packets with their synchronization source intact

Contributing source (CSRC)

- A source of a stream of RTP packets that has contributed to the combined stream produced by an RTP mixer
- Mixers insert the list of contributing sources in the packets they generate

.

RTP packets: Structure

Header

- Fixed
- Maybe followed by one header extension if extension bit is set

Payload

- Encoded stream samples

.

RTP packets: Structure

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

P: Padding
X: Header Extension
CC: CSRC count
M: Marker of record boundary

PT: Payload type; mapping can be specified by profile of the application

Sequence number: for each packet can be used by the receiver to detect loss or restore sequence.

Timestamp: The timestamp reflects the sampling instant of the first octet in the RTP data packet.

Demo: Wireshark traces

- Understand timestamp.
- Samples in a packet.
- Review sampling time.
- Reconstruct audio stream.

RTCP concepts …

Monitor:

- Application that receives RTCP packets sent by participants in an RTP session

Reports

- Reception quality feedback:
  - Lost packets
  - NTP Timestamp (synchronization)
  - Jitter
- Sent by RTP packets receivers (which may also be senders)

.

RTCP packets …

Examples of packets

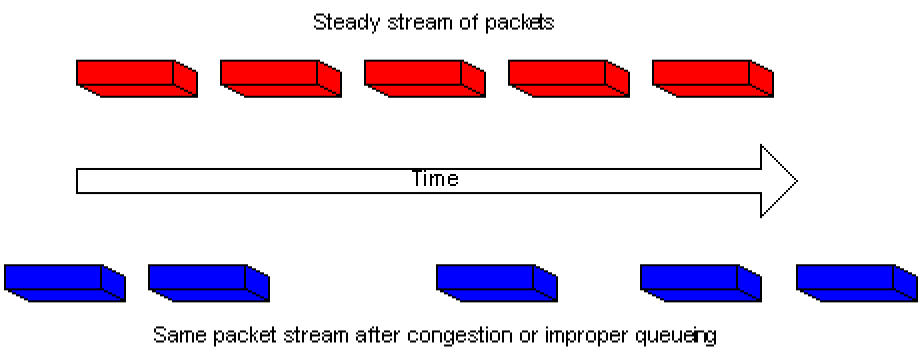
- Sender reports
- Receiver reports
- Bye

RTCP packets …

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
header |V=2|P|    RC   |   PT=SR=200   |             length            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         SSRC of sender                        |
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
sender |              NTP timestamp, most significant word             |
info   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |             NTP timestamp, least significant word             |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         RTP timestamp                         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                     sender's packet count                     |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                      sender's octet count                     |
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
report |                 SSRC_1 (SSRC of first source)                 |
block  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  1    | fraction lost |       cumulative number of packets lost       |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |           extended highest sequence number received           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                      interarrival jitter                      |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         last SR (LSR)                         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                   delay since last SR (DLSR)                  |
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
report |                 SSRC_2 (SSRC of second source)                |
block  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  2    :                               ...                             :
       +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
       |                  profile-specific extensions                  |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

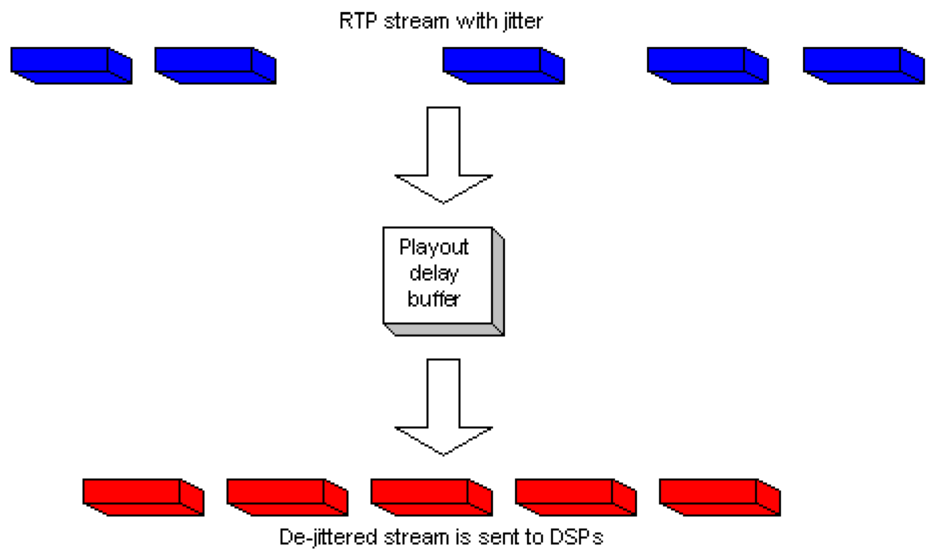
Jitter

- Jitter is defined as a variation in the delay of received packets.
- At the sending side, packets are sent in a continuous stream with the packets spaced evenly apart.
- Due to network congestion, improper queuing, or configuration errors, this steady stream can become lumpy, or the delay between each packet can vary instead of remaining constant.



