Journal of Mobile Multimedia, Vol. 5, No. 1 (2009) 012–028

NARROWCASTING FOR ARTICULATED PRIVACY AND ATTENTION IN SIP AUDIO CONFERENCING

SABBIR ALAM
R& D Dept. , Mobile Technika Inc.
Shinjuku-ku, Tokyo 162-0845; Japan
E·mail: sabbir@mobiletechnika.jp

MICHAEL COHEN & JULIÁN VILLEGAS
Spatial Media Group, University of Aizu
Aizu-Wakamatsu, Fukushima-ken 965-8580;
E·mail: {mcohen, d8091104}@u-aizu.ac.jp

ASHIR AHMED
Dept. of CSCE, Kyushu University
744 Moto'oka, Fukuoka 819-0395; Japan
E·mail: ashir@c.csce.kyushu-u.ac.jp

Received June 2, 2008
Revised September 1, 2008

In traditional conferencing systems, participants have little or no privacy, as their voices are by default shared with all others in a session. Such systems cannot offer participants the options of muting and deafening other members. The concept of narrowcasting can be applied to make these kinds of filters available in multimedia conferencing systems. Our system treats media sinks (in the simplest case, listeners) as full citizens, peers of the media sources (conversants' voices), and we defined therefore duals of `mute` & `select` : `deafen` & `attend`, which respectively block a sink or focus on it to the exclusion of others. In this article, we describe our prototyped application, which uses existing standard Session Initiation Protocol (SIP) methods to control fine-grained narrowcasting sessions. The runtime system considers the policy configured by the participants and provides a policy evaluation algorithm for media mixing and delivery. We have integrated a "virtual reality"-style interface with this SIP backend to display and control articulated narrowcasting with figurative avatars.

Keywords: Narrowcasting, Privacy, SIP, Conferencing, Audio

1 Introduction and Related Research

In multimedia conferencing, media streams are exchanged between participants upon session establishment by setting up communication channels within a group. By default, each participant receives a combined stream obtained by mixing media transmitted by the other participants. Situations arise when a participant wants to select a subset of the conference participants to whom her media are sent or from whom streams are received. Media filters are necessary to allow privacy of the participants in the conference. In analogy to broad-, multi-, any-, and swarm-casting, narrowcasting is a technique for limiting and focusing information streams. Narrowcasting systems extend broad- and multicasting systems by allowing media streams to be filtered— for relevancy control, privacy, and user interface optimization. We describe four narrowcasting commands— `mute`, `deafen`, `select`,

and `attend`— to provide distributed privacy in SIP-based conferencing.

Extensive development has been carried out in the area of conference and floor control [12] [7]. Conventional features regarding media privacy in conferences are typically limited to scheduling and selecting the speaker. Advanced conferencing features such as adding/deleting participants, changing user agents or modes (like switching from a desktop to a mobile phone), changing media, authenticating or authorizing participants, granting privileges, controlling presentation of media, sidebars, passive participants, whisper/private messages, audio-only, and lecture mode are described in RFC 4597 [9]. Media privacy features allow participants to control their own information and to distribute their attention, based on secrecy, anonymity, and solitude [17].

A Call Whisper [4] feature allows a participant to talk privately to one or more participants in a group. This walkie-talkie-like feature creates a one-way voice or video communication for a limited period of time. The session terminates when the controller releases the "PTT" (push to talk) button, so such a system is not practical when a longer session or two-way communication session is necessary. Voice Chat [20] [11] allows participants to create one or more private audio conferences: although the communication channel in the private voice chat group provides two-way communication, participants can hear the main conference at low volume. Private conversation [19] offers a private video, voice, and text conversation session inside a main conference. It is similar to a Call Whisper feature, but adds two-way communication capability and text messaging. In a WebEx audio conference, a conference chairperson can selectively disable the microphones to allow only certain attendees to speak. An 'audio-only' option allows a moderator to revoke and restore speaking privileges to attendees, so that muted attendees can listen but not speak. WebEx participants can have a private chat with someone during a meeting. Whisper Coaching (`www.audiocodes.com`) allows a supervisor to listen to a main conference conversation while talking to a selected set of participants at the conference.

The privacy control allowed by these applications is rather blunt. In order to better control media privacy, we explore the concept of narrowcasting [10], design for prototyping in decentralized [2] and centralized [1] models with a collaborative virtual environment (CVE) [3]. In this article, we describe a mechanism and instance of "Media Server Component Model" architecture for policy-based media mixing with a centralized media mixer within the standard SIP [15] framework for multimedia conferencing systems. We have defined media privacy commands and developed a policy evaluation algorithm, a media mixing and delivery mechanism considering policy configured by conference participants. We deploy four narrowcasting commands— `mute`, `deafen`, `select`, and `attend`— to provide distributed privacy.

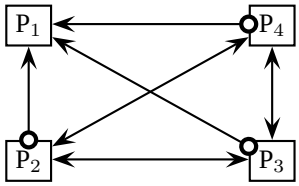
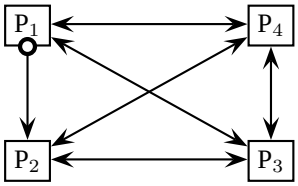
1.1 Media Privacy Control and Limitations

Mute is a popular feature for media privacy. It has three different varieties: self-mute, PBX-mute, and narrowcasting mute, juxtaposed in Table 1. Self-mute allows a user to withhold his media streams from other participants. In PBX-mute, a controller disables a participant's outgoing media to other participants. Narrowcasting mute refers to P2P control with which a participant (controller) can select another participant (controllee) to disallow the controllee's media towards the controller.

A call center scenario provides another example of media privacy: in instances when a first-tier agent cannot answer a customer's questions, the agent might have a private side-channel communication with a supervisor as back-up for realtime customer support, as shown in Fig. 1(a). Privacy control is invoked so that the Supervisor's media goes only to the agent, not to the customer, as shown in Fig. 1(b). Traditional conferencing systems do not generally provide such features. For instance,

`attend`, one of the narrowcasting commands, can accommodate such call center privacy control requirements.

