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

\[ A + C + D \]

\[ A + B + C \]

\[ A + B + D \]
Conference models: multicast transmit & receive
Conference models: multicast receive, unicast transmit
Conference models: central server

Can be call-out or dial-in
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>

April 2001
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

\[ X = A + B + C \]

\[ X - A = B + C \]

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

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

G.711 Mu

DVI

GSM

Play-out delay

Periodic timer interrupt

Mixed Linear Stream

Send to A
G.711 Mu

Send to B
DVI

Send to C
G.711 Mu

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>restricted conference digest authentication required</td>
<td>video</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Maximum 10 participants allowed</td>
<td></td>
</tr>
</tbody>
</table>

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

[Form]
Conference URL: [url]
Password: [password]
Description: [description]
Start time: [start_time]
End time: [end_time]
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**

<table>
<thead>
<tr>
<th>Conference URL (click to edit)</th>
<th>Participant</th>
<th>Privileges</th>
<th>Type</th>
<th>Status</th>
<th>Audio</th>
<th>Video</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>test <a href="http://example.com:8080">http://example.com:8080</a></td>
<td>*</td>
<td>send, receive, admin</td>
<td></td>
<td>notconnected</td>
<td>send-receive</td>
<td>send-receive</td>
<td>Update</td>
</tr>
<tr>
<td>test <a href="http://example.com:8080">http://example.com:8080</a></td>
<td>*</td>
<td>send, receive, admin</td>
<td></td>
<td>notconnected</td>
<td>send-receive</td>
<td>send-receive</td>
<td>Update</td>
</tr>
<tr>
<td>test <a href="http://example.com:8080">http://example.com:8080</a></td>
<td>*</td>
<td>send, receive, admin</td>
<td></td>
<td>notconnected</td>
<td>send-receive</td>
<td>send-receive</td>
<td>Update</td>
</tr>
<tr>
<td>test <a href="http://example.com:8080">http://example.com:8080</a></td>
<td>*</td>
<td>send, receive, admin</td>
<td></td>
<td>notconnected</td>
<td>send-receive</td>
<td>send-receive</td>
<td>Update</td>
</tr>
</tbody>
</table>
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>inbound (Mb/s)</th>
<th>outbound (Mb/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>&lt; 0.1</td>
<td>2.7</td>
<td>0.08</td>
<td>0.07</td>
</tr>
<tr>
<td>20</td>
<td>&lt; 1</td>
<td>6.0</td>
<td>0.08</td>
<td>1.37</td>
</tr>
<tr>
<td>40</td>
<td>2-3</td>
<td>9.6</td>
<td>0.08</td>
<td>2.81</td>
</tr>
<tr>
<td>60</td>
<td>5</td>
<td>13</td>
<td>0.08</td>
<td>4.25</td>
</tr>
<tr>
<td>80</td>
<td>10-15</td>
<td>17</td>
<td>0.08</td>
<td>5.69</td>
</tr>
<tr>
<td>100</td>
<td>35-50</td>
<td>22</td>
<td>0.08</td>
<td>7.13</td>
</tr>
<tr>
<td>120</td>
<td>50-70</td>
<td>26</td>
<td>0.08</td>
<td>8.59</td>
</tr>
</tbody>
</table>

April 2001
Performance for three-party conferences

Good quality up to 15 conferences

<table>
<thead>
<tr>
<th>Conference 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>&lt; 0.4</td>
<td>4.1</td>
<td>0.72</td>
<td>0.65</td>
</tr>
<tr>
<td>6</td>
<td>&lt; 2.0</td>
<td>5.7</td>
<td>1.44</td>
<td>1.30</td>
</tr>
<tr>
<td>9</td>
<td>7-13</td>
<td>7.3</td>
<td>2.16</td>
<td>1.94</td>
</tr>
<tr>
<td>12</td>
<td>15-20</td>
<td>9</td>
<td>2.88</td>
<td>2.60</td>
</tr>
<tr>
<td>15</td>
<td>25</td>
<td>10</td>
<td>3.60</td>
<td>3.24</td>
</tr>
<tr>
<td>18</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.