Network Working Group
Request for Comments: 5239
Category: Standards Track

M. Barnes
Nortel
C. Boulton
Avaya
O. Levin
Microsoft Corporation

June 2008

A Framework for Centralized Conferencing

Status of This Memo

Abstract

This document defines the framework for Centralized Conferencing. The framework allows participants using various call signaling protocols, such as SIP, H.323, Jabber, Q.931 or ISDN User Part (ISUP), to exchange media in a centralized unicast conference. The Centralized Conferencing Framework defines logical entities and naming conventions. The framework also outlines a set of conferencing protocols, which are complementary to the call signaling protocols, for building advanced conferencing applications. The framework binds all the defined components together for the benefit of builders of conferencing systems.

Table of Contents

1. Introduction

This document defines the framework for Centralized Conferencing. The framework allows participants using various call signaling protocols, such as SIP, H.323, Jabber, Q.931 or ISUP, to exchange media in a centralized unicast conference. Other than references to general functionality (e.g., establishment and teardown), details of these call signaling protocols are outside the scope of this document.

The Centralized Conferencing Framework defines logical entities and naming conventions. The framework also outlines a set of conferencing protocols, which are complementary to the call signaling protocols, for building advanced conferencing applications.

The Centralized Conferencing Framework is compatible with the functional model presented in the SIP Conferencing Framework [RFC4353]. Section 10 of this document discusses the relationship between the Centralized Conferencing Framework and the SIP Conferencing Framework, in the context of the Centralized Conferencing model presented in this document.

2. Conventions

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, [RFC2119] and indicate requirement levels for compliant implementations.

3. Terminology

This Centralized Conferencing Framework document generalizes, when appropriate, the SIP Conferencing Framework [RFC4353] terminology and introduces new concepts, as listed below. Further details and clarification of the new terms and concepts are provided in the subsequent sections of this document.

Active conference: The term "active conference" refers to a conference object that has been created and activated via the allocation of its identifiers (e.g., conference object identifier and conference identifier) and the associated focus. An active conference is created based on either a system default conference blueprint or a specific conference reservation.

Call Signaling protocol: The call signaling protocol is used between a participant and a focus. In this context, the term "call" means a channel or session used for media streams.

Conference blueprint: A conference blueprint is a static conference object within a conferencing system, which describes a typical conference setting supported by the system. A conference blueprint is the basis for creation of dynamic conference objects. A system may maintain multiple blueprints. Each blueprint is comprised of the initial values and ranges for the elements in the object, conformant to the data schemas for the conference information.

Conference control protocol (CCP): A conference control protocol provides the interface for data manipulation and state retrieval for the centralized conferencing data, represented by the conference object.

Conference factory: A conference factory is a logical entity that generates unique URI(s) to identify and represent a conference focus.

Conference identifier (ID): A conference identifier is a call signaling protocol-specific URI that identifies a conference focus and its associated conference instance.

Conference information: The conference information includes definitions for basic conference features, such as conference identifiers, membership, signaling, capabilities, and media types applicable to a wide range of conferencing applications. The conference information also includes the media and application-specific data for enhanced conferencing features or capabilities, such as media mixers. The conference information is the data type (i.e., the XML schema) for a conference object.

Conference instance: A conference instance refers to an internal implementation of a specific conference, represented as a set of logical conference objects and associated identifiers.

Conference object: A conference object represents a conference at a certain stage (e.g., description upon conference creation, reservation, activation, etc.), which a conferencing system maintains in order to describe the system capabilities and to provide access to the services available for each object independently. The conference object schema is based on the conference information.

Conference object identifier (ID): A conference object identifier is a URI that uniquely identifies a conference object and is used by a conference control protocol to access and modify the conference information.

Conference policies: Conference policies collectively refers to a set of rights, permissions, and limitations pertaining to operations being performed on a certain conference object.

Conference reservation: A conference reservation is a conference object, which is created from either a system default or client selected blueprint.

Conference state: The conference state reflects the state of a conference instance and is represented using a specific, well-defined schema.

Conferencing system: Conferencing system refers to a conferencing solution based on the data model discussed in this framework document and built using the protocol specifications referenced in this framework document.

Conference user identifier (ID): A unique identifier for a user within the scope of a conferencing system. A user may have multiple conference user identifiers within a conferencing system (e.g., to represent different roles).

Floor: Floor refers to a set of data or resources associated with a conference instance, for which a conference participant, or group of participants, is granted temporary access.

Floor chair: A floor chair is a floor control protocol compliant client, either a human participant or automated entity, who is authorized to manage access to one floor and can grant, deny, or revoke access. The floor chair does not have to be a participant in the conference instance.

Focus: A focus is a logical entity that maintains the call signaling interface with each participating client and the conference object representing the active state. As such, the focus acts as an endpoint for each of the supported signaling protocols and is responsible for all primary conference membership operations (e.g., join, leave, update the conference instance) and for media negotiation/maintenance between a conference participant and the focus.

Media graph: The media graph is the logical representation of the flow of media for a conference.

Media mixer: A media mixer is the logical entity with the capability to combine media inputs of the same type, transcode the media, and distribute the result(s) to a single or multiple outputs. In this context, the term "media" means any type of data being delivered over the network using appropriate transport means, such as RTP/ RTP Control Protocol (RTCP) (defined in [RFC3550]) or Message Session Relay Protocol (defined in [RFC4975]).

Role: A role provides the context for the set of conference operations that a participant can perform. A default role (e.g., standard conference participant) will always exist, providing a user with a set of basic conference operations. Based on system-specific authentication and authorization, a user may take on alternate roles, such as conference moderator, allowing access to a wider set of conference operations.

Sidebar: A sidebar is a separate conference instance that only exists within the context of a parent conference instance. The objective of a sidebar is to be able to provide additional or alternate media only to specific participants.

Whisper: A whisper involves a one-time media input to (a) specific participant(s) within a specific conference instance, accomplished using a sidebar. An example of a whisper would be an announcement injected only to the conference chair or to a new participant joining a conference.

4. Overview

A centralized conference is an association of endpoints, called conference participants, with a central endpoint, called a conference focus. The focus has direct peer relationships with the participants by maintaining a separate call signaling interface with each. Consequently, in this centralized conferencing model, the call signaling graph is always a star.

The most basic conference supported in this model would be an ad hoc, unmanaged conference, which would not necessarily require any of the functionality defined within this framework. For example, it could be supported using basic SIP signaling functionality with a participant serving as the focus; the SIP Conferencing Framework [RFC4353] together with the SIP Call Control Conferencing for User Agents [RFC4579] documents address these types of scenarios.

In addition to the basic features, however, a conferencing system supporting the centralized conferencing model proposed in this framework document can offer richer functionality, by including dedicated conferencing applications with explicitly defined capabilities, reserved recurring conferences, along with providing the standard protocols for managing and controlling the different attributes of these conferences.

The core requirements for centralized conferencing are outlined in [RFC4245]. These requirements are applicable for conferencing systems using various call signaling protocols, including SIP. Additional conferencing requirements are provided in [RFC4376] and [RFC4597].

The centralized conferencing system proposed by this framework is built around a fundamental concept of a conference object. A conference object provides the data representation of a conference during each of the various stages of a conference (e.g., creation, reservation, active, completed, etc.). A conference object is accessed via the logical functional elements, with whom a conferencing client interfaces, using the various protocols identified in Figure 1. The functional elements defined for a conferencing system described by the framework are a conference control server, floor control server, any number of Foci, and a notification service. A conference control protocol (CCP) provides the interface between a conference and media control client and the conference control server. A floor control protocol (e.g., Binary Floor Control Protocol (BFCP)) provides the interface between a floor control client and the floor control server. A call signaling protocol (e.g., SIP, H.323, Jabber, Q.931, ISUP, etc.) provides the interface between a call signaling client and a focus. A notification protocol (e.g. SIP Notify [RFC3265]) provides the interface between the conferencing client and the notification service.

A conferencing system can support a subset of the conferencing functions depicted in the conferencing system logical decomposition in Figure 1 and described in this document. However, there are some essential components that would typically be used by most other advanced functions, such as the notification service. For example, the notification service is used to correlate information, such as the list of participants with their media streams, between the various other components.

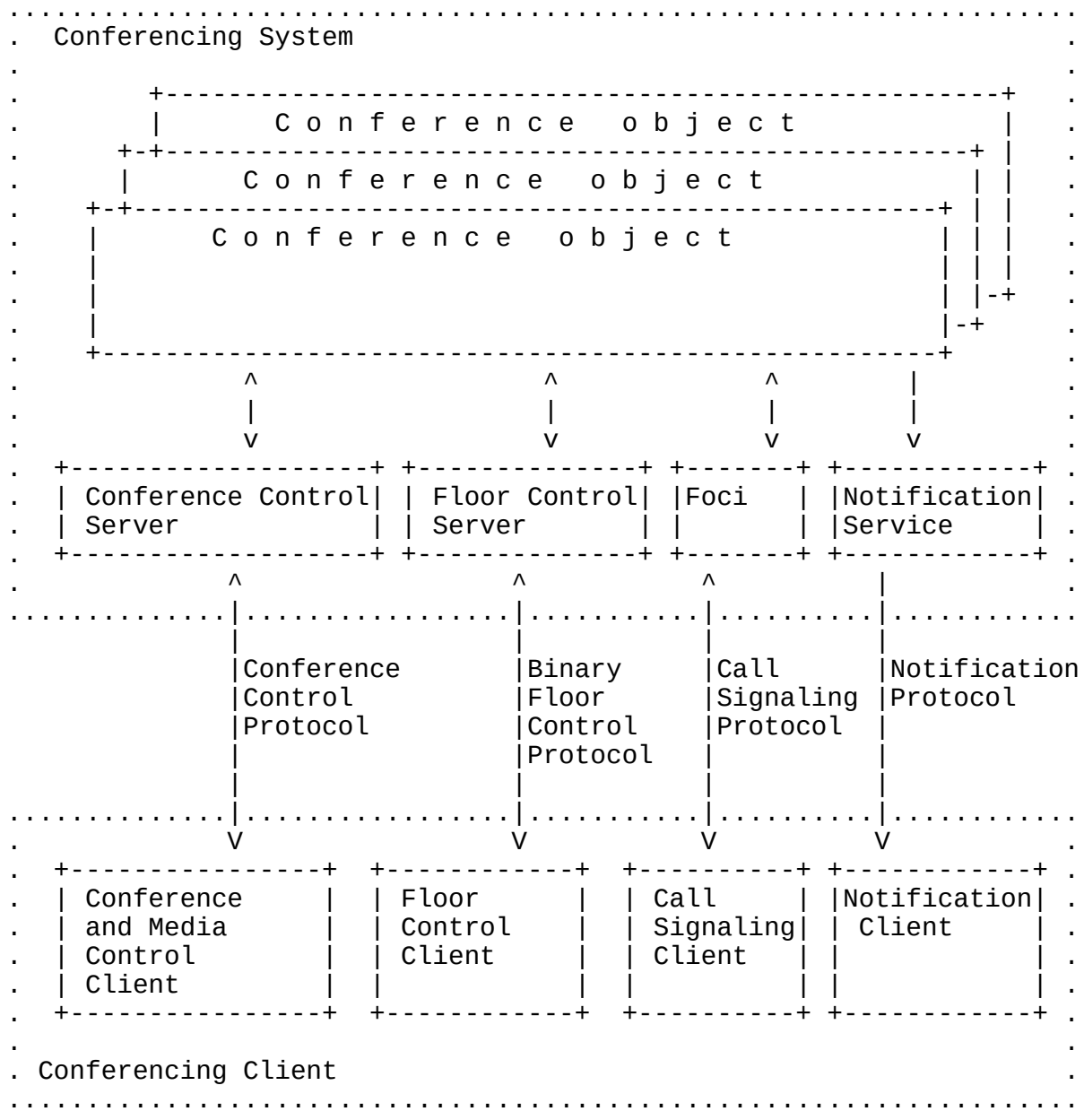


Figure 1: Conferencing System Logical Decomposition

The media graph of a conference can be centralized, decentralized, or any combination of both and potentially differ per media type. In the centralized case, the media sessions are established between a media mixer controlled by the focus and each one of the participants. In the decentralized (i.e., distributed) case, the media graph is a

multicast or multi-unicast mesh among the participants. Consequently, the media processing (e.g., mixing) can be controlled either by the focus alone or by the participants. The concepts in this framework document clearly map to a centralized media model. The concepts can also apply to the decentralized media case; however, the details of such are left for future study.

Section 5 of this document provides more details on the conference object. Section 6 defines the constructs and identifiers that MUST be implemented to manage the conference objects, instances, and users associated with a conferencing system. Section 7 of this document describes how a conferencing system is logically built using the defined high level data model and how the conference objects are maintained. Section 8 describes the fundamental conferencing mechanisms and provides a high level overview of the protocols. Section 9 then provides realizations of various conferencing scenarios, detailing the manipulation of the conference objects using the defined protocols. Section 10 of this document summarizes the relationship between this Centralized Conferencing Framework and the SIP Conferencing Framework.

5. Centralized Conferencing Data

The centralized conference data is logically represented by the conference object. A conference object is of type 'Conference information type', as illustrated in Figure 2. The conference information type is extensible.

