SIP for Telephony

…yet another set of SIP services…

“Supplementary Services”

- The easy ones first
  - Call Diversion
    - Intrinsic support (301/302 redirection)
  - Call Forwarding (unconditional, busy, no answer)
    - Performed by proxy and indicated by means of 181 Call Is Being Forwarded response
  - Call screening (for incoming and outgoing calls) at proxy
    - Respond with 403 Screening Failure when needed
  - Call Waiting
    - Implemented in endpoints
- More sophisticated SIP signaling required for
  - 3rd party call control
  - Call transfer
  - Call park and pickup
  - Message waiting
  - PSTN (ISUP) and PBX (QSIG) interworking
  - Conferencing
Third-Party Call Control (3PCC)

- Examples: “Click-to-dial”, conference bridge control, …
- Several approaches with different advantages / drawbacks
- Simplest call flow:

1. \( \text{INVITE} \) no SDP
2. OK SDP(A)
3. \( \text{INVITE} \) SDP(A)
4. OK SDP(B)
5. ACK SDP(B)
6. RTP media
7. \( \text{ACK} \)

Call Hold and Retrieve

- Call Hold and Retrieve
  - Media-level operation only (does not affect SIP call state!)
  - User agent sends re-INVITE to mute other party (a=inactive)
    - Repeated offer/answer exchange
    - “Hold” condition may apply into one direction only
    - a=sendonly indicates that the initiator will not pay attention to incoming media
  - Another round of re-INVITEs to re-establish media (a=sendrecv)

- Call Hold with Music on Hold
  - Create a separate dialog with music server
  - Connect the peer to the music server by forwarding SDP
    - Direct music stream to the peer on hold
    - Actually: UA putting peer on hold acts as third-party call controller
  - When taking user off hold, close the dialog to music server
    - Redirect peer to talk to local UA again
Call Hold (with Consultation)

A
- Active SIP dialog
  - INVITE
  - m=audio ...
  - a=sendonly
  - 200 OK
  - m=audio ...
  - a=sendonly

B
- No RTP (but RTCP)

C
- INVITE
- 100 Trying
- 200 OK
- RTP stream
- BYE
- 200 OK

RTP stream

Call Hold with Music on Hold

A
- Active SIP dialog
  - RTP stream
  - INVITE (SDP_MS)
  - 200 OK (SDP_A)
  - ACK
  - RTP stream (music)
  - INVITE (SDP_B)
  - 200 OK (SDP_A)
  - ACK

B

C
- Music Server
  - INVITE (Ø)
  - 100 Trying
  - 200 OK (SDP_MS)
  - ACK (SDP_A)
  - BYE
  - 200 OK
Call Transfer

- Part of Call Control Framework

- Uses basic SIP protocol features and extensions
  - REFER method to invoke another (INVITE) transaction
  - NOTIFY (with implicit subscription) to indicate success or failure
  - New Replaces: header to indicate substitution of an existing call
  - Refer-To: header to indicate to peer which target to contact
  - Referred-By: header to inform refer target about origin of call

- Supports numerous variants
  - Attended
  - Unattended
  - Intermediate three-way calling
  - Optional protection of transfer target

Simple Unattended Transfer

```
<table>
<thead>
<tr>
<th></th>
<th>Active SIP dialog</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>B</td>
</tr>
<tr>
<td></td>
<td>On-hold (INV/200 OK/ACK)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>REFER</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Refer-To: sip:<a href="mailto:c@tzi.org">c@tzi.org</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>202 Accepted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NOTIFY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Content-Type: message/sipfrag</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Content-Length: 6</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>INVITE</td>
</tr>
<tr>
<td></td>
<td>Referred-By: <a href="mailto:a@tzi.org">a@tzi.org</a></td>
</tr>
<tr>
<td></td>
<td>100 Trying</td>
</tr>
<tr>
<td></td>
<td>200 OK</td>
</tr>
<tr>
<td></td>
<td>ACK</td>
</tr>
<tr>
<td></td>
<td>200 OK</td>
</tr>
<tr>
<td></td>
<td>200 OK</td>
</tr>
<tr>
<td></td>
<td>BYE</td>
</tr>
<tr>
<td></td>
<td>200 OK</td>
</tr>
</tbody>
</table>
```
Call Park & Pickup