Table 1. Three Different Mute Operations

	Self-Mute	PBX Mute	Narrowcasting `mute`
Media Vector			
Media Distribution	$P_1 \leftarrow (P_2 + P_3 + P_4)$ $P_2 \leftarrow (P_3 + P_4)$ $P_3 \leftarrow (P_2 + P_4)$ $P_4 \leftarrow (P_2 + P_3)$		$P_1 \leftarrow (P_3 + P_4)$ $P_2 \leftarrow (P_1 + P_3 + P_4)$ $P_3 \leftarrow (P_1 + P_2 + P_4)$ $P_4 \leftarrow (P_1 + P_2 + P_3)$
Semantics	Self Control. P_1 mutes himself by turning off his mic so that no media goes to the media server, or P_1 can send a "self-mute" signal to the application server so that the media server simulates self-censorship.	Control by Admin. P_1 is muted by moderator. P_1's media is not mixed in the media server. P_1 is in a listen-only, or "lurker" (stealth) mode.	P2P Control. P_1 `mutes` P_2. P_2 may speak to everyone, but P_1 won't hear his voice.

During a conference, a participant might want to change the media— e.g., change voice to video, or transfer the call to another device or to a media storage server. Call hold or call mute can be invoked by a participant, but the floor control, i.e. the technique to select the speaker of the conference, is invoked by the administrator. A participant may want some selected participants to share his media during the session. Such features apply to received media streams as well. In a conference, multiple participants may want to control the media at once, which would conflict with each others policies.

The rest of the article is structured as follows. In section 2, we review SIP-based conferencing systems and models. In section 3, we explain the narrowcasting concept for media privacy. In section 4, we elaborate our design for a media mixing mechanism and implementation, including a figurative, "virtual reality"-style interface with avatar attributes denoting narrowcasting filters. Finally, the conclusion and outline of future research are presented in section 5.

2 SIP Conferencing

A traditional conferencing system using the PSTN (public switched telephone network) has limited features, implemented in a centrally controlled conference server. A more modern infrastructure such as SIP (Session Initiation Protocol) uses internet signaling and media streams. Due to the simplicity and flexibility of its control and management of multimedia conference services, we concentrate on SIP-based conferencing models.

A conference server and distributed participants are the major components of a centralized confer-

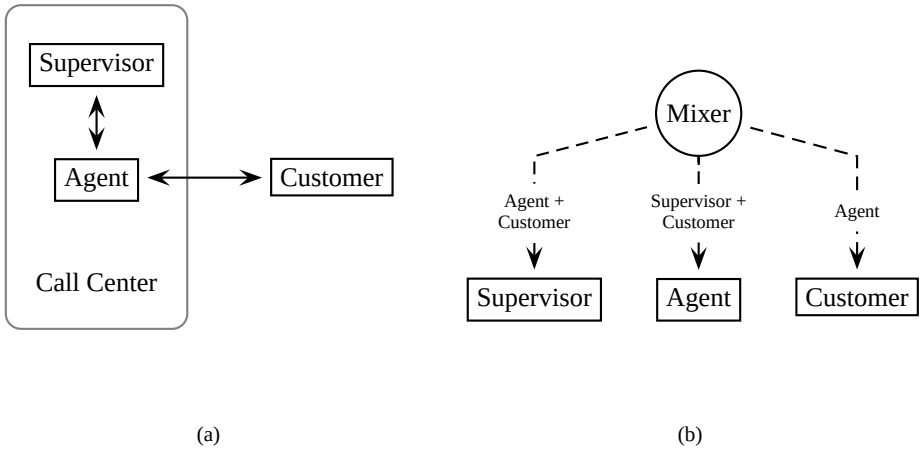


Fig. 1. Media Privacy: A Call Center Application Scenario

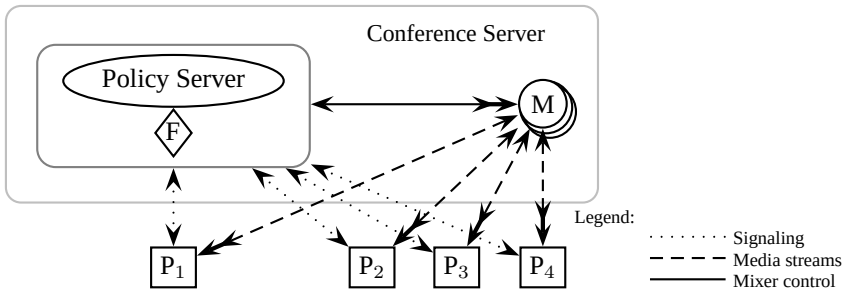


Fig. 2. Typical Conference Architecture

ence system (Fig. 2). A SIP conference server comprises a focus, policy server, and media mixer. The focus handles the conference control— creating, modifying, and terminating conferences. Conference policy is managed by the policy server, which can configure a media server. Mixing and distribution of media streams are the main functions of a media mixer, such as a "voice switch" for audio conferencing, which transmits some composite signals to the respective terminals, as suggested by the multiple arrowheads on the (dashed) return vectors. Value-added services— such as monitoring conference status, participant status, and billing— can be implemented inside or outside of this framework.

2.1 SIP Conference Model

There are two generic conference models: loosely and tightly coupled. In a loosely coupled model, there is neither a central point of control nor a conference server, whereas in a tightly coupled model, a centralized conference control server manages the conferences. The tightly coupled conferencing model can be further classified into six different types depending on the location of the focus and the mixer, as illustrated in Fig. 2, including the Media Server Component Model used for our proof-of-concept. These models are detailed by J. Rosenberg [13] and Y. Cho et. al. [5].

