Multimedia Sessions

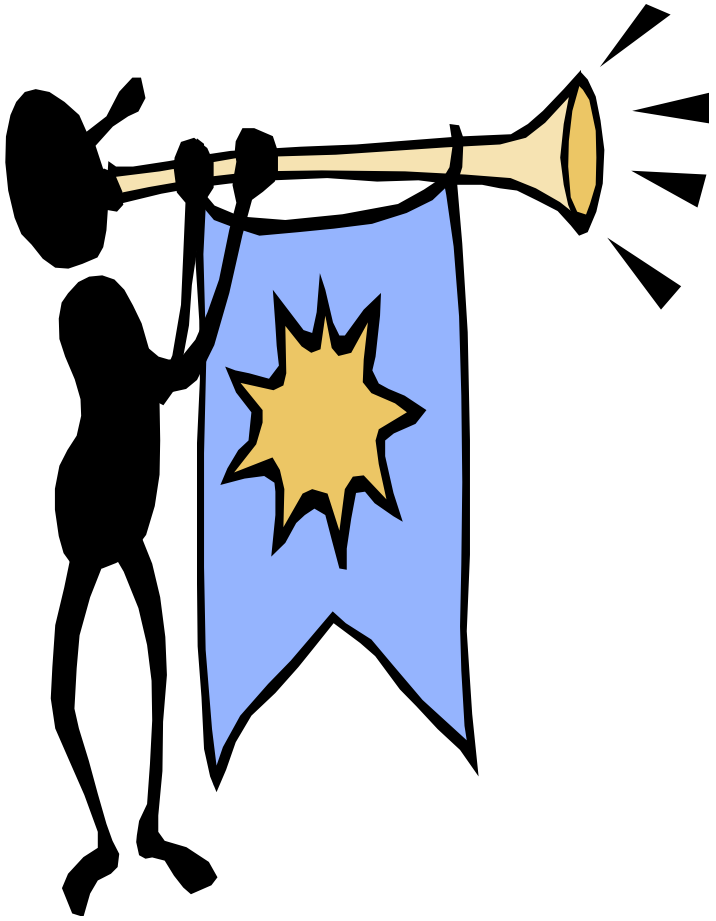
Roch H. Glitho

Multimedia Sessions

- Session Initiation Protocol (SIP)
- Conferencing Basics
- An advanced conferencing concept: Floor Control

SIP Session Signaling

Outline



1. Introduction
2. Core SIP
3. Selected Extensions

Introduction: Signaling vs Media

Signaling:

- Session establishment
- Session tear down
- Changes to the session
- Supplementary services

Media:

- Actual communication data: encoded voice stream, video stream,…

Introduction: SIP

Signaling Protocols:

- SIP and H.323

Media transport protocol:

- RTP

Why SIP?

SIP: Prime signaling system because adopted by all key next generation networks:

- 3GPP
- 3GPP2
- PacketCable:

SIP: Introduction

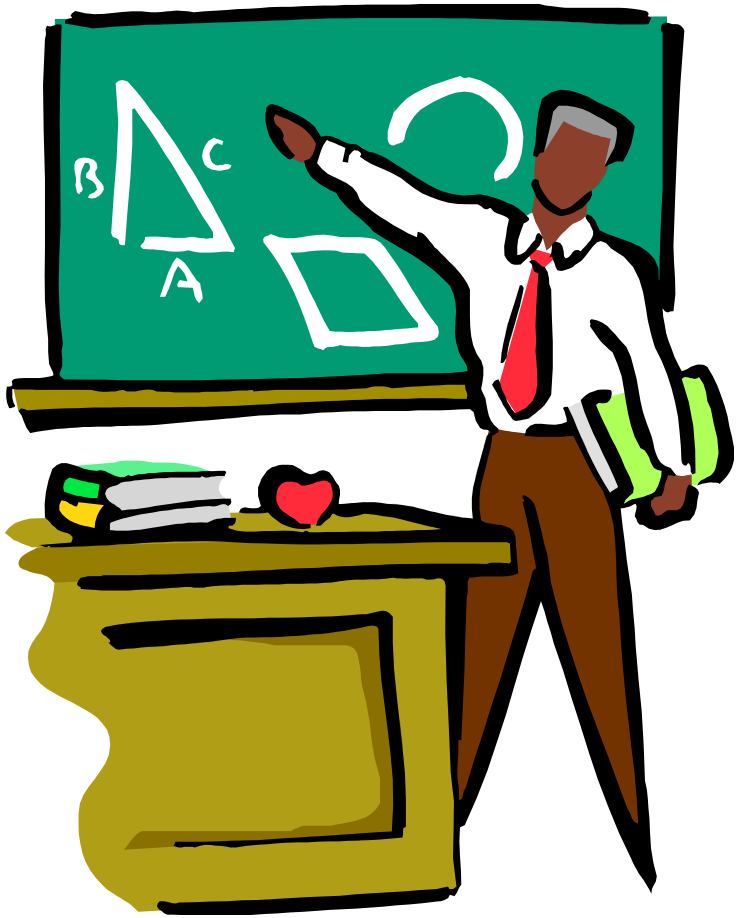
A set of IETF specifications including:

- **SIP core signalling:**
  - RFC 2543, March 1999
  - RFC 3261, June 2002 (Obsoletes RFC 2543)

- **SIP extensions (e.g. RFC 3265, June 2002 - Event notification)**
  - May have nothing to do with signalling
- IMS related extensions.

- **Used in conjunction with other IETF protocols**
  - QOS related protocol (e.g. RSVP)
  - Media transportation related protocol (e.g. RTP - RFC 1889)
  - Others (e.g. SDP - RFC 2327)

Session Initiation Protocol (SIP) - Core



1. Introduction
2. Functional entities
3. Messages
4. SDP
5. Examples

SIP: Introduction

SIP core Signaling

- A signalling protocol for the establishment, modification and tear down of multimedia sessions
- Based on HTTP

A few key features

- Text based protocol
- Client/server protocol (request/response protocol)

SIP: The Request

Request messages

- Methods for setting up and changing sessions
  - . INVITE
  - . ACK
  - . CANCEL
  - . BYE

- Others
  - . REGISTER (Registration of contact information)
  - . OPTIONS (Querying servers about their capabilities)

SIP: The Response

Response message

- Provisional
- Final

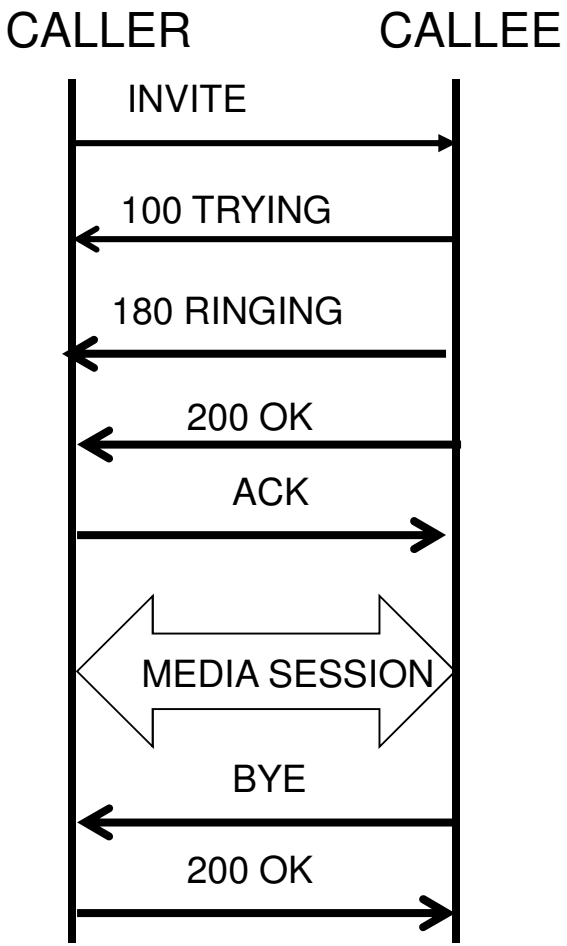
Examples of status code

1xx: Provisional

2xx: Success

6xx: Global failure

SIP: A basic peer to peer call scenario



SIP: The functional entities

User agents

- **End points, can act as both user agent client and as user agent server**
  - **User Agent Client: Create new SIP requests**
  - **User Agent Server: Generate responses to SIP requests**