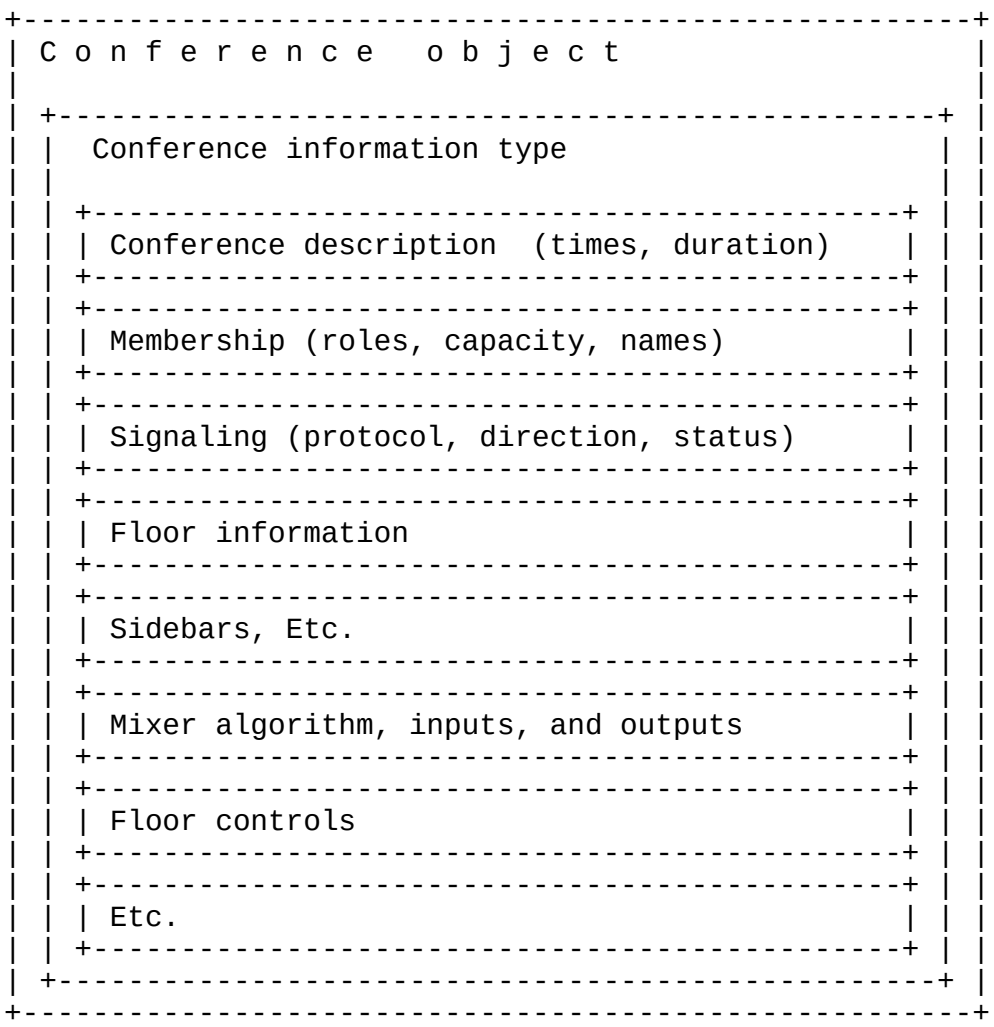


Figure 2: Conference Object Type Decomposition

In a system based on this conferencing framework, the same conference object type is used for representation of a conference during different stages of a conference, such as expressing conferencing system capabilities, reserving conferencing resources, or reflecting the state of ongoing conferences. Section 7 describes the usage semantics of the conference objects. The exact XML schema of the

conference object, including the organization of the conference information is detailed in a separate document [XCON-COMMON].

Along with the basic data model, as defined in [XCON-COMMON], the realization of this framework requires a policy infrastructure. The policies required by this framework to manage and control access to the data include local, system level boundaries associated with specific data elements, such as the membership, and the ranges and limitations of other data elements. Additional policy considerations for a system realization based on this data model are discussed in Section 5.2.

5.1. Conference Information

There is a core set of data in the conference information that is utilized in any conference, independent of the specific conference media nature (e.g., the mixing algorithms performed, the advanced floor control applied, etc.). This core set of data in the conference information contains the definitions representing the conference object capabilities, membership, roles, call signaling, and media status relevant to different stages of the conference life-cycle. This core set of conference information may be represented using the conference-type, as defined in the SIP conference event package [RFC4575]. Typically, participants with read-only access to the conference information would be interested in this core set of conference information only.

In order to support more complex media manipulations and enhanced conferencing features, the conference information, as defined in the data model [XCON-COMMON], contains additional data beyond that defined in the SIP conference event package [RFC4575]. The information defined in the data model [XCON-COMMON] provides specific media mixing details, available floor controls, and other data necessary to support enhanced conferencing features. This information allows authorized clients to manipulate the mixer’s behavior via the focus, with the resultant distribution of the media to all or individual participants. By doing so, a client can change its own state and/or the state of other participants in the conference.

New centralized conferencing specifications can extend the basic conference-type, as defined in the data model [XCON-COMMON], and introduce additional data elements to be used within the conference information type.

5.2. Conference policies

Conference policies collectively refers to a set of rights, permissions and limitations pertaining to operations being performed on a certain conference object.

The set of rights describes the read/write access privileges for the conference object as a whole. This access would usually be granted and defined in terms of giving the read-only or read/write access to clients with certain roles in the conference. Managing this access would require a conferencing system to have access to basic policy information to make the decisions, but doesn't necessarily require an explicit representation in the policy model. As such, for this framework document, the policies represented by the set of rights are reflected in the system realization (Section 7).

The permissions and limits require explicit policy mechanisms and are outside the scope of the data model [XCON-COMMON] and this framework document. However, there are some important policy considerations for a conferencing system. A conferencing system associates specific policies in the form of permissions and limitations with each user in a conferencing system. The permissions may vary depending upon the role associated with a specific conference user identifier. A conferencing system should provide a default user role that only allows participation in a conference through the default signaling means.

The conference object identifier provides access to the data associated with a specific conference. It is important to ensure that elements in the data have individual policy controls to provide flexibility in defining the various roles and specific data elements that may be manipulated by users with specific roles.

In addition, the conference notification interface allows specific data elements to be sent to users that register for such notifications. It is important that the appropriate access control is provided so that only users that are authorized to view specific data elements receive the data in the notifications.

6. Centralized Conferencing Constructs and Identifiers

This section provides details of the identifiers associated with the centralized conferencing framework constructs and the identifiers REQUIRED to address and manage the clients associated with a conferencing system. An overview of the allocation, characteristics, and functional role of the identifiers is provided.

6.1. Conference Identifier

The conference identifier (conference ID) is a call signaling protocol-specific URI that identifies a specific conference focus and its associated conference instance. A conference factory is one method for generating a unique conference ID, to identify and address a conference focus, using a call signaling interface. Details on the use of a conference factory for SIP signaling can be found in [RFC4579]. The conference identifier can also be obtained using the conference control protocol or other, including proprietary, out-of-band mechanisms. To realize the centralized conferencing framework in this document, a conferencing system is REQUIRED to support SIP as the default call signaling protocol. Other call signaling protocols (e.g., ISUP) are OPTIONAL.

6.2. Conference Object

A conference object provides the logical representation of a conference instance in a certain stage, such as a conference blueprint representing a conferencing system's capabilities, the data representing a conference reservation, and the conference state during an active conference. Each conference object is independently addressable through the conference control protocol interface (see Section 8.3). A conferencing system MUST provide a default blueprint representing the basic capabilities provided by that specific conferencing system.

Figure 3 illustrates the relationships between the conference identifier, the focus, and the conference object ID within the context of a logical conference instance, with the conference object corresponding to an active conference.

A conference object representing a conference in the active state can have multiple call signaling conference identifiers; for example, one for each call signaling protocol supported. There is a one-to-one mapping between an active conference object and a conference focus. The focus is addressed by explicitly associating unique conference IDs for each signaling protocol supported by the active conference object.

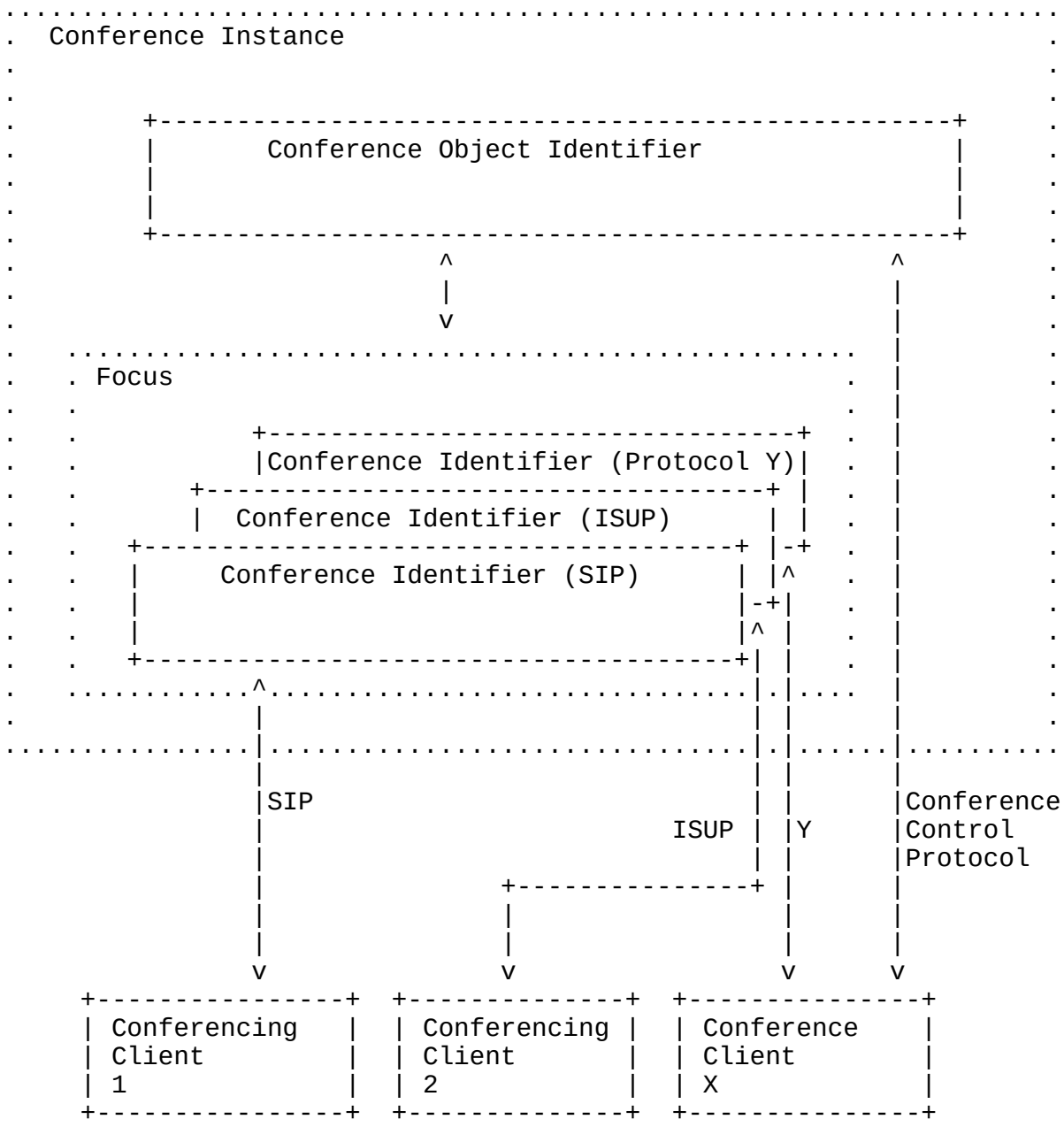


Figure 3: Identifier Relationships for an Active Conference

6.2.1. Conference Object Identifier

In order to make each conference object externally accessible, the conferencing system MUST allocate a unique URI per distinct conference object in the system. The conference object identifier is defined in [XCON-COMMON]. A conferencing system allocates a conferencing object identifier for every conference blueprint, for every conference reservation, and for every active conference. The distribution of the conference object identifier depends upon the specific use case and includes a variety of mechanisms, such as through the conference control protocol mechanism, the data model and conference package, or out-of-band mechanisms such as email.

When a user wishes to create or join a conference and the user does not have the conference object identifier for the specific conference, more general signaling mechanisms apply. A user may have a pre-configured conference object identifier to access the conferencing system or other signaling protocols may be used and the conferencing system maps those to a specific conference object identifier. Once a conference is established, a conference object identifier is REQUIRED for the user to manipulate any of the conferencing data or take advantage of any of the advanced conferencing features. The same notion applies to users joining a conference using other signaling protocols. They are able to initially join a conference using any of the other signaling protocols supported by the specific conferencing system, but the conference object identifier MUST be used to manipulate any of the conferencing data or take advantage of any of the advanced conferencing features. As mentioned previously, the mechanism by which the user learns of the conference object identifier varies and could be via the conference control protocol, using the data model and conference package or entirely out of band mechanisms such as email or a web interface.

The conference object identifier logically maps to other protocol-specific identifiers associated with the conference instance, such as the BFCP 'confid'. The mapping of the conference object identifier can be viewed to contain sensitive information in many conferencing systems. The conferencing system must ensure that the data is protected, that only authorized users can manipulate that information via the conferencing control protocol, and that only the appropriate users receive the information through the notification protocol. In general, this information would not be expected to be distributed to the average conference participant.

6.3. Conference User Identifier

Each user within a conferencing system MUST be allocated a unique conference user identifier. The conference user identifier is defined in [XCON-COMMON]. The conference user identifier is used in association with the conference object identifier to uniquely identify a user within the scope of conferencing system. There is also a requirement for identifying conferencing system users who may not be participating in a conference instance. Examples of these users would be a non-participating 'Floor Control Chair' or 'Media Policy Controller'. The conference user identifier is REQUIRED, in conference control protocol requests, to uniquely determine who is issuing commands, so that appropriate policies can be applied to the requested command.

A typical mode for distributing the user identifier is out of band during conferencing client configuration; thus, the mechanism is outside the scope of the centralized conferencing framework and protocols. However, a conferencing system MUST also be capable of allocating and distributing a user identifier during the first signaling interaction with the conferencing system, such as an initial request for blueprints or adding a new user to an existing conference using the conference control protocol. When a user joins a conference using a signaling-specific protocol, such as SIP for a dial-in conference, a conference user identifier MUST be assigned if one is not already associated with that user. While this conference user identifier isn't required for the participant to join the conference, it is REQUIRED to be allocated and assigned by the conferencing system such that it is available for use for any subsequent conference control protocol operations and/or notifications associated with that conference. For example, the conference user identifier would be sent in any notifications that may be sent to existing participants, such as the moderator, when this user joins.

The conference user identifier is logically associated with the other user identifiers assigned to the conferencing client for other protocol interfaces, such as an authenticated SIP user. The mapping of the conference user identifier to signaling specific user identifiers requires that methods for protecting and securing a user's identity are considered. Section 11.1 addresses "User Authentication and Authorization" and Section 11.2 addresses the "Security and Privacy of User Identity". In addition, the conferencing system MUST ensure the appropriate access control around any internal data structure that maintains this persistent data. This information would typically only be available to a conferencing system administrator.

7. Conferencing System Realization

Implementations based on this centralized conferencing framework can range from systems supporting ad hoc conferences, with default behavior only, to sophisticated systems with the ability to schedule recurring conferences, each with distinct characteristics, being integrated with external resource reservation tools, and providing snapshots of the conference information at any of the stages of the conference life-cycle.

A conference object is the logical representation of a conference instance at a certain stage, such as capabilities description upon conference creation, reservation, activation, etc., which a conferencing system maintains in order to describe the system capabilities and to provide access to the available services provided by the conferencing system. Consequently, this centralized conferencing framework does not mandate the actual usage of the conference object, but rather defines the general cloning tree concept and the mechanisms required for its realization, as described in detail in Section 7.1.