- Park a call from one endpoint, pick up from another
  - Requires remote URI and session identification
  - Park on the endpoint itself
  - Park elsewhere (refer to some “parking area”, e.g., a SIP server)
  - For pickup, endpoint contacts remote URI and provides session identification
    - Use Replace: header to substitute (early) dialog
    - Pickup UA needs to learn about dialog-specific information
      - Use SUBSCRIBE/NOTIFY with dialog event package
      - Other alternatives for shared state conceivable, too

- Pick up a call destined for another phone
  - Common PBX feature, e.g., in working groups

Call Pickup

```
A
  INVITE (AB-id)
  100 Trying
  180 Ringing

B
  INVITE (Replaces: AB-id)
  200 OK
  200 OK
  CANCEL (AB-id)
  200 OK
  487 Req. Term.

C
  SUBSCRIBE
  200 OK
  NOTIFY (AB-id)
  200 OK
  NOTIFY (AB-id)
  481 Does not exist
  RTP stream
```
Dialog Event Package

- Event package for INVITE-initiated dialogs ("dialog") at endpoints
  - Entities authenticated with same AoR as the endpoint
- XML-based extensible format

Parameters
- call-id:, from-tag:, to-tag: Uniquely identifies a SIP dialog
- with-sessd:

Contained elements
- State ("early", "trying", "proceeding", "terminated", "confirmed")
  - possibly with event and reason parameter
- Duration (since creation of the dialog state machine)
- Referred-By and Replaces
- Peers of the dialog ("local" and "remote")
  - Identity, target (SIP URI), SDP session description

NOTIFY message body MIME type: application/dialog-info+xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:ietf:params:xml:ns:dialog-info"
version="1" states="full">
  <dialog id="123456">
    <state>confirmed</state>
    <duration>274</duration>
    <local>
      <identity display="Alice">sip:alice@example.com</identity>
      <target uri="sip:alice@pc33.example.com">
        <param pname="isfocus" pval="true"/>
        <param pname="class" pval="personal"/>
      </target>
    </local>
    <remote>
      <identity display="Bob">sip:bob@example.org</identity>
      <target uri="sip:bobster@phone21.example.org"/>
    </remote>
  </dialog>
</dialog-info>
```
Further Supplementary Services

- Automatic redial ("call completion on busy subscriber")
  - Again: use dialog event package
  - Calling UA subscribes to dialog state of called party
  - Caller is notified when the call state at the called party changes
  - Calling UA can automatically re-invoke call setup (i.e., send INVITE)

- Click-to-dial
  - Server-side implementation
    - Client needs to provide its own URI and is then called back
    - E.g., by means of third party call control on server side
  - Client-side implementation
    - Plug-in for web browser (linked to "sip:" and "sips:" as URI schemes)
    - Configured with local IP phone (or 3PCC server)
    - Sends REFER message to IP phone with target SIP URI in Refer-To: header
    - May require explicit user confirmation (risk of malicious "dialer" pages otherwise)

Click to Dial

fork() &
execve()
Message Waiting Indication

- Asynchronously notify endpoint(s) about messages
  - Voice, fax, pager, multimedia, text (per RFC 3458)
- Defines a new SIP event package
  - Subscribe to one or more mailboxes
  - Results from many sources may be merged
- Content-Type: application/simple-message-summary
  - General indicator for new messages
  - Message type followed by new/old and (new-urgent/old-urgent)
  - Encoding as plain text
  - May include selected RFC 2822 message headers

Example

