XCON Working Group
Internet-Draft
Expires: April 25, 2005

G. Camarillo
Ericsson
J. Ott
Universitaet Bremen
K. Drage
Lucent Technologies

October 25, 2004

The Binary Floor Control Protocol (BFCP)
draft-ietf-xcon-bfcp-02.txt

Abstract

Floor control is a means to manage joint or exclusive access to shared resources in a (multiparty) conferencing environment. Thereby, floor control complements other functions -- such as conference and media session setup, conference policy manipulation,

and media control -- that are realized by other protocols.

This document specifies the Binary Floor Control Protocol (BFCP). BFCP is used between floor participants and floor control servers, and between floor chairs (i.e., moderators) and floor control servers.

Table of Contents

1. Introduction

Within a conference, some applications need to manage the access to a set of shared resources, such as the right to send media over a particular media stream. Floor control enables such applications to provide users with coordinated (shared or exclusive) access to these resources.

The Requirements for Floor Control Protocol [9] list a set of requirements that need to be met by floor control protocols. The Binary Floor Control Protocol (BFCP), which is specified in this document, meets these requirements.

The remainder of this document is organized as follows. Section 2 defines the terminology used throughout this document and Section 3 discusses the scope of BFCP (i.e., which tasks fall within the scope of BFCP and which ones are performed using different mechanisms). Section 4 provides a non-normative overview of BFCP operation and subsequent sections provide the normative specification of BFCP.

2. 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 [2] and indicate requirement levels for compliant implementations.

Media Participant: An entity that has access to the media resources of a conference (e.g., it can receive a media stream). In floor-controlled conferences, a given media participant is typically co-located with a floor participant, but does not need to. Third-party floor requests consist of having a floor participant request a floor for a media participant when they are not co-located.

Client: a floor participant or a floor chair that communicate with a floor control server using BFCP.

Conference Policy: The complete set of rules for a particular conference manipulated by the conference policy server. It includes the membership policy and the media policy. There is an instance of conference policy for each conference.

Conference Policy Control Protocol (CPCP): The protocol used by clients to manipulate the conference policy.

Floor: A permission to temporarily access or manipulate a specific shared resource or set of resources.

Floor Chair: A logical entity that manages one floor (grants, denies, or revokes a floor). An entity that assumes the logical role of a floor chair for a given transaction may assume a different role (e.g., floor participant) for a different transaction. The roles of floor chair and floor participant are defined on a transaction-by-transaction basis.

Floor Control: A mechanism that enables applications or users to gain safe and mutually exclusive or non-exclusive input access to the shared object or resource.

Floor Control Server: A logical entity that maintains the state of the floor(s) including which floors exists, who the floor chairs are, who holds a floor, etc. Requests to manipulate a floor are directed at the floor control server. The floor control server of a conference may perform other logical roles (e.g., floor participant) in another conference.

Floor Participant: A logical entity that requests floors, and possibly information about them, from a floor control server. An entity that assumes the logical role of a floor participant for a given transaction may assume a different role (e.g., a floor chair) for a different transaction. The roles of floor participant and floor chair are defined on a transaction-by-transaction basis. In floor-controlled conferences, a given floor participant is typically co-located with a media participant, but does not need to. Third-party floor requests consist of having a floor participant request a floor for a media participant when they are not co-located.

Media Policy: A set of rules manipulated by the conference policy server regarding the media composition of the conference. The media policy is used by the focus to determine the mixing characteristics for the conference. The media policy includes rules about which conference participants receive media from which other participants, and the ways in which that media is combined for each conference participant. In the case of audio, these rules can include the relative volumes at which each conference participant is mixed. In the case of video, these rules can indicate whether the video is tiled, whether the video indicates the loudest speaker, and so on.

Participant: An entity that acts as a floor participant, as a media participant, or as both.

3. Scope

As stated earlier, the BFCP is a protocol to coordinate access to shared resources in a conference following the requirements defined in [9]. Floor control complements other functions defined in the

conferencing framework [14], such as conference policy and media policy. In particular, it is the conference policy that defines which media streams and applications are floor-controlled, who is/are the respective floor chair(s), and how access to the floor is managed. Furthermore, it is up to the media policy to define which (if any) impact on media stream handling (e.g. switching or mixing) assignment of a floor to a media participant has.

The floor control protocol BFCP defined in this document only specifies a means to arbitrate access to floors. The rules and constraints for floor arbitration and the results of floor assignments are outside the scope of this document and defined by the conference and media policy, respectively.

Figure 1 shows the tasks that BFCP can perform.

```
                           +---------+
                           |  Floor  |
                           |  Chair  |
                           |         |
                           +---------+
                              ^   |
                              |   |
                 Notification |   | Decision
                              |   |
                              |   |
                   Floor      |   v
+-------------+   Request  +---------+              +-------------+
|    Floor    |----------->|  Floor  | Notification |    Floor    |
| Participant |            | Control |------------->| Participant |
|             |<-----------|  Server |              |             |
+-------------+ Granted or +---------+              +-------------+
                  Denied
```

Figure 1: Functionality provided by BFCP

BFCP provides a means:

- for floor participants to send floor requests to floor control servers.
- for floor control servers to grant or deny requests to access a given resource from floor participants.
- for floor chairs to send floor control servers decisions regarding floor requests.
- for floor control servers to keep floor participants and floor chairs informed about the status of a given floor or a given floor request.

Even though tasks that do not belong to the previous list are outside the scope of BFCP, some of these out-of-scope tasks relate to floor control and are essential to create floors and to establish BFCP connections between different entities. In the following subsections, we discuss some of these tasks and mechanisms to perform them.

3.1 Floor Creation

The conference policy for a particular conference contains the floors of the conference and the resource or resources associated with each floor. For example, a conference may have two floors: one controlling who can talk at a particular time and another controlling who can write on a shared whiteboard.

According to the definitions in Section 2, the conference policy is manipulated using a Conference Policy Control Protocol (CPCP), such as the one defined in [10]. Consequently, floor creation and termination is handled by CPCP. In addition, CPCP also handles the association of a given floor with a resource or a set of resources (e.g., media streams).

Additionally, the floor control server needs to stay up to date on changes on the conference policy (e.g., when a new floor is created). The floor control may use a mechanism such as the one defined in [13] to keep itself up to date.

3.2 Obtaining Information to Contact a Floor Control Server

A client needs a set of data in order to establish a BFCP connection to a floor control server. These data include the transport address of the server, the conference identifier, and the user identifier.

Clients can obtain this information in different ways. Two possibilities are using CPCP and using an offer/answer [8] exchange. How to use an SDP [6] offer/answer [8] exchange to obtain this information is described in [15].

3.3 Generating a Shared Secret

Authentication in BFCP is based on a shared secret between the client and the floor control server. So, there is a need for a mechanism to generate such a shared secret.

When the floor participant or the floor chair obtains the information needed to contact the BFCP floor control server over a secure channel (e.g., an offer/answer [8] exchange using SIP [7] protected using S/MIME), they can get the shared secret using the same channel.

If there is no secure channel available, a different mechanism needs to be used. For example, MIKEY [12] allows an offerer and an answerer to perform a Diffie-Hellman key exchange.

Shared secrets can also be generated and exchanged using out-of-band means.

3.4 Obtaining Floor-Resource Associations

Floors are associated with resources. For example, a floor that controls who talks at a given time has a particular audio stream as its associated resource. Associations between floors and resources are part of the conference policy, which is manipulated using CPCP.