De-Jitter Buffer

- The playout delay buffer must buffer these packets and then play them out in a steady stream to the digital signal processors (DSPs) to be converted back to an analog audio stream.



Adaptive Dejitter Buffering

- Fixed length
  - Store from 30 ms to 50 ms before start playing
- Adaptive
  - Recompute buffer size for each new Talkspurt (when the user is talking).
  - Playout delay adjustments made during periods of silence are less likely to be perceived

by users.

  - The playout delay is adjusted on a per-talkspurt basis by stretching or
  - compressing the silence between talkspurts.
  - The basic playout approach is to estimate
  - the end-to-end delay and use it to set the playout time of the first packet in a talkspurt.
  - Subsequent packets in the talkspurt will have the same total end-to-end delay.

Programmers viewpoint …

Standard APIs

- Ease application development by offering “high level” programmatic interfaces to protocols
- Enable the development of portable applications
- An example for media handling API
  - Java Media Framework (JMF)

.

Programmers viewpoint …

JMF key design goals

- Be easy to use
- Support capturing media data
- Enable the development of media streaming and conferencing applications in Java
- Enable customized solutions based on the existing API (e.g. higher level API)
- Provide access to raw media data
- Enable the development of customized downloadable de-multiplexers, mixers/translators and so on …

Programmers viewpoint ...

JMF RTP/RTCP APIs key design goals

- Be easy to use
- Support media data reception and transmission using RTP/RTCP
- Enable the development of media streaming and conferencing applications in Java

Programmers viewpoint …

JMF high level architecture

Media handling applications written in Java
Presentation APIs (e.g. start/stop) and processing APIs (e.g. encoding/decoding)
Plug-In APIs (e.g. interactions with codecs, multiplexers/de-multiplexers)

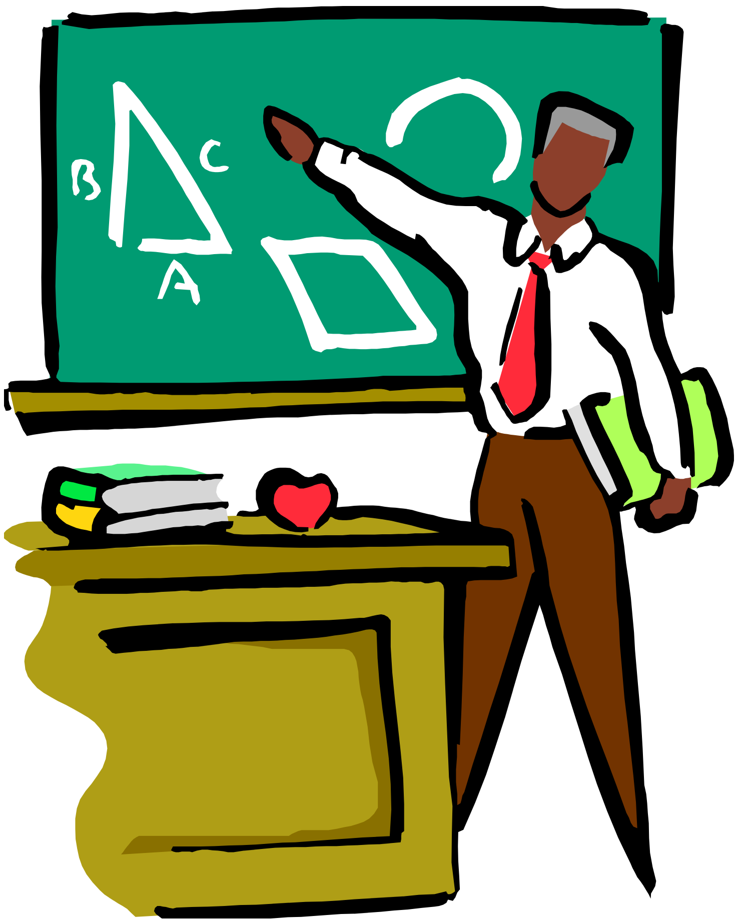
Programmers viewpoint …

JMF RTP/RTCP APIs key design goals

- Be easy to use
- Support media data reception and transmission using RTP/RTCP
- Enable the development of media streaming and conferencing applications in Java