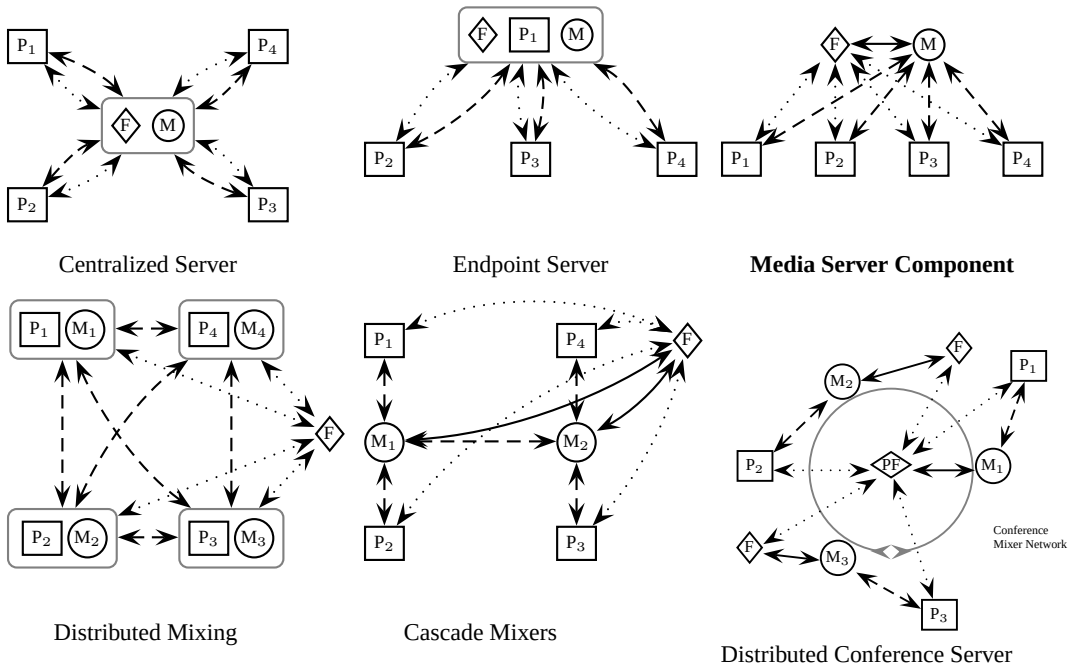


Fig. 3. Conferencing Models: 'P' indicates participant, 'F' indicates focus, 'M' indicates media mixer, and (in the last model) 'PF' indicates Primary Focus. Dotted lines indicate signaling, dashed lines indicate media transmission, and solid lines indicate mixer control.

2.2 SIP Conference Control

Conference control refers to the ability to manipulate the state of a session. A conference is represented by a unique URI (uniform resource identifer), usually a SIP URI, that identifies the focus of a conference. A conference URI can be emailed, sent in an instant message, linked on a web page, or obtained from some non-SIP mechanism. Conference control includes three primary functions:

- Creation: A participant joins a conference by sending an INVITE request to its focus ("dial-in") or by the focus sending an INVITE request to the participant ("dial-out"), citing the conference URI.

- Modification: A participant or focus can modify a session in a conference using a re-INVITE. For instance, when an audio conference extends to video, the focus re-INVITEs each participant adding a video media stream. A participant or focus may also put media streams on hold or take them off hold. Narrowcasting commands are applied to a session by selectively enabling its media streams.

- Termination: A privileged participant (typically a moderator or conference creator) closes a session by sending a BYE request to the focus. The focus then distributes a BYE request to all other participants in the conference, terminating the session.

3 Media Privacy: Narrowcasting Concept

In traditional conferencing systems, participants have little or no privacy, as their voices are by default shared with all others in a session. Such systems cannot offer participants the options of muting and deafening other members. The concept of narrowcasting can be applied to make these kinds of filters available in multimedia conferencing systems. Our system treats media sinks (in the simplest case, listeners) as full citizens, peers of the media sources (conversants' voices), and we defined therefore duals of `mute` & `select`: `deafen` & `attend`, which respectively block a sink or focus on it to the exclusion of others. Fig. 4 shows a famous Japanese carving which informally illustrates multimodal narrowcasting. Three monkeys— Mizaru (with covered eyes), Iwazaru (covered mouth), and Kikazaru (blocked ears)— manifest the notion of limiting media vectors. Mizaru can not see but can hear and speak; Iwazaru can not speak but can see and hear; Kikazaru can not hear but can speak and see.

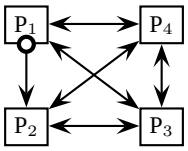
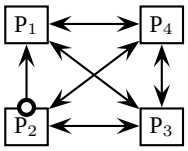
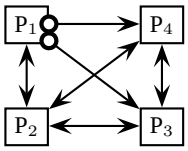
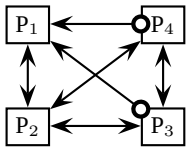
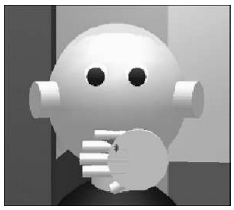
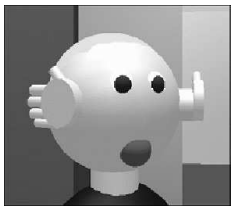
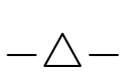
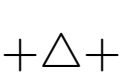
Fig. 4. Media Privacy (Narrowcasting Features)

For modern groupware situations like teleconferences, in which everyone can have presence across the global network, users want to shift and distribute attention (apathy) and accessibility/availability /exposure (privacy), and narrowcasting provides a formalization of such filters. The narrowcasting predicate calculus [6], shown in Fig. 5, is an appropriate basis for such a permission scheme.

The duality between source and sink operations is tight, and the semantics are identical: an object is inclusively enabled by default unless it is explicitly excluded (with $\overbrace{\texttt{mute}}^{\text{source}}$ or $\overbrace{\texttt{deafen}}^{\text{sink}}$) or peers of the same `self`/non-`self` class are explicitly included (with $\overbrace{\texttt{select}\,[\texttt{solo}]}^{\text{sources}}$ or $\overbrace{\texttt{attend}}^{\text{sinks}}$) when the respective object is not. Narrowcasting attributes are not mutually exclusive, and the dimensions are orthogonal. Because a source or a sink is active by default, invoking `exclude` and `include` operations simultaneously on an object results in its being disabled. For instance, a sink might be first `attend`ed, perhaps as a member of some non-singleton subset of a space's sinks, then later `deafen`ed, so that both attributes are simultaneously applied. (As audibility is assumed to be a revocable privilege, such a seemingly conflicted attribute state disables the sink, whose attention would be restored upon resetting its `deafen` flag.) Symmetrically, a source might be `select`ed and then `mute`d, akin to inclusion on a "short list" but relegated to back-up.

Narrowcasting audio commands are listed and their characteristics arrayed in Table 2. Our design allows each user to send or receive data streams to/from a specific recipients in a session. For easier understanding, we consider only audio streams in this article, but this design applies equally well to other media types.