Floor participants and floor chairs need to know which resources are associated with which floors. They can obtain this information using different mechanisms, such as CPCP or an offer/answer [8] exchange. How to use an offer/answer exchange to obtain these associations is described in [15].

> Note that floor participants perform offer/answer exchanges with the SIP focus of the conference. So, the SIP focus needs to obtain information about associations between floors and resources using a mechanism such as CPCP in order to be able to provide this information to a floor participant in an offer/answer exchange.

3.5 Policy Enforcement

A participant whose floor request is granted has the right to use (in a certain way) the resource or resources associated with the floor that was requested. For example, the participant may have the right to send media over a particular audio stream.

Nevertheless, holding a floor does not imply that others will not be able to use its associated resources at the same time, even if they do not have the right to do so. According to the definition in Section 2, the media policy of a conference is the one that determines which media participants can actually use the resources in the conference.

So, if the policy of a conference is to enforce floor control decisions, every change in the status of any floor needs to be reflected in the media policy of the conference. For example, the mixer only accepts media from the user who holds the floor.

A way to reflect the status of the floors in the media policy is to have the floor control server manipulate the media policy using CPCP. Nevertheless, there are other ways to enforce floor control policies,

such as having a floor control chair manipulate the media policy (using CPCP) only if there are misbehaving users trying to use a resource without holding its associated floor.

4. Overview of Operation

This section provides a non-normative description of BFCP operations. Section 4.1 describes the interface between floor participants and floor control servers and Section 4.2 describes the interface between floor chairs and floor control servers

BFCP messages, which use a TLV (Type-Length-Value) binary encoding, consist of a common header followed by a set of TLVs. The common header contains, among other information, a 32-bit conference identifier. Floor participants, media participants, and floor chairs are identified by a 16-bit user identifier, which is carried in a TLV.

There are two types of transactions in BFCP: client-initiated transactions and server-initiated transactions. Client-initiated transactions consist of a message from a client to the floor control server and a response from the floor control server to the client. Both messages can be related because they carry the same TRANSACTION-ID TLV. Server-initiated transactions consist of a single message, which has no TRANSACTION-ID TLV, from the floor control server to a client.

4.1 Floor Participant to Floor Control Server Interface

Floor participants request a floor by sending a FloorRequest message to the floor control server. BFCP supports third-party floor requests. That is, the floor participant sending the floor request need not be co-located with the media participant that will get the floor once the floor request is granted. FloorRequest messages carry the identity of the requester in a USER-ID TLV, and the identity of the beneficiary of the floor, in third party floor requests, in a BENEFICIARY-ID TLV.

> Third party floor requests can be sent, for example, by floor participants that have a BFCP connection to the floor control server but that are not media participants (i.e., they do not handle any media).

FloorRequest messages identify the floor or floors being requested by carrying their 16-bit floor identifiers in FLOOR-ID TLVs. If a FloorRequest message carries more than one floor identifier, the floor control server treats all the floor requests as an atomic package. That is, the floor control server either grants or denies

all the floors in the FloorRequest message.

Floor control servers respond to FloorRequest messages with FloorRequestInfo messages, which provide information about the status of the floor request. The first FloorRequestInfo message is the response to the FloorRequest message from the client, and therefore carries the same TRANSACTION-ID TLV as the FloorRequest.

Additionally, the first FloorRequestInfo message carries a FLOOR-REQUEST-ID TLV. Subsequent FloorRequestInfo messages related to the same floor request will carry the same FLOOR-REQUEST-ID TLV. This way, the floor participant can associate them with the appropriate floor request.

Messages from the floor participant related to a particular floor request also use the same FLOOR-REQUEST-ID TLV as the first FloorRequestInfo Message from the floor control server.

Figure 2 shows how a floor participant requests a floor, obtains it, and, at a later time, releases it. This figure illustrates the use, among other TLVs, of the TRANSACTION-ID and the FLOOR-REQUEST-ID TLVs.

```
  Floor Participant                                  Floor Control
                                                         Server
          |(1) FloorRequest                                |
          |TRANSACTION-ID: 123                             |
          |USER-ID: 234                                    |
          |FLOOR-ID: 543                                   |
          |----------------------------------------------->|
          |(2) FloorRequestInfo                            |
          |TRANSACTION-ID: 123                             |
          |USER-ID: 234                                    |
          |FLOOR-REQUEST-ID: 789                           |
          |FLOOR-ID: 543                                   |
          |REQUEST-STATUS: Pending                         |
          |<-----------------------------------------------|
          |(3) FloorRequestInfo                            |
          |USER-ID: 234                                    |
          |FLOOR-REQUEST-ID: 789                           |
          |FLOOR-ID: 543                                   |
          |REQUEST-STATUS: Accepted (1st in Queue)         |
          |<-----------------------------------------------|
          |(4) FloorRequestInfo                            |
          |USER-ID: 234                                    |
          |FLOOR-REQUEST-ID: 789                           |
```

```
             |FLOOR-ID: 543                                 |
             |REQUEST-STATUS: Granted                       |
             |<---------------------------------------------|
             |(5) FloorRelease                              |
             |TRANSACTION-ID: 154                           |
             |USER-ID: 234                                  |
             |FLOOR-REQUEST-ID: 789                         |
             |--------------------------------------------->|
             |(6) FloorRequestInfo                          |
             |TRANSACTION-ID: 154                           |
             |USER-ID: 234                                  |
             |FLOOR-REQUEST-ID: 789                         |
             |FLOOR-ID: 543                                 |
             |REQUEST-STATUS: Released                      |
             |<---------------------------------------------|
```

Figure 2: Requesting and releasing a floor

Figure 2 shows how a floor participant requests to be informed on the status of a floor. The first FloorInfo message from the floor control server is the response to the FloorInfoWanted message, and as such, carries the same TRANSACTION-ID TLV as the FloorInfoWanted message.

Subsequent FloorInfo messages consist of server-initiated transactions, and therefore carry no TRANSACTION-ID TLV. FloorInfo message (2) indicates that there are currently two floor requests for the floor whose Floor ID is 543. FloorInfo message (3) indicates that the floor requests with Floor Request ID 764 has been granted, while the floor request with Floor Request ID 635 is the first in the queue. FloorInfo message (4) indicates that the floor request with Floor Request ID 635 has been granted.

```
     Floor Participant                               Floor Control
                                                        Server
             |(1) FloorInfoWanted                           |
             |TRANSACTION-ID: 257                           |
             |USER-ID: 234                                  |
             |FLOOR-ID: 543                                 |
             |--------------------------------------------->|
             |(2) FloorInfo                                 |
             |TRANSACTION-ID: 257                           |
             |USER-ID: 234                                  |
             |FLOOR-ID:543                                  |
             |FLOOR-REQUEST-ID: 764                         |
```

```
             |FLOOR-ID: 543                                 |
             |BENEFICIARY-ID: 124                           |
             |REQUEST-STATUS: Accepted (1st in Queue)       |
             |FLOOR-REQUEST-ID: 635                         |
             |BENEFICIARY-ID: 154                           |
             |FLOOR-ID: 543                                 |
             |REQUEST-STATUS: Accepted (2nd in Queue)       |
             |<---------------------------------------------|
             |(3) FloorInfo                                 |
             |USER-ID: 234                                  |
             |FLOOR-ID:543                                  |
             |FLOOR-REQUEST-ID: 764                         |
             |FLOOR-ID: 543                                 |
             |BENEFICIARY-ID: 124                           |
             |REQUEST-STATUS: Granted                       |
             |FLOOR-REQUEST-ID: 635                         |
             |BENEFICIARY-ID: 154                           |
             |FLOOR-ID: 543                                 |
             |REQUEST-STATUS: Accepted (1st in Queue)       |
             |<---------------------------------------------|
             |(4) FloorInfo                                 |
             |USER-ID: 234                                  |
             |FLOOR-ID:543                                  |
             |FLOOR-REQUEST-ID: 635                         |
             |BENEFICIARY-ID: 154                           |
             |FLOOR-ID: 543                                 |
             |REQUEST-STATUS: Granted                       |
             |<---------------------------------------------|
```