Proxy servers

- Application level routers

Redirect servers

- Redirect clients to alternate servers

Registrars

- Keep tracks of users

SIP: The functional entities

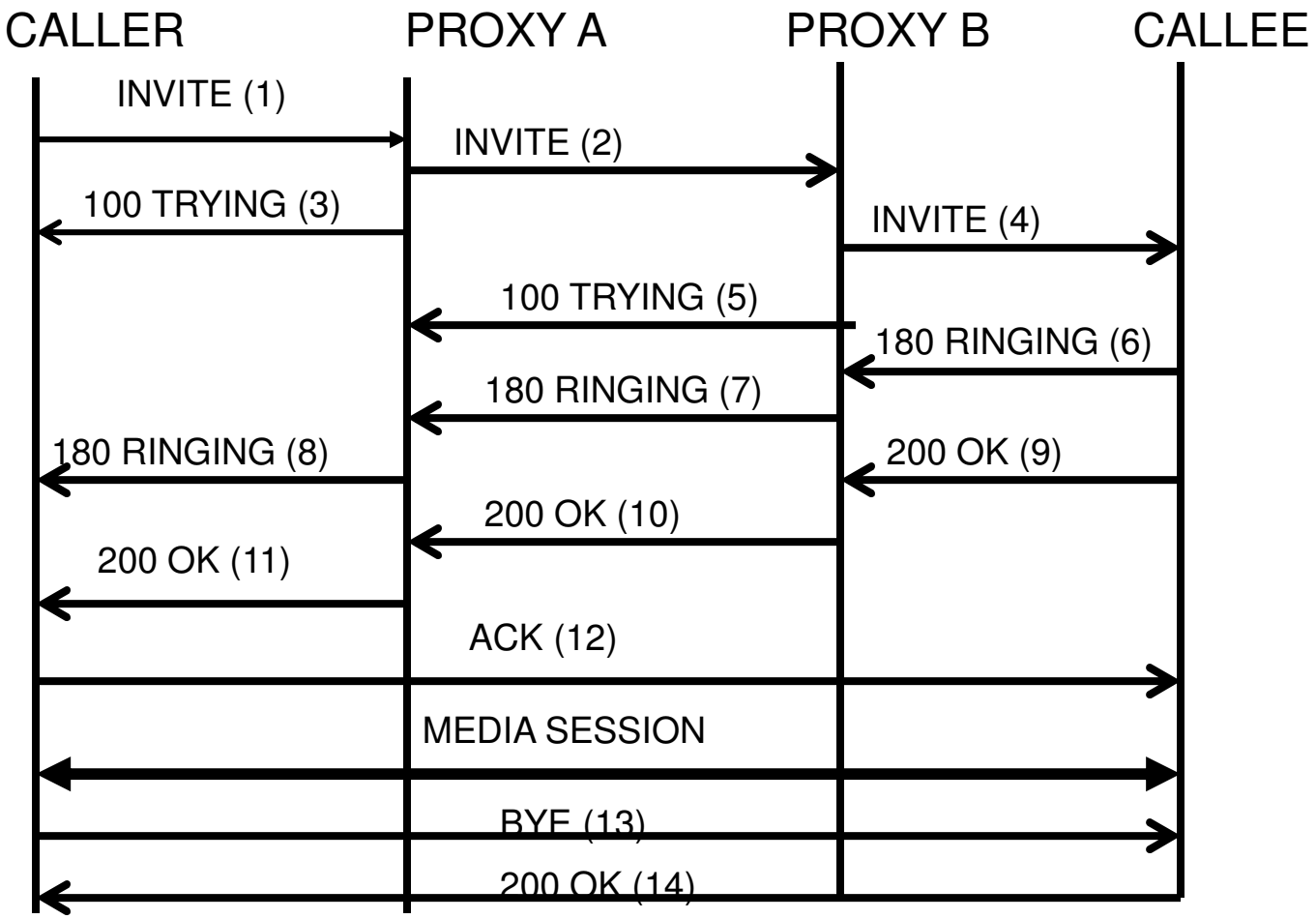
State-full proxy

- Keep track of all transactions between the initiation and the end of a transaction
- Transactions:
  - Requests sent by a client along with all the responses sent back by the server to the client

Stateless proxy

- Fire and forget

SIP: A call scenario



SIP: The messages

Generic structure

- Start-line
- Header field(s)
- Optional message body

Request message

- Request line as start line
  - . Method name
  - . Request URI
  - . Protocol version

Response message

- Status line as start line
  - . Protocol version
  - . Status code
  - . Reason phrase (Textual description of the code)

SIP: Examples of messages from the RFC

An example of an INVITE

```
INVITE sip:bob@biloxi.com SIP/2.0
  Via: SIP/2.0/UDP
 pc33.atlanta.com;branch=z9hG4bK776asdhds
  Max-Forwards: 70
  To: Bob <sip:bob@biloxi.com>
  From: Alice <sip:alice@atlanta.com>;tag=1928301774
  Call-ID: a84b4c76e66710@pc33.atlanta.com
  CSeq: 314159 INVITE
  Contact: <sip:alice@pc33.atlanta.com>
  Content-Type: application/sdp
  Content-Length: 142
```

SIP: Examples of messages from the RFC

An example of RESPONSE to the OPTIONS request

```
SIP/2.0 200 OK
   Via: SIP/2.0/UDP
  pc33.atlanta.com;branch=z9hG4bKhjhs8ass877
    ;received=192.0.2.4
   To: <sip:carol@chicago.com>;tag=93810874
   From: Alice <sip:alice@atlanta.com>;tag=1928301774
   Call-ID: a84b4c76e66710
   CSeq: 63104 OPTIONS
   Contact: <sip:carol@chicago.com>
   Contact: <mailto:carol@chicago.com>
   Allow: INVITE, ACK, CANCEL, OPTIONS, BYE
   Accept: application/sdp
   Accept-Encoding: gzip
   Accept-Language: en
   Supported: foo
   Content-Type: application/sdp
```

SDP

Session Description Protocol

- Convey the information necessary to allow a party to join a multimedia session
  - Session related information
  - Media related information
- Text based protocol

- No specified transport
  - Messages are embedded in the messages of the protocol used for the session
    - Session Announcement Protocol (SAP)
    - Session Initiation Protocol (SIP)

SDP

Session Description Protocol

Use with SIP

- Negotiation follows offer / response model
- Message put in the body of pertinent SIP messages
  - INVITE Request / response
  - OPTIONS Request / response

SDP

Session Description Protocol

- <Type> = <Value>
- Some examples
  - Session related
    - v= (protocol version)
    - s= (Session name)
  - Media related
    - m= (media name and transport address)
    - b= (bandwidth information)

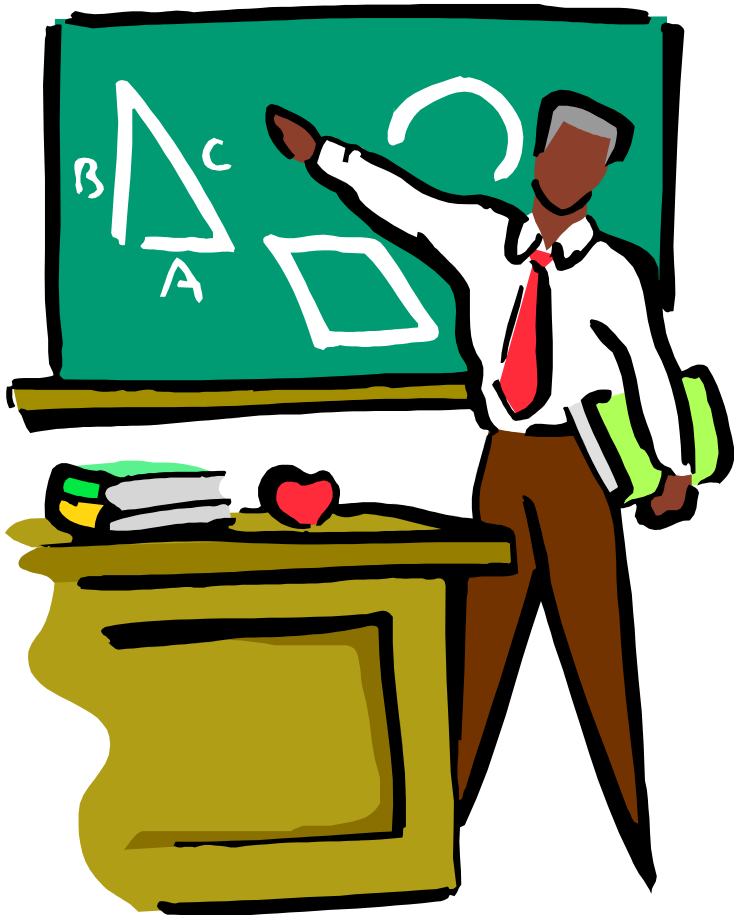
SDP: Examples of messages from the RFC …

Session Description Protocol

An example from the RFC …

v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu (Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 31
m=application 32416 udp wb
a=orient:portrait

SIP – Selected Extensions



1. Event framework
2. INFO method

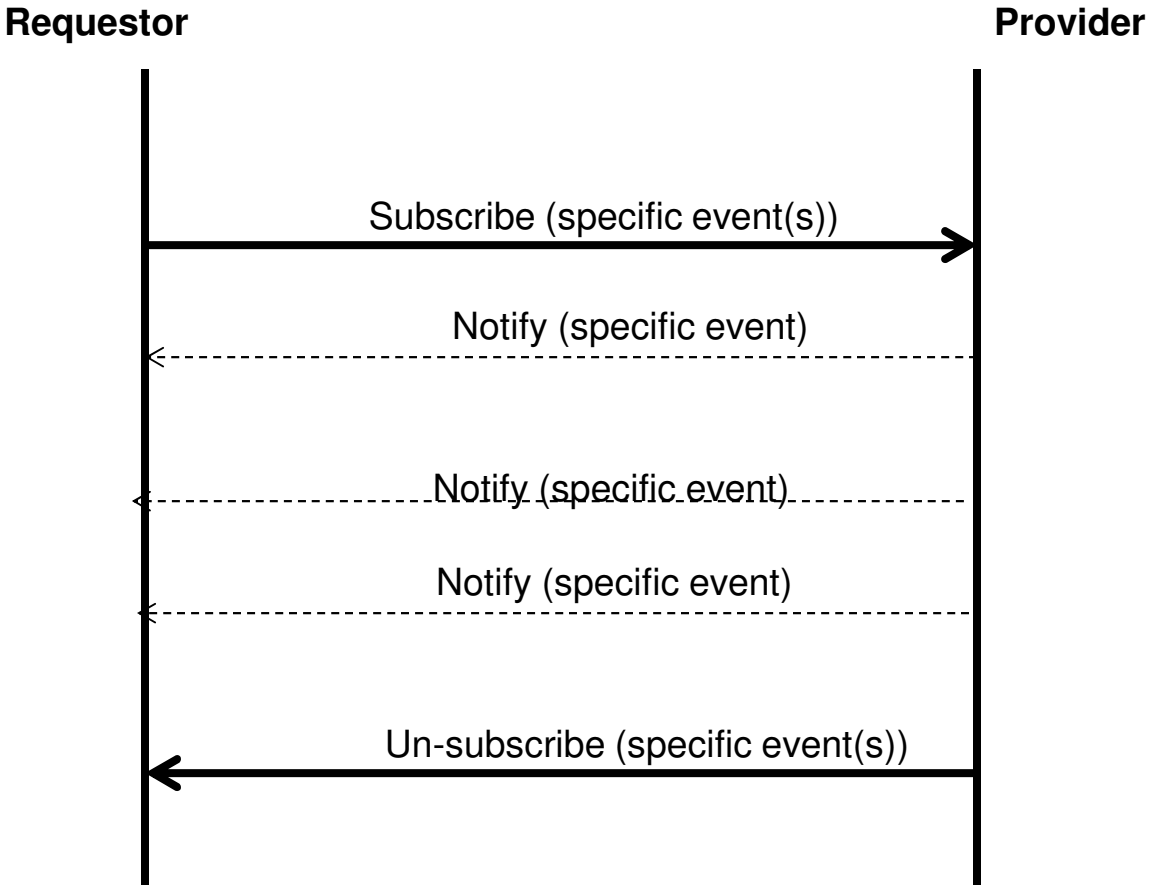
Event Notification

Motivation

- Necessity for a node to be asynchronously notified of happening (s) in other nodes
  - Busy / not busy (SIP phones)
    - A client A can call again a client B when notified that B is now not busy
  - On-line / Off-line
    - Buddy list