```
Messages-Waiting: yes
Message-Account: jo@tzi.org
Voice-Message: 4/8 (1/2)
Text-Message: 238/42116 (0/1)
```

Interfacing to the PSTN

1. PSTN — PSTN
   - SIP Proxy
   - Border element maps SS7 to SIP and vice versa

2. SIP — PSTN
Interfacing to the PSTN

- Preserve feature transparency
  - transport SS7 information (ISUP MIME type)
    - Conversion between different ISUP versions to be done on gateways
- Provide enough routing information to find callee
  - (partially) translate ISUP to SIP
    - Support for tel:-URLs to indicate Called Party Number
      - Address resolution using ENUM or TRIP
- Enable early media (e.g., in-band alerting and announcements)
- Convey additional information during call
  - PSTN – SIP – PSTN case
    - INFO method (RFC 2976)
- SIP to PSTN mapping
  - Call flows for basic interoperation (RFC 3372, 3398, 3666)
  - RFC 3578: Support for Overlap Sending

Basic SIP-ISUP Mapping

For exclusive use with TKK Netlab course S-38.3152 Networked Multimedia Protocols and Services
Example for SIP and Overlap Sending (1)

- Approach 1: Collect digits in gateway
  - Use timeout
  - Collect minimal number of digits

Example for SIP and Overlap Sending (2)

- Approach 2: Send multiple INVITEs
  - Use timeout, possibly collect minimal number of digits
  - Wait for feedback from the SIP network
  - CANCEL obsolete INVITEs
INFO Method

- Transmit application-layer information during call (RFC 2976)
  - Use SIP signaling path of current session
  - Information is carried in message headers or body
    - Specific MIME types defined for, e.g., ISUP (application/isup)
  - No change of (SIP-related) call state

---

183 Session Progress Message

- INFO not applicable before call is established
- ISUP mapping requires inband data prior to final response

→ provisional response 183

- Additional information in message body
  - SDP description of early media stream (e.g., for announcements)
- Message headers also conceivable

- May be made reliable (100rel)
  - So that sender can count on synchronized state
  - PRACK message to confirm receipt of 183
Early Media Support

- **Early media** during call setup (SIP INVITE)
  - One-way transmission to report progress
  - Announcements, specific dial tones, …
  - No charge
  - Inhibit local alerting at calling user agent

- **Problem: Media negotiation**
  - Send SDP message in provisional response → *fast setup*
  - Create SDP from initial INVITE’s capability set → What if not suitable for early media session?
  - Calling UA will establish *early* media session → Cannot decline session or change codecs
  - UPDATE may be used for modifying media session

SIP-PSTN Call With In-band Alerting

<table>
<thead>
<tr>
<th>SIP UA</th>
<th>SIP Proxy</th>
<th>SIP—PSTN Gateway</th>
<th>PSTN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INVITE</td>
<td>INVITE</td>
<td></td>
<td>IAM</td>
</tr>
<tr>
<td>100 Trying</td>
<td>100 Trying</td>
<td>100 Trying</td>
<td>ACM</td>
</tr>
<tr>
<td>183 Progress</td>
<td>183 Progress</td>
<td>Audio</td>
<td>ANM</td>
</tr>
<tr>
<td>Δ</td>
<td>RTP</td>
<td>Audio</td>
<td></td>
</tr>
<tr>
<td>200 OK</td>
<td>200 OK</td>
<td>Audio</td>
<td></td>
</tr>
<tr>
<td>Δ</td>
<td>ACK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Δ</td>
<td>⬇ RTP</td>
<td>Audio</td>
<td></td>
</tr>
<tr>
<td>Δ</td>
<td>ACK</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For exclusive use with TKK Netlab course S-38.3152 Networked Multimedia Protocols and Services
SIP Interworking with QSIG (ISDN PBXes)

- Interconnecting a SIP-based signaling environment with a traditional (ISDN) PBX
  - Relevant for stepwise migration from traditional to IP telephony
  - First, create SIP islands connected to the PBX
    - Old PBX runs as "master"
  - Then, switch trunk from old PBX to (IP-based) SIP server
    - And run old PBXes as "slaves"
  - Finally, throw away PBX and old phones
- Gateway function needed for semantic mapping between SIP and QSIG
  - Call signaling (stateful operation)
  - Numbering plans
  - Support for "overlap sending"
- Naturally limits available functionality

Gateway function needed for semantic mapping between SIP and QSIG

- Call signaling (stateful operation)
- Numbering plans
- Support for "overlap sending"
- Naturally limits available functionality

SIP-QSIG: Call Setup (en-bloc dialing)

- SETUP (+358…2460)
- CALL PROCEEDING
- ALERTING
- CONNECT
- CONNECT ACK
- DISCONNECT
- RELEASE
- RELEASE COMP.
- INVITE sip:+358…2460@
- 100 Trying
- 180 Ringing
- PRACK
- 200 OK
- 200 OK
- ACK
- BYE
- 200 OK

For exclusive use with TKK Netlab course S-38.3152 Networked Multimedia Protocols and Services
SIP-QSIG: Call Setup (overlap sending 1)

PISN
- SETUP (+358)
- SETUP ACK
- INFORMATION (…)
- INFORMATION (2460)
- CALL PROCEEDING
- ALERTING
- CONNECT
- CONNECT ACK

Audio

Gateway
- INVITE sip:+358...2460@
- 100 Trying
- 180 Ringing
- PRACK
- 200 OK
- 200 OK
- CONNECT
- CONNECT ACK
- ACK

Audio

SIP

SIP-QSIG: Call Setup (overlap sending 2)

PISN
- SETUP (+358)
- SETUP ACK
- INFORMATION (…)
- INFORMATION (2460)
- CALL PROCEEDING
- ALERTING
- CONNECT
- CONNECT ACK

Audio

Gateway
- INVITE sip:+358...@...
- 484
- ACK
- INVITE sip:+358...2460@...
- 100 Trying
- 180 Ringing
- PRACK
- 200 OK
- 200 OK

Audio

SIP