Figure 3: Obtaining status information about a floor

FloorInfo messages contain information about the floor requests they carry. For example, FloorInfo message (4) indicates that the floor request with Floor Request ID 635 has as the beneficiary (i.e., the participant that holds the floor when a particular floor request is granted) the participant whose User ID is 154. The floor request applies only to the floor whose Floor ID is 543. That is, this is not a multi-floor floor request.

4.2 Floor Chair to Floor Control Server Interface

Figure 4 shows a floor chair instructing a floor control server to grant a floor. Note, however, that although the floor control server needs to take into consideration the instructions received in ChairAction messages (e.g., granting a floor), it does not necessarily need to perform them exactly as requested by the floor chair. The operation that the floor control server performs depends

on the ChairAction message and on the internal state of the floor control server.

For example, a floor chair may send a ChairAction message granting a floor which was requested as part of an atomic floor request operation that involved several floors. Even if the chair responsible for one of the floors instructs the floor control server to grant the floor, the floor control server will not grant it until the chairs responsible for the other floors agree to grant them as well. In another example, a floor chair may instruct the floor control server to grant a floor to a participant. The floor control server needs to revoke the floor from its current holder before granting it to the new participant.

So, the floor control server is ultimately responsible to keep a coherent floor state using instructions from floor chairs as input to this state.

```
     Floor Chair                                    Floor Control
                                                       Server
           |(1) ChairAction                             |
           |TRANSACTION-ID: 769                         |
           |USER-ID: 357                                |
           |FLOOR-ID: 543                               |
           |FLOOR-REQUEST-ID: 635                       |
           |REQUEST-STATUS: Granted                     |
           |------------------------------------------->|
           |(2) ChairActionAck                          |
           |TRANSACTION-ID: 769                         |
           |USER-ID: 357                                |
           |<-------------------------------------------|
```

Figure 4: Chair instructing the floor control server

5. Packet Format

BFCP packets consist of an 8-byte fixed header followed by attributes. All the protocol values MUST be sent in network byte order.

5.1 FIXED-HEADER Format

The following is the FIXED-HEADER format.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Ver |Reserved |  Primitive    |        Payload Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Conference ID                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: FIXED-HEADER format

Ver: the 3-bit version field MUST be set to 1 to indicate this version of BFCP.

Reserved: at this point, the 5 bits in the reserved field SHOULD be set to zero by the sender of the message and MUST be ignored by the receiver.

Primitive: this 8-bit field identifies the main purpose of the message. The following primitive values are defined:

Value	Primitive	Direction
0	FloorRequest	P -> S
1	FloorRelease	P -> S
2	FloorRequestInfoWanted	P -> S ; Ch -> S
3	FloorRequestInfo	P <- S ; Ch <- S
4	FloorInfoWanted	P -> S ; Ch -> S
5	FloorInfo	P <- S ; Ch <- S
6	ChairAction	Ch -> S
7	ChairActionAck	Ch <- S
8	Hello	P -> S ; Ch -> S
9	HelloAck	P <- S ; Ch <- S
10	Error	P <- S ; Ch <- S

S:	Floor Control Server
P:	Floor Participant
Ch:	Floor Chair

Table 1: BFCP primitives

Payload Length: this 16-bit field contains length of the message in 4-byte units excluding the fixed header.

Conference ID: this 32-bit identifies the conference the message belongs to.

5.2 Attribute Format

BFCP attributes are encoded in TLV (Type-Length-Value) format. TLVs are 32-bit aligned.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type     |M|    Length     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                                                               |
/                       Attribute Contents                      /
/                                                               /
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: TLV format

Type: this 7-bit field contains the type of the attribute. The following attribute types are defined:

Type	Attribute
0	FLOOR-ID
1	USER-ID
2	BENEFICIARY-ID
3	TRANSACTION-ID
4	FLOOR-REQUEST-ID
5	HUMAN-READABLE-INFO
6	DIGEST
7	REQUEST-STATUS
8	ERROR-CODE
9	USER-DISPLAY-NAME
10	USER-URI
11	PRIORITY
12	NONCE
13	SUPPORTED-TLVS

Table 2: BFCP attributes

M: the 'M' bit, known as the Mandatory bit, indicates whether support of the attribute is required. If an unrecognized attribute with the 'M' bit set is received, the message is rejected.

Length: this 8-bit field contains the length of the attribute in

bytes, excluding any padding defined for specific attributes. The Type, 'M' bit, and Length fields are included.

Attribute Contents: the contents of the different TLVs are defined in the following sections.

5.2.1 FLOOR-ID

The following is the format of the FLOOR-ID attribute.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |0 0 0 0 0 0 0|M|0 0 0 0 0 1 0 0|           Floor ID            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: FLOOR-ID format

Floor ID: this field contains a 16-bit value that uniquely identifies a floor within a conference.

5.2.2 USER-ID

The following is the format of the USER-ID attribute.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |0 0 0 0 0 0 1|M|0 0 0 0 0 1 0 0|            User ID            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: USER-ID format

User ID: this field contains a 16-bit value that uniquely identifies a user within a conference.

5.2.3 BENEFICIARY-ID

The following is the format of the BENEFICIARY-ID attribute.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0 0 0 0 0 1 0|M|0 0 0 0 0 1 0 0|        Beneficiary ID         |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 9: BENEFICIARY-ID format

Beneficiary ID: this field contains a 16-bit value that uniquely identifies a user within a conference.

5.2.4 TRANSACTION-ID

The following is the format of the TRANSACTION-ID attribute.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0 0 0 0 0 1 1|M|0 0 0 0 0 1 0 0|        Transaction ID         |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 10: TRANSACTION-ID format

Transaction ID: this field contains a 16-bit value that allows users to match a given message with its response.

5.2.5 FLOOR-REQUEST-ID

The following is the format of the FLOOR-REQUEST-ID attribute.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0 0 0 0 1 0 0|M|0 0 0 0 0 1 0 0|       Floor Request ID        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 11: FLOOR-REQUEST-ID format

Floor Request ID: this field contains a 16-bit value that indentifies a floor request at the floor control server.

5.2.6 HUMAN-READABLE-INFO

The following is the format of the HUMAN-READABLE-INFO attribute.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |0 0 0 0 1 0 1|M|    Length     |                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
    |                                                               |
    /                             Text                              /
    /                                               +-+-+-+-+-+-+-+-+
    |                                               |    Padding    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 12: HUMAN-READABLE-INFO format

Text: this field contains UTF-8 [5] encoded text.

In some situations, the contents of the Text field may be generated by an automaton. If such automaton has information about the preferred language of the receiver of a particular HUMAN-READABLE-INFO TLV, it MAY use this language to generate the Text field.

