A Framework and Data Model for Centralized Conferencing
draft-ietf-xcon-framework-01

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on January 16, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

This document defines the framework for Centralized Conferencing (XCON), which is applicable to participants using different call control signaling protocols and exchanging media over networks with potentially different characteristics. The XCON Framework defines the XCON data model, the XCON logical entities, the naming
conventions, and outlines the standard conferencing protocols (complementary to the call control signalling protocols) for building advanced conferencing applications. The framework binds all the defined components together for the benefit of conferencing system builders.
# Table of Contents

1. Introduction ...................................................... 4  
2. Conventions and Terminology .................................... 4  
3. Overview .......................................................... 7  
4. XCON Data Model .................................................. 9  
   4.1 Common Conference Information ............................... 11  
   4.2 Conference Template ......................................... 12  
   4.3 Conference policies ......................................... 12  
5. XCON Constructs and Identifiers ................................ 13  
   5.1 Conference Identifier ....................................... 13  
   5.2 Conference Instance ......................................... 13  
   5.3 Conference Object ........................................... 14  
       5.3.1 Conference Object Identifier ......................... 16  
   5.4 Conference User Identifier ................................ 17  
6. Conferencing System Realization ................................ 18  
   6.1 Cloning Tree ................................................ 19  
   6.2 Ad-hoc Example .............................................. 21  
   6.3 Advanced Example ........................................... 22  
   6.4 Scheduling a ‘Conference Object for Reservation’ ....... 24  
7. Conferencing Mechanisms .......................................... 28  
   7.1 Call Control Signalling ...................................... 28  
   7.2 Notifications ............................................... 28  
   7.3 Conference Control Protocol ................................ 28  
       7.3.1 CPCP .................................................. 29  
       7.3.2 CCCP ................................................ 30  
       7.3.3 CSCP ............................................... 30  
       7.3.4 NETCONF .......................................... 30  
       7.3.5 SOAP ............................................... 31  
   7.4 Floor Control ............................................... 31  
8. Conferencing Scenario Realizations .............................. 33  
   8.1 Participant Manipulations .................................... 33  
   8.2 Media Manipulations ......................................... 35  
   8.3 Sidebar Manipulations ...................................... 36  
   8.4 Whispering or Private Messages ............................. 38  
   8.5 Conference Announcements and Recordings .................. 38  
9. Relationships between SIPPING and XCON Frameworks .......... 38  
10. Security Considerations ......................................... 38  
11. IANA Considerations ............................................ 40  
12. Acknowledgements ............................................... 40  
13. Open Issues .................................................... 41  
14. Changes since last Version ..................................... 41  
15. References ...................................................... 43  
   15.1 Normative References ....................................... 43  
   15.2 Informative References .................................... 43  
Authors’ Addresses .................................................. 45  
Intellectual Property and Copyright Statements .................. 47
1. Introduction

This document defines the framework for Centralized Conferencing (XCON), which is applicable to participants using various call signalling protocols (such as SIP, H.323, Jabber, HTML, PSTN, etc.) and exchanging media over networks with potentially different characteristics.

The XCON Framework defines the XCON data model, the XCON logical entities, the naming conventions, and outlines the standard conferencing protocols (complementary to the call control signalling protocols) for building advanced conferencing applications. The framework binds all the defined components together for the benefit of conferencing system builders.

The XCON Framework is compatible with the functional model presented in the SIPPING Conferencing Framework [9]. Section 9 of this document discusses the relationship between the XCON Framework and the SIPPING Conferencing framework, in the context of the XCON architecture.

2. Conventions and Terminology

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, RFC 2119 [1] and indicate requirement levels for compliant implementations.

The XCON Framework document both generalizes, when appropriate, the SIPPING Conferencing Framework [9] terminology and introduces new concepts as listed below.

Active Conference: The term Active Conference refers to a Conference Object that’s been created (for example, using a "blueprint") and "activated" via the allocation of its identifiers (i.e. Conference Object Identifier, Conference Identifier) and the associated Focus.

Call (Control) Signalling: The protocol used between a participant and a Focus. In this context, the term "call" means a channel or session used for media streams establishment. Protocol examples include, but are not limited to, SIP, H.323, MSRP, Jabber, HTML and PSTN signalling. Other than references to general functionality required (e.g. establishment and teardown), details of these protocols are outside the scope of this document.
Common Conference Information: The data type (i.e. the XML schema) which is used to represent the fixed information part of a Conference Object. It includes a common set of definitions for basic conference features, such as conference identifiers, membership, signalling, capabilities, media, etc.

Conference Control Protocol (CCP): A protocol used by clients to manipulate a Conference Object [Section 7.3].

Conference Factory: A logical entity that, upon request, generates unique URI(s) to identify and represent a conference Focus. The Conference Factory is typically used in the conference creation process via a signalling interface and non-signalling methods such as Conference Control Protocol [Section 7.3] and proprietary mechanisms.

Conference Identifier (ID): The call signalling protocol-specific URI that uniquely identifies a conference Focus and its associated Conference Instance.

Conference Instance: An instantiation of a Conference Object. The Conferencing System associates a Conference Instance with the Conference Identifier(s). There is a one-to-one mapping between a Conference Instance and a conference Focus.

Conference Object: A representation of 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 available services provided by the Conferencing System. All Conference Objects are of type ‘Conference Object Type’ which is comprised of two distinct sub components; the ‘Common Conference Information’ and the ‘Conference Template’ (see definitions in this section for details).

Conference Object Identifier (ID): A [TBD schema name] URI which uniquely identifies a Conference Object and is being used by a Conference Control Protocol [Section 7.3].