For exclusive use with TKK Netlab course S-38.3152 Networked Multimedia Protocols and Services
Quality of Service

- Best-effort quality potentially too poor for IP telephony
  → need support for resource reservation in SIP

- Some applications need multi-phase call-setup
  - Resource reservations to assure certain QoS
  - Establish particular security relationships
    - Extend security services to media channels
    - Avoid fraud and theft-of-service

→ Establish call only if preconditions are met
  - No charge for defect calls
  - No inconsistent state of involved endpoints
    (alert user only if resource reservation succeeded, …)
  - Fallback options: e.g. use codecs with lower bandwidth

SIP Preconditions (RFC 3312)

- SDP extension handles negotiation of QoS parameters
  - SDP attributes describe preconditions to be met
  - Current status (curr:) vs. desired status (des:)
  - Either party may request confirmation on current state (conf:)

'a=curr:' type status direction
'a=des:' type status strength direction
'a=conf:' type status direction

- Offer/answer to create common view on both UAs state
  - Typically INVITE, UPDATE, PRACK
  - Option tag precondition
  - New response code 580 Precondition Failure
  - Mark failed and unknown preconditions in SDP answer
QoS Parameter Negotiation (Overview)

INVITE

prereq: caller

183 Session Progress
preconditions
confirmation request

PRACK

200 OK (PRACK)

Resource reservation

UPDATE

prereq: callee

200 OK (UPDATE)

180 Ringing

Caller sends state description
Update desired state.
Need caller to reserve in send direction → confirmation request

Acknowledge answer
Independent reservations made

Confirmation from caller after successful resource reservation
Callee signals successful reservation. User is being alerted.

Example Negotiation of QoS Parameters

Confirmation request

v=0
o=jo 7849 2873246 IN IP4 ruin.inf...
s=SIP call
t=0 0
c=IN IP4 134.102.218.1
m=audio 52392 RTP/AVP 98 99
a=rtpmap:98 L8/8000
a=rtpmap:99 L16/8000
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
m=video 59485 RTP/AVP 31
a=rtpmap:31 H261/90000
a=curr:qos local send
a=curr:qos remote none
a=des:qos optional local sendrecv
a=des:qos optional remote sendrecv

Confirmation response

v=0
o=cabo 2552 892834 IN IP4 dmn.inf...
s=SIP call
t=0 0
c=IN IP4 134.102.218.46
m=audio 50239 RTP/AVP 98 99
a=rtpmap:98 L8/8000
a=rtpmap:99 L16/8000
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
a=conf:qos e2e recv
m=video 56112 RTP/AVP 31
a=rtpmap:31 H261/90000
a=curr:qos local send
a=curr:qos remote send
a=des:qos failure local sendrecv
a=des:qos optional remote sendrecv
Successful Reservation

- INVITE indicates support for reliable provisional responses
- Turn off local user alerting
- Choose appropriate subset of preconditions offered by sender
- Include confirmation request

- Reliable provisional response must be acknowledged

- Initiator of confirmation request awaits UPDATE message or CANCEL
- Continue call setup

Failed Reservation

- INVITE contains preconditions with confirmation request
- Indicate support for reliable provisional responses

- Turn off local user alerting
- No reservation results

- Reliable provisional response must be acknowledged

- Callee's resource reservation fails
- Send error response to caller
- Failed call setup complete
Other Preconditions

- **Security ("sec")**
  - Users do not want to send unprotected RTP over the wire
  - Negotiate a (set of) crypto algorithm(s) for the media streams and exchange keying material (e.g., using MIKEY)
  - DO NOT ring the phone before the security association is established and data can be encrypted and decrypted
    - Precondition helps to avoid clipping (if one party answers the phone too quickly)

- **Connectivity preconditions ("conn")**
  - Endpoints want to ensure mutual reachability for media streams
  - UDP packets
    - DO NOT ring the phone before ICE connectivity checks have passed
  - TCP / SCTP connection establishment
    - DO NOT ring the phone before

SIP Conferencing

- **Motivators:** n-way calling, video conferencing, …
  - Tightly coupled conferences

- **Different conferencing models preserved**
  - Except for “fully meshed” conference: complexity just not worth it!

- **Terminology**
  - Trying to avoid already overloaded terms as much as possible

- **Set of protocols**
  - Definition of basic building blocks (SIP and other)
  - Sample combinations to implement conferencing services

- **Focus on basic SIP model**
  - Additional developments in XCON currently in progress
SIP Conferencing

- MUST work with basic SIP support only
  - No awareness of conference
  - Just a point-to-point call with minimal means for control
  - Possibly augmented by out-of-band control (e.g. HTTP)

- Member types (SIP UA)
  - Conference-unaware: plain-old SIP device
  - Conference-aware: supports conferencing features
  - Focus: (one) center of a conference
  - Anonymous: Visible but unidentified participant
  - Invisible: Participant whose presence is not known

Conferences and URIs

- Conference types
  - Basic: just plain SIP, no further means for control
  - Complex: some conferencing features provided
  - Cascaded: several foci concatenated in a conference
  - Sidebar: conference as (logical) part of another