Padding: one, two, or three bytes of padding added so that the contents of the HUMAN-READABLE-INFO TLV is 32-bit aligned. The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver. If the TLV is already 32-bit aligned, no padding is needed.

5.2.7 DIGEST

The following is the format of the DIGEST attribute.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0 0 0 0 1 1 0|M|0 0 0 1 1 0 0 0|                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
     |                                                               |
     +                                                               +
     |                                                               |
     +                          HMAC-SHA1                            +
     |                                                               |
     +                                                               +
     |                                                               |
     +                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                               |            Padding            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 13: DIGEST format

HMAC-SHA1: this 20-byte field contains an HMAC-SHA1 [1] of the BFCP message. Its calculation is described in Section 9.

Padding: two bytes of padding added so that the contents of the HMAC-SHA1 TLV is 32-bit aligned. The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver.

5.2.8 REQUEST-STATUS

The following is the format of the REQUEST-STATUS attribute.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0 0 0 0 1 1 1|M|0 0 0 0 0 1 0 0|Request Status |Queue Position |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 14: REQUEST-STATUS format

Request Status: this 8-bit field contains the status of the request, as described in the following table.

Value	Status
0	Pending
1	Accepted
2	Granted
3	Denied
4	Cancelled
5	Released
6	Revoked

Table 3: Request Status values

Queue Position: this 8-bit field contains, when applicable, the position of the floor request in the floor request queue at the server. If the Request Status value is different from Accepted, the floor control server does not implement a floor request queue, or the floor control server does not want to provide the client with this information, all the bits of this field SHOULD be set to zero.

A floor request is in Pending state if the floor control server needs to contact a floor chair in order to accept the floor request, but has not done it yet. Once the floor control chair accepts the floor request, the floor request is moved to the Accepted state.

5.2.9 ERROR-CODE

The following is the format of the ERROR-CODE attribute.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |0 0 0 1 0 0 0|M|    Length     |           Error Code          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                     Error Specific Details                    |
   /                                                               /
   /                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               |            Padding            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 15: ERROR-CODE format

Error Code: this 16-bit field contains an error code from the following table.

Value	Meaning
0	Conference does not Exist
1	Authentication Failed
2	Unknown Mandatory TLV
3	Floor Request ID Does Not Exist
4	Unauthorized Operation
5	User does not Exist
6	Invalid Nonce
7	DIGEST TLV Required
8	Invalid Floor ID
9	You have Already Reached the Maximum Number of Ongoing Floor Requests for this Floor

Table 4: Error Code meaning

Error Specific Details: Present only for certain Error Codes. In this document, only for Error Code 2 (Unknown Mandatory TLV). For Error Code 2, this field contains the Types of the TLVs (which were present in the message that triggered the Error message) that were unknown to the receiver, encoded as follows.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Unknown TLV Type         |      Unknown TLV Type         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
/                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               |      Unknown TLV Type         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Unknown TLV Type         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 16: Unknown TLVs format

Padding: one, two, or three bytes of padding added so that the contents of the ERROR-CODE TLV is 32-bit aligned. If the TLV is already 32-bit aligned, no padding is needed.

The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver. Note all the Error Codes defined in this document but Error Code 2, result in a TLV which is already 32-bit aligned (i.e., no need of padding). Error Code 2 results in a TLV

that may need 2 bytes of padding.

5.2.10 USER-DISPLAY-NAME

The format of the USER-DISPLAY-NAME attribute is the same as the HUMAN-READABLE-INFO attribute (still, they have different attribute types). The Text field in the USER-DISPLAY-NAME attribute contains the name of the user.

5.2.11 USER-URI

The format of the USER-URI attribute is the same as the HUMAN-READABLE-INFO attribute (still, they have different attribute types). The Text field in the USER-URI attribute contains the URI of the user.

5.2.12 PRIORITY

The following is the format of the PRIORITY attribute.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |0 0 0 1 0 1 1|M|0 0 0 0 0 1 0 0|   Priority    |   Reserved    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 17: PRIORITY format

Priority: the higher the 8-bit value, the more priority is requested for a given floor request.

Reserved: at this point, the 8 bits in the reserved field SHOULD be set to zero by the sender of the message and MUST be ignored by the receiver.

5.2.13 NONCE

The following is the format of the NONCE attribute.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |0 0 0 1 1 0 0|M|    Length     |          Nonce Value          |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 18: NONCE format

Nonce Value: this 16-bit field contains a nonce.

5.2.14 SUPPORTED-TLVS

The following is the format of the SUPPORTED-TLVS attribute.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |0 0 0 1 1 0 1|M|    Length     |         Supported TLV         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Supported TLV         |         Supported TLV         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    /                                                               /
    /                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                               |            Padding            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 19: SUPPORTED-TLVS format

Supported TLV: these fields contain the Types of the TLVs that are supported by the floor control server.

Padding: two bytes of padding added so that the contents of the SUPPORTED-TLVS TLV is 32-bit aligned. If the TLV is already 32-bit aligned, no padding is needed.

The Padding bits SHOULD be set to zero by the sender and MUST be ignored by the receiver.

5.3 Message Format

This section contains the normative ABNF [3] of the BFCP messages.

5.3.1 FloorRequest

Floor participants request a floor by sending a FloorRequest message to the floor control server. The following is the format of the

FloorRequest message:

```
FloorRequest =   (FIXED-HEADER)
                 (TRANSACTION-ID)
                 (USER-ID)
                 [BENEFICIARY-ID]
                *(FLOOR-ID)
                 [HUMAN-READABLE-INFO]
                 [PRIORITY]
                 [NONCE]
                 [DIGEST]
```

Figure 20: FloorRequest format

5.3.2 FloorRelease

Floor participants release a floor by sending a FloorRelease message to the floor control server. Floor participants also use the FloorRelease message to cancel pending floor requests. The following is the format of the FloorRelease message:

```
FloorRelease =   (FIXED-HEADER)
                 (TRANSACTION-ID)
                 (USER-ID)
                 (FLOOR-REQUEST-ID)
                 [NONCE]
                 [DIGEST]
```

Figure 21: FloorRelease format

5.3.3 FloorRequestInfoWanted

Floor participants and floor chairs request information about a floor request by sending a FloorRequestInfoWanted message to the floor control server. The following is the format of the FloorRequestInfoWanted message:

```
   FloorRequestInfoWanted =   (FIXED-HEADER)
                              (TRANSACTION-ID)
                              (USER-ID)
                              [BENEFICIARY-ID]
                              [FLOOR-REQUEST-ID]
                              [NONCE]
                              [DIGEST]
```

Figure 22: FloorRequestInfoWanted format

5.3.4 FloorRequestInfo

The floor control server informs floor participants and floor chairs about the status of their floor requests by sending them FloorRequestInfo messages. The following is the format of the FloorRequestInfo message:

```
   FloorRequestInfo =     (FIXED-HEADER)
                          (TRANSACTION-ID)
                          (USER-ID)
                          [BENEFICIARY-ID]
                          [USER-DISPLAY-NAME]
                          [USER-URI]
                     1*(  (FLOOR-REQUEST-ID)
                        1*(FLOOR-ID)
                          [HUMAN-READABLE-INFO]
                          [PRIORITY]
                          (REQUEST-STATUS)     )
                          [NONCE]
```