Conference policies: A term which is used to collectively refer to a virtual set of rights, permissions and limitations pertaining to operations being performed on a certain Conference Object. The term is described in more details in Section 4.3.


Conferencing System: The term used to refer to a conferencing solution (system) that is based on the set of specifications and is built using the protocols and the data model defined by the XCON working group within the IETF.

Conference Template: The data type (i.e. the XML schema) which is used to represent the variable information part of the Conference Object. It can be included in the Conference Object to represent enhanced conferencing capabilities (e.g. media mixers, etc.) and/or user interface abstraction.
Floor: A term which is used to refer to a set of data or resources associated with a Conference Instance, for which a conference participant (or group of participants) is granted temporary input access.

Floor Chair: A Floor control protocol compliant client (human participant or automated entity) who is authorized to manage access to one floor (grants, denies, or revokes a floor). The floor chair does not have to be a participant in the Conference Instance.

Focus: Focus is a logical entity that maintains the call signalling interface between each participating client (i.e. participant) and the Conference Instance. As such, the Focus acts as an endpoint for each of the supported signalling 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. There is a one-to-one mapping between the Focus and its Conference Instance. Focus is addressed by explicitly associated unique conference URIs for each signalling protocol, supported for its Conference Instance.

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

Media Mixer: A logical entity that has the capability to combine media inputs of the same type (or/and 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/RTCP (defined in RFC 3550[7]), Message Session Relay Protocol (defined in [25]), etc.

Registered Conference Document: An official standards document (RFC) that defines and registers a Conference Template schema with the appropriate standards body (IANA). A ‘Registered Conference Document’ also includes any complimentary textual information in relation to the conference template schema and its meaning.

Role: Represents differing membership categories that a conference participant can assume within a conference. Each category has a different set of conference operations that a participant can perform. A default (e.g. Standard Conference Participant) ‘Role’ will always exist which provides a standard user with a set of basic conference operations. Any user with appropriate authentication and authorization may assume a role that has a wider set of conference operations that are otherwise not available (to a standard Conference Participant) – e.g. A ‘Role’ called ‘Conference Moderator’ may exist that has additional conference operations such as ‘modify conference end time’.
3. Overview

A centralized conference is an association of endpoints (called conference participants) with a central endpoint (called a conference Focus) where the Focus has direct peer-wise relationships with the participants by maintaining a separate call control signalling interface with each. Consequently, in this tightly coupled model, the call control signalling graph is always a star.

The most basic conference supported 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 signalling functionality with a participant serving as the Focus; the SIPPING Conferencing Framework [9] together with the SIP Call Control Conferencing for User Agents[15] documents address these types of scenarios.

An XCON-compliant Conferencing System, in addition to the basic features, can offer richer functionality including dedicated conferencing applications with explicitly defined capabilities, reserved reoccurring conferences, and the standard protocols for managing and controlling different conference aspects.

The core requirements for tightly managed centralized conferencing are outlined in [10]. These requirements are applicable for conferencing systems using various call control signalling protocols, not restricted to SIP alone. Additional conferencing requirements are provided in [12], [13], and [14].

The XCON architecture is built around a fundamental concept of a Conference Object. A Conference Object is a logical representation of a Conference Instance. For conference creation, a Conference Object provides a "blueprint" representing the System Capabilities. A conference object also provides the logical representation of a conference during each of the various stages of a conference (e.g. reservation, active, completed, etc.). A Conference Object is accessible using XCON protocols (e.g. a Conference Control Protocol [Section 7.3].

A Conferencing System can support any subset of the conferencing functions depicted in the Conferencing System logical decomposition in Figure 1 and described in this document. Nevertheless, typically advanced functions could not be implemented without implementing others. For example, the Notification Service is an essential component required for implementing almost any advanced functionality.
and being used, among other things, for correlation of information
(such as list of participants with their media streams) between
otherwise distinct functions.

Figure 1: Conferencing System Logical Decomposition.

The media graph of a conference can be centralized, de-centralized,
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 de-centralized (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 de-centralized media case, however, the details of such are left for future study by the XCON WG charter.

Section 4 of this document provides more details on the Conference Object. Section 5 provides an overview of the identifiers necessary to address and manage the Conference Objects, Instances and Users associated with a Conferencing System. Section 6 of this document describes how a Conferencing System is logically built using the defined data model and how the Conference Objects are maintained. Section 7 describes the fundamental conferencing mechanisms and provides a high level overview of the XCON protocols. Section 8 then provides realizations of various conferencing scenarios, detailing the manipulation of the Conference Objects using XCON defined protocols. Section 9 of this document summarizes the relationship between the XCON Framework and the SIPPING Conferencing Framework.

4. XCON Data Model

The XCON data model defined in this framework has no strict separation between conference membership, conference media information and the related policies (i.e. the capabilities and limitations for each). This approach meets the requirement in many conference control operations to enable synchronized access to the conference policies as a whole, to the conference state as a whole, and for receiving notifications about changes to either.

A Conference Object is of type ‘Conference Object Type’ which is comprised of two distinct components: the ‘Common Conference Information Type’ and the ‘Conference Template Type’, as illustrated in Figure 2. Each of these types is a placeholder for including potentially multiple sub-types.
Since, in an XCON-compliant system the same Conference Object Type is used for representation of a conference for different purposes, such as expressing Conferencing System capabilities, reserving conferencing resources or reflecting the state of ongoing conferences, each of the two components (i.e., the Common Conference Information and the Conference Template) is syntactically optional to be included in a particular Conference Object. Section 6 describes the usage semantics of the Conference Objects.