Event Notification

Conceptual framework



Event Notification

The SIP Event Notification Framework

- Terminology
  - Event package:
    - Events a node can report
    - Not part of the framework – Part of other RFCs
  - Subscriber
  - Notifier
- New Messages
  - Subscribe
    - Need to be refreshed
    - Used as well for un-subscribing (expiry value put to zero)
  - Notify

Event Notification

The SIP Event Notification Framework

- More on the methods
  - New headers
    - Event
    - Allow-Events
    - Subscription state

Event Notification

An example of use: REFER Method

- Recipient should contact a third party using the URI provided in the CONTACT field
  - Call transfer
  - Third party call control
- Handled as Subscribe / notify
  - REFER request is considered an implicit subscription to REFER event
    - Refer-TO: URI to be contacted
    - Expiry determined by recipient and communicated to sender in the first NOTIFY
    - Recipient needs to inform sender of the success / failure in contacting the third party

Event Notification

Another example of use: Presence

- Dissemination/consumption of presence information (e.g. on/off, willingness to communicate, device capabilities, preferences)
  - Numerous applications
    - Multiparty sessions initiated when a quorum is on-line
    - News adapted to device capabilities
- Several standards including SIMPLE (SIP based)
  - Handled as Subscribe / notify in SIMPLE
    - Watchers / presentities
      - Explicit subscriptions
      - Explicit notifications

INFO Method

Allow the exchange of non signalling related information during a SIP dialog

- Semantic defined at application level
- Mid-call signalling information
  - DTMF digits with SIP phones
- Info carried as
  - Headers and/or
  - Message body

References

Core SIP

- **SIP core signalling:**
- H. Schulzrinne, an J. Rosenberg, SIP: Internet Centric Signaling, IEEE Communications Magazine, October 2000
- RFC 3261, June 2002 (Obsoletes RFC 2543)
- RFC 2327 (SDP)

SIP extensions

No overview paper

- **RFC 3265, 3515 (Event framework)**
- **RFC 2976 (INFO Method)**

Conferencing

Basics

- Introduction
- Signaling Protocols
- Media control protocols

Part I: Introduction, signalling and media control protocols

- Introduction
  - What is multiparty multimedia session
  - Technical components
- Signaling protocols
  - SIP
- Media control protocols
  - SIP based protocols

- Introduction
  - What is multiparty multimedia session
  - How to implement
  - Protocols involved
  - Classifications

Multiparty multimedia session

- The conversational exchange of multimedia content between several parties
  - About multimedia
    - Audio, video, data, messaging



  - About participants
    - Any one who wants to participates the conference


How – thinking from a real life case

- When organizing a conference or a meeting, what to do?

Deciding topics, participants, time, agenda and booking a conf room → Policy control

Inviting participants and getting their confirmation to attend → Signaling

Starting the conference: let people seat down in the room and prepare the projector, microphone, player → Media control

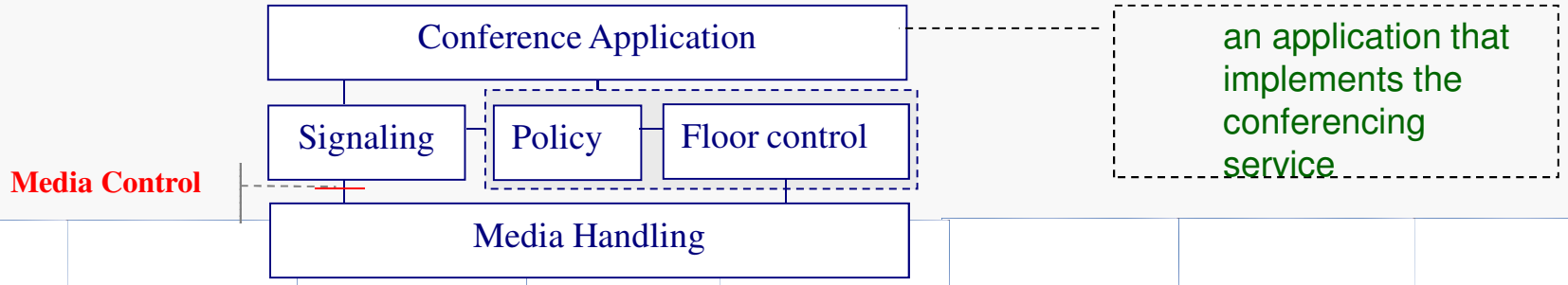
During a conference:

-- talking, discussing; presenting, playing a video to everybody, translating → Media handling

-- being a chair and deciding who can talk next → Floor control

How – technical components

- Signaling
  - Session establishment, modification and termination
  - Capability negotiation
- Media
  - Media handling: media transmission, mixing, trans-coding
  - Media control: stands when there is a separation of signaling and media mixing entities
- Conference control
  - Conference policy: conference arrangement, admission control, participant management, voting
  - Floor control: allows users of share resources such as video and audio without access conflicts.



Protocols involved

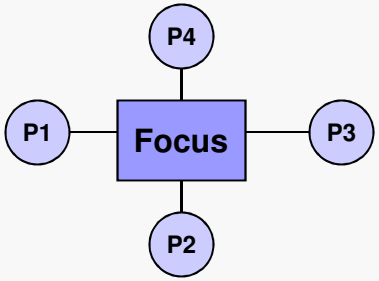
- Signaling
  - H.323, SIP (Session Initiation Protocol)
- Media
  - Media control: Megaco (Media Gateway Control protocol), SIP based media control – NetAnn/SIP MSCML (Media Server Control Markup Language), SIP media control channel framework
  - Media transport: RTP/RTCP, SRTP
- Conference control
  - Policy control: CPCP (conference policy control protocol), XCAP
  - Floor control: BFCP (Binary Floor Control Protocol), TBCP (Talk Burst Control Protocol)
    - Floor server control: FSCML (Floor Server Control Markup Language)

Classifications

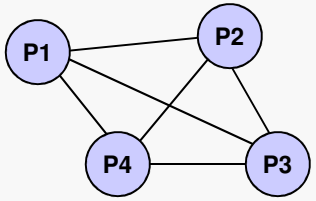
- Open/close
- Pre-arranged/ad hoc
- With/without sub-conferencing (i.e. sidebar)
- With/without floor control
- Topology: centralized, distributed, hybrid