Ad hoc and advanced conferencing examples are provided in Section 7.2 and Section 7.3, with the latter providing additional description of the conference object in terms of the stages of a conference, to support scheduled and other advanced conference capabilities. The scheduling of a conference based on these concepts and mechanisms is then detailed in Section 7.4

As discussed in Section 5.2, the overall policy in terms of permissions and limitations is outside the scope of this framework document. The policies applicable to the conference object as a whole in terms of read/write access would require a conferencing system have access to basic policy information to make the decisions. In the examples in this section, the policies are shown logically associated with the conference objects to emphasize the general requirement for policy functionality necessary for the realization of this framework.

7.1. Cloning Tree

The concept defined in this section is a logical representation only, as it is reflected through the centralized conferencing mechanisms: the URIs and the protocols. Of course, the actual system realization can differ from the presented model. The intent is to illustrate the role of the logical elements in providing an interface to the data, based on conferencing system and conferencing client actions, and describe the resultant protocol implications.

Any conference object in a conferencing system is created by either being explicitly cloned from an existing parent object or being implicitly cloned from a default system conference blueprint. A conference blueprint is a static conference object used to describe a typical conference setting supported by the system. Each system can maintain multiple blueprints, typically each describing a different conferencing type using the conference information format. This document uses the "cloning" metaphor instead of the "inheritance" metaphor because it more closely fits the idea of object replication, rather than a data type re-usage and extension concept.

The cloning operation needs to specify whether or not the link between the parent and child needs to be maintained in the system. If no link between the parent and child exists, the objects become independent and the child is not impacted by any operations on the parent object nor subject to any limitations of the parent object.

Once the new object is created, it can be addressed by a unique conference object URI assigned by the system, as described in Section 6.2.1. By default, the newly created object contains all the data existing in the parent object. The newly created object can expand the data it contains, within the schema types supported by the parent. It can also restrict the read/write access to its objects. However, unless the object is independent, it cannot modify the access restrictions imposed by the parent object.

Any piece of data in the child object can be independently accessed and, by default, can be independently modified without affecting the parent data.

Unless the object is independent, the parent object can enforce a different policy by marking certain data elements as "parent enforceable". The values of these data elements cannot be changed by directly accessing the child object, nor can they be expanded in the child object alone.

Figure 4 illustrates an example of a conference (Parent B), which is created independent of its Parent (Parent A). Parent B creates two child objects, Child 1 and Child 2. Any of the data elements of Parent B can be modified (i.e., there are no "parent enforceable" data elements), and depending upon the element, the changes will be reflected in Child 1 and Child 2 , whereas changes to Parent A will not impact the data elements of Parent B. Any "parent enforceable" data elements, as defined by Parent B, cannot be modified in the child objects.

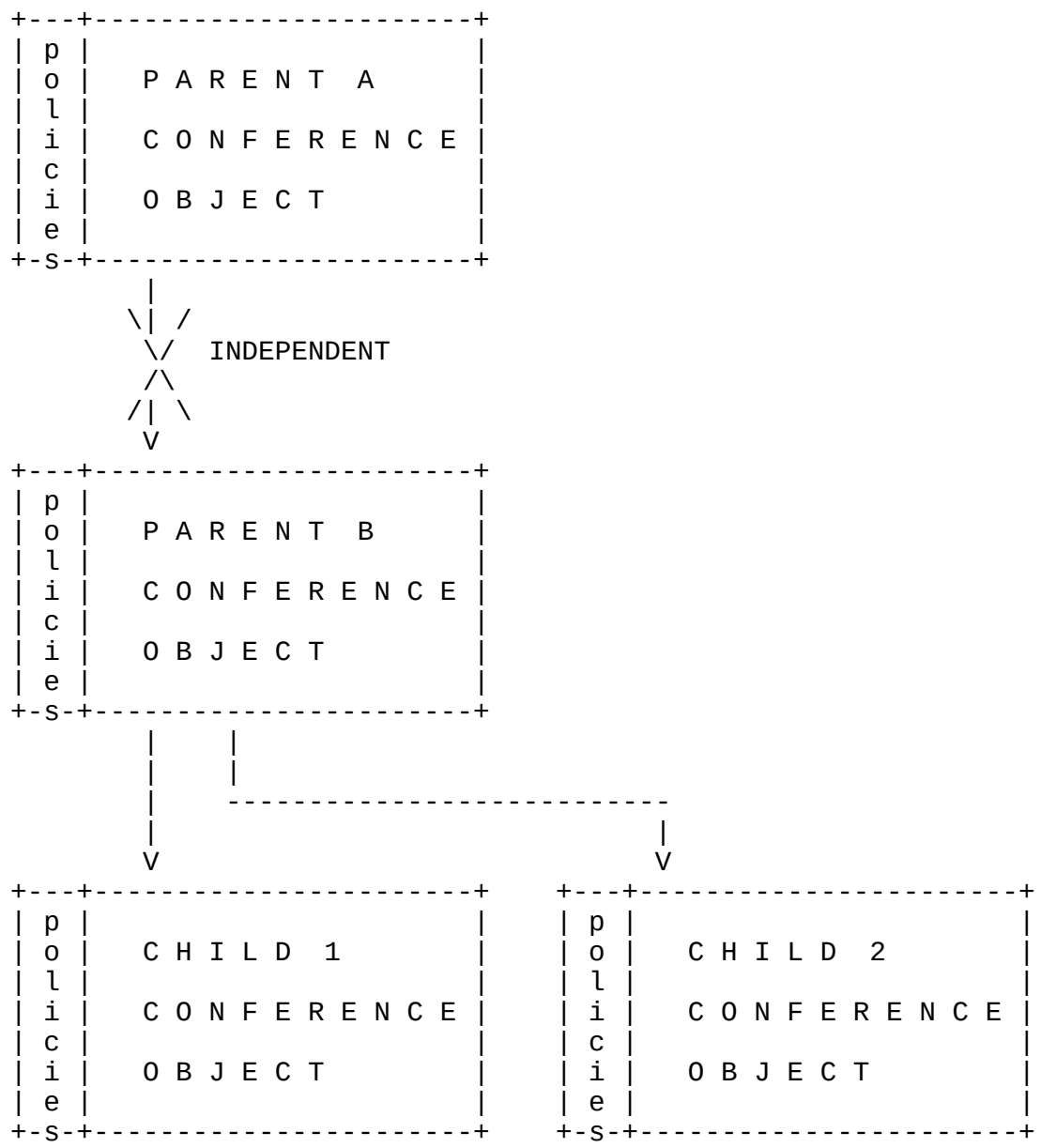


Figure 4: The Cloning Tree

Using the defined cloning model and its tools, the following sections show examples of how different systems based on this framework can be realized.

7.2. Ad Hoc Example

Figure 5 illustrates how an ad hoc conference can be created and managed in a conferencing system. A client can create a conference by establishing a call signaling channel with a conference factory, as specified in Section 6.1. The conference factory can internally select one of the system supported conference blueprints based on the requesting client privileges and the media lines included in the Session Description Protocol (SDP) body.

The selected blueprint with its default values is copied by the server into a newly created conference object, referred to as an 'Active Conference'. At this point, the conference object becomes independent from its blueprint. A new conference object identifier, a new conference identifier, and a new focus are allocated by the server.

During the conference lifetime, an authorized client can manipulate the conference object, by performing operations such as adding participants, using the conference control protocol.

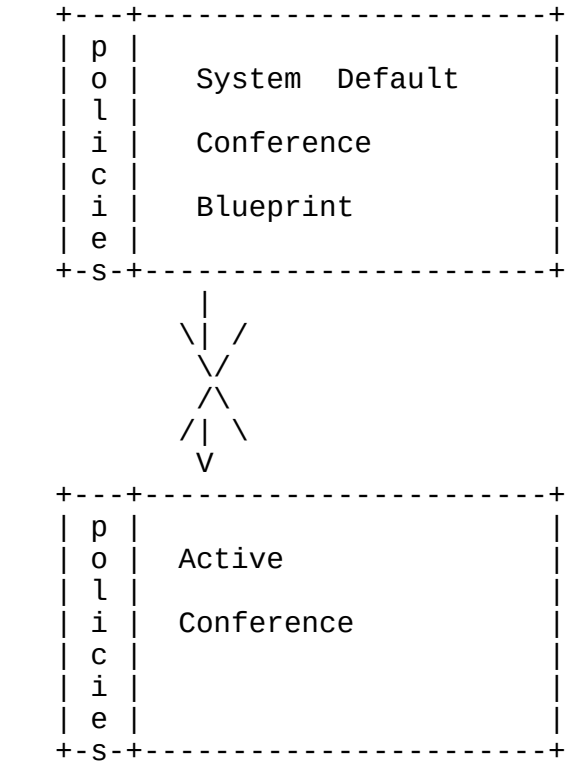


Figure 5: Conference Ad-hoc Creation and Lifetime

7.3. Advanced Example

Figure 6 illustrates how a recurring conference can be specified according to system capabilities, scheduled, reserved, and managed in a conferencing system. A client would first query a conferencing system for its capabilities. This can be done by requesting a list of the conference blueprints the system supports. Each blueprint contains a specific combination of capabilities and limitations of the conference server in terms of supported media types (e.g., audio, video, text, or combinations of these), participant roles, maximum number of participants of each role, availability of floor control, controls available for participants, availability and type of sidebars, the definitions and names of media streams, etc.

The selected blueprint with its default values is cloned by the client into a newly created conference object, referred to as a conference reservation, that specifies the resources needed from the system for this conference instance. At this point, the conference reservation becomes independent from its blueprint. The client can also change the default values, within the system ranges, and add additional information, such as the list of participants and the conference 'start' time, to the conference reservation.

At this point, the client can ask the conference server to create new conference reservations by attaching the conference reservation to the request. As a result, the server can allocate the needed resources, create the additional conference objects for the child conference reservations, and allocate the conference object identifiers for all -- the original conference reservation and for each child conference reservation.

From this point on, any authorized client is able to access and modify each of the conference objects independently. By default, changes to an individual child conference reservation will affect neither the parent conference reservation, from which it was created, nor its siblings.

On the other hand, some of the conference sub-objects, such as the maximum number of participants and the participants list, can be defined by the system as parent enforceable. As a result, these objects can be modified by accessing the parent conference reservation only. The changes to these objects can be applied automatically to each of the child reservations, subject to local policy.

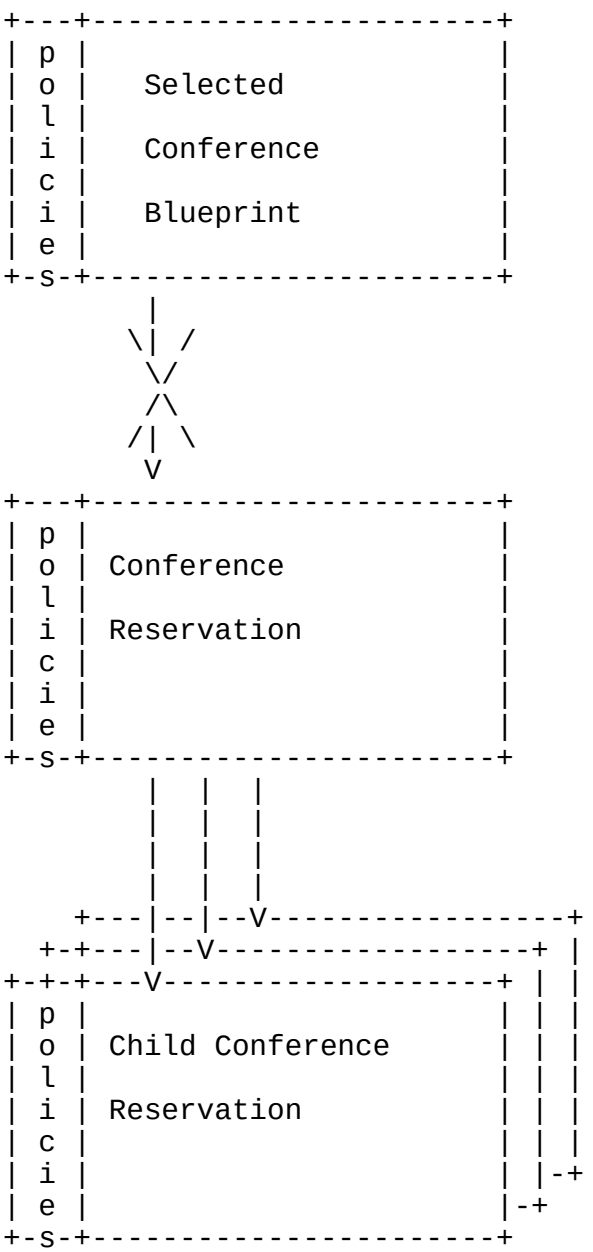


Figure 6: Advanced Conference Definition, Creation, and Lifetime

When the time comes to schedule the conference reservation, either via the system determination that the 'start' time has been reached or via client invocation, an active conference is cloned based on the conference reservation. As in the ad hoc example, the active conference is independent from the parent, and changes to the conference reservation will not impact the active conference. Any

desired changes must be targeted towards the active conference. An example of this interaction is shown in Section 9.1.

7.4. Scheduling a Conference

The capability to schedule conferences forms an important part of the conferencing system solution. An individual conference reservation typically has a specified 'start' and 'end' time, with the times being specified relative to a single specified 'fixed' time (e.g., 'start' = 09.00 GMT, 'end'= 'start'+2), subject to system considerations. In most advanced conferencing solutions, it is possible to not only schedule an individual occurrence of a conference reservation, but also schedule a series of related conferences (e.g., a weekly meeting that starts on Thursday at 09.00 GMT).

To be able to achieve such functionality, a conferencing system needs to be able to appropriately schedule and maintain conference reservations that form part of a recurring conference. The mechanism proposed in this document makes use of the "Internet Calendaring and Scheduling Core Object" specification defined in [RFC2445] in union with the concepts introduced in Section 5 for the purpose of achieving advanced conference scheduling capability.

Figure 7 illustrates a simplified view of a client interacting with a conferencing system. The client is using the conference control protocol to add a new conference reservation to the conferencing system by interfacing with the conference control server. A CCP request contains a valid conference reservation and reference by value to an "iCal" object that contains scheduling information about the conference (e.g., 'start' time, 'end' time).