[Editor’s Note: The exact XML schema of the Conference Object (i.e.
the organization of the Common Conference Information and the
Conference Template) is an open issue (Discussion Point 4 on the
mailing list), which has been summarized as the three potential
alternatives below:

1. The top-level information is the template section, and it
contains a subsection that is the common conference information.
This has the advantage that the schema can all be well defined:
because the common conference information is known at the time the
template is developed, the appropriate schema definition can be
inserted into the template schema. The downside is that this setup
requires navigation of the template information to get to the common
information, which is likely to be manipulated most frequently.

2. The top-level information is the common conference information,
which contains the template information. This has the advantage that
clients don’t even need to care about the template being used to
allow rendering and control over basic conference functionality (which
will suffice for many clients (e.g. those with limited screen). The
downside is that the common conference information schema must
include an extension point to allow new templates to hook into the
schema. This may make schema validation more difficult.

3. The template information and common conference information are
carried as separate objects (e.g. using multipart MIME). This
provides the benefits of allowing completely separate schema,
straightforward schema validation, and easy access to the common
conference information. The downside is that any mechanism for
separating the schema is going to add some amount of protocol
complexity and the need for synchronization between the two data
objects. Note that it has been argued that it is increasingly
difficult to find a potential client of the XCON protocols that
doesn’t already support multipart MIME).

The model put forth in this document is the result of the consensus
reached during the XCON second interim meeting in Boston and depicts
option 2 above. With slight adaptations it can also support option
3. ]

4.1 Common Conference Information

The common conference information section contains the core
information that is utilized in any conference and can be abstracted
regardless of the specific conference media nature (e.g. the mixing
algorithms performed, the advanced floor control applied, etc.).
Typically, participants with read-only access to the conference
information would be interested in this common information only.
The basic common conference information can be represented using the conference-info-type schema as defined in [11]. This schema contains the definitions for representation of the Conference Object capabilities, membership, roles, call control signalling and media statuses relevant to different stages of the conference life-cycle.

New XCON specifications can extend the basic conference-info-type and introduce additional schemas to be used within the common conference information type placeholder.

4.2 Conference Template

The concept of a "Conference Template" is introduced to abstract the complexity and the details of the "mixer" and other conferencing features and to allow for easy user interface abstraction for advanced Conferencing Systems.

Each Conference Template needs to be registered with IANA. The IANA registration needs to point to an RFC having the text description of the feature behavior and optionally the XML definition allowing the feature presentation, configuration, and management. RFCs of this kind are referred by XCON documents as a "Registered Conference Document".

Typically, a conference template would contain the information about the specific media mixing details, the associated client roles and the available floor controls. This information would allow authorized clients to manipulate mixer’s behavior and the resultant distribution of the media to all or individual participants. By doing so, a client can change its own state and/or state of other participants in the conference.

A conference template can also include an abstract user interface definition in terms of sliders, radio boxes, etc. for simplifying user interaction with a specific non-trivial feature.

4.3 Conference policies

Conference policies is the term used to collectively refer to a virtual set of rights, permissions and limitations pertaining to operations being performed on a certain Conference Object.

The virtual 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. For details see [TBD].
The permissions and limits are specified as an integral part of the Conference Object Type, with data objects containing the allowed ranges for other data objects (e.g. maximum number of participants) and lists of clients allowed to perform certain operations on a conference object. For example, the "allowed to join" list of participants is consulted to decide who is allowed to join. The entries in the list can specify the identity of an individual user (joe@example.com), a role, a domain (*@example.com), etc. For details see [TBD].

A more general rule mechanism, beyond the functionality provided by the permissions and limits, is an item for future study for the XCON WG.

5. XCON Constructs and Identifiers

This section provides details of the identifiers associated with the XCON constructs (e.g. Conference Object and Conference Instance) and the identifiers necessary 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.

5.1 Conference Identifier

The Conference Identifier (ID) is the call signalling protocol-specific URI that uniquely identifies a Conference Instance and a conference Focus. The Conference Factory is the logical entity that generates the unique Conference ID to identify and represent a conference Focus. The Conference Factory is typically used in the conference creation process via a signalling interface and non-signalling methods such as Conference Control Protocol [Section 7.3] and proprietary mechanisms.

5.2 Conference Instance

A Conference Instance is an internal implementation construct of a conference, which is accessible by call signalling means, thus no explicit identifier is required. Note that a Conference Instance can have more than a single Conference Identifier, for example, for each call signalling protocol supported. There is a one-to-one mapping between a Conference Instance and a conference Focus. The Focus is addressed by explicitly associating unique Conference IDs for each signalling protocol supported by its Conference Instance.

A single Conference Instance can be internally mapped to a single or multiple Conference Objects; each of them accessible by a Call Control protocol.
5.3 Conference Object

A Conference Object is the logical representation of a Conference Instance at a certain stage, such as a "blueprint" representing a Conferencing System's capabilities, the data representing a reserved or scheduled conference, or the conference state during an active conference. The Conferencing System exposes this data as separate objects to allow independent access. Consequently, the XCON specifications allow the association of multiple Conference Objects with a single Conference Instance.

Figure 3 illustrates the relationships between the Conference Identifier, the Focus and the Conference Object Identifier within the context of a logical Conference Instance, with the Conference Object corresponding to an ongoing conference (i.e. a conference in the active state). The Conference Object is identified by a single or a set of call signaling Conference Identifiers, with a one-to-one mapping, as shown in the figure.

The Conference Objects corresponding to additional conference stages are addressing using CCP [Section 7.3]. CCP will define the necessary logical naming conventions for addressing additional Conference Objects, within the context of the cloning tree concept introduced in Section 6.
Figure 3: Conference Identifier, Focus, Conference Instance and Conference Object Identifier Relationships.
5.3.1 Conference Object Identifier

In order to make each Conference Object externally accessible, the Conferencing System allocates a unique URI per distinct Conference Object in the system, as defined in [ref:TBD]. A Conference Control Protocol [as outlined in Section 7.3] can then be used for directly manipulating a particular Conference Object and for obtaining its current state.