- Focus
  - Signaling center of a conference

- Conference URI
  - Identifies a focus
  - (isFocus parameter may indicate this)

- Factory URI
  - for automated conference creation
    - Yields a dynamically generated conference URI in return
Terminology and Model

Conference Policy Control Protocol
CPCP

Binary Floor Control Protocol
BFCP

SIP Dialog

SIP Subscription
e.g. RTP Sessions

Conference Policy Server

Floor Control Server

Focus
Conference Notif. Service

Member-Ship & Media Policy

Participant
Mixer

Terminology and Model

Conference Policy Control Protocol
CPCP

Binary Floor Control Protocol
BFCP

SIP Dialog

SIP Subscription
e.g. RTP Sessions

Conference Policy Server

Floor Control Server

Focus
Conference Notif. Service

Member-Ship & Media Policy

Participant
Mixer
SIP Signaling Relationships

(RTP) Media Sessions

For exclusive use with TKK Netlab course S-38.3152 Networked Multimedia Protocols and Services
Conferencing Scenarios (1)

- Simple conferencing scenarios
  - Plain SIP only (RFC 3261, 3264)
- Extend point-to-point call
  - Works only with local focus; otherwise, new call required
- Ad-hoc conference
  - Automated creation at focus
  - IVR / DTMF for control
  - Audio for information about the conference and its members
- Reserved conference
  - Same as ad-hoc
  - Use external means for reservation and configuration (e.g. web)

Conferencing Scenarios (2)

- Advanced conferencing scenarios:
  - Support for Call Transfer
  - Support means to communicate information from focus to UA
  - Optional: means to manipulate conference and media policy
- Extend point-to-point call
- Join / create a conference based upon an existing dialog
- Ad-hoc conference
- Reserved conference
- Make use of additional conferencing features
Sample Conferencing Features

- Invite participants (dial-in, dial-out), expel participants
- Authenticate new participants by members
- Obtain conference and media policy information
- Manipulate membership/conference policy
  - Participant privileges, participant management (black list, white list)
  - Floor control
- Explicit media control (media policy)
  - Configure media distribution
  - Add / remove media sessions
- Floor control
  - Manage access to conference resources
  - Executed by floor control protocol, governed by floor control policy
- Create, control, and terminate sidebars
  - Separate conference vs. media policy

Membership Policy

Define, retrieve/notify, modify, and act upon...

- Formal rules for the conference
  - Conference creation, termination, (policy) modification
  - Access control: black list, white list, rules for authentication
  - Privileges of individual participants
  - Visibility of the conference and its members
  - Access to floor and media policy (defined separately)
- General conference attributes
- Participation management
  - Invite, expel
- ...

For exclusive use with TKK Netlab course S-38.3152 Networked Multimedia Protocols and Services
Media Policy

- **Mixer model**
  - Input switch: collecting & selecting input streams from participants
    - Possibly transcoding and other per-media functions
  - Mixing topologies describing *mixing policies*
  - Output switch: selecting & distributing output streams to participants
    - Possibly transcoding and other per-media functions
- **Mixer may be centralized or not**
- **Media policy defines how incoming streams are processed, combined, and then distributed**
  - Individual mixing functions may be defined per participant
  - Common mixing functions may be defined for the conference
  - Mixing function may take into account “events” from other components

SIP Signaling Building Blocks

- **Membership control**
  - Initiation of conferences: INVITE
  - Inviting / adding to conferences: INVITE, REFER
  - Leaving a conference: BYE
  - Expelling from a conference: REFER (method="BYE")

- **Conference control**
  - State change notifications: SUBSCRIBE / NOTIFY
    - Dialog event package, conference event package, ...
  - Conference / media policy control
    - Might use XCAP data manipulation framework discussed in SIMPLE context

- **Other**
  - Determine focus URI OPTIONS

- **Augmented by other protocols**
  - Floor control: BFCP
  - HTTP/HTML access to a web page for human interaction
    - Conference control, reservation, group management, etc.
Conferencing Event Package (1)

- Event package for notifying UAs about conference state
  - XML-based encoding of conference status
  - Goes together with dialog state
- Allows minimally enhanced SIP UAs to learn about
  - General conference information (focus URI, state, version)
  - Description (display text, subject, free text, keywords)
    - Conference URIs (for participation and streaming)
    - Service URIs (web page, conference recording, event subscription)
    - Maximum user count
    - Available media (display text, type, status)
  - Host information (display text, lifetime, web page, URIs)
  - Conference state (user count, active, locked)
  - Users (entity == user URI, state)
    - Display text, Associated AoRs, roles, languages, cascaded focus vs. endpoint
    - Endpoint: further detailed status information
- Notification only: other mechanisms for conf. state manipulations

Conferencing Event Package (2)