Table 2. Narrowcasting Commands

	P_1 `mutes` P_2	P_1 `deafens` P_2	P_1 `selects` P_2	P_1 `attends` P_2
Media Vectors				
Semantics	Block the media stream coming from a source.	Block media streams going to a sink.	Limit the projected sound to particular sources.	Limit the received sound to particular sinks.
Situation	Participant wants to block media from specific participants.	Participant wants to block media to specific participants.	Participant wants to receive media only from particular participants.	Participant wants to send media to only specific participants.
Figurative Avatars				
Mobile Icons				
Media Distribution	$\begin{bmatrix} \times & 1 & 1 & 1 \\ \mathbf{0} & \times & 1 & 1 \\ 1 & 1 & \times & 1 \\ 1 & 1 & 1 & \times \end{bmatrix}$	$\begin{bmatrix} \times & \mathbf{0} & 1 & 1 \\ 1 & \times & 1 & 1 \\ 1 & 1 & \times & 1 \\ 1 & 1 & 1 & \times \end{bmatrix}$	$\begin{bmatrix} \times & 1 & 1 & 1 \\ \mathbf{1} & \times & 1 & 1 \\ \mathbf{0} & 1 & \times & 1 \\ \mathbf{0} & 1 & 1 & \times \end{bmatrix}$	$\begin{bmatrix} \times & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 1 & \times & 1 & 1 \\ 1 & 1 & \times & 1 \\ 1 & 1 & 1 & \times \end{bmatrix}$

The general expression of inclusive selection is:

$$\texttt{active}(\text{object x}) = \neg\texttt{exclude}(\text{x}) \wedge (\exists\, \text{y}\ (\texttt{include}(\text{y}) \wedge (\texttt{self}(\text{y}) \Leftrightarrow \texttt{self}(\text{x}))) \Rightarrow \texttt{include}(\text{x})). \quad (1)$$

So, for `mute` and `select` (`solo`), the relation is:

$$\texttt{active}(\text{source x}) = \neg\texttt{mute}(\text{x}) \wedge (\exists\, \text{y}\ (\texttt{select}(\text{y}) \wedge (\texttt{self}(\text{y}) \Leftrightarrow \texttt{self}(\text{x}))) \Rightarrow \texttt{select}(\text{x})), \quad (1a)$$

`mute` explicitly turning off a source, and `select` disabling the complement of the selection (in the spirit of "anything not mandatory is forbidden"). For `deafen` and `attend`, the relation is:

$$\texttt{active}(\text{sink x}) = \neg\texttt{deafen}(\text{x}) \wedge (\exists\, \text{y}\ (\texttt{attend}(\text{y}) \wedge (\texttt{self}(\text{y}) \Leftrightarrow \texttt{self}(\text{x}))) \Rightarrow \texttt{attend}(\text{x})). \quad (1b)$$

Fig. 5. Formalization of narrowcasting and selection functions in predicate calculus notation, where '$\neg$' means "not," '$\wedge$' means conjunction (logical "and"), '$\exists$' means "there exists," '$\Rightarrow$' means "implies," and '$\Leftrightarrow$' means mutual implication (equivalence).

In this section, we formally define four narrowcasting commands. In the following expressions, P_a denotes the actor (controller), P_o the object (controllee), P_i a sender of the media (source), and P_j a receiver of the media (sink) for a, i, j, $o \in \{1..\mathrm{n}\}$, where n is the total number of participants. In the absence of narrowcasting commands, a default sound scape presented to a sink P_j is composited from the distributed sources, excluding the receiver's own source stream, as formalized by

$$\mathrm{P}_j \leftarrow \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j \quad (1)$$

3.1 `Mute`

The narrowcasting command `mute` blocks media coming from a source. The mute in traditional systems is a self-mute function which allows a user to withhold his/her media from other participants, but the modern `mute` is a control function that can select another participant (or a group of participants) to disallow media towards the controller, still allowing other participants to hear the controllee. The $\sum$ operator composites media from the respective participants.

$$\mathrm{P}_j \leftarrow \begin{cases} \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j - \mathrm{P}_o & \text{when } \mathrm{P}_j = \mathrm{P}_a \text{ or } \mathrm{P}_a = \mathrm{P}_0, \\ \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j & \text{otherwise.} \end{cases} \quad (2)$$

The example modeled by the matrix in the first column of Table 2 illustrates when P_1 `mutes` another participant P_2. In this example, n=4, P_a=P_1 (the controller), and P_o=P_2 (the controllee). Due to this operation, P_1 will not receive any media from P_2. (This is actually a simplification of the evaluation performed by our prototype, since our model supports multipresence, the designation

by a single human user of possibly multiple iconic representatives in an interface. Such complicated subtleties are beyond the scope of this article.)

3.2 `Deafen`

`Deafen` is a sink-related media privacy command that blocks media streams to a selected participant. For example, if Bob (P_1) wants to share his media with everyone in a conference except Alice (P_2), then Alice will not receive any streams from Bob if Bob `deafen`s Alice. (Transposing the participants one can realize an equivalent operation, P_2 `mute`s P_1.) The second column of Table 2 shows the media relationship among four participants.

$$\mathrm{P}_j \leftarrow \begin{cases} \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j - \mathrm{P}_a & \text{when } \mathrm{P}_j = \mathrm{P}_o, \\ \phi & \text{when } \mathrm{P}_a = \mathrm{P}_o, \\ \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j & \text{otherwise.} \end{cases} \tag{3}$$

Again in this example, $a = 1$ and $o = 2$.