Figure 23: FloorRequestInfo format

5.3.5 FloorInfoWanted

Floor participants and floor chairs request information about a floor or floors by sending a FloorInfoWanted message to the floor control server. The following is the format of the FloorRequest message:

```
   FloorInfoWanted =   (FIXED-HEADER)
                       (TRANSACTION-ID)
                       (USER-ID)
                      *(FLOOR-ID)
                       [NONCE]
                       [DIGEST]
```

Figure 24: FloorInfoWanted format

5.3.6 **FloorInfo**

The floor control server informs floor participants and floor chairs about the status (e.g., the current holder) of a floor by sending them FloorInfo messages. The following is the format of the FloorInfo message:

```
   FloorInfo        =        (FIXED-HEADER)
                             [TRANSACTION-ID]
                             (USER-ID)
                             [FLOOR-ID]
                          *( (FLOOR-REQUEST-ID)
                             [BENEFICIARY-ID]
                             [USER-DISPLAY-NAME]
                             [USER-URI]
                            *(FLOOR-ID)
                             [HUMAN-READABLE-INFO]
                             [PRIORITY]
                             (REQUEST-STATUS)      )
                             [NONCE]
```

Figure 25: FloorInfo format

5.3.7 **ChairAction**

Floor chairs send instructions to floor control servers by sending ChairAction messages. The following is the format of the ChairAction message:

```
   ChairAction  =   (FIXED-HEADER)
                    (TRANSACTION-ID)
                    (USER-ID)
                  1*(FLOOR-ID)
                    (FLOOR-REQUEST-ID)
                    (REQUEST-STATUS)
                    [HUMAN-READABLE-INFO]
                    [NONCE]
                    [DIGEST]
```

Figure 26: ChairAction format

5.3.8 ChairActionAck

Floor control servers confirm that they have accepted a ChairAction message by sending a ChairActionAck message. The following is the format of the ChairActionAck message:

```
   ChairActionAck  =   (FIXED-HEADER)
                       (TRANSACTION-ID)
                       (USER-ID)
                       [NONCE]
```

Figure 27: ChairActionAck format

5.3.9 Hello

Floor participants and floor chairs check the liveness of floor control servers by sending a Hello message. The following is the format of the Hello message:

```
   Hello          =  (FIXED-HEADER)
                     (TRANSACTION-ID)
                     (USER-ID)
                     [NONCE]
                     [DIGEST]
```

Figure 28: Hello format

5.3.10 HelloAck

Floor control servers confirm that they are alive on reception of a Hello message by sending a HelloAck message. The following is the

format of the HelloAck message:

```
HelloAck       =  (FIXED-HEADER)
                  (TRANSACTION-ID)
                  (USER-ID)
                  (SUPPORTED-TLVS)
                  [NONCE]
```

Figure 29: HelloAck format

5.3.11 Error

Floor control servers inform floor participants and floor chairs about errors processing requests by sending them Error messages. The following is the format of the Error message:

```
Error               =   (FIXED-HEADER)
                        (TRANSACTION-ID)
                        (USER-ID)
                        (ERROR-CODE)
                        [NONCE]
                        [HUMAN-READABLE-INFO]
```

Figure 30: Error format

6. Transport

BFCP entities exchange BFCP messages using TCP connections. TCP provides an in-order reliable delivery of a stream of bytes. Consequently, message framing is implemented in the application layer. BFCP implements application-layer framing using TLVs.

If a floor control server detects that the TCP connection towards one of the floor participants is lost, it is up to the local policy of the floor control server what to do with the pending floor requests of the floor participant. The floor control server MAY cancel all the floor participant's floor requests or it MAY keep them while the TCP connection is re-established. Connection re-establishment is handled in different ways depending on how the client obtains information to contact the floor control server (as described in Section 3.2, two possibilities are CPCP and an offer/answer exchange).

7. Lower-Layer Security

BFCP relies on lower-layer security mechanisms to provide replay and integrity protection, and confidentiality. BFCP floor control servers MUST support TLS [4], and BFCP clients (which include both floor participants and floor chairs) SHOULD support TLS. Any BFCP entity MAY support other security mechanisms.

BFCP entities that implement TLS MUST support, at a minimum, the TLS TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite .

8. Protocol Transactions

In BFCP, there are two types of transactions: client-initiated transactions and server-initiated transactions (notifications). Client-initiated transactions consist of a request from a client to a floor control server and a response from the floor control server to the client. The request carries a TRANSACTION-ID TLV which the floor control server copies into the response. Clients use Transaction ID values to match responses with previously-issued requests.

Server-initiated transactions consist of a single message from a floor control server to a client. Since they do not trigger any response, server-initiated transactions do not have Transaction IDs associated with them.

8.1 Client Behavior

A client starting a client-initiated transaction MUST set the Conference ID in the FIXED-HEADER of the message to the Conference ID for the conference that the client obtained previously.

The client MUST set the Transaction ID value in the TRANSACTION-ID TLV to a number which MUST NOT be reused in another message from the client until a response from the server is received for the transaction. The client uses the Transaction ID value to match this message with the response from the floor control server.

8.2 Server Behavior

A floor control server sending a response within a client-initiated transaction MUST copy the Conference ID, the TRANSACTION-ID TLV, and the USER-ID TLV from the request received from the client into the response. Server-initiated transactions MUST NOT contain a TRANSACTION-ID TLV.

9. Authentication and Authorization

BFCP uses the DIGEST TLV to provide client authentication. The DIGEST TLV contains an HMAC-SHA1 [1] of the BFCP message. The use of SHA1 implies that the length of the HMAC is 20 bytes. The text used as input to HMAC is the BFCP message, including the FIXED-HEADER, up to and including the TLV preceding the DIGEST TLV. This text is then padded with zeroes so as to be a multiple of 64 bytes. As a result, the DIGEST TLV MUST be the last attribute in any BFCP message. The key used as input to HMAC is the secret shared between the server and the user identified by the USER-ID TLV in the message.

9.1 Client Behavior

Clients can authenticate floor control servers by checking the floor control server's certificate when the TLS connection is established between them.

To achieve client authentication, a client needs to prove to the floor control server that the client can produce a DIGEST TLV for a message using their shared secret and that the message is fresh (to avoid replay attacks). Clients prove the freshness of a message by including a NONCE TLV in the message. The NONCE TLV is the second to last TLV in the message (the last one is the DIGEST TLV).

The nonce in the NONCE TLV is provided by the floor control server using an out-of-band mechanism (e.g., using an offer/answer exchange as described in [15]), or in an Error response -- typically with Error Code 7 (DIGEST TLV Required) or 6 (Invalid Nonce).

A client that obtains a nonce out-of-band SHOULD add a NONCE TLV and a DIGEST TLV to the first message it sends to the floor control server. Furthermore, if a client generates a message without this TLV and receives an Error response with Error Code 7 (DIGEST TLV Required), the client SHOULD re-send the message with a DIGEST TLV and a NONCE TLV with the nonce received in the Error response.

If after sending a message with a DIGEST TLV, a client receives an Error response with Error Code 6 (Invalid Nonce), the client SHOULD re-send the message using the new nonce received in the Error response. If the Error Code is 1 (Authentication Failed) instead, the client MUST NOT send further messages to the floor control server until it has obtained a different (hopefully valid) shared secret than the one used in the original message.

If a client receives a nonce in a message from the floor control server, the client SHOULD add a NONCE TLV with this nonce and a DIGEST TLV to its next message to the floor control server.