```
<conference-info version="1" state="partial"
    entity="sip:3402934234@conf.example.com">
  <users>
    <user entity="sip:carol@chicago.example.com" state="full">
      <display-text>Carol</display-text>
      <endpoint entity="sip:carol@client.chicago.example.com">
        <status>connected</status>
        <joining-method>dialed-out</joining-method>
        <media id="1">
          <display-text>Main Audio</display-text>
          <type>audio</type>
          <src-id>583398</src-id>
          <status>sendrecv</status>
        </media>
        <media id="2">
          <type>video</type>
          <src-id>345212</src-id>
          <status>sendrecv</status>
        </media>
      </endpoint>
    </user>
  </users>
</conference-info>
```
Call Flow Example: Conference Creation

INVITE sip:conf-factory
180 Ringing
200 OK Contact:conf-id
ACK
Media sessions
SUBSCRIBE sip:conf-id
200 OK
NOTIFY
200 OK

Call Flow Example: User Joining

Alice
Media sessions
NOTIFY
200 OK
Media sessions
Bob
INVITE sip:conf-id
180 Ringing
200 OK Contact:conf-id
ACK
Carol
Media sessions
SUBSCRIBE sip:conf-id
200 OK
NOTIFY
200 OK
Call Flow Example: Adding a User

Alice  
Media sessions  
REFER sip:conf-id  
Refer-To: carol

INVITE Contact:conf-id

Focus  
Media sessions

202 Accepted
NOTIFY
200 OK
NOTIFY
200 OK

Bob  
Media sessions

Carol  
Media sessions

SUBSCRIBE sip:conf-id  
200 OK
NOTIFY  
200 OK

Call Flow Example: Removing a User

Alice  
Media sessions  
REFER sip:conf-id  
Refer-To: carol?m=BYE

BYE sip:carol

Focus  
Media sessions

202 Accepted
NOTIFY
200 OK

Bob  
Media sessions

Carol  
Media sessions

NOTIFY
200 OK
NOTIFY
200 OK
### Call Flow Example: User Leaving