The Conference Object URI acts as a top level identifier for an ‘umbrella style’ construct within the Conference System for the purpose of identifying incoming connections for complimentary XCON protocols (e.g. BFCP). This implicit binding provides a structured mapping of the various XCON protocols with the associated Conference Object Identifier. As an example, Figure 4 illustrates how BFCP connections fall under the general Conference Object identifier umbrella.

![Conference Object Mapping Diagram]

The Conference Object Identifier acts as the top level key in the identification process. The BFCP protocol, as defined in [24], defines the ‘conf-id’ identifier which represents a conference instance within Floor Control. BFCP also defines the ‘floorid’ identifier for specific floors within a Conference instance. When created within the Conference System, the ‘conf-id’ has a 1:1 mapping to the unique XCON Conference Object Identifier. Using the BFCP...
example, the Conference System is able to map the floor request to the associated Conference Object.

This umbrella style association can be applied to any supplementary mechanisms that are applied to the XCON model defined in this document as long as a unique reference per conference instance is available that can be mapped to a Conference Object.

[Editor’s Note: The URI schema name per TBD must be registered with IANA.]

5.4 Conference User Identifier

Section 5.3.1 discusses the concept of an umbrella mechanism for associating various protocol identifiers with a top level XCON identifier. This section outlines a similar umbrella mechanism for the purpose of correlating users of supplementary XCON protocols under one universal XCON identity within an XCON Conference System.

It is important to emphasize that the Conference User Identifier being described must be in the context of the Conference System. This is due to the requirement for identity of Conference System users who may not be participating in a Conference Instance. Examples being a non participating 'Floor Control Chair' or 'Media Policy' Controller. Users can then be associated with Conference Objects as defined in Section 5.3.1. This association enables the Conference System to associate and enforce user level policies at both a system level and Conference Object level.

Each user within a Conference System is allocated a unique Conference User Identifier, as defined in [TBD]. This identifier acts as a top level identifier, in a similar fashion to that defined for the Conference Object Identifier described in Section 5.3.1. User identifiers defined in other protocols that are used within a Conference Instance fall underneath the top level identifier and enable the Conference System to correlate and map authentication under the single user umbrella.

Figure 5 illustrates an example using the Conference User Identifier in association with the user identity defined for BFCP and SIP Digest user identity as defined in RFC3261[4]
Within a Conference System, a user is identified by a single Conference User Identifier. Any other XCON mechanisms that contain a protocol specific user ID can be associated with the 'Conference User Identifier', as illustrated in Figure 5. This mechanism allows conference systems to manage and relate system wide user identities in relation to specific Conference Objects and helps in the enforcement of system wide policies.

6. Conferencing System Realization

XCON-compliant implementations can range from systems supporting ad-hoc conferences, with default behavior only, to sophisticated systems with the ability to schedule re-occurring 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, the XCON specifications don’t mandate the actual usage of the Conference Object, but rather defines the general cloning tree concept and the mechanisms required for its realization, which is described in detail in Section 6.1.

Adhoc and advanced conferencing examples are provided in Section 6.2
and Section 6.3, 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 6.4

6.1 Cloning Tree

The concept defined in this section is a logical representation only, as it is reflected through the XCON mechanisms: the URIs and the protocols. Of course, the actual system realization can differ from the presented model and doesn’t require physical copying of the information, etc.

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 object. This document uses the "cloning" metaphor instead of the "inheritance" metaphor because it more closely fits the object replication and extension concept, rather than data type re-usage and extension concept.

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

By default, all the data existing in the parent object is copied to the newly created 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 5.3.1

By default, 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 relax the access relative to its parent’s access.

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 objects can not be changed by
directly accessing the child object; neither can they be expanded in the child object alone.

In the future, XCON specifications may introduce logical relationships, in addition to the "parent enforceable", between elements being cloned from one another.
6.2 Ad-hoc Example

Figure 7 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 control signalling channel with a Conference Factory as specified in Section 5.1. The Conference Factory can internally select one of the system supported Conference Object blueprints based on the requesting client privileges and the media lines included in the 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 (such as adding participants) using the Conference Control Protocol [Section 7.3].
6.3 Advanced Example

Figure 8 illustrates how a recurring conference can be specified according to system capabilities, scheduled, reserved, and managed in a conferencing system.

Firstly, a client would query a Conferencing System for its capabilities. This can be done by requesting a list of the Conference Object "blueprints" (or templates) the system supports. Each blueprint can contain a specific combination of capabilities and limitations of the conference server in terms of supported media types (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.

A Client may need to query any templates that it doesn’t understand and then make a decision on compatibility. Interface hints need to be included as per [21]. The client then selects which specific
template to use and retrieves the template from the server itself (and NOT from some centralized repository).

The selected blueprint with its default values is copied by the client into a newly created Conference Object, referred to as a ‘Conference Object for Reservation’, that specifies the resources needed from the system for this Conference Instance. At this point the ‘Conference Object for 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 Object for Reservation’.

At this point the client can ask the conference server to create a new conference reservation by attaching the ‘Conference Object for Reservation’ to the request. As a result, the server can allocate the needed resources, create the additional Conference Objects for each conference occurrence (referred to as a ‘Conference Object for Occurrence’) and allocate the Conference Object identifiers for all – the ‘Conference Object for Reservation’ and for each ‘Conference Object for Occurrence’.

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 ‘Conference Object for Occurrence’ will affect neither the ‘Conference Object for Reservation’ 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-forcible. As a result, these objects can be modified by accessing the ‘Conference Object for Reservation’ only. The changes to these objects can be applied automatically to each of the ‘Conference Object for Occurrence’s (subject to local policy).
Figure 8: Advanced Conference Definition, Creation, and Lifetime.