- Signaling protocols
  - IETF: SIP
    - Conferencing models
    - Scenarios

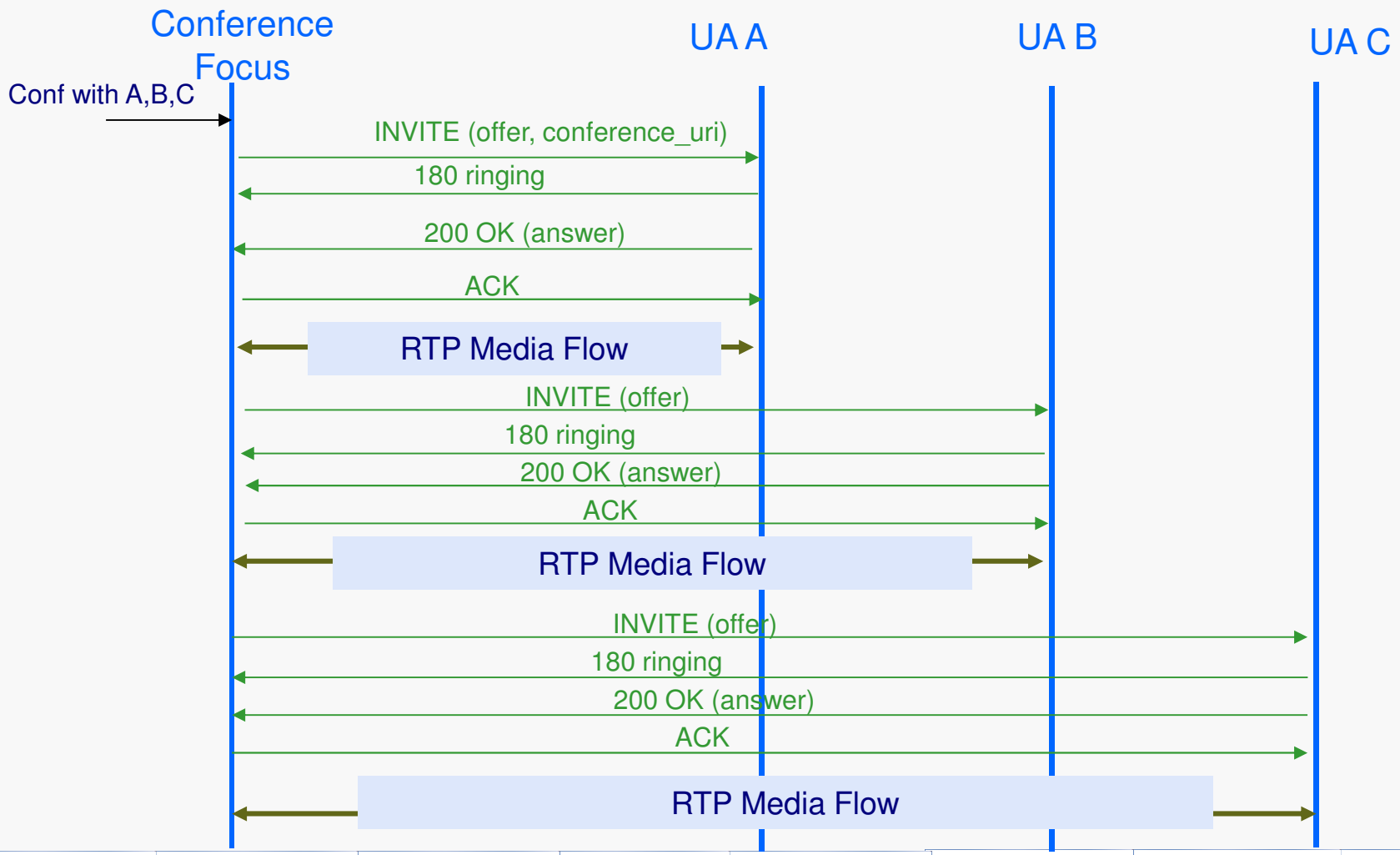
SIP conferencing models



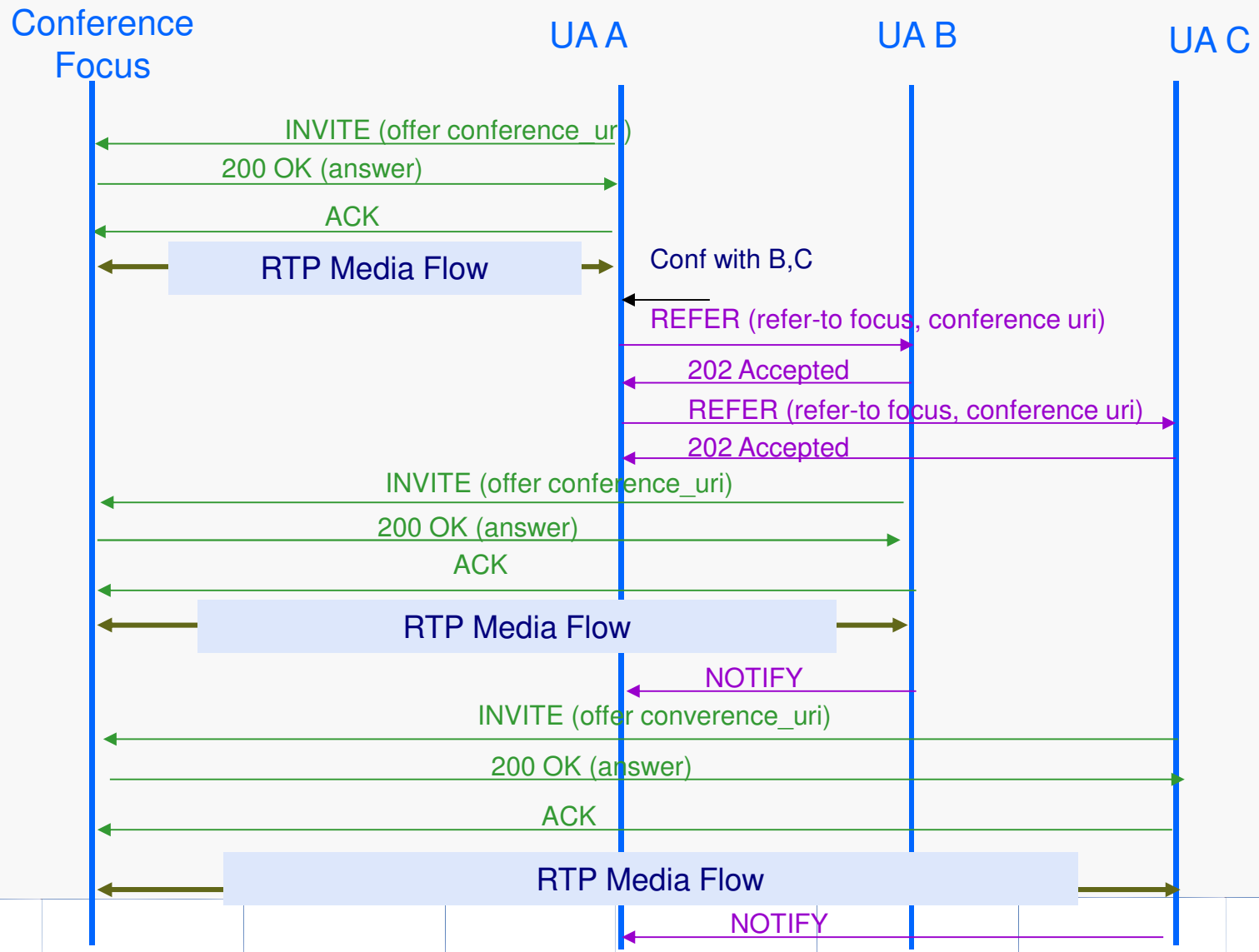
- Tightly coupled conference
  - Dial-In Conference
    - End point invite conference server which handle the media mixing
  - Dial-Out Conference
    - Server invite all the parties into a conference
  - Ad-hoc Centralized Conference
    - Two party setup conference directly, other party added through a conference server
- Loosely coupled
  - central signaling with multicast media
- Fully distributed



SIP conference example – dial out



SIP conference example – dial in



- Media control protocols
  - SIP Based Media Control
    - MSCML
    - SIP media control channel framework

- SIP based media control protocols
  - MSCML (RFC 5022)

What is MSCML

- Defined initially by RFC 4722, replaced by RFC 5022
- provides services to users at an application level, services specified in user part of SIP Request URI, control between AS and MS
- Provide IVR and advanced conference service, as well as fax
- Command oriented, request/response protocol

Basic concept

- There are three type of MSCML message, request, response, notification

```
<?xml version="1.0" encoding="utf-8"?>
  <MediaServerControl version="1.0">
    <request>
      ... request body ...
    </request>
  </MediaServerControl>
```

```
<?xml version="1.0" encoding="utf-8"?>
  <MediaServerControl version="1.0">
    <response>
      ... response body ...
    </response>
  </MediaServerControl>
```

- MSCML messages are located in the body of SIP Request messages. Each SIP request can only embed on MSCML message
  - SIP request messages: INVITE, INFO
  - ‘conf’ and ‘ivr’ in SIP request URI specify the message type

MSCML main commands

- Main requests
  - Conference related
    - <configure_conf>
    - <configure_leg>
    - <configure_team>
  - IVR related
    - <play>
    - <playcollect>
    - <prompt>
    - <playrecord>
    - <stop>
  - Event/signal (within a dialog)
    - <subscribe>
    - <notification>
    - <signal>
- Response
  - ID: optional
  - Request Type: e.g. <play>
  - Code: 2XX, 4XX, 5XX
  - Text: human readable

MSCML conference management

- Configure_conference is mandatory: creating a control leg for conference

```
<?xml version="1.0" encoding="utf-8"?>
  <MediaServerControl version="1.0">
    <request>
      <configure_conference reservedtalkers="120"
        reserveconfmedia="yes"/>
    </request>
  </MediaServerControl>
```

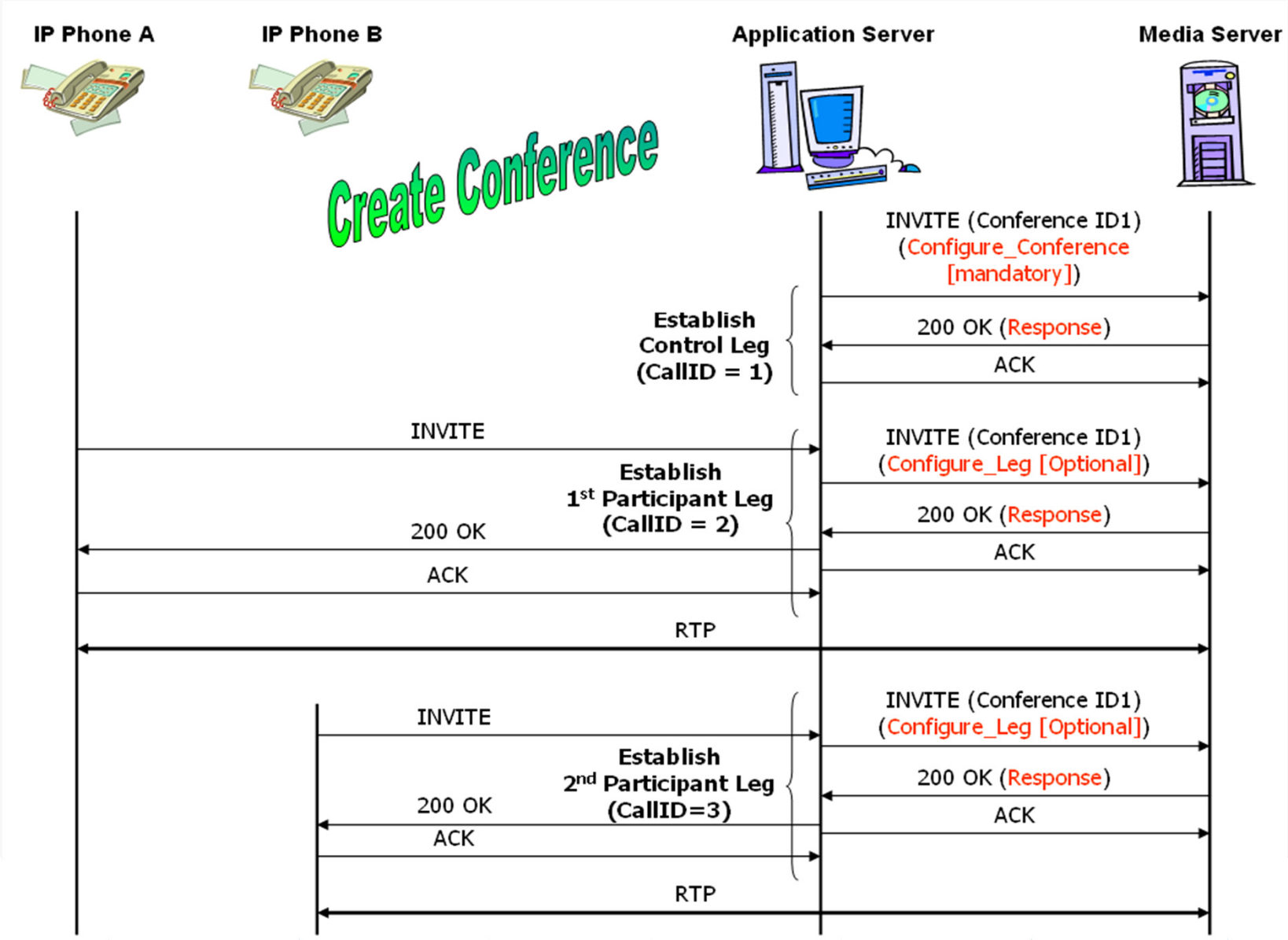
- Configure_leg is a control leg for a dialog. It can configure the dialog's media mode

```
<?xml version="1.0" encoding="utf-8"?>
  <MediaServerControl version="1.0">
    <request>
      <configure_leg mixmode="mute"/>
    </request>
  </MediaServerControl>
```