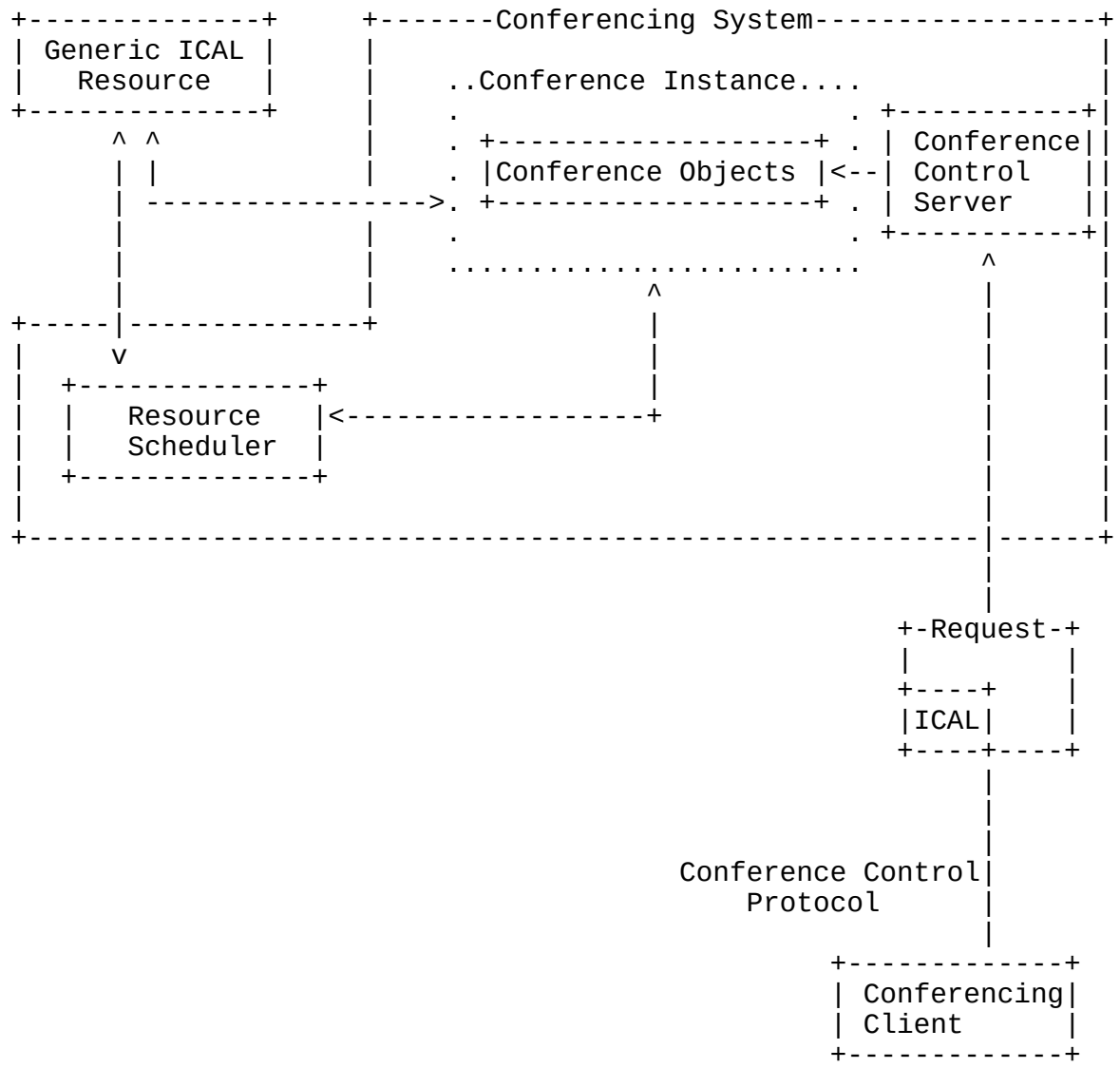


Figure 7: Resource Scheduling

A CCP request to create a new conference reservation is validated, including the associated iCal object, and the resultant conference reservation is created. The conference reservation is uniquely represented within the conferencing system by a conference object identifier (e.g., xcon:hd87928374), as introduced in Section 6.2.1 and defined in [XCON-COMMON]. This unique URI is returned to the client and can be used to reference the conference reservation, if any future manipulations are required (e.g., alter 'start' time), using a CCP request.

The previous example explains how a client creates a basic conference reservation using an iCal reference in association with a conference control protocol. Figure 7 can also be applied when explaining how a series of conferences are scheduled in the system. The description is almost identical with the exception that the iCal definition that is included in a CCP request represents a series of recurring conference instances (e.g., conference 'start' time, 'end' time, occur weekly). The conferencing system will treat this request the same as the first example. The CCP request will be validated, along with the associated iCal object, and the conference reservation is created. The conference reservation and its conference object ID, created for this example, represent the entire series of recurring conference instances rather than a single Conference. If the client uses the conference object ID provided and a CCP request to adjust the conference reservation, every conference instance in the series will be altered. This includes all future occurrences, such as a conference scheduled as an infinite series, subject to the limitations of the available calendaring interface.

A conferencing system that supports the scheduling of a series of conference instances should also be able to support manipulation within a specific range of the series. A good example is a conference reservation that has been scheduled to occur every Monday at 09.00 GMT. For the next three weeks only, the meeting has been altered to occur at 10.00 GMT in an alternative venue. With Figure 7 in mind, the client will construct a CCP request whose purpose is to modify the existing conference reservation for the recurring conference instance. The client will include the conference object ID provided by the conferencing system to explicitly reference the conference reservation within the conferencing system. A CCP request will contain all the required changes to the conference reservation (e.g., change of venue).

The conferencing system matches the incoming CCP request to the existing conference reservation but identifies that the associated iCal object only refers to a range of the existing series. The conferencing system creates a child, by cloning the original conference reservation, to represent the altered conference instances within the series. The cloned child object is not independent of the original parent object, thus preventing any potential conflicts in scheduling (e.g., a change to the whole series ''start' time'). The cloned conference reservation, representing the altered series of conference instances, has its own associated conference object ID that is returned to the client using a CCP response. This conference object ID is then used by the client to make any future alterations on the newly defined sub-series. This process can be repeated any number of times as the newly returned conference object ID representing an altered (cloned) series of conference instances, can

itself be manipulated using a CCP request for the newly created conference object ID . This provides a flexible approach to the scheduling of recurring conference instances.

8. Conferencing Mechanisms

8.1. Call Signaling

The focus is the central component of the conference. Participants interface with the focus using an appropriate call signaling protocol (CSP). Participants request to establish or join a conference using the CSP. After checking the applicable policies, a focus then either accepts the request, sends a progress indication related to the status of the request (e.g., for a parked call while awaiting moderator approval to join), or rejects that request using the call signaling interface.

During an active conference, a conference control protocol can be used to affect the conference state. For example, CCP requests to add and delete participants are communicated to the focus and checked against the conference policies. If approved, the participants are added or deleted using the call signaling to/from the focus.

8.2. Notifications

A conferencing system is responsible for implementing a conference notification service. The conference notification service provides updates about the conference instance state to authorized parties, including participants. A model for notifications using SIP is defined in [RFC3265] with the specifics to support conferencing defined in [RFC4575].

The conference user identifier and associated role are used by the conferencing system to filter the notifications such that they contain only information that is allowed to be sent to that user.

8.3. Conference Control Protocol

The conference control protocol provides for data manipulation and state retrieval for the centralized conferencing data, represented by the conference object. The details of the conference control protocol are provided in separate documents.

8.4. Floor Control

A floor control protocol allows an authorized client to manage access to a specific floor and to grant, deny or revoke access of other conference users to that floor. Floor control is not a mandatory

mechanism for a conferencing system implementation, but it provides advanced media input control features for conference users. A mechanism for floor control within a conferencing system is defined in the "Binary Floor Control Protocol (BFCP)" specification [RFC4582].

Within this framework, a client supporting floor control needs to obtain information for connecting to a floor control server to enable it to issue floor requests. This connection information can be retrieved using information provided by mechanisms such as negotiation using the SDP [RFC4566] offer/answer [RFC3264] exchange on the signaling interface with the focus. Section 11.3 provides a discussion of client authentication of a floor control server.

As well as the client to the floor control server connection information, a client wishing to interact with a floor control server requires access to additional information. This information associates floor control interactions with the appropriate floor instance. Once a connection has been established and authenticated (see [RFC4582] for authentication details), a specific floor control message requires detailed information to uniquely identify a conference, a user, and a floor.

The conference is uniquely identified by the conference object ID per Section 6.2.1. This conference object ID must be included in all floor control messages. When the SDP model is used as described in [RFC4583], this identifier maps to the 'confid' SDP attribute.

Each authorized user associated with a conference object is uniquely represented by a conference user ID per Section 6.3. This conference user ID must be included in all floor control messages. When using SDP offer/answer exchange to negotiate a floor control connection with the focus using the call signaling protocol, the unique conference user identifier is contained in the 'userid' SDP attribute, as defined in [RFC4583].

A media session within a conferencing system can have any number of floors (0 or more) that are represented by the conference identifier. When using SDP offer/answer exchange to negotiate a floor control connection with the focus using the call signaling interface, the unique conference identifier is contained in the 'floorid' SDP attribute, as defined in [RFC4583], e.g., a=floorid:1 m-stream:10 . Each 'floorid' attribute, representing a unique floor, has an 'm-stream' tag containing one or more identifiers. The identifiers represent individual SDP media sessions (as defined using 'm=' from SDP) using the SDP 'Label' attribute, as defined in [RFC4574].

9. Conferencing Scenario Realizations

This section addresses how advanced conferencing scenarios, many of which have been described in [RFC4597], are realized using this centralized conferencing framework. The objective of this section is to further illustrate the model, mechanisms, and protocols presented in the previous sections and also serves to validate that the model, mechanisms, and protocols are sufficient to support advanced conferencing scenarios.

The scenarios provide a high level primitive view of the necessary operations and general logic flow. The details shown in the scenarios are for illustrative purposes only and don't necessarily reflect the actual structure of the conference control protocol messages nor the detailed data, including states, which are defined in separate documents. It should be noted that not all entities impacted by the request are shown in the diagram (e.g., focus), but rather the emphasis is on the new entities introduced by this centralized conferencing framework.

9.1. Conference Creation

There are different ways to create a conference. A participant can create a conference using call signaling means only, such as SIP detailed in [RFC4579]. For a conferencing client to have more flexibility in defining the characteristics and capabilities of a conference, a conferencing client would implement a conference control protocol client. By using a conference control protocol, the client can determine the capabilities of a conferencing system and its various resources.

Figure 8 provides an example of one client "Alice" determining the conference blueprints available for a particular conferencing system and creating a conference based on the desired blueprint.

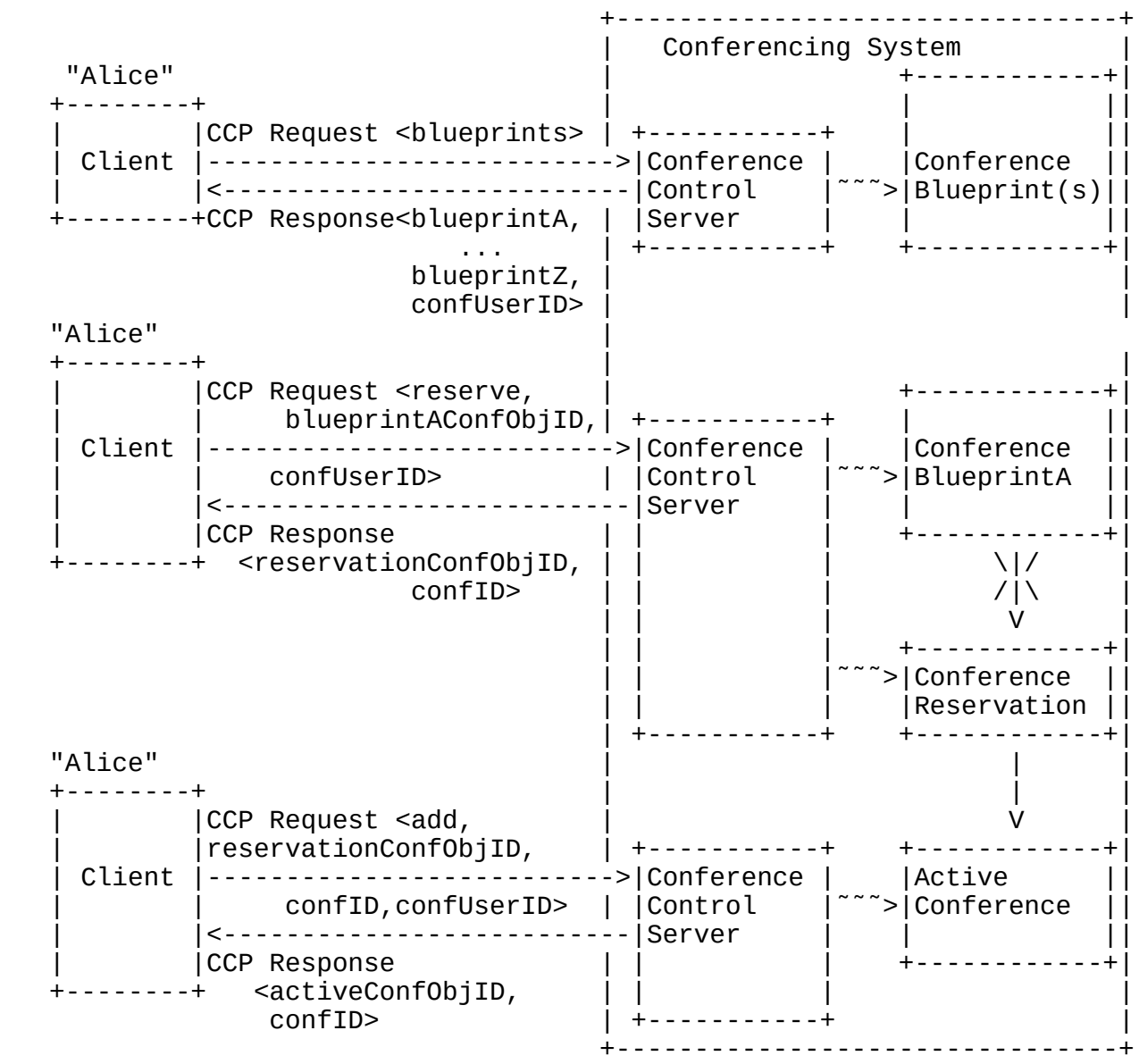


Figure 8: Client Creation of Conference Using Blueprints

Upon receipt of the conference control protocol request for blueprints, the conferencing system would first authenticate "Alice" (and allocate a conference user identifier, if necessary) and then ensure that "Alice" has the appropriate authority based on system policies to receive any blueprints supported by that system. Any blueprints that "Alice" is authorized to use are returned in a response, along with the conference user ID.