3.3 `Select (Solo)`

The privacy command `select` limits received media to particular sources. For instance, students might `select` a teacher to avoid distractions. P_1 will receive media only from P_2 if P_1 `select`s P_2, implicitly `mut`ing the complement of the selection. The third column of Table 2 shows the media relationships among four participants; two vectors are disabled in this case.

$$\mathrm{P}_j \leftarrow \begin{cases} \mathrm{P}_o & \text{when } \mathrm{P}_j = \mathrm{P}_a, \\ \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j & \text{otherwise.} \end{cases} \tag{4}$$

3.4 `Attend`

`Attend` is the other including command for media privacy, limiting received sound to a particular recipient. If Alice `attend`s Bob, only Bob will hear Alice, since other participants are implicitly `deafen`ed. The rightmost column of Table 2 shows the media relationship among four participants; again two media vectors are suppressed.

$$\mathrm{P}_j \leftarrow \begin{cases} \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j & \text{when } \mathrm{P}_j = \mathrm{P}_o, \\ \sum_{i=1}^{n} \mathrm{P}_i - \mathrm{P}_j - \mathrm{P}_a & \text{otherwise.} \end{cases} \tag{5}$$

4 System Design and Implementation

The main required functions for media policy configuration and control are policy configuration, policy evaluation, and media mixing and distribution. The Media Server Component Model (top right of (Fig. 2) selected for our implementation comprises a centralized focus (collocated with the policy server), a centralized mixer, and participants. The architecture, elaborated in Fig. 6, embeds policy configuration, media mixing, and a CVE interface within a SIP framework. All the components in this

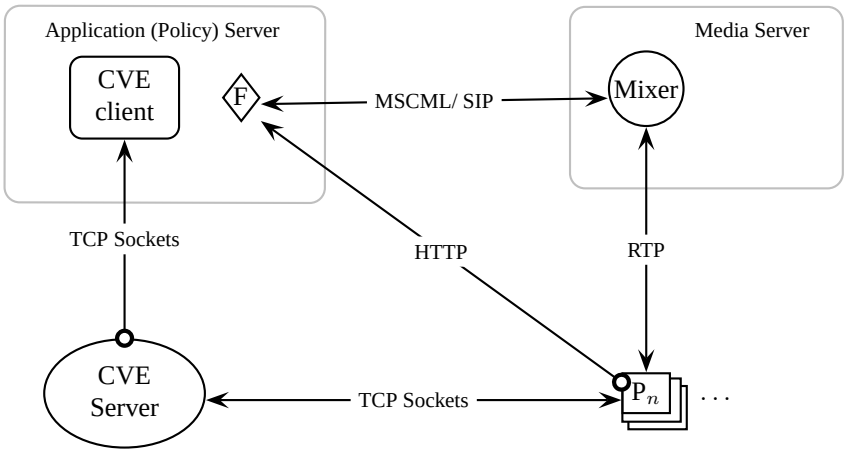


Fig. 6. Media Server Component Model with Collaborative Virtual Environment Integration

architecture are standard SIP user agents extended with additional user interfaces needed for media policy configuration and control. The communication protocols XCAP (**E**xtensible Markup Language Configuration Access Protocol) and MSCML (Media Server Control Markup Language) [8] are IETF standards.

4.1 Policy Configuration

In an extended SIP framework, conference participants could configure privacy by sending requests to a policy server using XCAP [14], a standardized way to use HTTP to store, retrieve, and manipulate configuration and application data in XML format. In our proof-of-concept, participants set policies using various (browser, figurative, or mobile) GUIs to invoke narrowcasting commands on specified controllees, and control is via TCP sockets or HTTP directly (without XCAP).

4.2 Policy Evaluation by Application Server

An application server performs three major functions to evaluate policy:

Evaluating policies configured by each participant: The policy from each participant can be logically compiled into a matrix, as shown in Table 3, where entry c_{ij} of the matrix represents connectivity of source i to sink j, and the main diagonal is populated by "don't care"s. Each participant (P_1, P_2, . . . , P_n), where n is the total number of participants, logically sets permissions in authorized cells. Since a media relationship ultimately factors at least two participants, a source and a sink, each cell contains policies from both. For example, $P_1 \rightarrow P_2$, i.e. media sourced at P_1 and sunk at P_2, has policy involvement of both P_1 and P_2: P_1 sets permissions about whether or not to send media to P_2, and at the same time, P_2 sets permissions about whether or not to receive such media. The policy then is evaluated depending on the combined relationship between P_1 and P_2.

Responding to participants regarding changes made in the policy: A policy evaluation report (confirming success or alerting failure of a configuration request) could be sent to participants via standard XCAP response codes.

Table 3. Policy Matrix $P = [\mathrm{p}_{ij}]$

	P_1	P_2	$\cdots$	P_n
P_1		$\mathrm{P}_1(\mathrm{P}_1 \rightarrow \mathrm{P}_2)$ $\mathrm{P}_2(\mathrm{P}_1 \rightarrow \mathrm{P}_2)$	$\cdots$	$\mathrm{P}_1(\mathrm{P}_1 \rightarrow \mathrm{P}_n)$ $\mathrm{P}_n(\mathrm{P}_1 \rightarrow \mathrm{P}_n)$
P_2	$\mathrm{P}_2(\mathrm{P}_2 \rightarrow \mathrm{P}_1)$ $\mathrm{P}_1(\mathrm{P}_2 \rightarrow \mathrm{P}_1)$		$\cdots$	$\mathrm{P}_2(\mathrm{P}_2 \rightarrow \mathrm{P}_n)$ $\mathrm{P}_n(\mathrm{P}_2 \rightarrow \mathrm{P}_n)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
P_n	$\mathrm{P}_n(\mathrm{P}_n \rightarrow \mathrm{P}_1)$ $\mathrm{P}_1(\mathrm{P}_n \rightarrow \mathrm{P}_1)$	$\mathrm{P}_n(\mathrm{P}_n \rightarrow \mathrm{P}_2)$ $\mathrm{P}_2(\mathrm{P}_n \rightarrow \mathrm{P}_2)$	$\cdots$	

Sending requests to a media mixer for necessary media mixing: After compiling the media policies, the system determines which media streams need to be mixed and delivered to whom. Using standard MSCML the policy server instructs the media mixer to perform the necessary mixing.

4.3 Media Mixing and Distribution

The media server receives MSCML requests from a policy configuration server. According to the accumulated state, it performs the necessary mixing and delivers these streams to subscribed participants. The maximum number of mixes, the power set of the participants excluding the empty and universal sets, is

$$\sum_{i=1}^{n-1} {}_n\mathrm{C}_i = 2^n - 2. \tag{6}$$

Therefore, for n = 3, 4, 5, the maximum number of mixes would be 6, 14, and 30, respectively. However, depending on participants' media privacy requests, the actual number of mixes might be fewer.

Fig. 7 illustrates narrowcasting media distribution among four participants when P_1 `mutes` P_2 and `deafens` P_4. All participants send their media to the media mixer. The media server mixes only the necessary streams and delivers them back to the appropriate recipients.