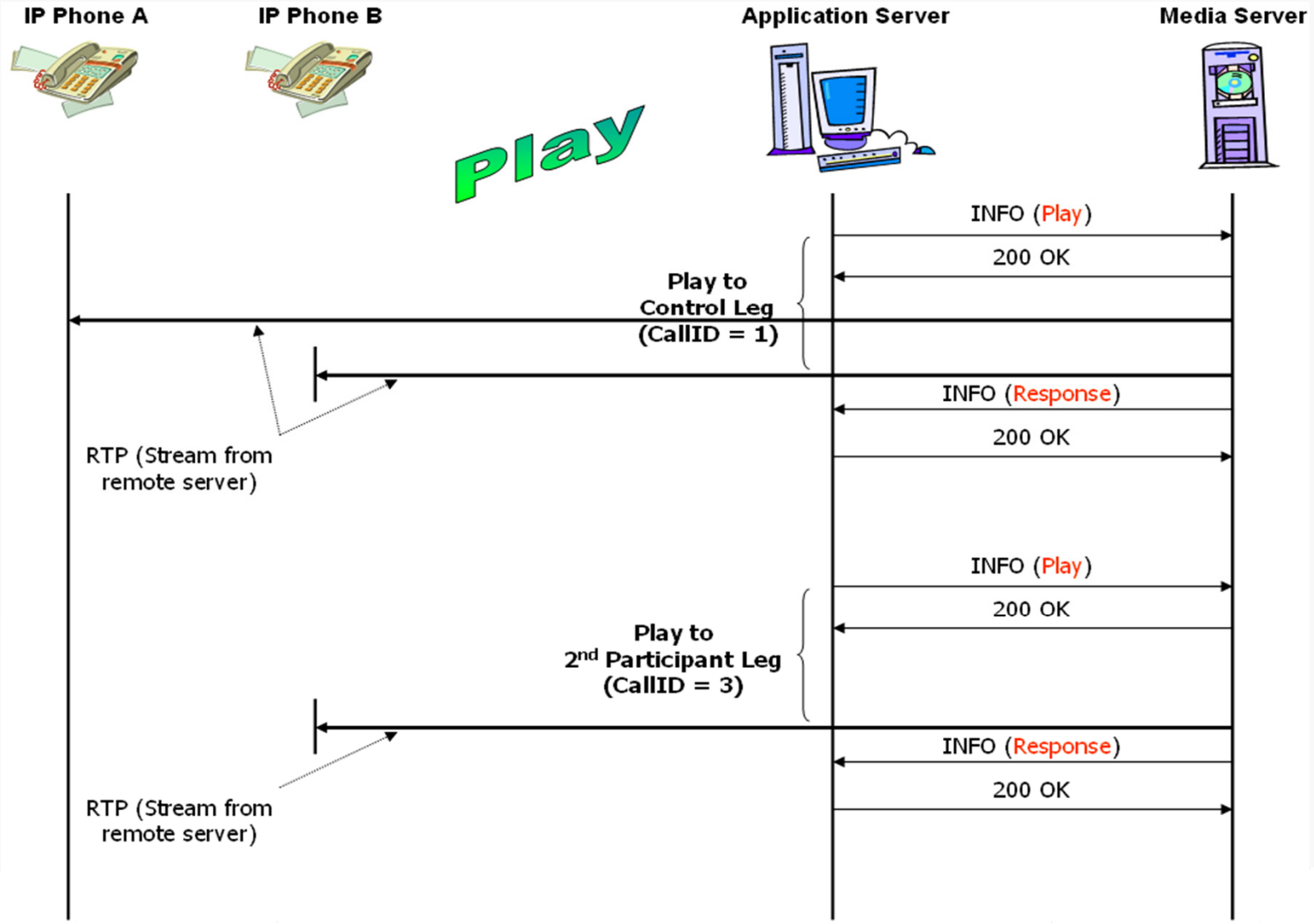
- Can play a prompt to a conference or to a specific leg
- Conference terminates by sending a BYE to conference control leg
- BYE to a leg will just remove a participant

```
<?xml version="1.0" encoding="utf-8"?>
  <MediaServerControl version="1.0">
    <request>
      <play>
        <prompt>
          <audio
url="http://prompts.example.net/en_US/welcome.au"/>
        </prompt>
      </play>
    </request>
  </MediaServerControl>
```

MSCML conference example – Create



MSCML conference example – play a prompt



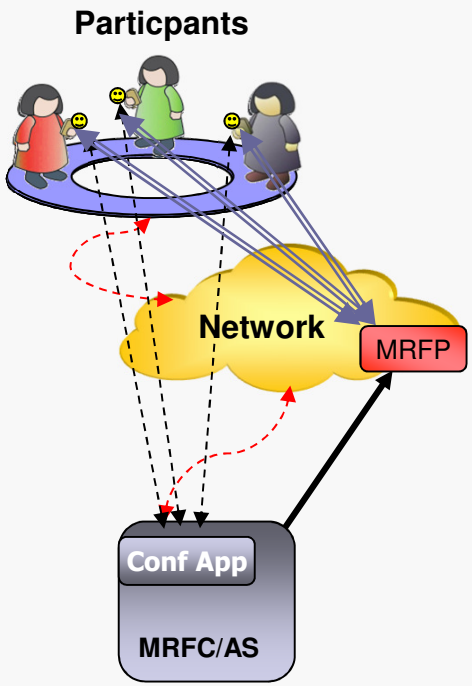
Questions

Part-II

Multiparty multimedia session

- Is the conversational exchange of multimedia content among several parties.

- It has 3 main building blocks:
  - Signaling
    - H.323, SIP
  - Media control and handling
    - Megaco/H.248, NetAnn/SIP-MSCML
    - RTP
  - Conference control
    - Policy control: CPCP (conference policy control protocol), XCAP
    - Floor control



An Advanced Concept: Floor Control

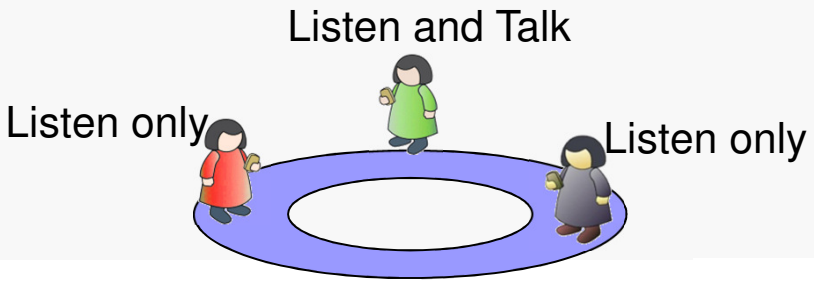
- Definition
- Architecture
- Protocols

Definition

Floor control: a mechanism that enables the management of the joint or exclusive access to the shared resources inside a conference

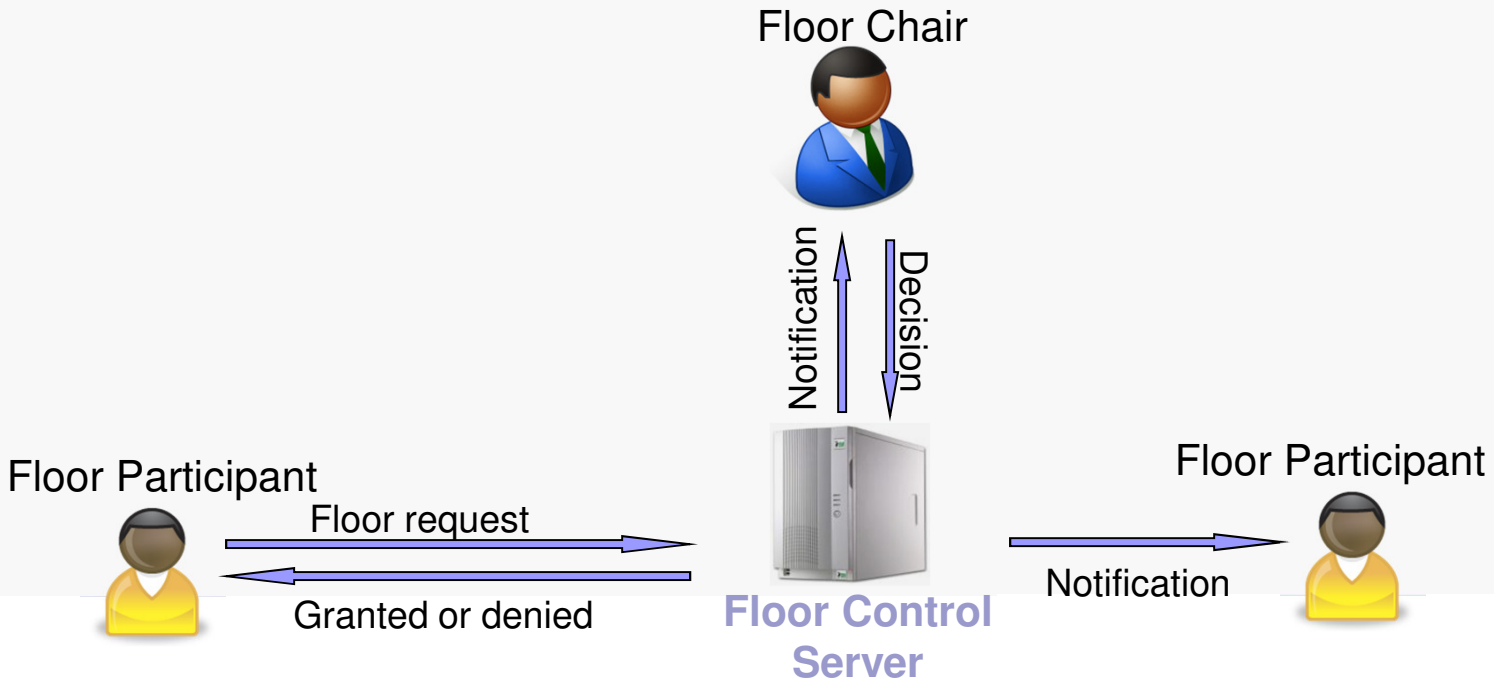
.e.g. audio/video channels, slide bar presentation

Floor: “A temporary permission to access or manipulate a specific shared resource or set of resources”.