Hechmi KHLIFI/ Roch H. Glitho

Megaco / H.248



1. Introduction
2. Genesis
3. Concepts
4. Protocol
5. Call cases

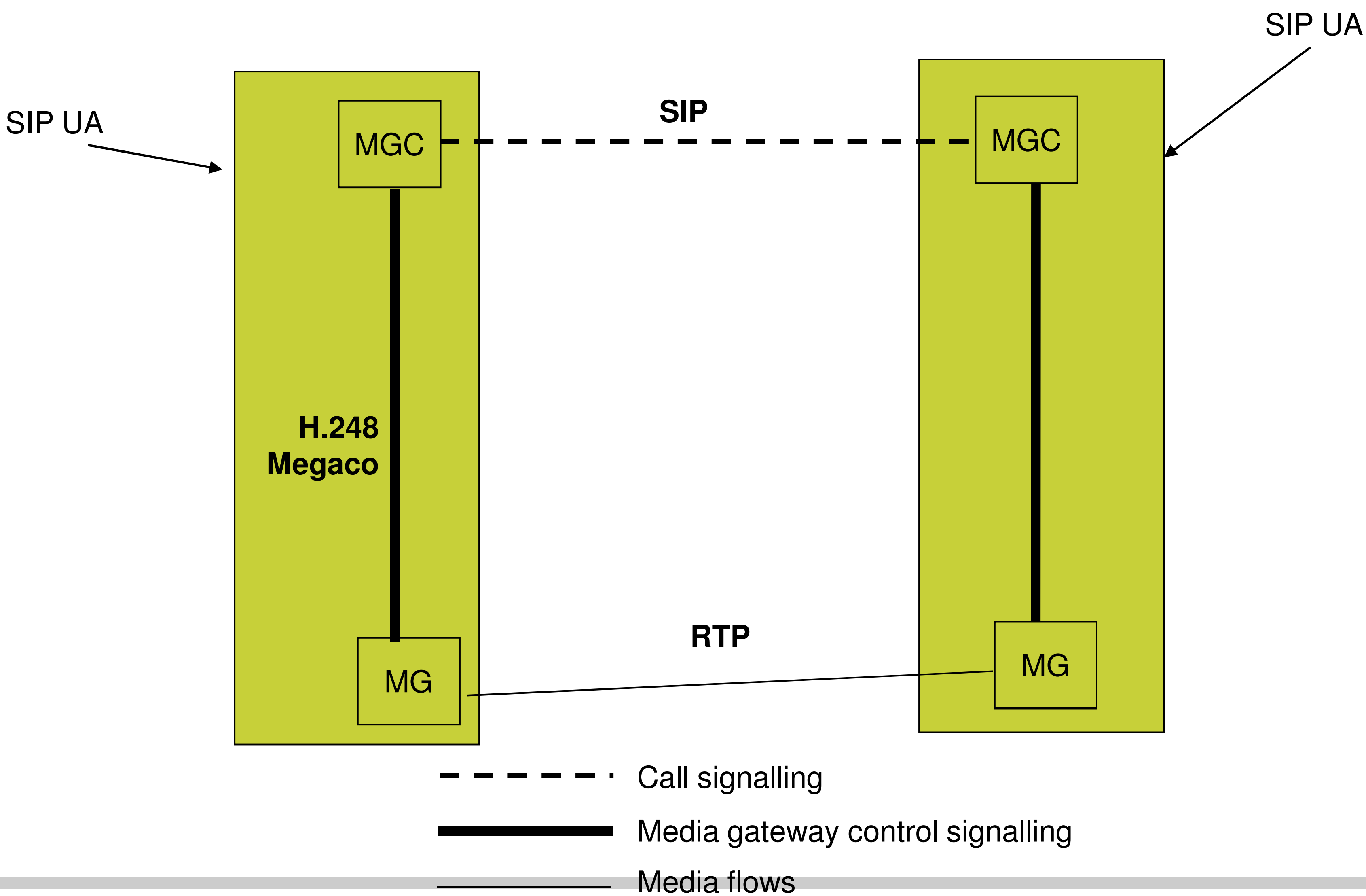
Megaco/H.248: Introduction

MEdia GAteway COntrol Protocol [RFC3015]

- Media Gateway control protocol
- H.248 is ITU-T reference for the same protocol
- Protocol for controlling telephony gateway and terminals (IP Phones)