```
Alice  Focus  Bob  Carol
Media sessions <-> Media sessions

BYE sip:conf-id
200 OK
ACK

NOTIFY
200 OK

Media sessions

Terminate conference
e.g. by non-SIP means
```

### Binary Floor Control Protocol (BFCP)

- Support access control to “floors” aka. media sessions
  - Session identification via SDP attributes
  - Shared or exclusive access (equivalent to read or write lock)
    - Limited number of simultaneous floor holders
  - Clients (=participants) issue floor requests or release the floor
    - Requests identify floor(s) concerned and may include hint message
  - Server manages incoming requests and maintain floor state
    - Queuing of requests possible
    - Notifies clients about state changes (granting, revoking floor, floor holder)
    - Governed by some conference policy rules (beyond BFCP scope)
  - Floor chair(s)
    - Manage (grant, revoke, ...) floors
    - One or more chairs per floor – one chair for all floors

- Protocol runs on top of TCP or TLS
  - Negotiated via SDP offer/answer (connection-oriented media)
  - Binary TLV encoding
  - Request–response protocol + asynchronous notifications
  - Transaction identifiers for individual operations
BFCP Overview

The floor chair may also be co-located with the floor control server.
The floor chair may be an automaton executing a predefined policy.

BFCP and SDP

- BFCP negotiation: another media stream
  - Determine connection setup parameters and the floor control server
    - Uses m=application (a=control is not a top-level MIME type)
    - a=floorctrl:c-s (c-only, s-only)
  - Media session identification for floor control
  - Media level attributes for floor control
  - Refer to SDP label attribute at media level

