Centralized Conferencing using SIP

Kundan Singh, Gautam Nair, Henning Schulzrinne
Dept. of Computer Science
Columbia University
New York, New York
{kns10,gnair,schulzrinne}@cs.columbia.edu

IP telephony workshop – Columbia University, New York
April 2, 2001
Overview

- conferencing models
- centralized conferencing server
- design issues
- measurement results
Conference models: end system mixing

A+B+C

A+B+D

A+C+D

April 2001
Conference models: multicast transmit & receive
Conference models: multicast receive, unicast transmit

Diagram:

1. Connection from A to B
2. Connection from B to C
3. Connection from D to RTP translator

RTP translator

Nodes A, B, C, D
Conference models: central server

can be call-out or dial-in

\[ \Sigma \]

A + B

C

A + C

B + C

SIP

A + B

RTP

A

B

C
Conference models: full mesh

A invites B
A invites C
C invites B
C invites D
D invites A, B
Conference models – complexity

$I$ active senders, $N$ participants

<table>
<thead>
<tr>
<th>Properties</th>
<th>central</th>
<th>full mesh</th>
<th>mcast</th>
<th>uni rx, mcast tx</th>
<th>end mixing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Star</td>
<td>full mesh</td>
<td>mcast tree</td>
<td>star+mcast tree</td>
<td>ad-hoc</td>
</tr>
<tr>
<td>Server proc.</td>
<td>$O(M + N)$</td>
<td>n/a</td>
<td>n/a</td>
<td>$O(M + N)$</td>
<td>n/a</td>
</tr>
<tr>
<td>Endpoint proc.</td>
<td>$O(1)$</td>
<td>$O(M)$</td>
<td>$O(M)$</td>
<td>$O(1)$</td>
<td>variable</td>
</tr>
<tr>
<td>Server bw</td>
<td>$O(M + N)$</td>
<td>n/a</td>
<td>n/a</td>
<td>$O(M)$</td>
<td>n/a</td>
</tr>
<tr>
<td>Endpoint bw</td>
<td>$O(1)$</td>
<td>$O(M)$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>variable</td>
</tr>
<tr>
<td>Scaling</td>
<td>medium</td>
<td>medium</td>
<td>large</td>
<td>large</td>
<td>medium</td>
</tr>
<tr>
<td>Heterogen. UA</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes (partially)</td>
</tr>
<tr>
<td>Own media?</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>
Central server

- conference identified by SIP URL, e.g., staffmeet@cs.columbia.edu
- simple end points
- centralized control
- ad-hoc conferences useful for three-party calls
- can create ad-hoc conference, e.g., sip:letsmeet-adhoc@a.servers.com, use REFER to get others to add themselves to that conference
- for ad-hoc, conference lasts until last one leaves
Columbia sipconf conferencing server

- central-server model for mixing
- mixes audio streams, replicates RTP (and UDP) streams
- audio (G.711, DVI, GSM) - others can be easily added
- works for video and text chat - packet replication
Mixing heterogeneous streams

\[ X = A + B + C \]

\[ X - A = B + C \]

\[ X - B = E \]$\rightarrow$ Send to A

\[ G.711 \text{ Mu} \]

\[ DVI \]

\[ GSM \]

\[ E = \text{Audio Encoder} \]

\[ D = \text{Audio Decoder} \]

April 2001
**Example: Columbia software conference server**

<table>
<thead>
<tr>
<th>Conference URL (Click to edit)</th>
<th>Description</th>
<th>Duration</th>
<th>Participants</th>
<th>Media</th>
</tr>
</thead>
<tbody>
<tr>
<td>demo</td>
<td>Demonstration of sipconf audio conference server.</td>
<td>always on</td>
<td>anyone can join</td>
<td>audio</td>
</tr>
<tr>
<td>test</td>
<td>Testing</td>
<td>always on</td>
<td>Maximum 10 participants allowed</td>
<td>audio video</td>
</tr>
</tbody>
</table>

Click [here](#) to setup a new conference.
Example: Columbia software conference server

![Conference Configuration Screen]

- **Conference URL:** demo
- **Password:** leave blank
- **Description:** Demonstration of sipconf audio
- **Start time:** 0000-00-00 00:00
- **End time:** 9999-12-31 00:00
- **Authentication:** none
- **Conference Type:** public
- **Participant List Type:** public
- **Max number of participants:** unlimited
- **Supported media type:** audio, video, chat

*Create New, Update Existing, or Delete Existing*
Example: Columbia software conference server
Design and implementation issues

- Packetization time
- Scalability
  - server farm
  - multi-stage servers
  - dedicated hardware
- inactivity detection
- multi-protocol server
## Performance for single conference

PARC Ultra 10 with 350 MHz CPU:

<table>
<thead>
<tr>
<th>Participants</th>
<th>CPU (%)</th>
<th>memory (MB)</th>
<th>bandwidth (Mb/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>inbound</td>
<td>outbound</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>&lt; 0.1</td>
<td>2.7</td>
<td>0.08</td>
</tr>
<tr>
<td>20</td>
<td>&lt; 1</td>
<td>6.0</td>
<td>0.08</td>
</tr>
<tr>
<td>40</td>
<td>2-3</td>
<td>9.6</td>
<td>0.08</td>
</tr>
<tr>
<td>60</td>
<td>5</td>
<td>13</td>
<td>0.08</td>
</tr>
<tr>
<td>80</td>
<td>10-15</td>
<td>17</td>
<td>0.08</td>
</tr>
<tr>
<td>100</td>
<td>35-50</td>
<td>22</td>
<td>0.08</td>
</tr>
<tr>
<td>120</td>
<td>50-70</td>
<td>26</td>
<td>0.08</td>
</tr>
</tbody>
</table>

April 2001
Performance for three-party conferences

Good quality up to 15 conferences

<table>
<thead>
<tr>
<th>Conferences</th>
<th>Participants</th>
<th>CPU (%)</th>
<th>Memory (MB)</th>
<th>Bandwidth (Mb/s) inbound</th>
<th>Bandwidth (Mb/s) outbound</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>9</td>
<td>&lt; 0.4</td>
<td>4.1</td>
<td>0.72</td>
<td>0.65</td>
</tr>
<tr>
<td>6</td>
<td>18</td>
<td>&lt; 2.0</td>
<td>5.7</td>
<td>1.44</td>
<td>1.30</td>
</tr>
<tr>
<td>9</td>
<td>27</td>
<td>7-13</td>
<td>7.3</td>
<td>2.16</td>
<td>1.94</td>
</tr>
<tr>
<td>12</td>
<td>36</td>
<td>15-20</td>
<td>9</td>
<td>2.88</td>
<td>2.60</td>
</tr>
<tr>
<td>15</td>
<td>45</td>
<td>25</td>
<td>10</td>
<td>3.60</td>
<td>3.24</td>
</tr>
<tr>
<td>18</td>
<td>54</td>
<td>30</td>
<td>12</td>
<td>4.32</td>
<td>3.89</td>
</tr>
</tbody>
</table>
Centralized conferencing - conclusion

- need many different models of conferencing
- as long as no multicast, central server is good for medium to large scale
- trade-off infrastructure vs. complexity vs. scaling
- can be handled by existing SIP mechanisms. Works with H.323 also.