9.2 Floor Control Server Behavior

Before accepting any BFCP message, the floor control server SHOULD authenticate the client. If the floor control server receives a message without DIGEST TLV from an unauthenticated client, the floor control server responds with an Error message with Error Code 7 (DIGEST TLV Required). The floor control message MUST include a NONCE TLV with a nonce value that is unguessable by attackers.

When a floor control server receives a BFCP message with a DIGEST TLV, it checks whether the NONCE TLV carries a nonce which was generated by the floor control server for this client and which still has not expired. If the nonce is not valid, authentication is considered to have failed, in which case the floor control server SHOULD return an Error message with Error Code 6 (Invalid Nonce) with a new nonce in a NONCE TLV.

If the nonce is valid, the floor control server calculates the HMAC-SHA1 [1] of the message excluding the DIGEST TLV. The key used as input to HMAC is the secret shared between the server and the user identified by the USER-ID TLV in the message. If the resulting value is the same as the one in the DIGEST TLV, authentication is considered successful.

If the resulting value is different than the one in the DIGEST TLV, authentication is considered to have failed, in which case the server SHOULD return an Error message, as described in Section 13.7, with Error Code 1 (Authentication Failed). Messages from a client that cannot be authenticated MUST NOT be processed further.

Floor control servers may include a NONCE TLV in their responses to provide the nonce to be used in the next message by the client. However, when TLS is used, floor control servers typically authenticate only the first message sent over the TLS connection.

After authenticating a BCFP message, the floor control server checks whether or not the client is authorized to perform the operation it is requesting. If the client is not authorized to perform the operation being requested, the floor control server generates an Error message, as described in Section 13.7, with an Error code with a value of 4 (Unauthorized Operation). Messages from a client that cannot be authorized MUST NOT be processed further.

10. Floor Participant Operations

This section specifies how floor participants can perform different operations, such as requesting a floor, using the protocol elements described in earlier sections. Section 11 specifies operations that

are specific to floor chairs, such as instructing the floor control server to grant or revoke a floor, and Section 12 specifies operations that can be performed by any client (i.e., both floor participants and floor chairs).

10.1 Requesting a Floor

A floor participant that wishes to request one or more floors does so by sending a FloorRequest message to the floor control server.

10.1.1 Sending a FloorRequest Message

The ABNF in Section 5.3.1 describes the TLVs that a FloorRequest message can contain. In addition, the ABNF specifies normatively which of these TLVs are mandatory, and which ones are optional.

The floor participant sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 8.1. Additionally, the floor participant follows the rules in Section 9.1 which relate to the authentication of the message (i.e., to the DIGEST TLV).

The floor participant must insert a USER-ID TLV, which will be used by the floor control server to authenticate and authorize the request. If the sender of the FloorRequest message (identified by the USER-ID TLV) is not the participant that would eventually get the floor (i.e., a third party floor request), the sender SHOULD add a BENEFICIARY-ID TLV to the message identifying the beneficiary of the floor.

> Note that the name space for both the User ID and the Beneficiary ID is the same. That is, a given participant is identified by a single 16-bit value that can be used in USER-ID and in BENEFICIARY-ID TLVs.

The floor participant must insert at least one FLOOR-ID TLV in the FloorRequest message. If the client inserts more than one FLOOR-ID TLVs, the floor control server will treat all the floor requests as an atomic package. That is, the floor control server will either grant or deny all the floors in the FloorRequest message.

The floor participant may use a HUMAN-READABLE-INFO TLV to state the reason why the floor or floors are being requested. The Text field in the HUMAN-READABLE-INFO TLV is intended for human consumption.

The floor participant may request the server to handle the floor request with a certain priority using a PRIORITY TLV.

10.1.2 Receiving a Response

A message from the floor control server is considered to be a response to the FloorRequest message if the message from the floor control server has the same Conference ID, Transaction ID, and User ID as the FloorRequest message, as described in Section 8.1.

The successful processing of a FloorRequest message at the floor control server involves generating one or several FloorRequestInfo messages. The floor participant obtains a Floor Request ID in a FLOOR-REQUEST-ID TLV in the first FloorRequestInfo message from the floor control server. Subsequent FloorRequestInfo messages from the floor control server regarding the same floor request will carry the same Floor Request ID as the initial FloorRequestInfo message. This way, the floor participant can associate subsequent incoming FloorRequestInfo messages with the ongoing floor request.

The floor participant obtains information about the status of the floor request in the REQUEST-STATUS TLV of each of the FloorRequestInfo messages received from the floor control server. If the Request Status value is Granted, all the floors that were requested in the FloorRequest message have been granted. If the Request Status value is Denied, all the floors that were requested in the FloorRequest message have been denied. The HUMAN-READABLE-INFO TLV, if present, provides extra information which the floor participant MAY display to the user.

A floor request is considered to be ongoing while it is in the Pending, Accepted, or Granted states.

If the response is an Error message, the floor control server could not process the FloorRequest message for some reason, which is described in the Error message.

10.2 Cancelling a Floor Request and Releasing a Floor

A floor participant that wishes to cancel an ongoing floor request does so by sending a FloorRelease message to the floor control server. The FloorRelease message is also used by floor participants that hold a floor and would like to release it.

10.2.1 Sending a FloorRelease Message

The ABNF in Section 5.3.2 describes the TLVs that a FloorRelease message can contain. In addition, the ABNF specifies normatively which of these TLVs are mandatory, and which ones are optional.

The floor participant sets the Conference ID in the FIXED-HEADER and

the TRANSACTION-ID TLV following the rules given in Section 8.1. Additionally, the floor participant follows the rules in Section 9.1 which relate to the authentication of the message (i.e., to the DIGEST TLV). The floor participant must insert a USER-ID TLV, which will be used by the floor control server to authenticate and authorize the request.

Note that the FloorRelease message is used to release a floor or floors that were granted and to cancel ongoing floor requests (from the protocol perspective both are ongoing floor requests). Using the same message in both situations helps resolve the race condition that occurs when the FloorRelease message and the FloorGrant message cross each other on the wire.

The floor participant uses the FLOOR-REQUEST-ID that was received in the response to the FloorRequest message that the FloorRelease message is cancelling.

Note that if the floor participant requested several floors as an atomic operation (i.e., in a single FloorRequest message), all the floors are released as an atomic operation as well (i.e., all are released at the same time).

10.2.2 Receiving a Response

A message from the floor control server is considered to be a response to the FloorRelease message if the message from the floor control server has the same Conference ID, Transaction ID, and User ID as the FloorRequest message, as described in Section 8.1.

If the response is a FloorRequestInfo message, the Request Status value in the REQUEST-STATUS-TLV will be Cancelled or Released.

If the response is an Error message, the floor control server could not process the FloorRequest message for some reason, which is described in the Error message.

It is possible that the FloorRelease message crosses on the wire with a FloorRequestInfo message from the server with a Request Status different from Cancelled or Released. In any case, such a FloorRequestInfo message will not be a response to the FloorRelease message, because its Transaction ID will not match that of the FloorRelease.

11. Chair Operations

This section specifies how floor chairs can instruct the floor control server to grant or revoke a floor using the protocol elements

described in earlier sections.

Floor chairs that wish to send instructions to a floor control server do so by sending a ChairAction message.

11.1 Sending a ChairAction Message

The ABNF in Section 5.3.7 describes the TLVs that a ChairAction message can contain. In addition, the ABNF specifies normatively which of these TLVs are mandatory, and which ones are optional.