```plaintext
m=application 50000 TCP/TLS/BFCP *
a=setup:passive
a=connection:new
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
```
BFCP Example (1)

Participant 1

FloorRequest
Transaction ID: 4711
User ID: 1234
Floor ID: 1

FloorRequestStatus
Transaction ID: 4711
User ID: 1234

FloorRequestInformation
Transaction ID: 4711
User ID: 1234
Floor ID: 1
Status: pending, 1st in queue

FloorRequestStatus
... FloorRequestInformation
Transaction ID: 4711
User ID: 1234
Floor ID: 1
Status: granted

Participant 2

FloorControl
Server

FloorChair

ChairAction
Transaction-ID: 943
User ID: 1234, Floor ID: 1
FloorRequest ID: 4711
Status: granted

ChairActionACK ...

FloorRequestStatus
... FloorRequestInformation
Transaction ID: 4711
User ID: 1234
Floor ID: 1
Status: granted

FloorRequestStatus
Transaction ID: 0
User ID: 1234

FloorRequestInformation
Transaction ID: 4711
User ID: 1234
Floor ID: 1
Status: pending, 1st in queue

BFCP Example (2)

Participant 1

FloorQuery
Transaction ID: 99
Bene ID: 314
Floor ID: 1

Participant 2

FloorControl
Server

FloorChair

FloorStatus
Transaction ID: 99
User ID: 314

FloorRequestInformation
Beneficiary ID: 1234
Floor ID: 1
Status: granted

ChairAction
Transaction-ID: 4711
User ID: 1234
Floor ID: 1
Status: pending, 1st in queue

ChairActionACK ...

For exclusive use with TKK Netlab course S-38.3152 Networked Multimedia Protocols and Services
Conference Policy Control Protocol

- In progress…
- Needs to address different aspects
  - Data model for conference and media policy, etc.
  - Protocol for manipulating this data
    - Minimal set of semantics-independent operations on a data structure or
    - Semantic operations that are cause data manipulation (as a side effect)
- Several proposals around
  - Conference State Change Protocol (CSCP)
    - Realized as extensions to the Binary Floor Control Protocol (BFCP)
  - Centralized Conference Control Protocol ("C3P")
    - XML-based protocol elements; may be carried over various transports (including SIP, SOAP, …)
  - COMP: Conference Object Manipulation Protocol
    - Object manipulation based upon web services + SIP events
  - CCMP: Centralized Conference Manipulation
    - SOAP-based
- Presently under evaluation

SIP & Multimedia

- Text-based conferencing
  - Supported by SDP media descriptions
  - Considerations exist for audio-to-text translation service (and vice versa)
- Video
  - SDP description readily available for negotiation
  - Conference and media policy gain weight ("who sees whom?")
  - Open issue: time-critical video control "commands"
  - Other remote access features (e.g., camera control) unresolved
- Application sharing (in the very beginning at this point)
  - Old ITU-T T.120: shared whiteboard, file transfer, and generic app sharing
    - Could be launched from within a SIP session
    - Example: Microsoft NetMeeting
  - GUI "remoting"
    - Newly formed Widget Description Exchange (WIDEX) WG in the IETF
  - Some ideas around RTP-based application sharing
    - Immature at this point
  - Other
    - VNC? X Window System-based approaches?
Remaining old fragments
Passing Session State Information

- HERFP: inconsistent information about session status
  - Need to tell initiating UAC before dialog state update
  - “Tunnel” 4xx error response in reliable provisional response

  → 155 Please Update

- Inquires session status update from UAC
  - Reason header (RFC 3326)
    - numeric reason code, SIP error code, SIP error phrase
  - UAC can send appropriate information in following request
    (e.g. update credentials, SDP capability set, ...)
  - Dialog state may not be affected

- Issue: when to send 155?
  - UAC/user interaction vital
  - Establishing security associations, early media sessions, ...

155 Example: Heterogeneous Error Responses

UAC sends credentials
UAS A alerts user
155 Example: Heterogeneous Error Responses
UAS Proxy UAS A UAS B
INVITE INVITE INVITE
155 155 200 (COMET) 180 (INVITE) 180 (INVITE)
200 (INVITE) 200 (INVITE) 200 (INVITE) 180 (INVITE)
CANCEL 487 (INVITE) ACK ACK

Proxy forks, UAS A requests authentication
SIP/2.0 155 Please Update
Reason: 7;
triggered=SIP/2.0;
cause=401;
text="Unauthorized"
WWW-Authenticate: ...

UPDATE sip:.... SIP/2.0
Authorization: ...

Proxy cancels second branch
Transaction complete

UAS B alerts user