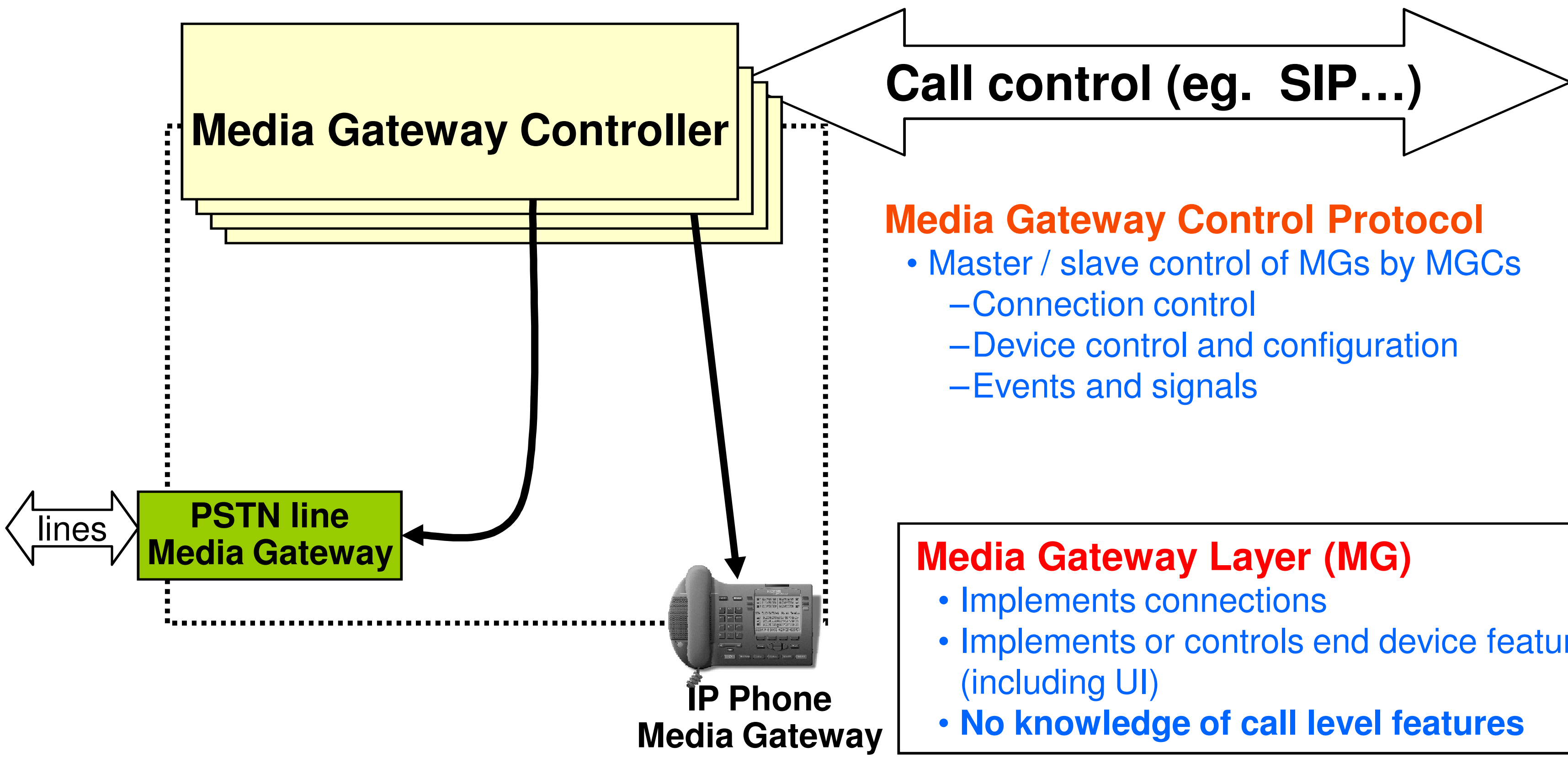
.