6.4 Scheduling a ‘Conference Object for Reservation’

Scheduling conferences forms an important part of the Conferencing System solution. The concept of an individual Conference Instance
and its relationship to a specific Conference Object is introduced in Section 5.2 and Section 5.3. A basic Conference Instance represents a single occurrence that 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 conference instance, 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 Instances that form part of a recurring conference schedule. The mechanism proposed in this document makes use of the 'Internet Calendaring and Scheduling Core Object' specification defined in RFC2445[8] in union with the concepts introduced in Section 4 for the purpose of achieving advanced conference scheduling capability.

Figure 9 illustrates a simplified view of a Client interacting with a Conference System. The Client is using the Conference Control Protocol (Section 7.3) to add a new Conference Instance to the Conference System by interfacing with the Conference Control Server. A Conference Control Protocol request contains a valid 'Conference Object for Reservation' and reference by value to an 'iCal' object which contains scheduling information about the conference instance (e.g. start time, end time).
Figure 9: Resource Scheduling

A successful creation of a Conference Instance using the Conference Control Protocol results in a new ‘Conference Object for Reservation’. A Conference Control Protocol request is validated, including the associated iCal object, and the resultant ‘Conference Object for Reservation’ is created. The Conference Object is uniquely represented within the Conference System by Conference Object Identifier (e.g. xcon:hd87928374) as discussed in Section 5.3. The unique URI is returned to the client and can be used to reference the ‘Conference Object for Reservation’
representation if any future manipulations are required (e.g. Alter
start time) using a Conference Control Protocol.

The previous example explains how a client creates a basic
‘Conference Object for Reservation’ using an iCal reference in
association with a Conference Control Protocol. Figure 9 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 Conference Control
Request represents a series of recurring Conference Instances (e.g.
conference start time, end time, occur weekly). The Conference
system will treat this request the same as the first example. The
protocol request will be validated, along with the associated iCal
object, and the ‘Conference Object for Reservation’ will be created.
The ‘Conference Object for Reservation’ and the 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 Conference Control
Protocol to adjust the ‘Conference Object for 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 series range. A good example might be that a ‘Conference
Object for Reservation’ 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 9 in mind, the client will construct a Conference Control
request whose purpose is to modify the existing ‘Conference Object
for Reservation’ representing the recurring Conference Instance. The
Client will include the Conference Object ID provided by the
Conference System to explicitly reference the ‘Conference Object for
Reservation’ within the Conference System. A Conference Control
request will contain all the required changes to the ‘Conference
Object for Reservation’ (e.g. Change of venue). The Conference
System matches the incoming request to the existing ‘Conference
Object for Reservation’ but identifies that the associated iCal
object only refers to a range of the existing series. The Conference
System creates a child clone of the original ‘Conference Object for
Reservation’ to represent the altered Conference Instances within
the Series. The cloned ‘Conference Object for Reservation’
representing the altered series of Conference Instances has its own
associated Conference Object ID which is returned to the Client using
a Conference Control Protocol. 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 the Conference Control Protocol and the newly created Conference Object ID representation. This provides a flexible approach to the scheduling of recurring Conference instances.

7. Conferencing Mechanisms

7.1 Call Control Signalling

The focus is the central component of the conference. Participants interface with the focus using an appropriate Call Control Signalling protocol. Participants request to establish or join a conference using the Call control signalling interface. A focus then either accepts (after checking the policies), sends a progress indication (e.g. for a parked call while awaiting moderator approval to join) or rejects that request using the call control signalling interface.

During an active conference, a Conference Control Protocol [Section 7.3] can be used to affect the conference state. For example, Conference Control Protocol requests to add and delete participants will be communicated to the focus and checked against the conference policies. If approved, the participants will be added or deleted using the call signalling to/from the focus.

7.2 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 (e.g. participants) according to [11].

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.

7.3 Conference Control Protocol

The XCON working group charter includes the development of a protocol or a set of protocols for controlling the state of a Conference Object and for retrieving its state. The nature of this protocol is currently under discussion in the XCON working group. The proposals span from data manipulation (management-like) protocols (CPCP and NETCONF) to semantic-oriented (CCCP and CSCP). Details of the proposed protocols are in the sections below. The following paragraphs summarize the fundamental issues around the selection of the protocol(s). [Editor’s Note: Discussion Point 5 on the XCON WG
It is recognized that semantic manipulations make for a cleaner protocol design, with the disadvantage that extensions to the underlying data model require extensions to the protocol used to manipulate it. Syntactic manipulations allow for extensions to the data model without requiring protocol extensions, with the disadvantage that the server generally has to infer intent from data manipulations instead of having intent explicitly signaled.

It is worth noting that one portion of the data to be manipulated, the Common Conference Information, will not be extended, and would naturally lend itself to semantic manipulation. Another part of the data, the Conference Template, is intended to be extended, and would naturally lend itself to syntactic manipulation. However, there has been a stated desire to use a single protocol (and presumably a single mode of operation within this protocol) to manipulate all conference object state (common and template).

The third statement in the paragraph above makes the first two solution options mutually exclusive. A proposal was made that by allowing more than one protocol, a hybrid approach could be taken such that CPCP and CSCP can both be used, since they are based on other protocols that are likely to be supported by the clients (XCAP and BFCP, respectively). However, the very rough consensus of the WG supports a single protocol for Conference Control and SOAP has most recently been put forth as that protocol. A basic overview of SOAP in the context of Conference control is provided in Section 7.3.5.