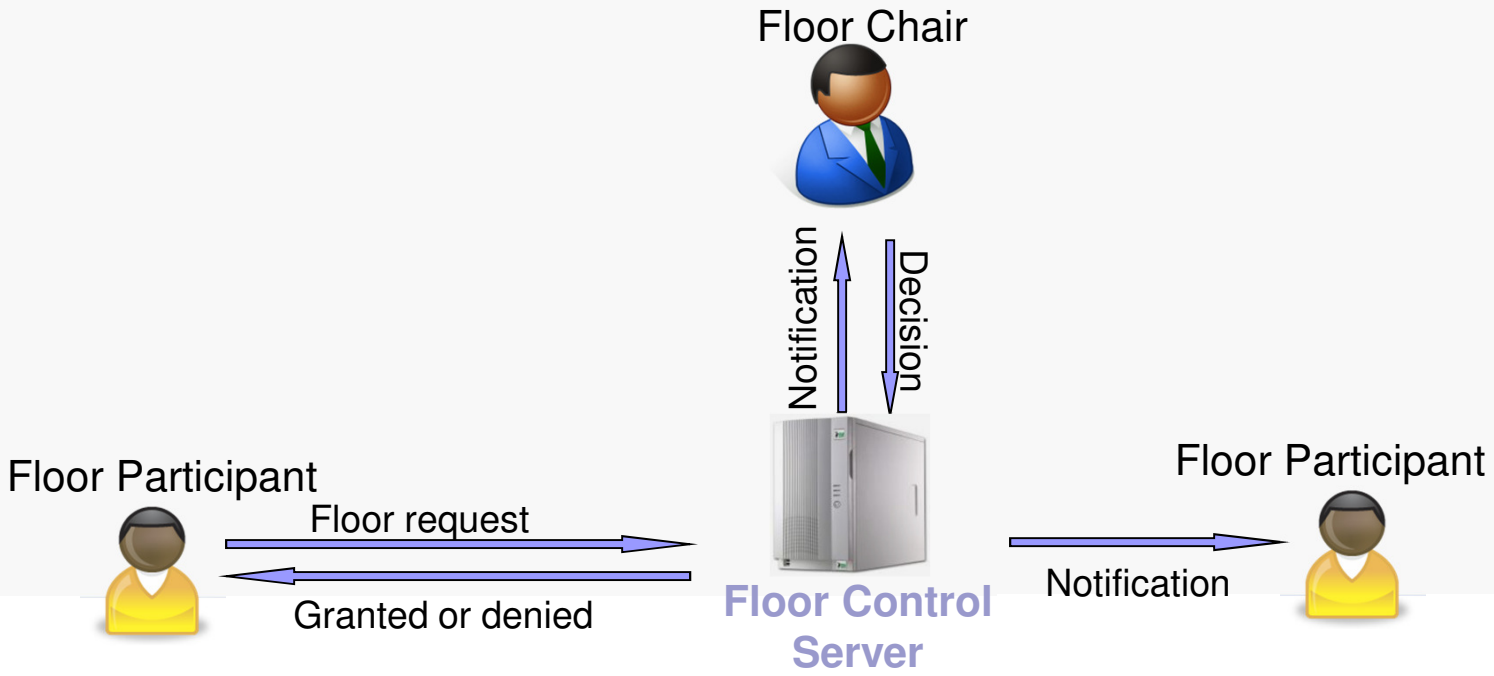
Architecture

- Three entities are involved in floor control:
  - Floor participant
  - Floor chair
  - Floor Control Server (FCS)



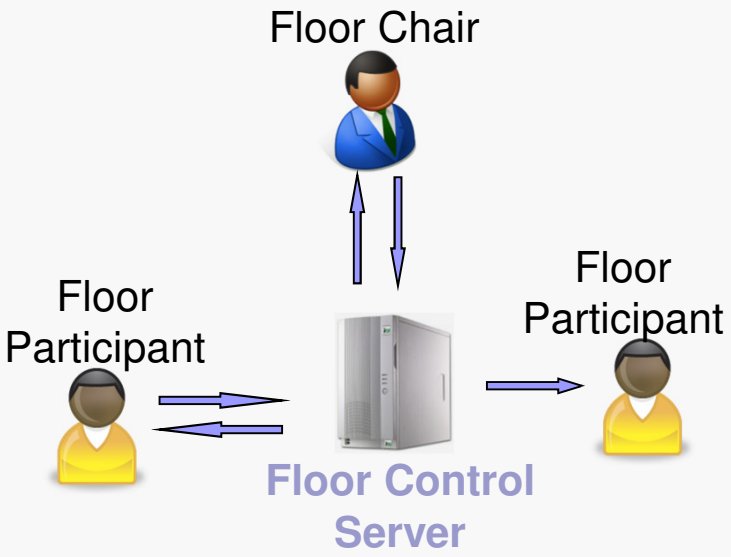
Architecture

- Two main algorithms
  - First come First Serve (FCFS)
  - Chair moderated



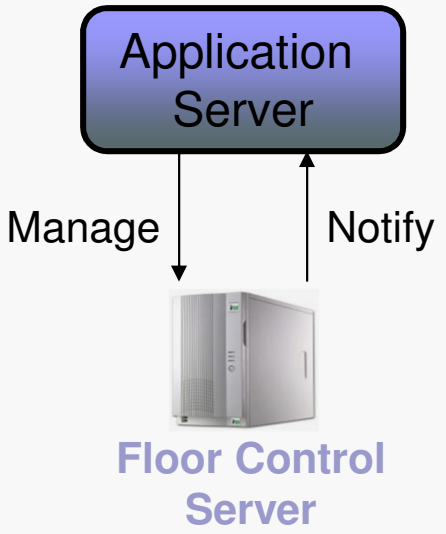
Protocols

1. Establish floor control connections between the different entities



2. Coordinate access to shared resources

3. Control the FCS
   - Create/terminate floor
   - Add participant/resource to floor
   - Remove participant/resource from floor



Protocols

- Establish floor control connections between the different entities
  - SIP/SDP (RFC 4583, RFC 5239)
- Coordinate access to shared resources
  - Binary Floor Control Protocol (BFCP)
  - Talk Burst Control Protocol (TBCP)
- Control the FCS
  - Megaco/H.248
  - SIP Floor Server Control Markup Language (SIP-FSCML)

Establish floor control connections between the different entities

- **Examples** of an offer sent by a conference server to a client

```
m=application 50000 TCP/TLS/BFCP *
a=setup:passive
a=connection:new
a=fingerprint:SHA-1 \
   4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
a=floorctrl:s-only
a=confid:4321
a=userid:1234
a=floorid:1 m-stream:10
a=floorid:2 m-stream:11
m=audio 50002 RTP/AVP 0
a=label:10
m=video 50004 RTP/AVP 31
a=label:11
```

Establish floor control connections between the different entities

- **Examples** of an answer returned by the client

```
m=application 9 TCP/TLS/BFCP *
a=setup:active
a=connection:new
a=fingerprint:SHA-1 \
   3D:B4:7B:E3:CC:FC:0D:1B:5D:31:33:9E:48:9B:67:FE:68:40
   :E8:21
a=floorctrl:c-only
m=audio 55000 RTP/AVP 0
m=video 55002 RTP/AVP 31
```

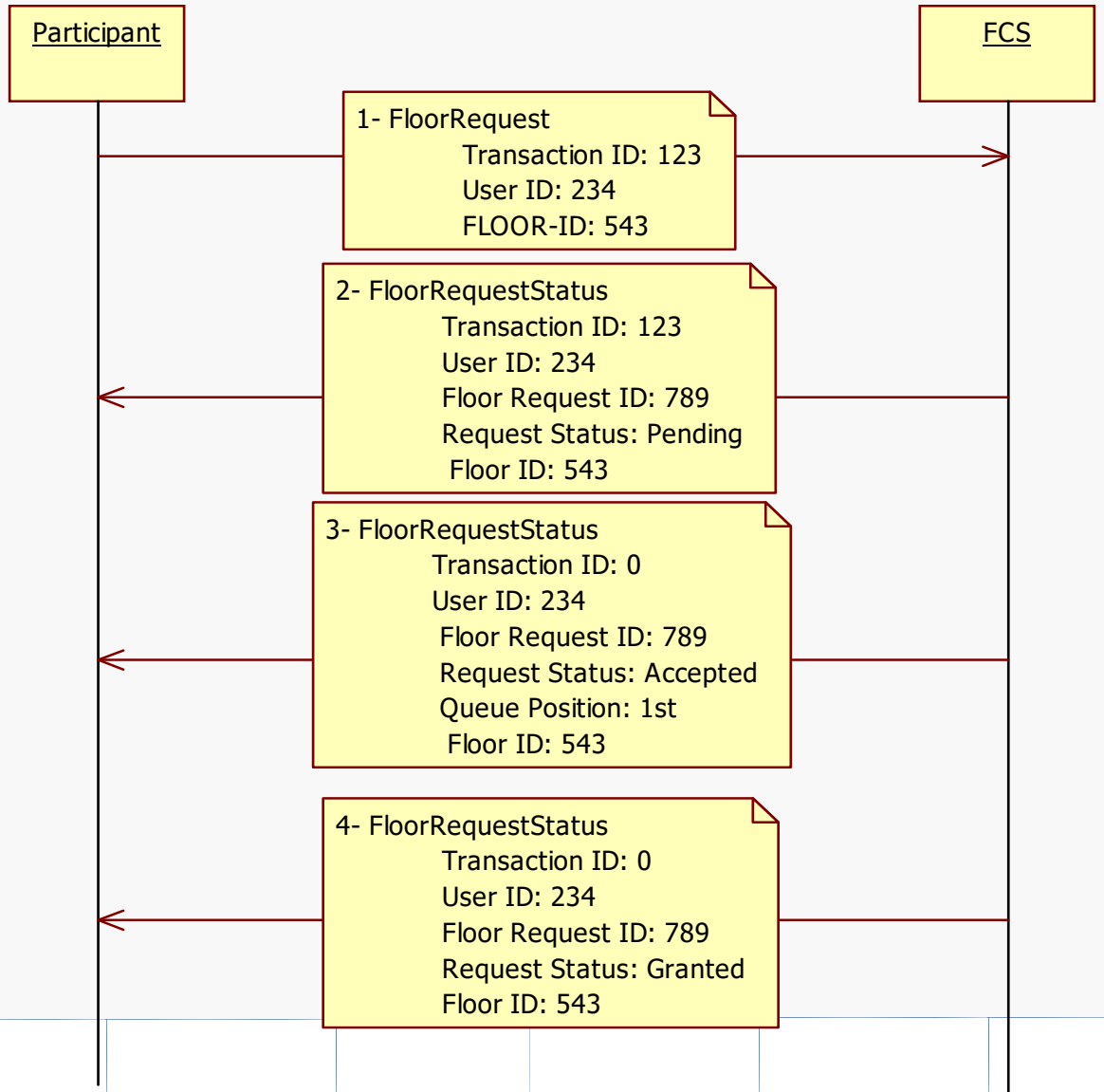
Coordinate access to shared resources

- Binary Floor Control Protocol (BFCP)
  - Standardized in RFC 4582
  - Negotiation of BFCP connections within SIP/SDP
    - Standardized in RFC 4583
  - Advantages
    - Fast (binary encoded)
    - Secure
    - Reliable (over TCP)
    - Provides all the floor control functionalities