4.4 Mixing Configuration and Observation

Our prototype environment comprises a SIP server (BEA WebLogic SIP Server), an application server (BEA WebLogic Workshop), a media server (Dialogic/Cantata Snowshore IP Media Server), and four SIP clients (X-lite). We implemented narrowcasting commands `mute`, `deafen`, `attend`, and (partially) `select`, integrating these filter functions into the application server. Fig. 8 shows the control and media streams among a participant, application server, and media mixer when applying a narrowcasting command.

The following trace shows the MSCML code sent from the policy configuration (application) server to the media mixer when P_1 `deafens` P_2. In each block, the first chunk is the SIP headers and the second chunk, the body, is the MSCML payload. P_1 makes a private group with P_3 and P_4 so P_1, P_3,

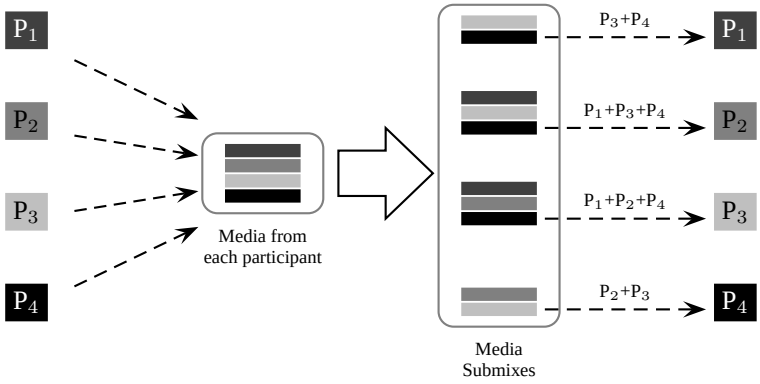


Fig. 7. Media Mixing and Delivery (P_1 `mutes` P_2 and `deafens` P_4)

and P_4 can hear each other, but P_2 cannot hear P_1. The application server evaluates the policy and sends it to the media server.

```
# Note: irrelevant headers are elided
INFO sip:192.168.1.12:5060 SIP/2.0
To: <sip:conf=conference_0@192.168.1.12>;tag=1168487679
Content-Length: 350
From: <sip:P1@192.168.1.11>;tag=ee2d88d1
Content-Type: application/mediaservercontrol+xml
Max-Forwards: 66

<?xml version='1.0'?>
<MediaServerControl version="1.0">
 <request>
  <configure_leg mixmode="private" id="sip:P1@192.168.1.11">
   <configure_team action="add">
    <teammate id="sip:P4@192.168.1.11"/>
    <teammate id="sip:P3@192.168.1.11"/>
   </configure_team>
  </configure_leg>
 </request>
</MediaServerControl>
```

The media server confirms the configured media mixing and delivery, also using MSCML.

```
INFO sip:app-1w4n5gq0kbcgv@192.168.1.11:5060;
transport=udp;wlsscid=-7ba6e82e1ed19297;lr SIP/2.0
Via: SIP/2.0/UDP 192.168.1.12:5060
To: <sip:P1@192.168.1.11>;tag=ee2d88d1
From: sip:conf=conference_0@192.168.1.12;tag=1168487679
Content-Type: application/mediaservercontrol+xml
```

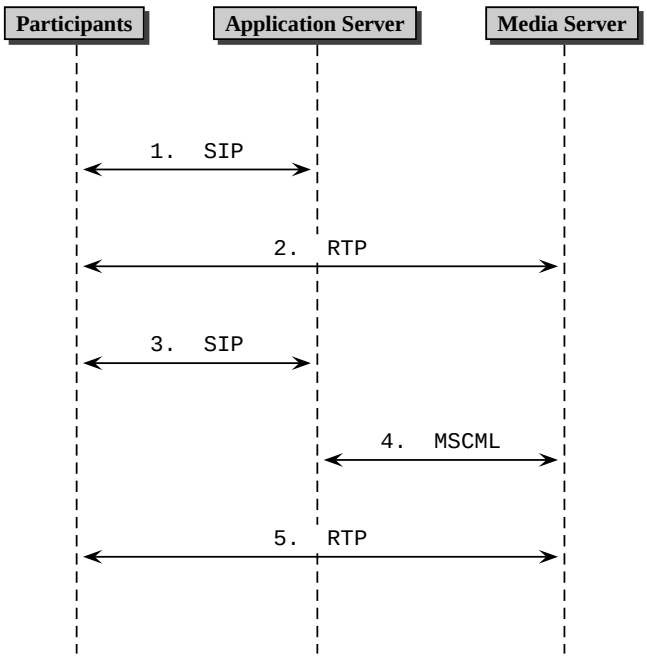


Fig. 8. Communication Flow Between SIP Entities: A default configuration (`1.`) establishes a normal session (`2.`), but it can be adjusted (`3.`) to reconfigure (`4.`) the mixes returned to the participants (`5.`).

```
Content-Length: 281

<?xml version="1.0"?>
<MediaServerControl version="1.0">
 <response request="configure_leg" code="200" text="OK">
  <team id="sip:P1@192.168.1.11" numteam="2">
   <teammate id="sip:P3@192.168.1.11"/>
   <teammate id="sip:P4@192.168.1.11"/>
  </team>
 </response>
</MediaServerControl>
```

The MSCML configuration and audibility are shown in Table 4 when P_1 `deafens` P_2.