Media gateway control vs. SIP



Architecture

Media Gateway Control Layer (MGC)
- **Contains all call control intelligence**

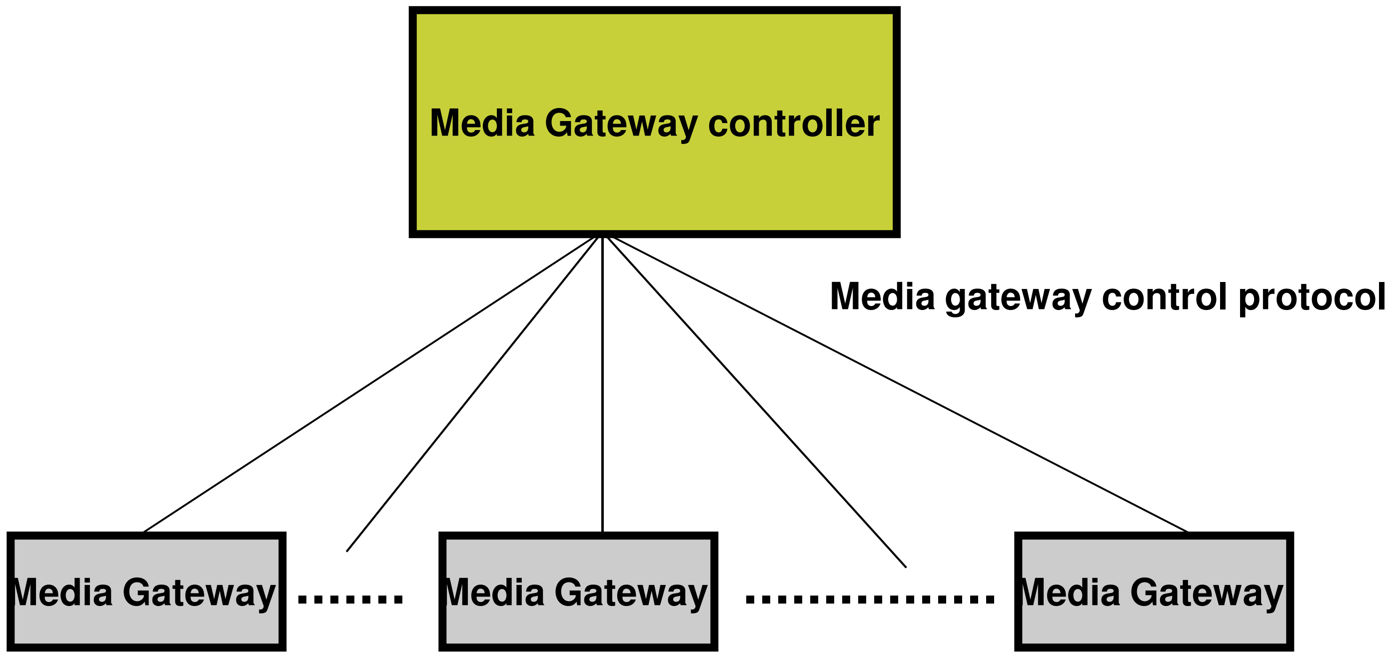


Media Gateway Control Protocol
- Master / slave control of MGs by MGCs
  - Connection control
  - Device control and configuration
  - Events and signals

Media Gateway Layer (MG)
- Implements connections
- Implements or controls end device features (including UI)
- **No knowledge of call level features**

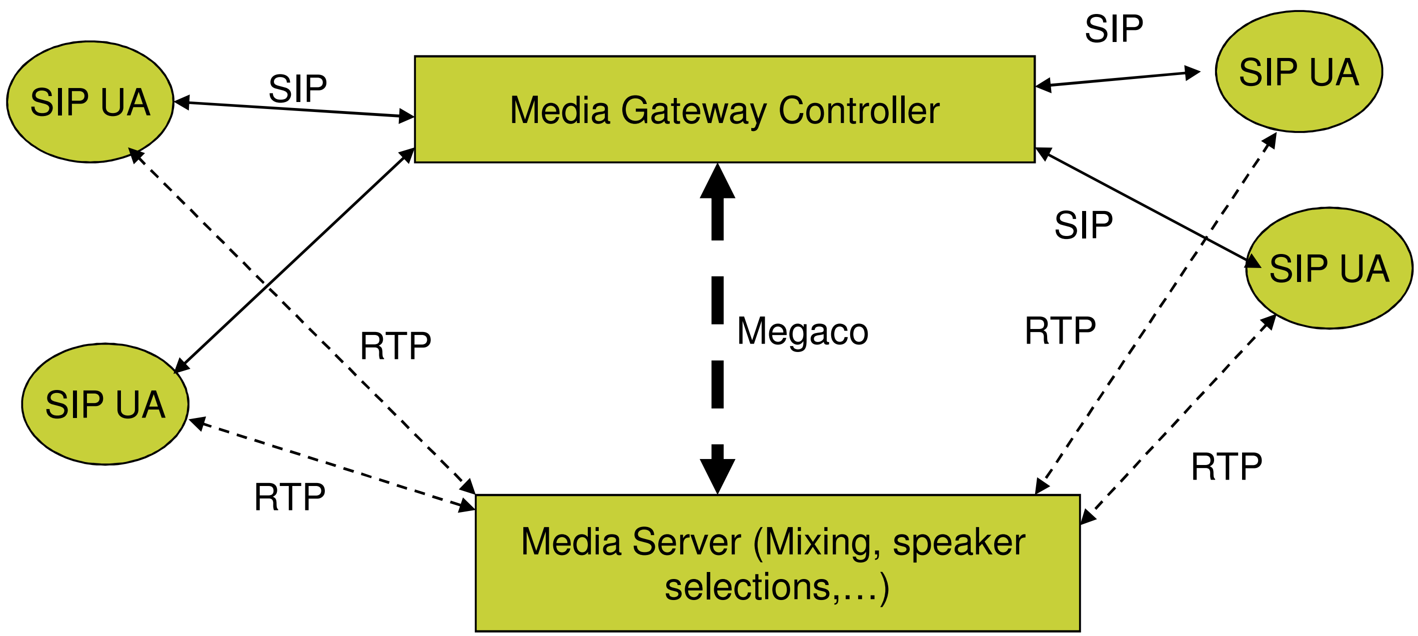
Megaco/H.248 : Motivations

- Improves scalability
  - Enables applications to share expensive media processing equipment
  - Basis for Vendor Independent Network deployment



Megaco/H.248 : Example of Application

- Conferencing Server



Megaco/H.248: Genesis

A long history starting in 1998

- Simple Gateway Control Protocol (SGCP)
  - Text based encoding, limited command set
- IP Device Control Protocol (IPDCP)
  - A few more features to SGCP
- Media Gateway Control Protocol (MGCP)
  - Merge of SGCP and IPDC
- Media gateway Decomposition Control Protocol (MDCP)
  - Binary encoded
- Megaco / H.248 (Joint IETF / ITU-T specifications)
  - A compromise
    - Both text based and binary encoding
    - A wide range of transport protocols(e.g. UDP, TCP, SCTP)