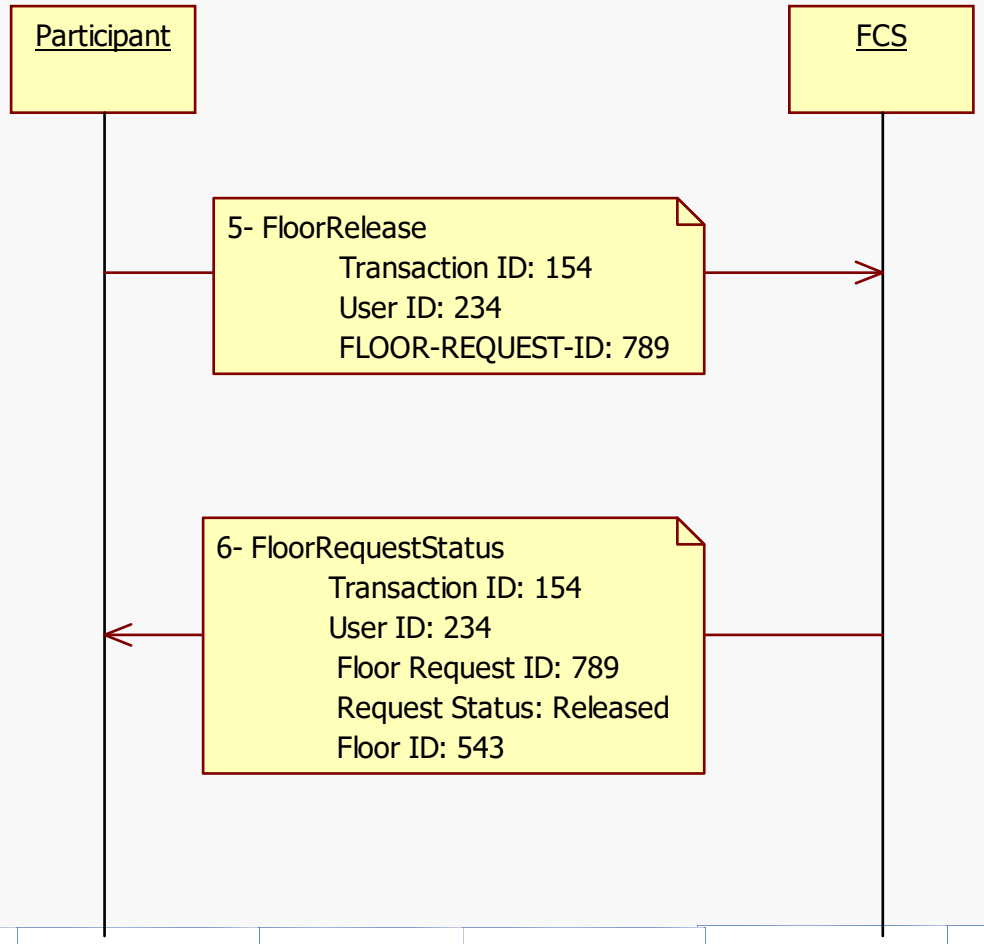
Binary Floor Control Protocol

- Protocol operations and messages/primitives
  - Participant operations
    - Request a floor (FloorRequest)
    - Cancel a floor request (FloorRelease)
    - Release a Floor (FloorRelease)
  - Chair operations
    - Grant a floor (ChairAction)
    - Deny a floor (ChairAction)
    - Revoke a floor (ChairAction)
  - Participant/Chair
    - Requesting Information about Floors (FloorQuery)
    - Requesting Information about Floor Requests (FloorRequestQuery)
    - Requesting Information about a User (UserQuery)
    - Obtaining the Capabilities of a Floor Control Server (Hello)
  - FCS operations
    - Handles the participant and chair requests (FloorRequestStatus, FloorStatus, UserStatus, ChairActionAck, HelloAck, Error)

Binary Floor Control Protocol



Binary Floor Control Protocol



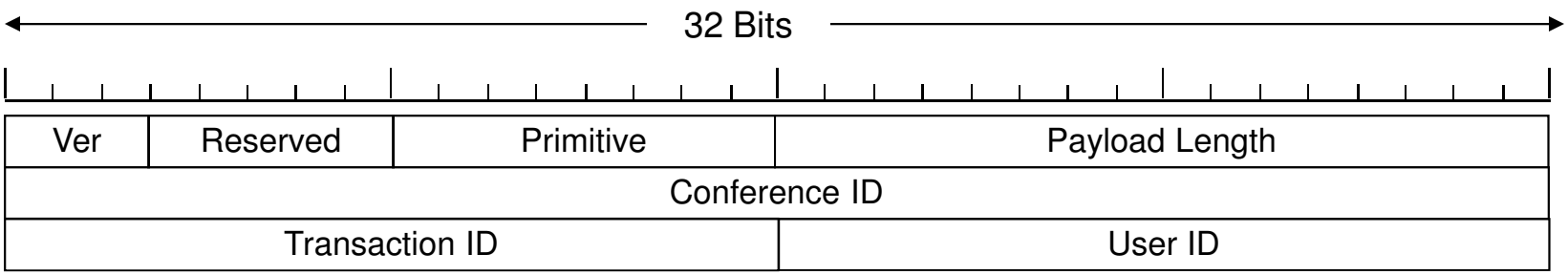
Binary Floor Control Protocol

- Packet Format
  - BFCP messages
    - Consist of a common header followed by a set of attributes.
    - Use a TLV (Type-Length-Value) binary encoding
    - Floor participants, media participants, and floor chairs are identified by 16-bit user identifiers.
    - BFCP supports nested attributes (i.e., attributes that contain attributes).
      - Referred to as grouped attributes.

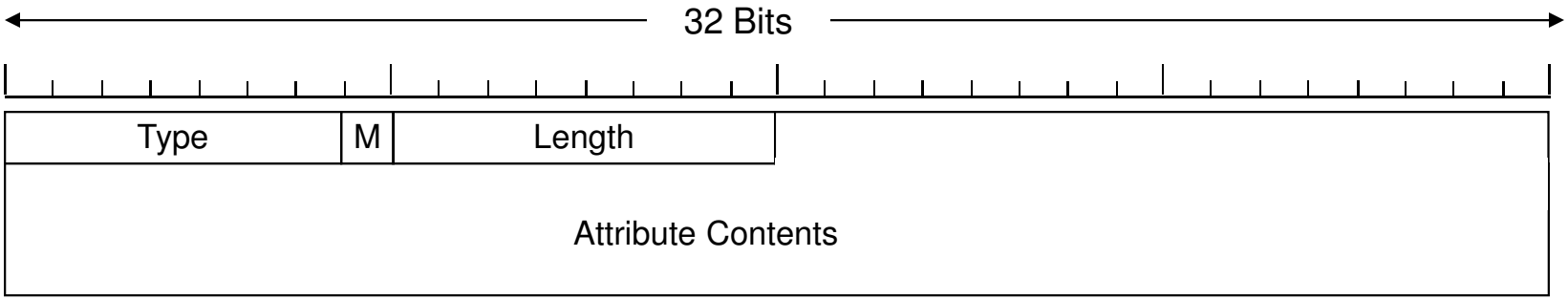
Binary Floor Control Protocol

- Packet Format

Common-header format



Attribute format



Binary Floor Control Protocol

- Primitives

Value	Primitive	Direction
1	FloorRequest	P -> S
2	FloorRelease	P -> S
3	FloorRequestQuery	P -> S ; Ch -> S
4	FloorRequestStatus	P <- S ; Ch <- S
5	UserQuery	P -> S ; Ch -> S
6	UserStatus	P <- S ; Ch <- S
7	FloorQuery	P -> S ; Ch -> S
8	FloorStatus	P <- S ; Ch <- S
9	ChairAction	Ch -> S
10	ChairActionAck	Ch <- S
11	Hello	P -> S ; Ch -> S
12	HelloAck	P <- S ; Ch <- S
13	Error	P <- S ; Ch <- S

S: Floor Control Server P: Floor Participant Ch: Floor Chair

Talk Burst Control Protocol

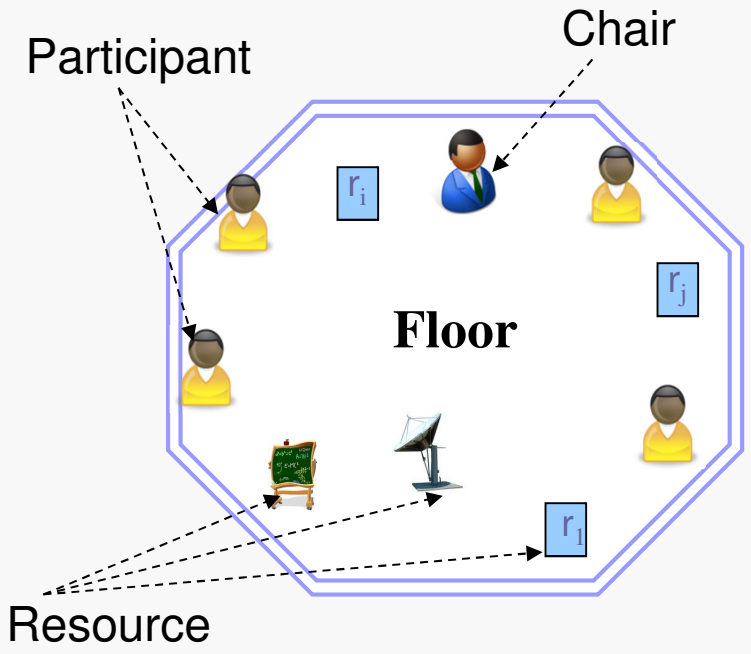
- TBCP
  - Defined by the OMA (Open Mobile Alliance)
  - Uses the application extension features of RTCP (RTP Control Protocol) in order to invoke floor control within the POC (Push to talk Over Cellular) environment.
- Typical TBCP messages include:
  - Talk Burst Granted
  - Talk Burst Request Message
  - Talk Burst Deny Message
  - Talk Burst Release Message
  - Talk Burst Taken
  - Talk Burst Idle
  - Talk Burst Revoke
- Advantages
  - Fast
  - Secure
- Disadvantages
  - Only provides basic floor control functionalities (e.g. no chair supported).