4.5 Narrowcasting in Virtual Environments

Virtual environments are characterized, in contrast to general multimedia systems, by the explicit notion of the position (location and orientation) of the perspective presented to respective users. Often such vantage points are modeled by the standpoints and directions of objects in a virtual space. These representatives might be more or less symbolic (abstract icons) or figurative (avatars), but act as delegates of human users. Icons and avatars can deify embodied virtuality, treating abstract presence as a user interface object, symbols and manifestations of sources and sinks. We have developed worksta-

Table 4. MSCML Configuration: P_1 `deafen`s P_2

Participant	ID	Team Members	Mixmode	Hears
P_1	P_1	P_3,P_4	Private	P_2+P_3+P_4
P_2	P_2	None	Full	P_3+P_4
P_3	P_3	P_1,P_4	Full	P_1+P_2+P_4
P_4	P_4	P_1,P_3	Full	P_1+P_2+P_3

tion [10] and mobile interfaces to manipulate narrowcasting attributes in virtual spaces via a Java3D interface [18] [16]. This "virtual reality"-style interface features perspective displays of virtual rooms and spaces with figurative avatars, each of which can be associated with an audio source, corresponding to the voice of a corresponding user. A participant can rearrange the locations of avatars in virtual spaces and designate sinks, through whose ears the resulting spatialized soundscape is heard. Also, each participant can apply narrowcasting attributes to the avatars, altering the sound mix. Recalling the monkeys in Fig. 4, Fig. 9 illustrates the visual cues used for narrowcasting, including a hand covering the mouth of the `muted` avatar and hands clapped over the ears of a `deafened` avatar. An action taken by a participant is communicated using a CVE client/server architecture, which framework allows multimodal clients to exchange status data through the network. Clients currently include sound spatializers, telepresence applications, turnoramic and panoramic browsers, music visualizers, motion platforms, and mobile interfaces.

A bridge between CVE clients and SIP-based narrowcasting allows distributed multimodal interfaces. The results of narrowcasting operations are expressed aurally by the SIP-based mixer and visually by the graphical interfaces. Besides the previously-described web- and workstation-based interfaces, we have also prototyped a mobile narrowcasting display and control, shown in Fig. 10, although we have not yet integrated the mobile audio stream, so it is currently more useful as a colocated "remote control" than as a truly mobile application. Symbolic representations of narrowcasting operations were developed for the mobile interface by flattening figurative 3D avatars to 2.5D icons, as seen in the second-last row of Table 2. In the mobile GUI, narrowcasting attributes' graphical displays are triply encoded— by position (before the "mouth" for `m`ute and `s`elect, straddling the "ears" for `d`eafen and `a`ttend), symbol ('+' for include & '-' for exclude), and color (green for assert & red, yellow, and orange for inhibit- by self, other, and implicitly, respectively).

The bridge between the interfaces and the SIP-based backend is a 'read-only' CVE client embedded in the SIP application server. When the policy server is launched, the embedded client connects to a CVE session server and opens a channel for each member in the conference. Every time a user enables or disables one of the narrowcasting attributes, the action is relayed to the embedded CVE client. As each message is received, the client invokes the necessary methods to reflect the changed status in the SIP conference.

4.6 System Performance

The narrowcasting control is basically light-weight: the commands are typically infrequent, and each of them is easily processed by an application server. For excluding narrowcasting commands (`deafen` and `mute`), the time complexity is constant (O(1)), independent of the number of participants in a session. For including narrowcasting commands (`select` and `attend`), in which the

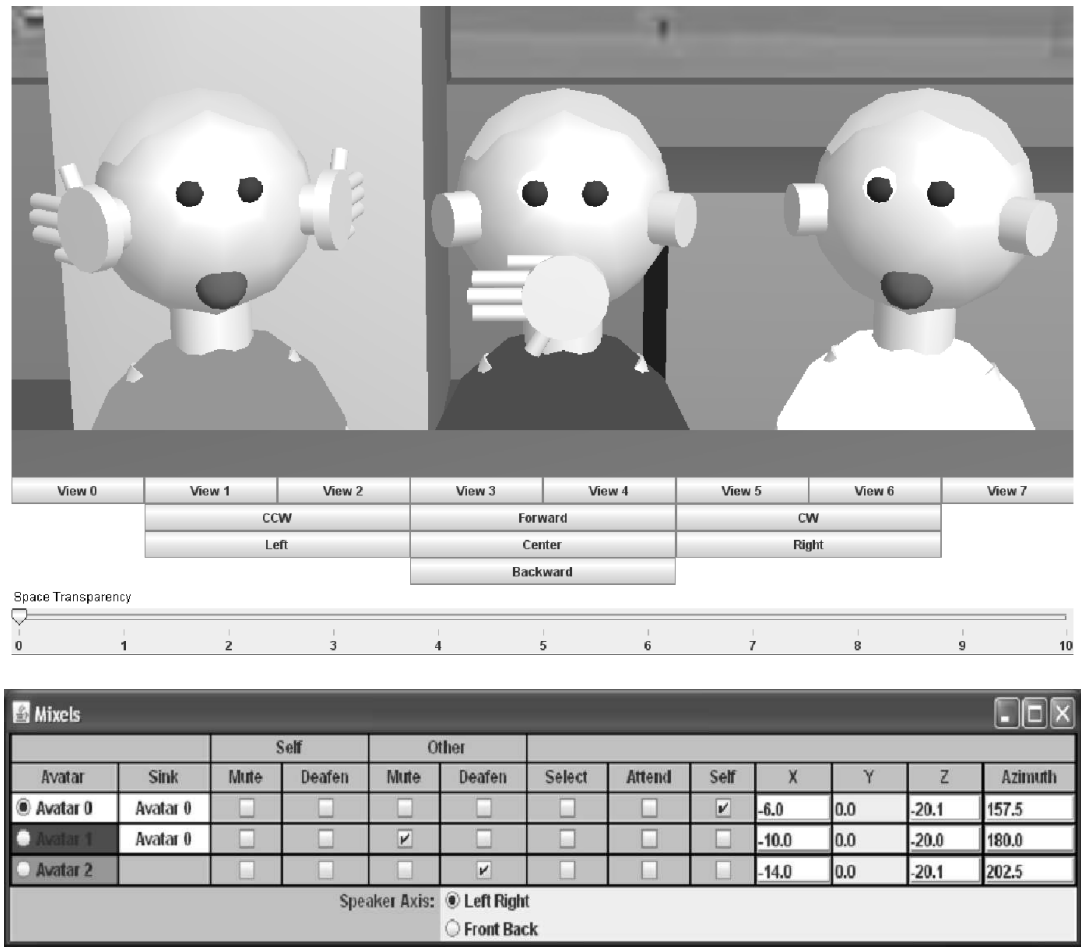


Fig. 9. Narrowcasting Control in Virtual Environment: P_1 (avatar 0, right) `mutes` P_2 (avatar 1, middle) and `deafens` P_3 (avatar 2, left).

connectivity state of the complement of a selection needs to be adjusted, the time complexity is O(n), linear in the number of participants. The configuration for the IP Media Server used in our experiments supports up to 100 clients. Even though our laboratory testbed uses a much smaller user pool, typically about four, there is no reason not to assume that the signaling protocol can keep up with practical realtime demands and support the same number of session participants.