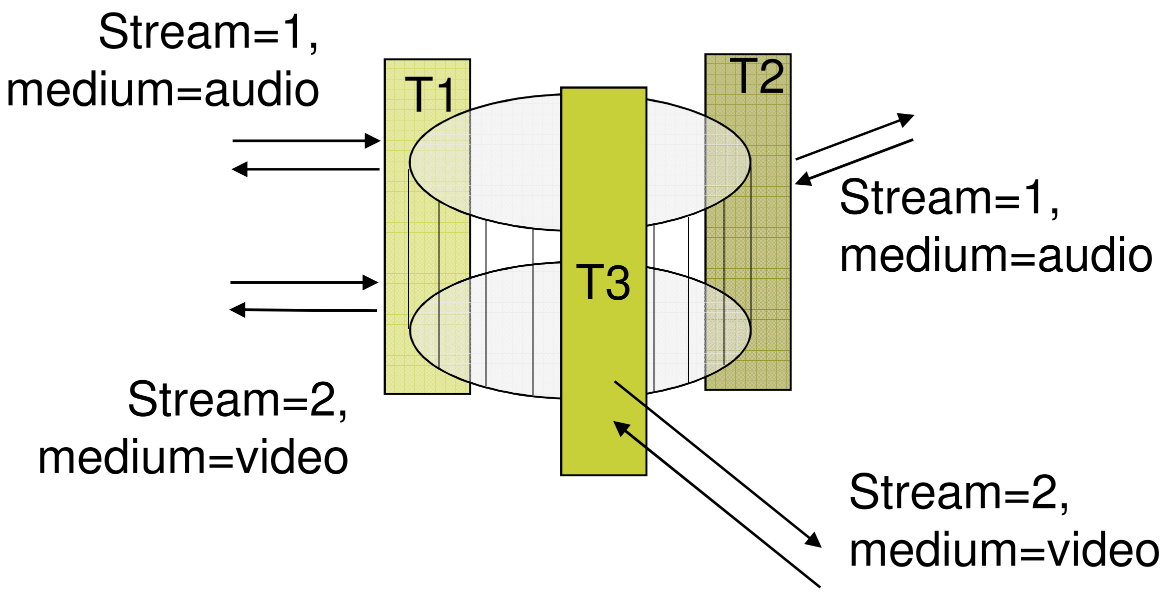
Megaco/H.248: Connection Model

```
      +------------------------------------------------------+
      |Media Gateway                                         |
      | +-------------------------------------------------+  |
      | |Context                       +-------------+   |  |
      | |                              | Termination |   |  |
      | |                              |-------------|   |  |
      | |  +-------------+          +->| SCN Bearer  |<---+->
      | |  | Termination |  +------+ |  |   Channel   |   |  |
      | |  |-------------|  |      |---+ +-------------+   |  |
   <-+--->| RTP Stream  |---|   *  |                     |  |
      | |  |             |  |      |---+ +-------------+   |  |
      | |  +-------------+  +------+ |  | Termination |   |  |
      | |                           |  |-------------|   |  |
      | |                           +->| SCN Bearer  |<---+->
      | |                              |   Channel   |   |  |
      | |                              +-------------+   |  |
      | +-------------------------------------------------+  |
      |                                                      |
```

Megaco/H.248: Concepts

- Termination
  - Identifies an end point for media flows
  - Implements Signals, and generates Events
  - Can appear in at most one context.
  - Permanent (provisioned) terminations can exist outside a context
- Context
  - Defines communication between Terminations
  - Contains 1 or more Terminations
  - Supports multiple streams
- Stream
  - A context can have multiple streams, each typically for a medium, e.g. audio, video, etc
  - The MGC specifies which streams a given termination supports

Concepts Example



Megaco/H.248: Concepts - Termination

Source or sink of media

- Persistent (circuit switched) or ephemeral (e.g. RTP)
- IDs
  - Unique or wildcard mechanism (ALL or CHOOSE)
- Properties/descriptors
  - Unique ids
  - Default values
  - Categorization
    - Common (I.e. termination state properties) vs. stream specific
    - For each media stream
      - Local properties
      - Properties of received streams
      - Properties of transmitted streams
    - Mandatory vs. optional
      - Options are grouped in packages

Megaco/H.248: Concepts - Termination

Examples of properties/descriptors

- Streams
  - Single bidirectional stream
    - Local control: Send only – send/receive …
    - Local: media received
    - Remote: media sent
- Events
  - To be detected by the MG and reported to the controller
    - On hook / Off hook transition
- Signals
  - To be applied to a termination by the MG
    - Tones
    - Announcements
- Digit map
  - Dialling plan residing in the MG
  - Detect and report events received on a termination ..

Megaco/H.248: Concepts - Context

Context (mixing bridge)

- Who can hear/see/talk to whom
- Association between terminations
- May imply
  - Conversion (RTP stream to PSTN PCM and vice versa)
  - Mixing (audio or video)
  - Null context
    - Terminations that are not associated with no other termination (e.g. idle circuit switched lines)
  - Topology
  - Precedence

.