Upon receipt of the conference control protocol response containing the blueprints, "Alice" determines which blueprint to use for the conference to be created. "Alice" creates a conference object based on the blueprint (i.e., clones) and modifies applicable fields, such as membership list and 'start' time. "Alice" then sends a request to the conferencing system to create a conference reservation based upon the updated blueprint.

Upon receipt of the conference control protocol request to "reserve" a conference based upon the blueprint in the request, the conferencing system ensures that the blueprint received is a valid blueprint (i.e., the values of the various field are within range). The conferencing system determines the appropriate read/write access of any users to be added to a conference based on this blueprint (using membership, roles, etc.). The conferencing system uses the received blueprint to clone a conference reservation. The conferencing system also reserves or allocates a conference ID to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the reservation through the conference instance.

Upon receipt of the conference control protocol response to reserve the conference, "Alice" can now create an active conference using that reservation or create additional reservations based upon the existing reservations. In this example, "Alice" has reserved a meetme conference bridge. Thus, "Alice" provides the conference information, including the necessary conference ID, to desired participants. When the first participant, including "Alice", requests to be added to the conference, an active conference and focus are created. The focus is associated with the conference ID received in the request. Any participants that have the authority to manipulate the conference would receive the conference object identifier of the active conference object in the response.

9.2. Participant Manipulations

There are different ways to affect a participant state in a conference. A participant can join and leave the conference using call signaling means only, such as SIP. This kind of operation is called "1st party signaling" and does not affect the state of other participants in the conference.

Limited operations for controlling other conference participants (a so called "3rd party control") through the focus, using call signaling only, may also be available for some signaling protocols. For example, "Conferencing for SIP User Agents" [RFC4579] shows how SIP with REFER can be used to achieve this functionality.

In order to perform richer conference control, a user client needs to implement a conference control protocol client. By using a conference control protocol, the client can affect its own state, the state of other participants, and the state of various resources (such as media mixers) that may indirectly affect the state of any of the conference participants.

Figure 9 provides an example of one client "Alice" impacting the state of another client "Bob". This example assumes an established conference. In this example, "Alice" wants to add "Bob" to the conference.

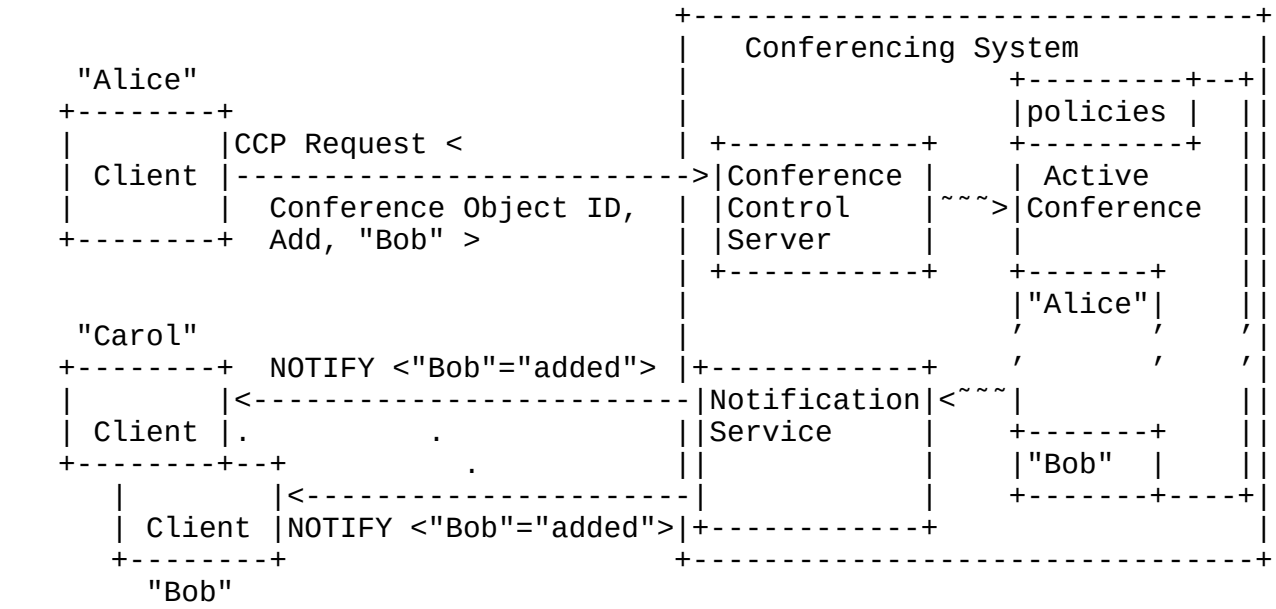


Figure 9: Client Manipulation of Conference - Add a Party

Upon receipt of the conference control protocol request to "add" a party ("Bob") in the specific conference as identified by the conference object ID, the conferencing system ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. The conferencing system must also determine whether "Bob" is already a user of this conferencing system or whether he is a new user.

If "Bob" is a new user for this conferencing system, a Conference User Identifier is created for Bob. Based upon the addressing information provided for "Bob" by "Alice", the call signaling to add "Bob" to the conference is instigated through the focus.

Once the call signaling indicates that "Bob" has been successfully added to the specific conference, per updates to the state, and depending upon the policies, other participants (including "Bob") may be notified of the addition of "Bob" to the conference via the conference notification service.

9.3. Media Manipulations

There are different ways to manipulate the media in a conference. A participant can change its own media streams by, for example, sending re-INVITE with new SDP content using SIP only. This kind of operation is called "1st party signaling" and they do not affect the state of other participants in the conference.

In order to perform richer conference control, a user client needs to implement a conference control protocol client. By using a conference control protocol, the client can manipulate the state of various resources, such as media mixers, which may indirectly affect the state of any of the conference participants.

Figure 10 provides an example of one client "Alice" impacting the media state of another client "Bob". This example assumes an established conference. In this example, the client, "Alice" whose Role is "moderator" of the conference, wants to mute "Bob" on a medium-size multi-party conference, as his device is not muted (and he’s obviously not listening to the call) and background noise in his office environment is disruptive to the conference.

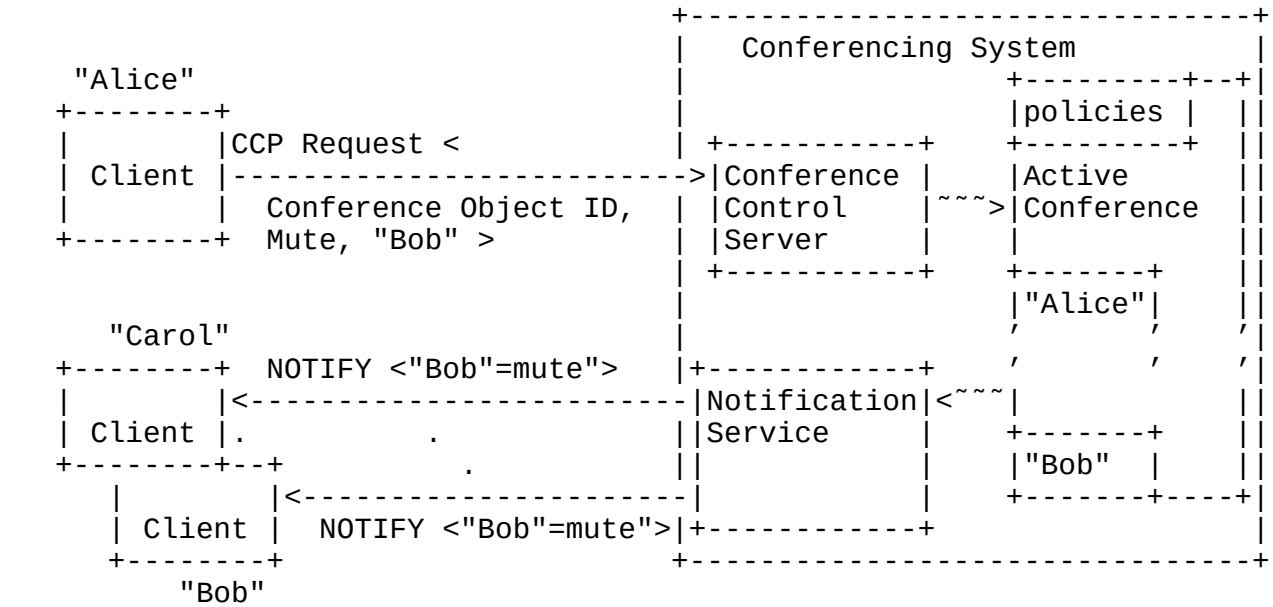


Figure 10: Client Manipulation of Conference - Mute a Party

Upon receipt of the conference control protocol request to "mute" a party ("Bob") in the specific conference as identified by the conference object ID, the conference server ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. "Bob's" status is marked as "recvonly" and the conference object is updated to reflect that "Bob's" media is not to be "mixed" with the conference media.

Depending upon the policies, other participants (including "Bob") may be notified of this change via the conference notification service.

9.4. Sidebar Manipulations

A sidebar can be viewed as a separate Conference instance that only exists within the context of a parent conference instance. Although viewed as an independent conference instance, it can not exist without a parent. A sidebar is created using the same mechanisms employed for a standard conference, as described in Section 7.1.

A conference object representing a sidebar is created by cloning the parent associated with the existing conference and updating any information specific to the sidebar. A sidebar conference object is

implicitly linked to the parent conference object (i.e., it is not an independent object) and is associated with the parent conference object identifier, as shown in Figure 11. A conferencing system manages and enforces the parent and appropriate localized restrictions on the sidebar conference object (e.g., no members from outside the parent conference instance can join, sidebar conference cannot exist if parent conference is terminated, etc.).

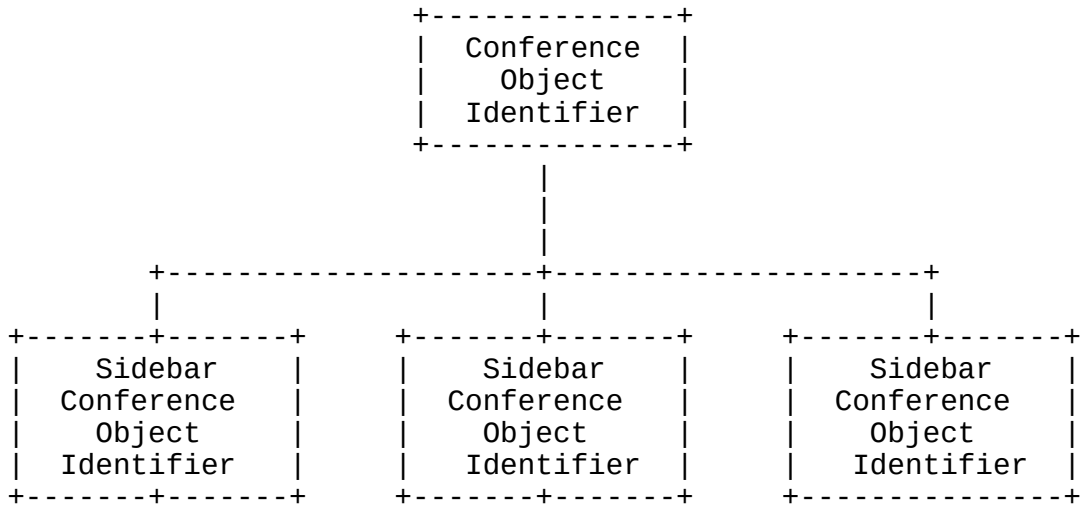


Figure 11: Conference Object Mapping

Figure 11 illustrates the relationship between a conference object and associated sidebar conference objects within a conferencing system. Each sidebar conference object has a unique conference object identifier, as described in Section 6.2.1. The main conference object identifier acts as a top level identifier for associated sidebars.

A sidebar conference object identifier follows many of the concepts outlined in the cloning tree model described in Section 7.1. A sidebar conference object contains a subset of members from the original conference object. Properties of the sidebar conference object can be manipulated by a Conference Control Protocol using the unique conference object identifier for the sidebar. It is also possible for the top level conference object to enforce policy on the sidebar object (similar to parent enforceable, as discussed in Section 7.1).

9.4.1. Internal Sidebar

Figure 12 provides an example of one client "Alice" involved in active conference with "Bob" and "Carol". "Alice" wants to create a sidebar to have a side discussion with "Bob" while still viewing the video associated with the main conference. Alternatively, the audio from the main conference could be maintained at a reduced volume. "Alice" initiates the sidebar by sending a request to the conferencing system to create a conference reservation based upon the active conference object. "Alice" and "Bob" would remain on the roster of the main conference, such that other participants could be aware of their participation in the main conference, while an internal-sidebar conference is occurring.