The floor chair sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 8.1. Additionally, the floor chair follows the rules in Section 9.1 which relate to the authentication of the message (i.e., to the DIGEST TLV). The floor chair must insert a USER-ID TLV, which will be used by the floor control server to authenticate and authorize the request.

The ChairAction message contains instructions that apply to one or more floors within a particular floor request. The floor or floors are identified by FLOOR-ID TLVs and the floor request is identified by a FLOOR-REQUEST-ID TLV, which are carried in the ChairAction message.

> For example, if a floor request consists of two floors that depend on different floor chairs, each floor chair will grant its floor within the floor request. Once both chairs have granted their floor, the floor control server will grant the floor request as a whole. On the other hand, if one of the floor chairs denies its floor, the floor control server will deny the floor request as a whole, regardless of the other floor chair's decision.

The floor chair provides the new status for one or more floors within the floor request using a REQUEST-STATUS TLV. If the new status of the floor request is Accepted, the floor chair MAY use the Queue Position field to provide a queue position for the floor request. If the floor chair does not wish to provide a queue position, all the bits of the Queue Position field SHOULD be set to zero. The floor chair SHOULD use the Status Revoked to revoke a floor that was granted (i.e., Granted status) and the Status Denied to reject floor requests in any other status (e.g., Pending and Accepted).

> Note that a floor request may involve several floors and that a ChairAction message may only deal with a subset of these floors (e.g., if a single floor chair is not authorized to manage all the floors). In this case, the REQUEST-STATUS that the floor chair provides in the ChairAction message might not be the actual status

that the floor request gets at the server. The floor control server will combine the instructions received from the different floor chairs to come up with the actual status of the floor request.

The floor chair may use a HUMAN-READABLE-INFO TLV to state the reason why the floor or floors are being accepted, granted, or revoked. The Text in the HUMAN-READABLE-INFO TLV is intended for human consumption.

11.2 Receiving a Response

A message from the floor control server is considered to be a response to the ChairAction message if the message from the server has the same Conference ID, Transaction ID, and User ID as the ChairAction message, as described in Section 8.1.

A ChairActionAck message from the floor control server confirms that the floor control server has accepted the ChairAction message. An Error message indicates that the floor control server could not process the ChairAction message for some reason, which is described in the Error message.

12. General Client Operations

This section specifies operations that can be performed by any client. That is, they are not specific to floor participants or floor chairs. They can be performed by both.

12.1 Requesting Information about Floors

A client can obtain information about the status of a floor or floors in different ways, which include using BFCP and using out-of-band mechanisms. Clients using BFCP to obtain such information use the procedures described in this section.

Clients request information about the status of one or several floors by sending a FloorInfoWanted message to the floor control server.

12.1.1 Sending a FloorInfoWanted Message

The ABNF in Section 5.3.5 describes the TLVs that a FloorInfoWanted message can contain. In addition, the ABNF specifies normatively which of these TLVs are mandatory, and which ones are optional.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 8.1. Additionally, the client follows the rules in Section 9.1 which

relate to the authentication and the protection of the integrity of the message (i.e., to the DIGEST TLV). The client must insert a USER-ID TLV, which will be used by the floor control server to authenticate and authorize the request.

The client inserts in the message all the Floor IDs it wants to receive information about. The floor control server will send periodic information about all these floors. If the client does not want to receive information about a particular floor any longer, it sends a new FloorInfoWanted message removing the FLOOR-ID of this floor. If the client does not want to receive information about any floor any longer, it sends a FloorInfoWanted message with no FLOOR-ID TLV.

12.1.2 Receiving a Response

A message from the floor control server is considered to be a response to the FloorInfoWanted message if the message from the floor control server has the same Conference ID, Transaction ID, and User ID as the FloorRequest message, as described in Section 8.1.

On reception of the FloorInfoWanted message, the floor control server will respond with a FloorInfo message or with an Error message. If the response is a FloorInfo message, it will contain information about one of the floors the client requested information about. If the client did not include any FLOOR-ID TLV in its FloorInfoWanted message, the FloorInfo message from the floor control server will not include any either.

FloorInfo messages which carry information about a floor contain a FLOOR-ID TLV that identifies the floor. After this TLV, FloorInfo messages contain information about existing (one or more) floor request that relate to that floor. The information about each particular floor request consist of a FLOOR-REQUEST-ID TLV that identifies the floor request followed by a set of TLVs that provide information about the floor request.

After the first FloorInfo, the floor control server will continue sending FloorInfo messages periodically informing the client about changes on the floors the client requested information about.

12.2 Requesting Information about Floor Requests

A client can obtain information about the status of one or several floor requests in different ways, which include using BFCP and using out-of-band mechanisms. Clients using BFCP to obtain such information use the procedures described in this section.

Clients request information about the current status of one or several floor requests by sending a FloorRequestInfoWanted message to the floor control server.

Requesting information about a particular floor request is useful in a number of situations. For example, on reception of a FloorRequest message, a floor control server may choose to return FloorRequestInfo messages only when the floor request changes its state (e.g., from Accepted to Granted), but not when the floor request advances in its queue. In this situation, if the user requests it, the floor participant can use a FloorRequestInfoWanted message to poll the floor control server for the status of the floor request.
FloorRequestInfoWanted messages can also be used to request information on all the floor requests associated with a floor participant. For example, a floor participant, after experiencing connectivity problems (e.g., its TCP connection with the floor control server was down for a while and eventually was re-established), may need to request information about all the still existing floor requests associated to the floor participant.

12.2.1 Sending a FloorRequestInfoWanted Message

The ABNF in Section 5.3.3 describes the TLVs that a FloorRequestInfoWanted message can contain. In addition, the ABNF specifies normatively which of these TLVs are mandatory, and which ones are optional.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 8.1. Additionally, the client follows the rules in Section 9.1 which relate to the authentication of the message (i.e., to the DIGEST TLV). The client must insert a USER-ID TLV, which will be used by the floor control server to authenticate and authorize the request.

If the client wants to request the status of a single floor request, it MUST insert a FLOOR-REQUEST-ID TLV that identifies the floor request at the floor control server.

The client can also request information about all the ongoing floor requests associated with a particular participant. In this case, the client MUST NOT insert a FLOOR-REQUEST-ID TLV. If the beneficiary of the floor requests the client is requesting information about is not the client issuing the FloorRequestInfoWanted message (which is identified by the USER-ID TLV in the message) the client MUST insert a BENEFICIARY-ID TLV.

12.2.2 Receiving a Response

A message from the floor control server is considered to be a response to the FloorRequestInfoWanted message if the message from the floor control server has the same Conference ID, Transaction ID,a nd User ID as the FloorRequestInfoWanted message, as described in Section 8.1.

If the response is a FloorRequestInfo message, the client obtains information about the status of the FloorRequest the client requested information about in a REQUEST-STATUS TLVs. If the client requested information about several floor requests, the FloorRequestInfo message will carry several FLOOR-REQUEST-ID TLVs. Each FLOOR-REQUEST-ID TLV will be followed by TLVs (which will include a REQUEST-STATUS TLV) providing information about the floor request identified by the FLOOR-REQUEST-ID TLV.

If the response is an Error message, the floor control server could not process the FloorRequestInfoWanted message for some reason, which is described in the Error message.

12.3 Obtaining the Capabilities of a Floor Control Server

A client that wishes to obtain the capabilities of a floor control server does so by sending a Hello message to the floor control server.