### 7.3.1 CPCP

A Conference Policy Control Protocol (CPCP) is a data manipulation protocol being proposed as a standard means of storing and manipulating the conference policy. According to CPCP, the conference policy is comprised of the rules associated with a specific Conference Instance. Requirements for the CPCP are defined in the CPCP Requirements document [13]. The Conference Policy Control Protocol document [17] defines the XML Schema for the conference policy data elements.

The privileges as to which users are allowed to read and/or manipulate a specific Conference Instance are defined in a separate Extensible Markup Language (XML) Schema[19]. This schema is based on the common policy model being used for geographic privacy information and for presence information.

A separate document [18] proposes XCAP as one protocol mechanism for storing and manipulating this conferencing policy data. XCAP is a
HTTP 1.1 based protocol that allows clients to read, write, modify and delete application data stored in XML format on a server. One of the main advantages of XCAP is that it maps XML document elements and attributes to HTTP URIs that can be directly accessed by HTTP.

7.3.2 CCCP

A Centralized Conferencing Control Protocol [20] is a semantic-oriented protocol defined to allow participants or otherwise authorized entities to directly manipulate an active Conference Instance. CCCP is defined as a set of transactions issued over a reliable transport protocol. A transaction consists of a Request carrying the required information in an XML body and a corresponding Response carrying an appropriate XML body.

CCCP requests are submitted to the CCCP server and can be prioritized and queued, based on the CCCP client Role and the requested primitives. CCCP requires no single lock per document, and the CCCP server can locally implement an optimization strategy independent of CCCP client behavior.

7.3.3 CSCP

A Conference State Change protocol [26] is a client server protocol used to change the state of a conference object. CSCP extends the BFCP protocol [16] with new commands. A client sends the server a request representing a sequence of commands. Each command can set, get, add, or delete a single field in the conference object. Changes to the conference object are performed on a hierarchal set of elements and unique attributes within those elements. A series of changes can be pipelined in a single BFCP message. The server executes each action in series. If one of them fails, the server returns an error for the action that failed and does not execute any of the actions after that. Each individual action is atomic but the pipelined series is not.

The item to which a command applies is specified by a unique ID and, where appropriate, attribute name. The ID is an unsigned 32 bit integer called the Element ID. The server discovery of the Element ID is outside of CSCP. Typically this is done by using the conference notification service per Section 7.2. Each field in the data received in the notification contains a unique field ID attribute that can be used in BFCP requests.

7.3.4 NETCONF

The Network Configuration (NETCONF) protocol [27] defines a simple mechanism through which a network device can be managed,
configuration data information can be retrieved, and new
configuration data can be uploaded and manipulated. The protocol
allows the device to expose a full, formal, application programming
interface (API).

NETCONF is proposed as the mechanism for object content manipulation
and state retrieval for the XCON data. NETCONF provides a flexible
configuration retrieval mechanism, with extensibility. It allows for
incremental configuration and commits. NETCONF supports stored
configurations (e.g. for startup, running, etc.). It also supports
XPath and subtree filtering. With NETCONF, there are no constraints
on the configuration content.

7.3.5 SOAP

The SOAP protocol is fundamentally an XML messaging scheme, capable
of supporting remote procedure calls. SOAP defines a simple message
format composed of a "header" and a "body" contained within an
"envelope". The "header" contains meta-information relating to the
message, and can be used to indicate such things as store-and-forward
behaviour or transactional characteristics. In addition, SOAP
encoding is optimized for ease of automated deserialization.

SOAP is proposed as the mechanism for object content manipulation and
state retrieval for the XCON data. In general, SOAP is a good fit
for Conference Control, essentially because of its remote procedure
call characteristics and its inherently synchronous and client-driven
nature.

7.4 Floor Control

[Editor’s Note: This text still requires further updating to reflect
new model and pending new WG agreements. Amongst the things TODO
are:

1. Need to be able to fetch from the Conference Object the
credentials required for BFCP. This includes the conference id, user
id, and password.

4. Evaluation of an alternative proposal [TBD as a stand alone draft
as well] for using Templates as the means for correlating Floor
Control identifiers.

5. Still need to condense this section – propose to add a scenario
to section 8 and thus remove some of the details, leaving this
description a very basic overview and mapping of the identifiers.
}
A mechanism for floor control within a Conferencing System is defined in the 'Binary Floor Control Protocol' specification [16]. Floor control is not a mandatory mechanism for a Conferencing System implementation but provides advanced media input control features for participants.

An XCON compliant 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[2] offer/answer[5] exchange on the signalling interface with the focus.

As well as the Client --> Floor 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 [16] for authentication details), a specific floor control message requires detailed information to uniquely identify a user, a floor and a conference. This information, defined in the next set of bullet points, can be obtained using methods such as negotiation using the SDP offer/answer exchange on the signalling interface with the focus:

- **Conference Object ID** - Each Conference Object has a unique identifier per Section 5.3.1, obtained from the Conference server, which MUST be included in all Floor messages. When the SDP model is used as described in [24] this identifier maps to the 'confid' SDP attribute.
- **Conference User Identifier** - Each user has a unique identifier within the Conference Object per Section 5.4, obtained from the Conference server, which MUST be included in all Floor messages. When using SDP offer/answer exchange to negotiate a Floor control connection with the focus using the call control signalling interface, the unique conference identifier is contained in the 'userid' SDP attribute, as defined in [24].
- **Floor ID** - A media session with a Conferencing System can have any number of 'Floors' (0 or more) that are represented by a unique identifier within the Conference Instance (as represented by Conference ID). When using SDP offer/answer exchange to negotiate a Floor control connection with the focus using the call control signalling interface, the unique conference identifier is contained in the 'floorid' SDP attribute, as defined in [24].
  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 [23].
Clients can authenticate a BFCP server using the TLS certificates. Requests submitted on a successfully created BFCP connection may additionally require digest authentication within the BFCP protocol to authenticate the Floor control client to the server. For this to take place a shared secret needs to be exchanged between the BFCP client/server entities. This can be achieved out of band using a mechanism such as the ‘k=’ SDP attribute. The shared secret can also be exchanged using un-specified ‘out of band’ mechanisms. When using Digest authentication of BFCP client messages the exchange of an active ‘Nonce’ is also required. This can be achieved using a BFCP request response interaction as defined in BFCP (A request is submitted without a Nonce TLV and the server generates an error response with either an Error Code 7 (DIGEST TLV Required) or 6 (Invalid Nonce) containing the valid nonce). The BFCP ‘Nonce’ value can also be obtained ‘out of band’ using information provided in the Offer/Answer exchange. As with the other SDP Floor attributes referenced in this section and defined in [24], the ‘nonce’ attribute can be inserted in the SIP response e.g. a=nonce:dhsa8hd0dwqj.

8. Conferencing Scenario Realizations

This section addresses how advanced conferencing scenarios, many of which have been described in [14], are realized using this XCON 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.

Section 6.2 and Section 6.3 provide examples of the creation of a basic adhoc conference and a more advanced scheduled conference.

8.1 Participant Manipulations

There are different ways to affect a participant state in a conference. A participant can join and leave the conference using call signalling means only, such as SIP. This kind of operation is called "1st party signalling" 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 signalling only, may also be available for some signalling protocols. For example, "Conferencing for SIP User Agents" [15] 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,
state of other participants, and state of various resources (such as media mixers) which may indirectly affect the state of any of the conference participants.

Figure 11 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.

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 XCON.

Figure 10: 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 Conference System ensures that "Alice" has the appropriate authority based on the policies associated with that specific Conference Object to perform the operation. The Conference System must also determine whether "Bob" is already a user of this Conference System or whether he is a new user.
If "Bob" is a new user for this conference system, a Conference User Identifier is created for Bob. Based upon the addressing information provided for "Bob" by "Alice", the call signalling to add "Bob" to the conference is instigated through the Focus.

Once the call signalling 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.

8.2 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 operations is called "1st party signalling" 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 11 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.

It should be noted that only entities impacted by the request are shown in the diagram (e.g. there is no Focus shown).
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 Template of the Conference Object (if included) 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.

8.3 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 6.1 A Conference object representing a Sidebar is created by cloning the parent associated with the existing conference, copying the information from the parent 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). A Conference 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 can not exist if parent Conference is terminated, etc.). The Sidebar mechanism uses the umbrella approach for association of XCON Conference Object Identifiers as introduced in other sections of this document.

Figure 12: Conference Object Mapping.

Figure 12 illustrates the relationship between a Conference Object and associated Sidebar Conference Objects within a Conference System. Each Sidebar Conference Object has a unique Conference Object Identifier as described in Section 5.3.1. 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 6.1. 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 (Section 7.3 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 6.1).

[Editor’s Note: this needs more detail - especially around cloning tree.]
8.4 Whispering or Private Messages

[Editor’s Note: TBD. Once we get full agreement on terminology and the basic ideas in the other sections, we’ll tackle this.]

8.5 Conference Announcements and Recordings

[Editor’s note: TBD. Use Cullen’s comments on the previous version of the doc.]

9. Relationships between SIPPING and XCON Frameworks

The SIPPING Conferencing Framework [9] 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 SIPPING Conferencing Framework are being used to illustrate how SIP [4] can be used as a signalling means in these conferencing systems. SIPPING Conferencing Framework does not define new conference control protocols to be used by the general conferencing system. It uses only basic SIP [4], the SIP Conferencing for User Agents [15], and the SIPPING Conference Package [9] for basic SIP conferencing realization.

The XCON framework defines a particular centralized Conferencing System and the logical entities implementing it. It also defines a particular XCON Data Model and refers to the standard protocols (beyond call signalling means) being defined by the XCON WG to be used among the XCON logical entities for implementing advanced conferencing features. The purpose of XCON working group and this framework is to achieve interoperability between the XCON 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 [4], the SIP Conferencing for User Agents [15], and the SIPPING Conference Package [9] are fully compatible with both Framework documents.

10. 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 7. The protocols used for manipulation and retrieval of confidential information MUST support a confidentiality and integrity mechanism. Similar requirements apply for the floor control protocols.

There are also security issues associated with the authorization to perform actions on the conferencing system to invoke specific capabilities. Section 4.3 discusses the policies associated with the Conference Object to ensure that only authorized entities are able to manipulate the data to access the capabilities. The final set of issues involves the privacy and security of the identity of a user in the conference.

Many policy authorization decisions are based on the identity of the user or the Role that a user may have. There are several ways that a user might authenticate its identity to the system. The conferencing system may know about specific users and assign passwords to the users. The users may also be authenticated by knowing a particular Conference ID and a PIN for it. Sometimes, a PIN is not required and the Conference ID is used as a shared secret. The call signalling means can provide a trusted form of the user identity or it might just provide a hint of the possible identity and the user still needs to provide some authentication to prove it has the identity that was provided as a hint in the call signalling. This may be in the form of an IVR system or other means. The goal for the Conferencing System is to figure out a user identity and a Role for any attempt to send a request to the Conferencing System.

When a Conferencing System presents the identity of authorized users, it may choose to provide information about the way the identity was proven to or verified by the system. A user may also come as a completely unauthenticated user into the system — this fact needs also 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 authenticates 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
The term password is used to mean the usual, that is to say a reasonable sized, in number of bits, hard to predict shared secret. Today users often have passwords with more than 30 bits of randomness in them. 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). When Conference 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 Conference 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.