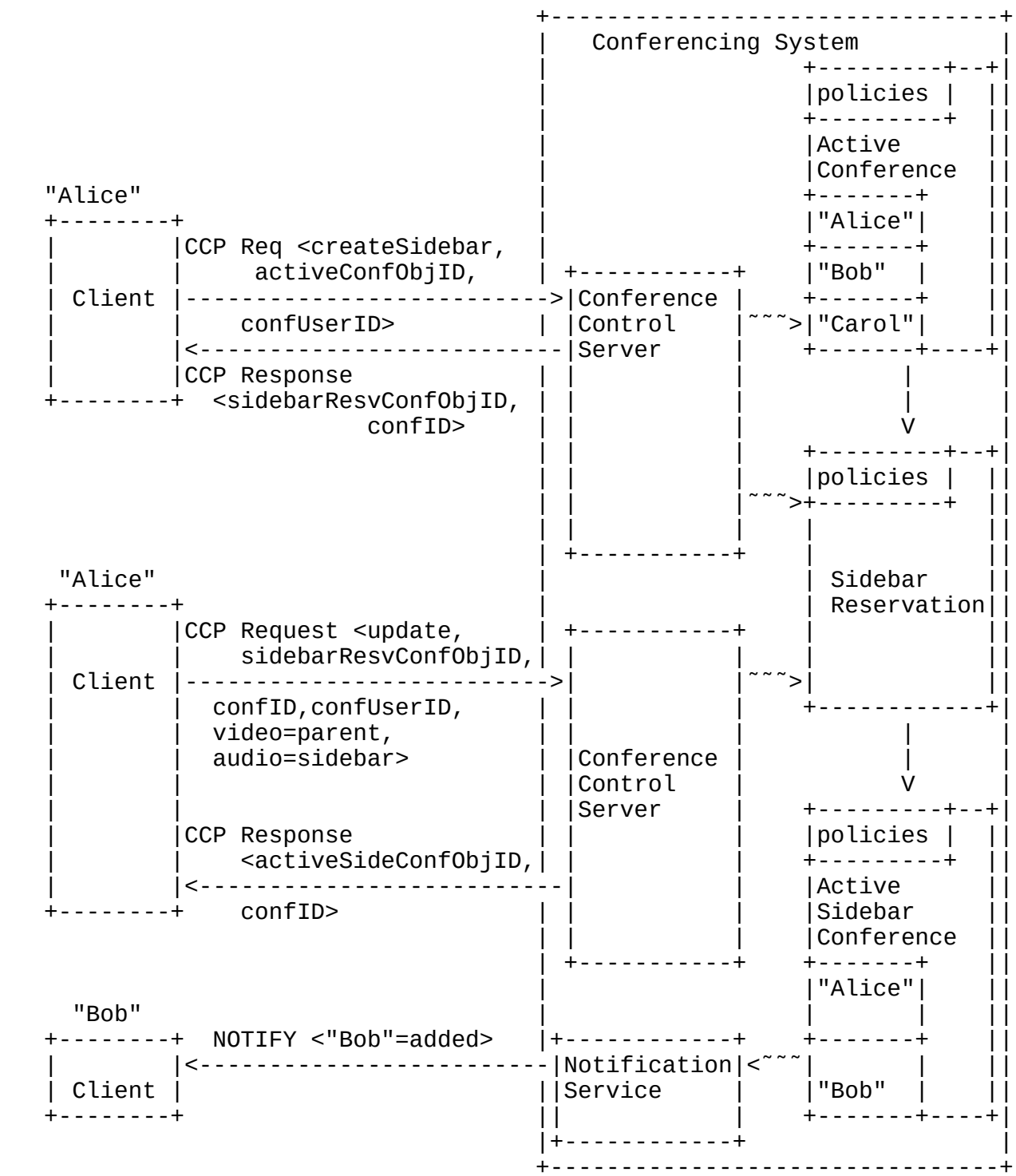


Figure 12: Client Creation of a Sidebar Conference

Upon receipt of the conference control protocol request to "reserve" a new sidebar conference, based upon the active conference received in the request, the conferencing system uses the received active conference to clone a conference reservation for the sidebar. As discussed previously, the sidebar reservation is NOT independent of the active conference (i.e., parent). The conferencing system also reserves or allocates a conference ID to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the sidebar reservation through the conference instance.

Upon receipt of the conference control protocol response to reserve the conference, "Alice" can now create an active conference using that reservation or create additional reservations based upon the existing reservations. In this example, "Alice" wants only "Bob" to be involved in the sidebar, thus she manipulates the membership. "Alice" also only wants the video from the original conference and wants the audio to be restricted to the participants in the sidebar. Alternatively, "Alice" could manipulate the media values to receive the audio from the main conference at a reduced volume. "Alice" sends a conference control protocol request to update the information in the reservation and to create an active conference.

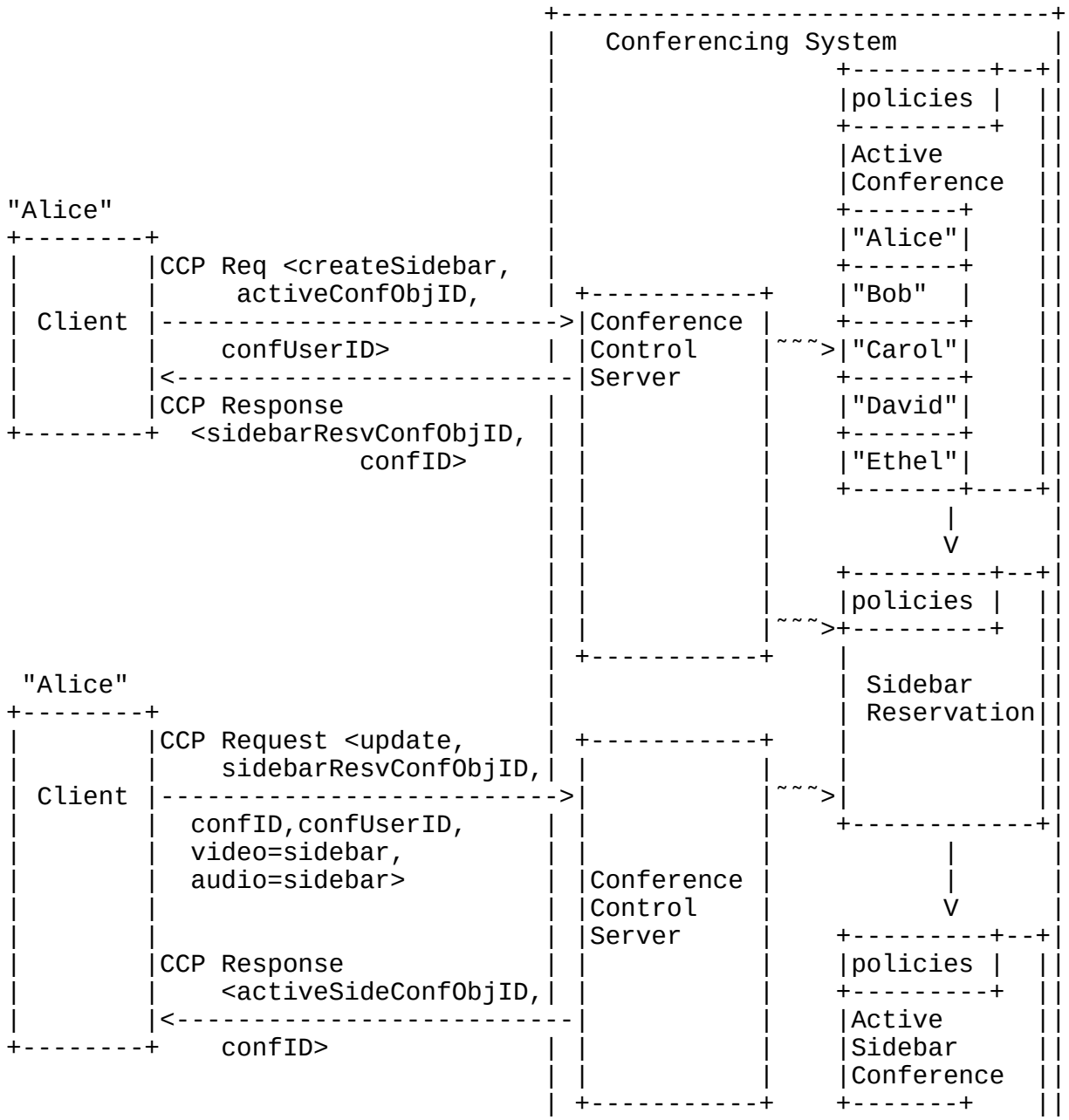
Upon receipt of the conference control protocol request to update the reservation and to create an active conference for the sidebar, as identified by the conference object ID, the conferencing system ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. The conferencing system must also validate the updated information in the reservation, ensuring that a member like "Bob" is already a user of this conferencing system.

Depending upon the policies, the initiator of the request (i.e., "Alice") and the participants in the sidebar (i.e., "Bob") may be notified of his addition to the sidebar via the conference notification service.

9.4.2. External Sidebar

Figure 13 provides an example of one client "Alice" involved in an active conference with "Bob", "Carol", "David", and "Ethel". "Alice" gets an important text message via a whisper from "Bob" that a critical customer needs to talk to "Alice", "Bob", and "Ethel". "Alice" creates a sidebar to have a side discussion with the customer "Fred" including the participants in the current conference with the

exception of "Carol" and "David", who remain in the active conference. "Alice" initiates the sidebar by sending a request to the conferencing system to create a conference reservation based upon the active conference object. "Alice", "Bob", and "Ethel" would remain on the roster of the main conference in a hold state. Whether or not the hold state of these participants is visible to other participants depends upon the individual and local policy.



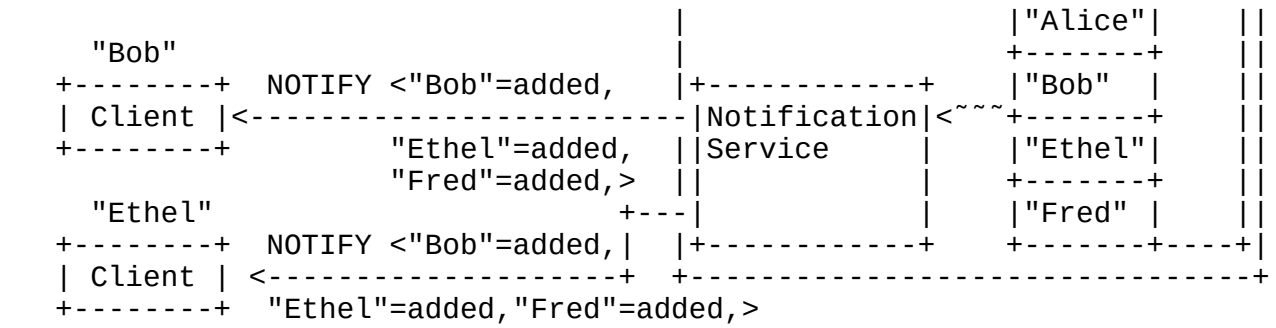


Figure 13: Client Creation of an External Sidebar

Upon receipt of the conference control protocol request to "reserve" a new sidebar conference, based upon the active conference received in the request, the conferencing system uses the received active conference to clone a conference reservation for the sidebar. As discussed previously, the sidebar reservation is NOT independent of the active conference (i.e., parent). The conferencing system also reserves or allocates a conference ID to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the sidebar reservation through the conference instance.

Upon receipt of the conference control protocol response to reserve the conference, "Alice" can now create an active conference using that reservation or create additional reservations based upon the existing reservations. In this example, "Alice" wants only "Bob" and "Ethel", along with the new participant "Fred" to be involved in the sidebar; thus, she manipulates the membership. "Alice" sets the media such that the participants in the sidebar don’t receive any media from the main conference. "Alice" sends a conference control protocol request to update the information in the reservation and to create an active conference.

Upon receipt of the conference control protocol request to update the reservation and to create an active conference for the sidebar, as identified by the conference object ID, the conferencing system ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. The conferencing system must also validate the updated information in the reservation, ensuring whether members like "Fred" are already a user of this conferencing system or whether he

is a new user. Since "Fred" is a new user for this conferencing system, a conference user identifier is created for "Fred". Based upon the addressing information provided for "Fred" by "Alice", the call signaling to add "Fred" to the conference is instigated through the focus.

Depending upon the policies, the initiator of the request (i.e., "Alice") and the participants in the sidebar (i.e., "Bob" and "Ethel") may be notified of his addition to the sidebar via the conference notification service.

9.5. Floor Control Using Sidebars

Floor control with sidebars can be used to realize conferencing scenarios such as an analyst briefing. In this scenario, the conference call has a panel of speakers who are allowed to talk in the main conference. The other participants are the analysts, who are not allowed to speak unless they have the floor. To request access to the floor, they have to join a new sidebar with the moderator and ask their question. The moderator can also whisper to each analyst what their status/position in the floor control queue, similar to the example in Figure 15.

Figure 14 provides an example of the configuration involved for this type of conference. As in the previous sidebar examples, there is the main conference along with a sidebar. "Alice" and "Bob" are the main participants in the conference, with "A1", "A2", and "A3" representing the analysts. The sidebar remains active throughout the conference, with the moderator, "Carol", serving as the chair. As discussed previously, the sidebar conference is NOT independent of the active conference (i.e., parent). The analysts are provided the conference object ID associated with the active sidebar when they join the main conference. The conferencing system also allocates a conference ID to be used for any subsequent manipulations of the sidebar conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the active sidebar conference through the conference instance. The analysts are permanently muted while in the main conference. The analysts are moved to the sidebar when they wish to speak. Only one analyst is given the floor at a given time. All participants in the main conference receive audio from the sidebar conference, as well as audio provided by the panelists in the main conference.

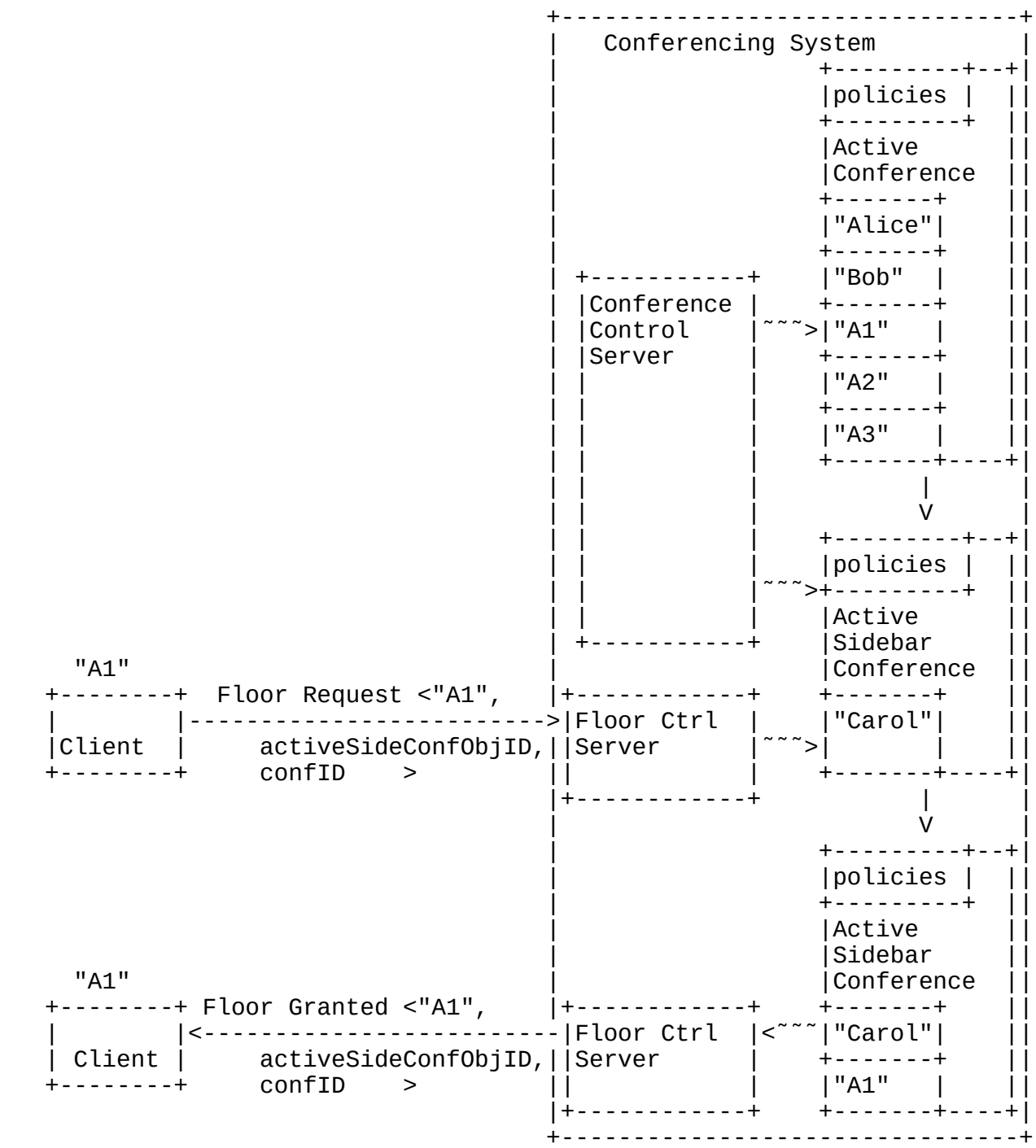


Figure 14: Floor Control with Sidebars

When "A1" wishes to ask a question, he sends a Floor Request message to the floor control server. Upon receipt of the request, the floor control server notifies the moderator, "Carol" of the active sidebar conference, who's serving as the floor chair. Note, that this signaling flow is not shown in the diagram. Since no other analysts have yet requested the floor, "Carol" indicates to the floor control server that "A1" may be granted the floor.