Megaco/H.248: Protocol - Commands

Command	Initiator	Description
Add	MGC	Adds a termination to a context.
Modify	MGC	Modifies a termination's properties, events, and signals.
Move	MGC	Moves a termination from one context to another.
Subtract	MGC	Removes a termination from its context.
AuditValue	MGC	Returns current state of properties, events, signals, and statistics.
AuditCapabilities	MGC	Returns all possible values for termination properties, events, and signals allowed by an MG.
Notify	MG	Informs MGC of event occurrence(s).
ServiceChange	MGC	Takes or places a termination(s) out of or in service.
.	MG	For registration and restart; notifies MGC termination(s) will be taken out of or returned to service.

Megaco/H.248: Events

- Events are detected at the MG and reported to the MGC
  - example: DTMF
- MGC controls what events it wants to learn about at any given time
  - sets the termination Events descriptor
-

.

Hechmi KHLIFI/ Roch H. Glitho

Megaco/H.248: Signals

- Signals cause things to happen on terminations
  - play a tone, display text, ...
- Specified in the Signals descriptor for a termination
  - MGC can specify duration of signal ahead of time or signal can play until explicitly stopped
- Signals stop playing when any event is detected unless MGC says otherwise.

.

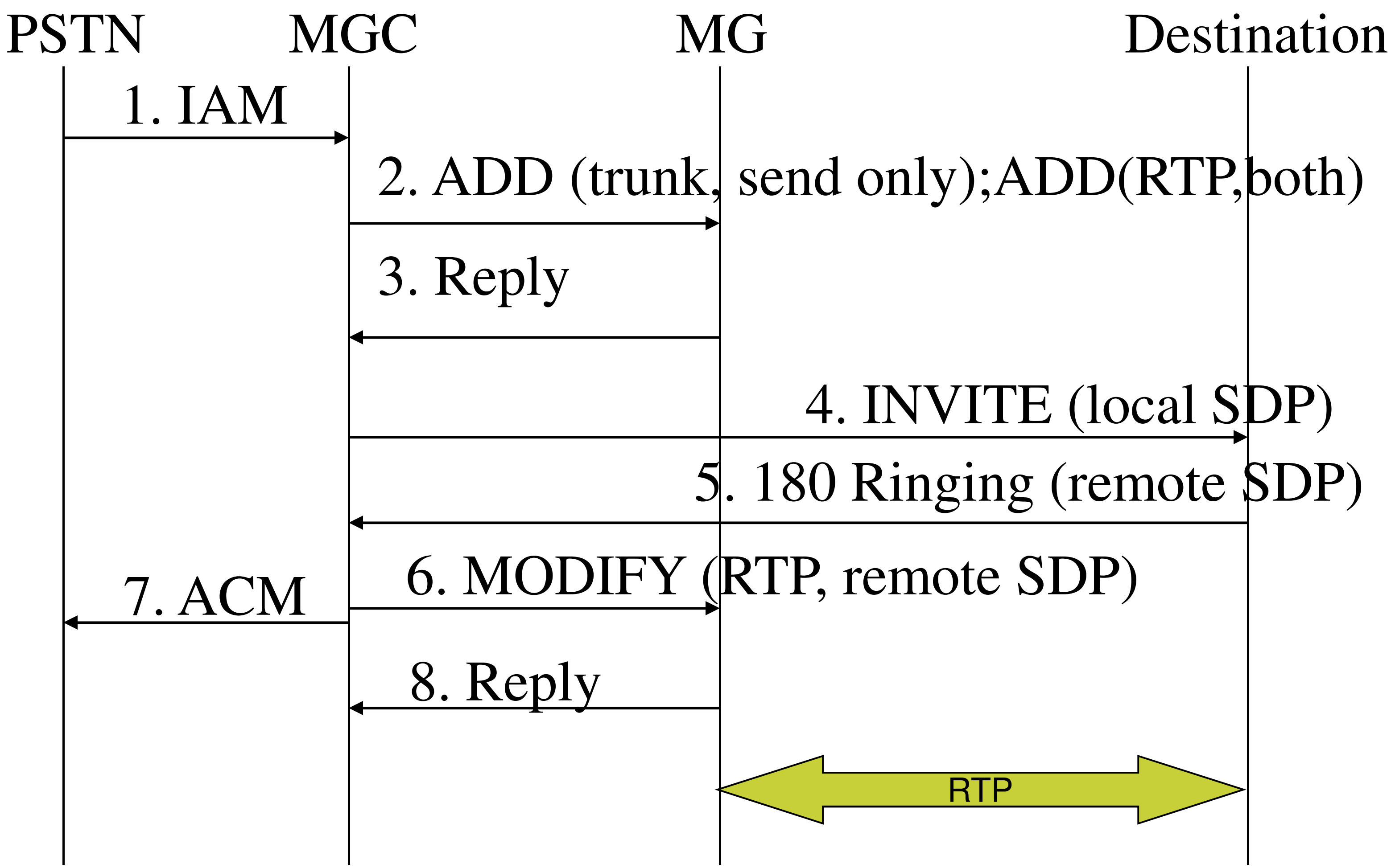
Megaco/H.248: Protocol - Transportation

Several alternatives

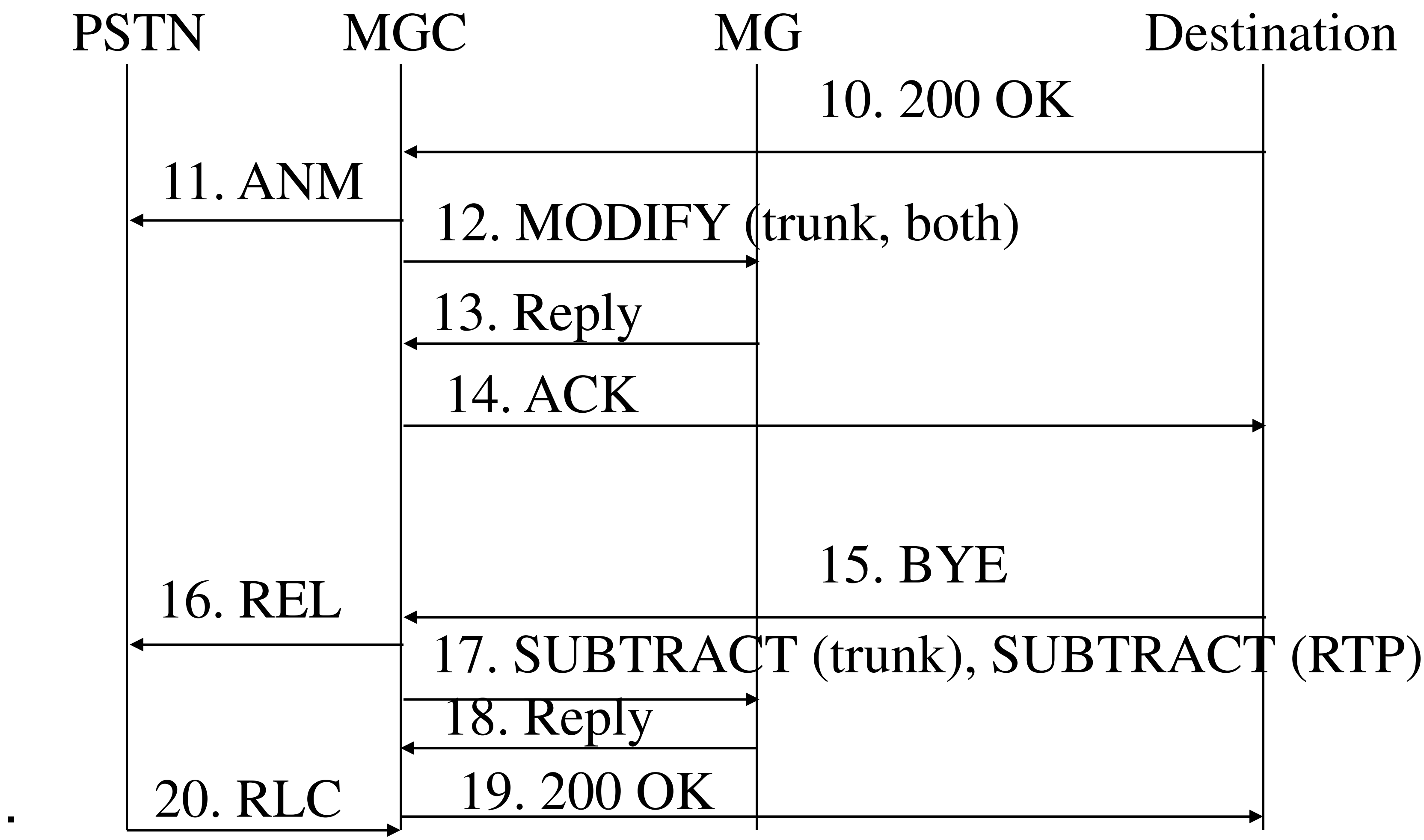
An example

- UDP/IP
  - Unreliable, timeouts / resends
  - At most once functionality required (Receivers should keep track of received commands)

Megaco/H.248: Message Flow 1



Megaco/H.248: Message Flow 2



Sample Message

```
MEGACO/1 <MGC1.gz.cn>
     Transaction=1432 {
     Context=$ {
     ADD=Ckt54/20 {
     Media= {
      LocalControl={mode=sendonly},
      Local={
     v=0
     c=TDM NUL xxxx
     m=audio 0 31kHz/I230 basic
            },
Remote={
     v=0
     c=TDM NUL xxxx
     m=audio 0 UDI/I230 basic
              }}
     }}}
```

References ...

1. Moderassi and S. Mohan, special issue, Advanced Signaling and Control in Next Generation Networks, IEEE Communications Magazine, October 2000 – Include papers on:

   - H.323
   - SIP

2. Additional references on Megaco/H.248

RFC 3525 (The protocol)

RFC 3054 (IP Phone)