5 Conclusion and Future Research

In this article, we have described an instance of Media Server Component Model architecture for policy-based media-mixing and narrowcasting within the standard SIP framework for multimedia conferencing systems. We have implemented privacy commands and a policy evaluation algorithm, a media mixing and delivery mechanism that considers the policy configured by the participants. The policy can be displayed and controlled via a 3D interface in which hands and other attributes (megaphones and ear trumpets) clapped over figurative avatars' mouths and ears represent media stream

Fig. 10. Mobile Narrowcasting Interface

filters. The popularity of applications like 'Second Life' extends the ways in which people interact. Such three-dimensional environments represent a fertile platform for virtual conferences, meetings, and concerts.

Future research includes allowing selection of multiple sources and sinks for narrowcasting commands and consideration of other conference architectures with multiple policy servers and media mixers. Such capability will be ported to mobile computers like smartphones. `Muffle` (partial `deafen`) and `muzzle` (partial `mute`) will enrich the narrowcasting state space. We will also generalize policy determination in metasessions with multiple simultaneous chatspaces, in which one has presence across multiple virtual spaces, each with several or many conversants, including "multipresence," allowing designation of multiple instances of "self."

Acknowledgements

This project was sponsored in part by a grant from the Japan Science and Technology Foundation. We also thank Dialogic/Cantata and BEA for providing us with the media and application servers for the experiment.

References

1. Mohammad Sabbir Alam, Michael Cohen, and Ashir Ahmed. Articulated Narrowcasting for Privacy and Awareness in Multimedia Conferencing Systems and Design for Implementation Within a SIP Framework. *JVRB: J. of Virtual Reality and Broadcasting*, 4(9), 2007.
2. Mohammad Sabbir Alam, Michael Cohen, and Ashir Ahmed. Narrowcasting— Controlling Media Privacy in SIP Multimedia Conferencing. In *Proc. IEEE CCNC: 4th Consumer Communications and Networking Conf.*, Las Vegas, January 2007.

3. Mohammad Sabbir Alam, Michael Cohen, and Ashir Ahmed. Narrowcasting: Implementation of Privacy Control in SIP Conferencing. In *Proc. ICME: Int. Conf. on Multimedia & Expo*, Beijing, July 2007.
4. Lance Berc, Hania Gajewska, and Mark Manasse. Pssst: Side Conversations in the Argo Telecollaboration System. In *Proc. UIST: 8th Annual ACM Symp. on User Interface and Software Technology*, pages 155–156, New York, NY, November 1995. ACM Press. ISBN 0-89791-709-X.
5. Yeong-Hun Cho, Moon-Sang Jeong, Jong-Tae Park, and Wee-Hyuk Lee. Distributed Management Architecture for Multimedia Conferencing Using SIP. In *Proc. DFMA: 1st Int. Conf. on Distributed Frameworks for Multimedia Applications*, Besancon, France, February 2005.
6. Michael Cohen. Exclude and include for audio sources and sinks: Analogs of mute & solo are deafen & attend. *Presence: Teleoperators and Virtual Environments*, 9(1):84–96, February 2000. ISSN 1054-7460; `www.u-aizu.ac.jp/~mcohen/welcome/publications/ie1.pdf`.
7. H.-P. Dommel and J. Garcia-Luna-Aceves. Floor Control for Activity Coordination in Networked Multimedia Applications. In *Proc. APCC: 2nd Asian-Pacific Conference on Communications*, Osaka, Japan, June 1995.
8. J. Van Dyke, E. Burger, and A. Spitzer. RFC 4722— Media Server Control Markup Language (MSCML) and Protocol, November 2006.
9. R. Even and N. Ismail. RFC 4825— Conferencing Scenarios, July 2006.
10. Owen Noel Newton Fernando, Kazuya Adachi, Uresh Duminduwardena, Makoto Kawaguchi, and Michael Cohen. Audio Narrowcasting and Privacy for Multipresent Avatars on Workstations and Mobile Phones. IEICE *Trans. on Information and Systems*, E89-D(1):73–87, January 2006.
11. Jonathan Kaplan and Nicole Yankelovich. Sun^{TM} Labs Meeting Suite, Executive Edition. In *Proc. HCIC: Human Computer Interaction Consortium*, Fraser, February 2006.
12. Petri Koskelainen, Henning Schulzrinne, and Xiaotao Wu. A SIP-based Conference Control Framework. In *Proc. NOSSDAV: 12th Int. Wkshp. on Network and Operating Systems Support for Digital Audio and Video*, pages 53–61, New York, NY, 2002. ACM Press. ISBN 1-58113-512-2.
13. J. Rosenberg. RFC 4353— A Framework for Conferencing with the Session Initiation Protocol, February 2006.
14. J. Rosenberg. RFC 4897— The Extensible Markup Language (XML) Configuration Access Protocol (XCAP), May 2007.
15. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. RFC 3261— SIP: Session Initiation Protocol, June 2002.
16. Henry Sowizral, Kevin Rushforth, and Michael Deering. *The Java 3D API Specification*. Addison-Wesley, second edition, 2000. ISBN 0-201-71041-2.
17. Richard A. Spinello. *Cyberethics Morality and Law in Cyberspace*. Jones and Bartlett Publishers, Sudbury, 2004. ISBN 0-7637-0064-9.
18. Aaron E. Walsh and Doug Gehringer. *Java 3D API Jump-Start*. Prentice-Hall, 2002. ISBN 0-13-034076-6.
19. C. C. Wang, J. Cahnbley, and J. W. Richardson. US Patent—Method and System for Providing a Private Conversation Channel in a Videoconference System, June 2003. Publication No. WO 2003/053034 A1.
20. Nicole Yankelovich, Jen McGinn, Mike Wessler, Jonathan Kaplan, Joe Provino, and Harold Fox. Private communications in public meetings. In *Proc. CHI: Human Factors in Computing Systems*, pages 1873–1876, Portland, OR, April 2005. ACM Press. ISBN 1-59593-002-7.