9.6. Whispering or Private Messages

The case of private messages can be handled as a sidebar with just two participants, similar to the example in Section 9.4.1, but rather than using audio within the sidebar, "Alice" could add an additional text based media stream to the sidebar. The other context, referred to as whisper, in this document refers to situations involving one time media targeted to specific user(s). An example of a whisper would be an announcement injected only to the conference chair or to a new participant joining a conference.

Figure 15 provides an example of one user "Alice" who's chairing a fixed length conference with "Bob" and "Carol". The configuration is such that only the chair is providing a warning when there are only 10 minutes left in the conference. At that time, "Alice" is moved into a sidebar created by the conferencing system and only "Alice" receives the announcement.

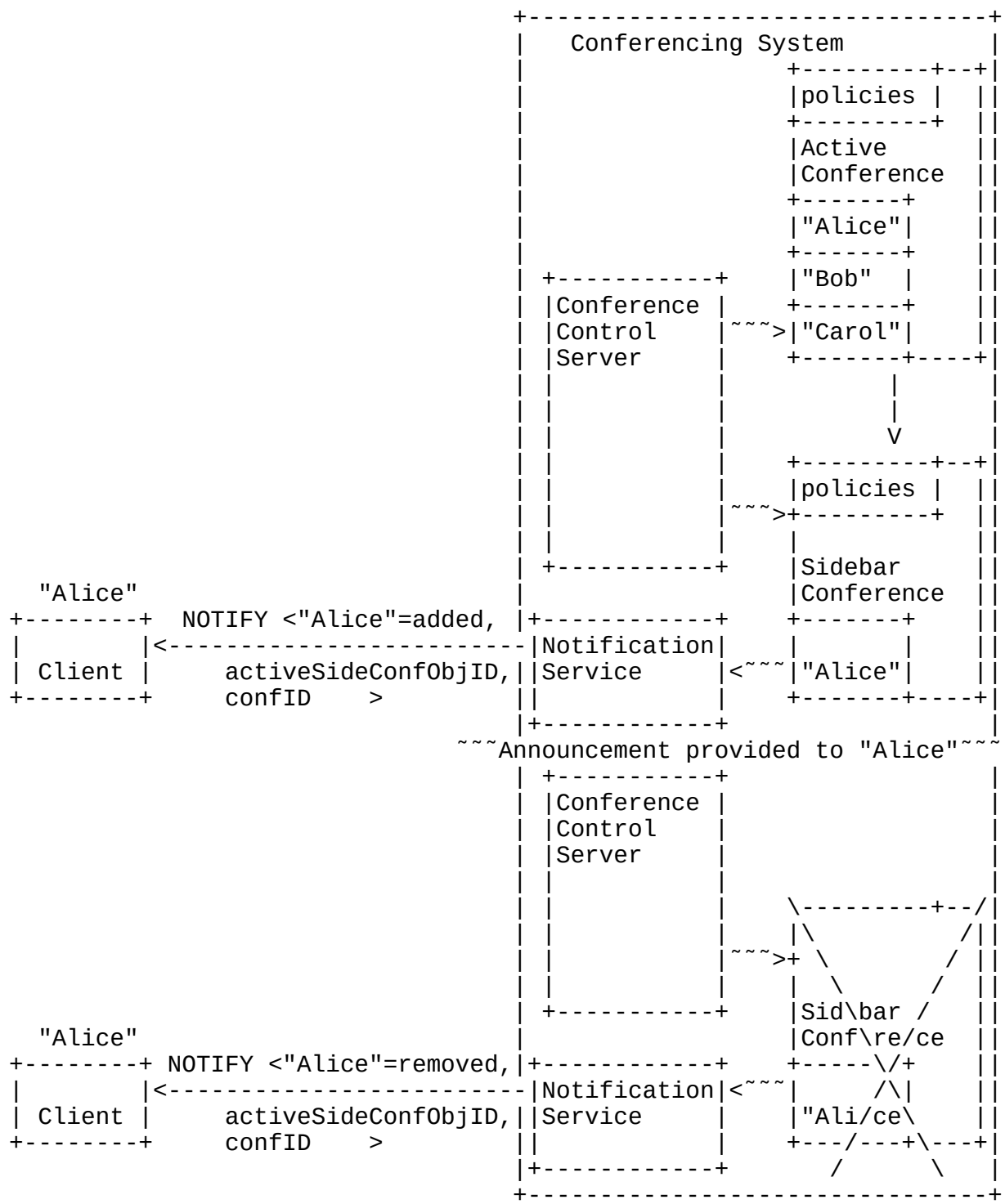


Figure 15: Whisper

When the conferencing system determines that there are only 10 minutes left in the conference which "Alice" is chairing, rather than creating a reservation as was done for the sidebar in Section 9.4.1, the conferencing system directly creates an active sidebar conference, based on the active conference associated with "Alice". As discussed previously, the sidebar conference is NOT independent of the active conference (i.e., parent). The conferencing system also allocates a conference ID to be used for any subsequent manipulations of the sidebar conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the active sidebar conference through the conference instance.

Immediately upon creation of the active sidebar conference, the announcement media is provided to "Alice". Depending upon the policies, "Alice" may be notified of her addition to the sidebar via the conference notification service. "Alice" continues to receive the media from the main conference.

Upon completion of the announcement, "Alice" is removed from the sidebar, and the sidebar conference is deleted. Depending upon the policies, "Alice" may be notified of her removal from the sidebar via the conference notification service.

9.7. Conference Announcements and Recordings

Each participant can require a different type of announcement and/or recording service from the system. For example, "Alice", the conference chair, could be listening to a roll call while "Bob" may be using a telephony user interface to create a sidebar. Some announcements would apply to all the participants such as "This conference will end in 10 minutes". Recording is often required to capture the names of participants as they join a conference, typically after the participant has entered an access code, as discussed in Section 9.8. These recorded names are then announced to all the participants as the new participant is added to the active conference.

An example of a conferencing recording and announcement, along with collecting the dual tone multi-frequency (DTMF), within the context of this framework, is shown in Figure 16.

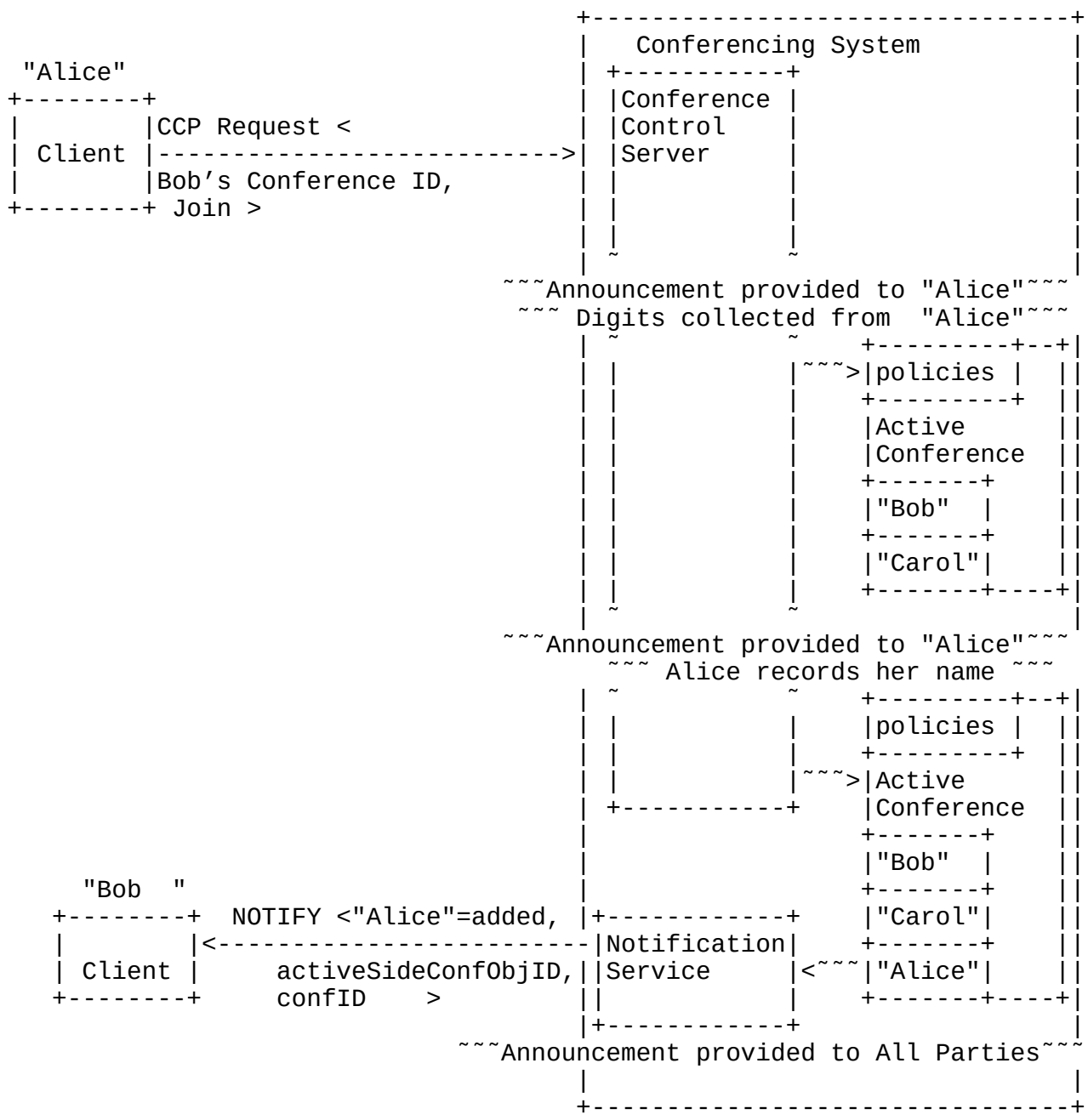


Figure 16: Recording and Announcements

Upon receipt of the conference control protocol request from "Alice" to join "Bob's" conference, the conferencing system maps the identifier received in the request to the conference object

representing "Bob's" active conference. The conferencing system determines that a password is required for this specific conference; thus, an announcement asking "Alice" to enter the password is provided to "Alice". Once "Alice" enters the password, it is validated against the policies associated with "Bob's" active conference. The conferencing system then connects to a server that prompts and records "Alice"'s name. The conferencing system must also determine whether "Alice" is already a user of this conferencing system or whether she is a new user.

If "Alice" is a new user for this conferencing system, a conference user identifier is created for "Alice". Based upon the addressing information provided by "Alice", the call signaling to add "Alice" to the conference is instigated through the focus.

Once the call signaling indicates that "Alice" has been successfully added to the specific conference, per updates to the state, and depending upon the policies, other participants (e.g., "Bob") are notified of the addition of "Alice" to the conference via the conference notification service, and an announcement is provided to all the participants indicating that "Alice" has joined the conference.

9.8. Monitoring for DTMF

The conferencing system also needs the capability to monitor for DTMF from each individual participant. This would typically be used to enter the identifier and/or access code for joining a specific conference.

An example of DTMF monitoring, within the context of the framework elements, is shown in Figure 16.

9.9. Observing and Coaching

The capability to observe a conference allows a participant with the appropriate authority to listen to the conference, typically without being an active participant and often as a hidden participant. When such a capability is available on a conferencing system, there is often an announcement provided to each participant as they join the conference indicating the call may be monitored. This capability is useful in the context of conferences, which might be experiencing technical difficulties, thus allowing a technician to listen in to evaluate the type of problem.

This capability could also apply to call center applications as it provides a mechanism for a supervisor to observe how the agent is handling a particular call with a customer. This scenario can be

handled by a supervisor adding themselves to the existing active conference, with a listen only audio media path. Whether the agent is aware of when the supervisor joins the call should be configurable.

Taking the supervisor capability one step further introduces a scenario whereby the agent can hear the supervisor, as well as the customer. The customer can still only hear the agent. This scenario would involve the creation of a sidebar involving the agent and the supervisor. Both the agent and supervisor receive the audio from the main conference. When the agent speaks, it is heard by the customer in the main conference. When the supervisor speaks, it is heard only by the agent in the sidebar conference.

An example of observing and coaching is shown in Figure 17. In this example, call center agent "Bob" is involved in a conference with customer "Carol". Since "Bob" is a new agent and "Alice" sees that he has been on the call with "Carol" for longer than normal, she decides to observe the call and coach "Bob" as necessary.