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 set select various options for revealing their identity to other users. A user can be "hidden" such that other users can not see they are involved in the conference, or they can be "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’ visibility (including "anonymous" and "hidden") may be displayed to the moderator or administrator, subject to a Conference System’s local policies. "Hidden" status is often used by robot participants of a conference and is also used in many call center conference situations.

11. IANA Considerations

This is an informational draft, with no IANA considerations required.

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, Cullen Jennings for providing input for the "Security Considerations" section and Keith Lantz for his careful reviews and constructive
13. Open Issues

Several open issues are identified in the body of the document. This list is intended as a summary to facilitate the tracking of the primary open issues that require WG input.


2. DP5. CCP Protocol.

14. Changes since last Version

Changes from WG 00 to 01::

- **Section 2** (Conventions and Terminology). Slight modifications to definitions of Call (control) signalling, Conference Identifier, Conference Instance, Conference Object.

- **Section 2** (Conventions and Terminology). Renaming of term "Registered Template Definition" to "Registered Conference Document" (per agreement at interim Meeting).

- **Section 3** (Next to the last paragraph on the media model). Clarified the text such that it doesn’t read that the focus performs media mixing. Changed "focus" to "media mixer controlled by the focus" in the 2nd sentence and "performed" to "controlled" in the 4th.

- **Section 5.** Rearranged the sub-sections a bit for better flow. First describe the Conference ID, then the Conference Instance, followed by the Conference Object, with the Conference Object Identifier described in a subsection of the Conference Object section. Added a diagram showing the relationship between Conference Identifier/Focus and Conference Object Identifier, within the context of a Conference Instance to the Conference Object section.

- **Section 6.1** (Cloning Tree). Rewording to clarify which operations apply to independent objects (and non-independent).

- **Section 6.3** (Advanced Example). Added additional text with regards to future conferences, introducing the concept of infinite series (which would be limited by calendaring interface).

- **Section 7.3** (Conference Control Protocol). Updated to include
reference to SOAP option.

- **Section 8.3** (sidebars) - reworded 1st paragraph to be more explicit about the XCON FW constructs used.

Changes from individual 02 to WG 00:

- few minor editorial changes

- **Section 2.** Removed second sentence of definition of Conference ID, as that’s now included/described in context in new Identifier section.

- **Section 3.** Clarified that TBD in Figure 1 is "Conference Control Protocol" (per Keith’s comment to be more explicit).

- **Section 4.1.** Identifiers. Moved this to a new section (Section 5).

- New section for Identifiers (Section 5), thus all section references beyond 4 are incremented in the new version.

- **Section 4.** Since section 4.1 was removed, section 4.2 became the body text for section 4.

- **Section 4.2.** Added "Floor Information" to Figure 2 as part of Common Conference Information, also added "Floor Control" to Conference Template (per text and Cullen’s draft).

- **Section 4.5.** Conference policies. Reworded to not introduce new terms, but use the general terms identified in the 1st paragraph.

- **Section 5.2.** Removed "Instance" from "Active Conference Instance" in Figure 4.

- **Section 5.3.** Added text clarifying that templates are retrieved from server (not central repository) (per DP3 conclusion).

- **Section 5.4.** Added text that there is a single time and that the times are all relative the one time (per DP1 conclusion).

- **Section 5.4.** Added text clarifying that changes to a series impact "all future occurrences (per DP1 discussion/conclusion).

- **Section 6.3** - Added subsections for discussion of CSCP and NETCONF as the CCP.

- **Section 6.4** - Floor Control. Removed Editor’s notes 2 and 3.
Condensed the text only slightly, but added explicit references to new identifier section.

- **Section 6.4.1** Moved to new Identifier section  (Section 5)
- **Section 7.1** - moved example to 7.2. Included a new (more appropriate example) in 7.1, although this may be too basic.
- **Section 7.3** - added some proposed text for Sidebars.

15. References

15.1 Normative References


15.2 Informative References


[10] Levin, O. and R. Even, "High Level Requirements for Tightly Coupled SIP Conferencing",
draft-ietf-sipping-conferencing-requirements-01 (work in progress), October 2004.


[16] Camarillo, G., "The Binary Floor Control Protocol (BFCP)",
draft-ietf-xcon-bfcp-04 (work in progress), May 2005.


[20] Levin, O. and G. Kimchi, "Centralized Conference Data Model",
draft-levin-xcon-cccpp-02 (work in progress), February 2005.

draft-jennings-xcon-media-control-02 (work in progress),
February 2005.

[22] Rosen, B., "SIP Conferencing: Sub-conferences and Sidebars",
draft-rosen-xcon-conf-sidebars-01 (work in progress),
July 2004.

[23] Levin, O. and G. Camarillo, "The SDP (Session Description
Protocol) Label Attribute",
draft-ietf-mmusic-sdp-media-label-01 (work in progress),
January 2005.

Binary Floor Control Protocol (BFCP) Streams",
draft-camarillo-mmusic-sdp-bfcp-00 (work in progress),

draft-ietf-simple-message-sessions-10 (work in progress),
February 2005.

(CSCP)"

draft-jennings-xcon-cscp-00 (work in progress),
February 2005.

[27] Enns, R., "NETCONF Configuration Protocol",
draft-ietf-netconf-prot-07 (work in progress), June 2005.

Authors' Addresses

Mary Barnes
Nortel
2201 Lakeside Blvd
Richardson, TX

Email: mary.barnes@nortel.com

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

Email: cboulton@ubiquitysoftware.com
Orit Levin
Microsoft Corporation
One Microsoft Way
Redmond, WA  98052

Email: oritl@microsoft.com
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.