Control the FCS

- SIP-FSCML is a non-standard alternative to H.248
  - Less complex
  - Easy to understand and use by SIP application developers.
- It follows SIP and XML paradigms.
- It enables a peer-to-peer communication model between the AS and the FCS.
  - This allows the FCS to be simultaneously used by multiple ASs.

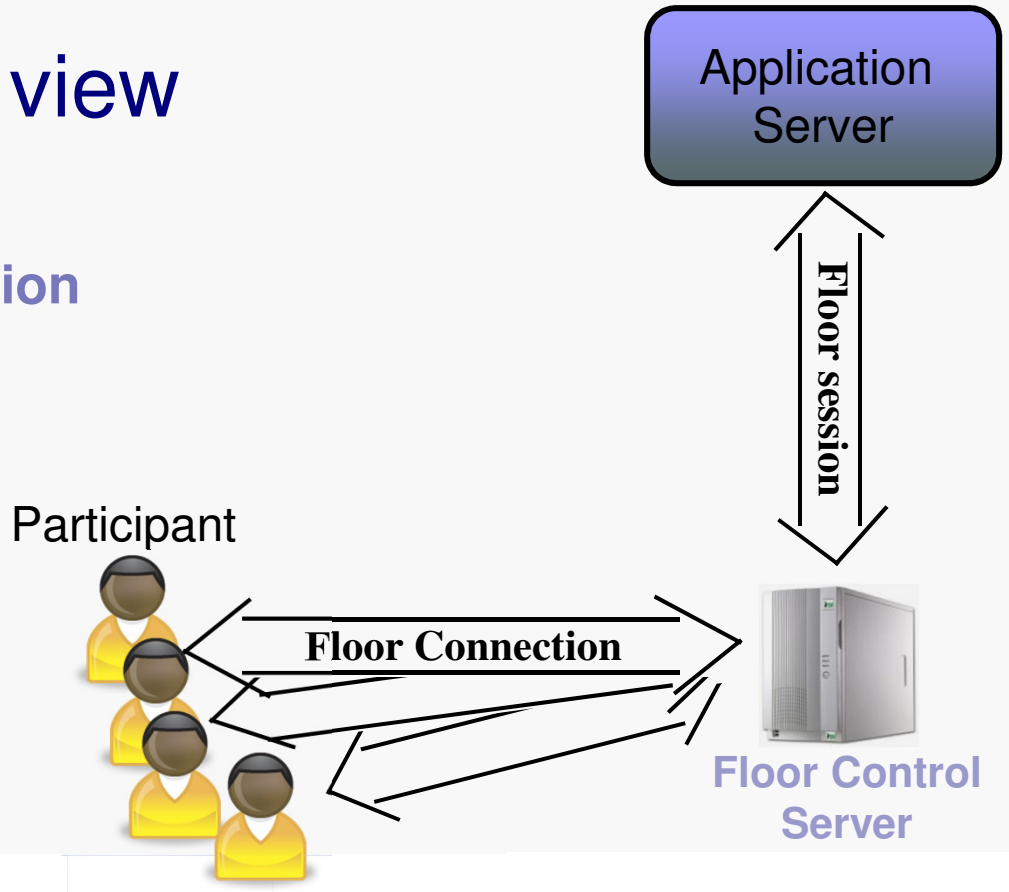
SIP Floor Server Control Markup Language

- Conceptual view
  - Floor
  - Floor Connection
  - Floor Session



SIP Floor Server Control Markup Language

- Conceptual view
  - Floor
  - Floor Connection
  - Floor Session



SIP Floor Server Control Markup Language

- The control session between the application and the FCS is opened through a SIP INVITE message.
- FSCML requests to the FCS are carried in SIP INFO messages
  - Each INFO message includes a single FSCML body
  - An FSCML body can carry any number of FSCML requests
- SIP-FSCML responses are transported in a separate INFO message
- SIP-FSCML Is a request-response protocol; with only final responses
- SIP-FSCML relies on SIP subscribe/notify mechanism, to allow applications subscribe to floor control related events

SIP Floor Server Control Markup Language

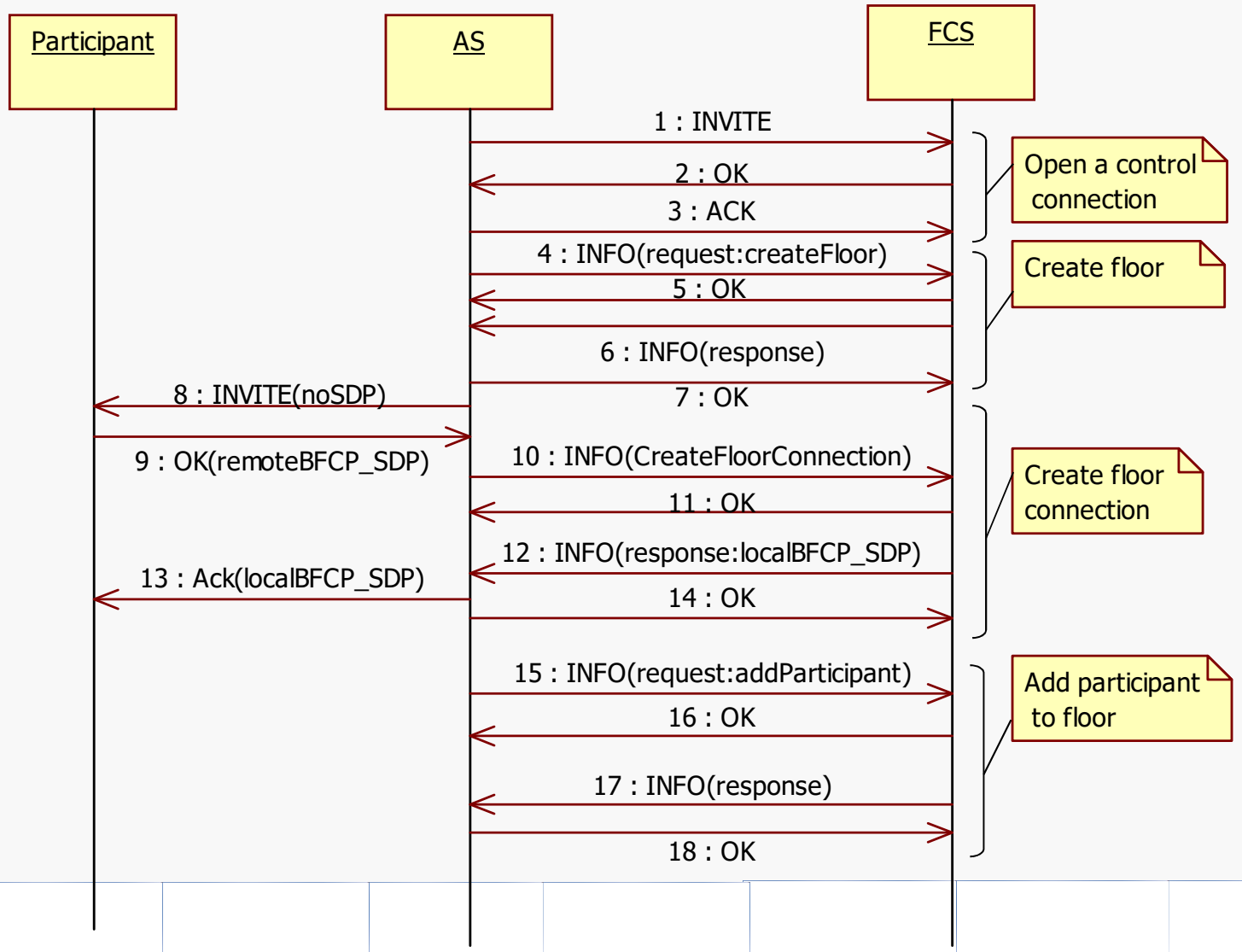
- SIP-FSCML operations
  - Open/close control connection
  - Create floor
  - Create floor Connection
  - Add/remove floor to/from a conference
  - Set/update Chair for a floor
  - Add/remove floor participant(s)
  - Set floor algorithm
  - add/remove media to/from a floor
  - Set maximum floor holders
  - Set maximum floor holding time

SIP Floor Server Control Markup Language

- Example of FSCML body

```
<FloorServerControl>
 <conferenceid>the conference ID</conferenceid>
 <request type=" CreateFloor">
   <floorid>the floor ID</floorid> (mandatory)
   <algorithm>the floor control algorithm</algorithm> (mandatory)
   <maxholders>max number of floor holders </maxholders>(optional default=1)
   <maxholdingtime>max time (in seconds) a participant can hold a floor, in case someone else
                   asked for it          </maxholdingtime>(optional, default 0=unlimited)
 </request>
 <request type=" SetChair">
   <floorid>floor whose chair should be set</floorid> (mandatory)
    <chairid>the chair ID</chairid>
 </request>
 <request type=" AddParticipant">
   <floorid>id of the floor to which to add</floorid> (mandatory)
   <participantid>the participant ID</ participantid > (mandatory)
 </request>
</ FloorServerControl>
```

SIP FSCML- A Scenario



Questions