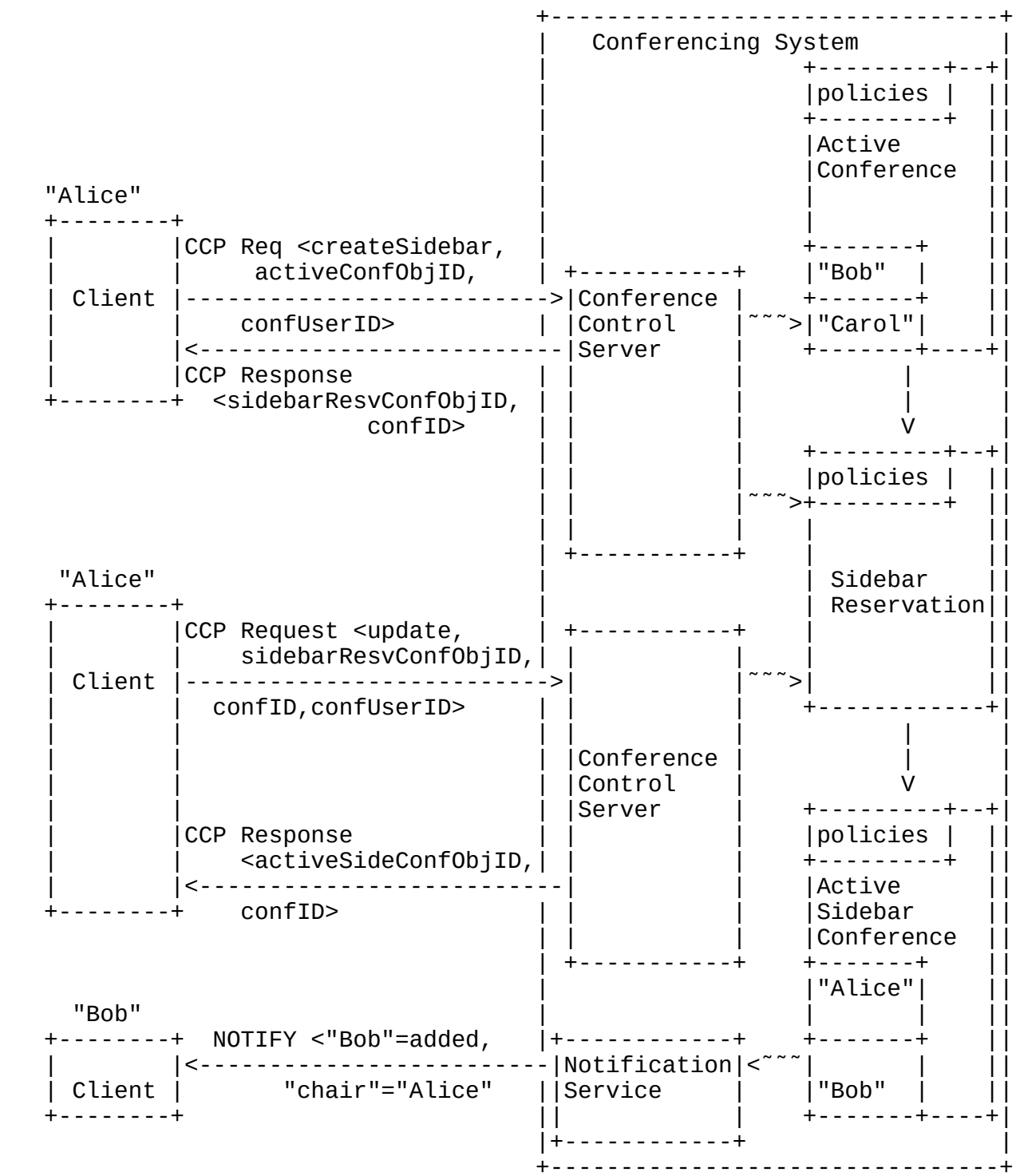


Figure 17: Supervisor Creating a Sidebar for Observing/Coaching

Upon receipt of the conference control protocol request from "Alice" to "reserve" a new sidebar conference, based upon the active conference received in the request, the conferencing system uses the received active conference to clone a conference reservation for the sidebar. The conferencing system also reserves or allocates a conference ID to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the sidebar reservation through the conference instance.

Upon receipt of the conference control protocol response to reserve the conference, "Alice" can now create an active conference using that reservation or create additional reservations based upon the existing reservations. In this example, "Alice" wants only "Bob" to be involved in the sidebar; thus, she manipulates the membership. "Alice" also wants the audio to be received by herself and "Bob" from the original conference, but wants any outgoing audio from herself to be restricted to the participants in the sidebar, whereas "Bob's" outgoing audio should go to the main conference, so that both "Alice" and the customer "Carol" hear the same audio from "Bob". "Alice" sends a conference control protocol request to update the information in the reservation and to create an active conference.

Upon receipt of the conference control protocol request to update the reservation and to create an active conference for the sidebar, as identified by the conference object ID, the conferencing system ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. Based upon the addressing information provided for "Bob" by "Alice", the call signaling to add "Bob" to the sidebar with the appropriate media characteristics is instigated through the focus.

"Bob" is notified of his addition to the sidebar via the conference notification service; thus, he is aware that "Alice", the supervisor, is available for coaching him through this call.

10. Relationships between SIP and Centralized Conferencing Frameworks

The SIP Conferencing Framework [RFC4353] provides an overview of a wide range of centralized conferencing solutions known today in the conferencing industry. The document introduces a terminology and logical entities in order to systemize the overview and to show the common core of many of these systems. The logical entities and the listed scenarios in the SIP Conferencing Framework are used to

illustrate how SIP [RFC3261] can be used as a signaling means in these conferencing systems. The SIP Conferencing Framework does not define new conference control protocols to be used by the general conferencing system. It uses only basic SIP [RFC3261], the SIP Conferencing for User Agents [RFC4579], and the SIP Conference Package [RFC4575] for basic SIP conferencing realization.

This centralized conferencing framework document defines a particular centralized conferencing system and the logical entities implementing it. It also defines a particular data model and refers to the set of protocols (beyond call signaling means) to be used among the logical entities for implementing advanced conferencing features. The purpose of the XCON Working Group and this framework is to achieve interoperability between the logical entities from different vendors for controlling different aspects of advanced conferencing applications.

The logical entities defined in the two frameworks are not intended to be mapped one-to-one. The two frameworks differ in the interpretation of the internal conferencing system decomposition and the corresponding operations. Nevertheless, the basic SIP [RFC3261], the SIP Conferencing for User Agents [RFC4579], and the SIP Conference Package [RFC4575] are fully compatible with both framework documents. The basis for compatibility is provided by including the basic data elements defined in [RFC4575] in the Conference Information Data Model for Centralized Conferencing (XCON) [XCON-COMMON]. User agents that only support [RFC4579] and do not support the Conferencing Control Protocol are still provided basic SIP conferencing, but cannot take advantage of any of the advanced features.

11. Security Considerations

There are a wide variety of potential attacks related to conferencing, due to the natural involvement of multiple endpoints and the many, often user-invoked, capabilities provided by the conferencing system. Examples of attacks include the following: an endpoint attempting to listen to conferences in which it is not authorized to participate, an endpoint attempting to disconnect or mute other users, and theft of service by an endpoint in attempting to create conferences it is not allowed to create.

There are several issues surrounding security of this conferencing framework. One set of issues involves securing the actual protocols and the associated authorization mechanisms. This first set of issues should be addressed in the specifications specific to the protocols described in Section 8 and policy control. The protocols used for manipulation and retrieval of confidential information need

to support a confidentiality and integrity mechanism. Similar requirements apply for the floor control protocols. Section 11.3 discusses an approach for client authentication of a floor control server. It is RECOMMENDED that all the protocols that interface with the conferencing system implement Transport Layer Security (TLS).

There are also security issues associated with the authorization to perform actions on the conferencing system to invoke specific capabilities. Section 5.2 discusses the policies associated with the conference object to ensure that only authorized entities are able to manipulate the data to access the capabilities. Another set of issues involves the privacy and security of the identity of a user in the conference, which is discussed in Section 11.2.

A final issue is related to Denial of Service (DoS) attacks on the conferencing system itself. In order to minimize the potential for DoS attacks, it is recommended that conferencing systems require user authentication and authorization for any client participating in a conference. It is recommended that the specific signaling and media protocols include mechanisms to minimize the potential for DoS.

11.1. User Authentication and Authorization

Many policy authorization decisions are based on the identity of the user or the role that a user may have. Conferencing systems typically require authentication of users to validate their identity. There are several ways that a user might authenticate its identity to the system. For users joining a conference using one of the call signaling protocols, the user authentication mechanisms for the specific protocol often suffice. For the case of users joining the conference via SIP signaling or using the conference control protocol, TLS is RECOMMENDED.

The conferencing system may also know (e.g., out-of-band mechanisms) about specific users and assign passwords to allow these users to be authorized. In some cases (e.g., Public Switched Telephone Network (PSTN) users), additional authorization may be required to allow the user to participate in the conference. This may be in the form of an Interactive Voice Response (IVR) system or other means. The users may also be authorized by knowing a particular conference ID and a Personal Identification (PIN) for it. Sometimes, a PIN is not required and the conference ID is used as a shared secret.

In the cases where a user is authorized via multiple mechanisms, it is up to the conferencing system to correlate (if desired) the authorization of the call signaling interface with other authorization mechanisms. A conferencing system can avoid the problem with multiple mechanisms by restricting the methods by which

a conference can be joined. For example, many conferencing systems that provide a web interface for conferences correlate the PSTN call signaling by forcing a dial-out mode for joining the conference. Thus, there is only the need for a single PIN or password to join the conference.

When a conferencing system presents the identity of authorized users, it may choose to provide information about the way the identity was proven or verified by the system. A user may also come as a completely unauthenticated user into the system -- this fact needs also to be communicated to interested parties.

When guest users interact with the system, it is often in the context of a particular conference. In this case, the user may provide a PIN or a password that is specific to the conferences and authorizes the user to take on a certain role in that conference. The guest user can then perform actions that are allowed to any user with that role.

The term password refers to the usual, reasonable sized and hard to predict shared secret. Today, users often have passwords containing up to 30 bits (8-16 characters) of entropy. A PIN is a special password case -- a shared secret that is only numeric and often contains a fairly small number of bits (often as few as 10 bits or 3 digits). When conferencing systems are used for audio on the PSTN, there is often a need to authenticate using a PIN. Typically, if the user fails to provide the correct PIN a few times in a row, the PSTN call is disconnected. The rate of making the calls and getting to the point to enter a PIN makes it fairly hard to do an exhaustive search of the PIN space even for 4 digit PINs. When using a high speed interface to connect to a conferencing system, it is often possible to do thousands of attempts per second and the PIN space could quickly be searched. Because of this, it is not appropriate to use PINs for authorization on any of the interfaces that provide fast queries or many simultaneous queries.

Once a user is authenticated and authorized through the various mechanisms available on the conferencing system, a conference user identifier is associated with any signaling specific user identifiers that may have been used for authentication and authorization. This conference user identifier may be provided to a specific user through the conference notification interface and will be provided to users that interact with the conferencing system using the conference control protocol. This conference user identifier is required for any subsequent operations on the conference object.

11.2. Security and Privacy of Identity

This conferencing system has an idea of the identity of a user, but this does not mean it can reveal this identity to other users, due to privacy considerations. Users can select various options for revealing their identity to other users. A user can be "hidden" such that other users can not see they are participants in the conference, "anonymous" such that users can see that another user is there, but not see the identity of the user, or they can be "public" where other users can see their identity. If there are multiple "anonymous" users, other parties will be able to see them as independent "anonymous" parties and will be able to tell how many "anonymous" parties are in the conference. Note, that the visibility to other participants is dependent on their roles. For example, users' identity (including "anonymous" and "hidden") may be displayed to the moderator or administrator, subject to a conferencing system's local policies. "Hidden" status is often used by automated or machine participants of a conference (e.g., call recording) and is also used in many call center situations.

Since a conferencing system based on this framework allocates a unique conference user identifier for each user of the conferencing system, it is not necessary to distribute any signaling specific user identifier to other users or participants. Access to any signaling specific user identifiers can be controlled by applying the appropriate access control to the signaling specific user identifiers in the data schema.

11.3. Floor Control Server Authentication

The floor control protocol contains mechanisms that clients can use to authenticate servers, and that servers can use to authenticate clients, as described in Section 9 of [RFC4582]. The precise mechanisms used for such authentication can vary depending on the call control protocol used. Clients using call control protocols that employ an SDP offer/answer model, such as SIP, use the mechanism described in Section 8 of [RFC4583]. Clients using other call control protocols make use of the mechanisms described in the BFCP Connection Establishment document [RFC5018].

12. Acknowledgements

This document is a result of architectural discussions among IETF XCON Working Group participants. The authors would like to thank Henning Schulzrinne for the "Conference Object Tree" proposal and general feedback, Cullen Jennings for providing input for the "Security Considerations" section, and Keith Lantz, Dave Morgan, Oscar Novo, Roni Even, Umesh Chandra, Avshalom Houri, Sean Olson,

Rohan Mahy, Brian Rosen, Pierre Tane, Bob Braudes, Gregory Sperounes, and Gonzalo Camarillo for their reviews and constructive input. In addition, the authors would like to thank Scott Brim for his gen-art review comments and Kurt Zeilenga for his secdir review comments.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

13.2. Informative References

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC3265] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC2445] Dawson, F. and Stenerson, D., "Internet Calendaring and Scheduling Core Object Specification (iCalendar)", RFC 2445, November 1998.

[RFC4245] Levin, O. and R. Even, "High-Level Requirements for Tightly Coupled SIP Conferencing", RFC 4245, November 2005.

[RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, February 2006.

[RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, August 2006.

[RFC4376] Koskelainen, P., Ott, J., Schulzrinne, H., and X. Wu, "Requirements for Floor Control Protocols", RFC 4376, February 2006.

[RFC4597] Even, R. and N. Ismail, "Conferencing Scenarios", RFC 4597, August 2006.

[RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", BCP 119, RFC 4579, August 2006.

[RFC4582] Camarillo, G., Ott, J., and K. Drage, "The Binary Floor Control Protocol (BFCP)", RFC 4582, November 2006.

[RFC4574] Levin, O. and G. Camarillo, "The Session Description Protocol (SDP) Label Attribute", RFC 4574, August 2006.

[RFC4583] Camarillo, G., "Session Description Protocol (SDP) Format for Binary Floor Control Protocol (BFCP) Streams", RFC 4583, November 2006.

[XCON-COMMON] Novo, O., Camarillo, G., Morgan, D., and R. Even, "Conference Information Data Model for Centralized Conferencing (XCON)", Work in Progress, March 2008.

[RFC4975] Campbell, B., Mahy, R., and C. Jennings, "The Message Session Relay Protocol (MSRP)", RFC 4975, September 2007.

[RFC5018] Camarillo, G., "Connection Establishment in the Binary Floor Control Protocol (BFCP)", RFC 5018, September 2007.

Authors' Addresses

Mary Barnes
Nortel
2201 Lakeside Blvd
Richardson, TX

EMail: mary.barnes@nortel.com

Chris Boulton
Avaya
Building 3
Wern Fawr Lane
St Mellons
Cardiff, South Wales CF3 5EA

EMail: cboulton@avaya.com

Orit Levin
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

EMail: oritl@microsoft.com

Full Copyright Statement

Intellectual Property