12.3.1 Sending a Hello Message

The ABNF in Section 5.3.9 describes the TLVs that a Hello message can contain. In addition, the ABNF specifies normatively which of these TLVs are mandatory, and which ones are optional.

The client sets the Conference ID in the FIXED-HEADER and the TRANSACTION-ID TLV following the rules given in Section 8.1. Additionally, the client follows the rules in Section 9.1 which relate to the authentication and the protection of the integrity of the message (i.e., to the DIGEST TLV). The client must insert a USER-ID TLV, which will be used by the floor control server to authenticate and authorize the request.

12.3.2 Receiving Responses

A message from the floor control server is considered a response to the Hello message by the client if the message from the floor control server has the same Conference ID, Transaction ID, and User ID as the Hello message, as described in Section 8.1.

If the response is a HelloAck message, the floor control server could process successfully the Hello message. The SUPPORTED-TLVS TLV indicates which TLVs are supported by the server.

If the response is an Error message, the floor control server could not process the Hello message for some reason, which is described in the Error message.

13. Floor Control Server Operations

This section specifies how floor control servers can perform different operations, such as granting a floor, using the protocol elements described in earlier sections.

On reception of a message from a client, the floor control server MUST check whether or not the value of the Conference ID matched an existing conference. If it does not, the floor control server SHOULD send an Error message, as described in Section 13.7, with Error code 0 (Conference does not Exist).

On reception of a message from a client, the floor control server follows the rules in Section 9.2, which relate to the authentication of the message.

On reception of a message from a client, the floor control server MUST check whether or not it understands all the mandatory ('M' bit set) TLVs in the message. If the floor control server does not understand all of them, the floor control server SHOULD send an Error message, as described in Section 13.7, with Error code 2 (Authentication Failed). The Error message SHOULD list the TLVs that were not understood.

13.1 Reception of a FloorRequest Message

On reception of a FloorRequest message, the floor control server follows the rules in Section 9.2 which relate to client authentication and authorization. If while processing the FloorRequest message, the floor control server encounters an error, it SHOULD generate an Error response following the procedures described in Section 13.7

BFCP allows floor participants to have several ongoing floor requests for the same floor (e.g., the same floor participant can occupy more than one position in a queue at the same time). A floor control server that only supports a certain number of ongoing floor requests per floor participant (e.g., one) can use Error Code 9 (You have Already Reached the Maximum Number of Ongoing Floor Requests for this Floor) to inform the floor

participant.

13.1.1 Generating the First FloorRequestInfo Message

The successful processing of a FloorRequest message by a floor control server involves generating one or several FloorRequestInfo messages, the first of which SHOULD be generated as soon as possible. If the floor control server cannot accept, grant, or deny the floor request right away (e.g., a decision from a chair is needed), it SHOULD use a Request Status value of Pending in the REQUEST-STATUS TLV of the first FloorRequestInfo message it generates.

The policy a floor control server follows to grant or deny floors is outside the scope of this document. A given floor control server may perform these decisions automatically while another may contact a human acting as a chair everytime a decision needs to be made.

The floor control server copies the Conference ID, the TRANSACTION-ID, and the USER-ID TLVs from the FloorRequest into the FloorRequestInfo, as described in Section 8.2. Additionally, the floor control server copies (if present) the BENEFICIARY-ID TLV from the FloorRequest into the FloorRequestInfo.

The floor control server MUST assign an identitifier that is unique within the conference to this floor request, and insert it in a FLOOR-REQUEST-ID TLV into the FloorRequestInfo message. This indentifier will be used by the floor participant (or by a chair or chairs) to refer to this specific floor request in the future.

The floor control server copies the FLOOR-ID TLVs from the FloorRequest into the FloorRequestInfo. These FLOOR-ID TLVs identify the floors being requested (i.e., the floors associated with this particular floor request).

The floor control server also copies (if present) the PRIORITY TLV from the FloorRequest into the FloorRequestInfo. The Priority value requested by the floor participant is only a hint, and does not necessarily need to be taken into consideration to decide whether to grant or not the floor request.

13.1.2 Generation of Subsequent FloorRequestInfo Messages

A floor request is considered to be ongoing as long as it is not in the Cancelled, Released, or Revoked states. If the REQUEST-STATUS TLV of the first FloorRequestInfo message generated by the floor control server did not indicate any of these states, the floor control server will need to send subsequent FloorRequestInfo

messages.

When the status of the floor request changes, the floor control floor control server SHOULD send new FloorRequestInfo messages with the appropriate Request Status. These FloorRequestInfo messages MUST contain a FLOOR-REQUEST-ID TLV equal to the one sent in the first FloorRequestInfo message, but MUST NOT contain any TRANSACTION-ID TLV. (The Floor Request ID identifies the floor request the FloorRequestInfo applies to.)

The FIXED-HEADER and the rest of the TLVs (expect for the HUMAN-READABLE-INFO TLV) are the same as in the first FloorRequestInfo message.

> The rate at which the floor control server sends FloorRequestInfo messages is a matter of local policy. A floor control server may choose to send a new FloorRequestInfo message every time the floor request moves in the floor request queue while another may choose to only send a new FloorRequestInfo message when the floor request is Granted or Denied.

The floor control server may add a HUMAN-READABLE-INFO TLV to any of the FloorRequestInfo messages it generates to provide extra information about its decisions regarding the floor request (e.g., why it was denied).

> Floor participants and floor chairs may request to be informed about the status of a floor following the procedures in Section 12.1. If the processing of a floor request changes the status of a floor (e.g., the floor request is granted and consequently the floor has a new holder), the floor control server needs to follow the procedures in Section 13.4 to inform the clients that have requested that information

The floor control server can discard the state information about a particular floor request when this reaches a status of Cancelled, Released, or Revoked.

13.2 Reception of a FloorRequestInfoWanted Message

On reception of a FloorRequestInfoWanted message, the floor control server follows the rules in Section 9.2 which relate to client authentication and authorization. If while processing the FloorRequestInfoWanted message, the floor control server encounters an error, it SHOULD generate an Error response following the procedures described in Section 13.7

The successful processing of a FloorRequestInfoWanted message by a

floor control server involves generating a FloorRequestInfo message, which SHOULD be generated as soon as possible.

The floor control server copies the Conference ID, the TRANSACTION-ID, and the USER-ID TLVs from the FloorRequestInfoWanted message into the FloorRequestInfo message, as described in Section 8.2.

13.2.1 Information on a Single Floor Request

If the FloorRequestInfoWanted message carries a FLOOR-REQUEST-ID, the sender of the message is requesting information about the floor request identified by the FLOOR-REQUEST-ID TLV. The floor control servre copies the FLOOR-REQUEST-ID TLV from the FloorRequestInfoWanted message into the FloorRequestInfo message.

The floor control server adds FLOOR-ID TLVs to the FloorRequestInfo message identifying the floors being requested (i.e., the floors associated with the floor request identified by the FLOOR-REQUEST-ID TLV).

The floor control server may also add a PRIORITY TLV with the Priority value requested for the floor request and a HUMAN-READABLE-INFO TLV with extra information about the floor request.

The floor control server adds a REQUEST-STATUS TLV with the current status of the floor request.

13.2.2 Information on the Floor Requests Associated to a Participant

If the FloorRequestInfoWanted message does not carry a FLOOR-REQUEST-ID TLV, the sender of the message is requesting information about all the floor requests from a given participant. This participant is identified by a BENEFICIARY-ID TLV or, in the absence of a BENEFICIARY-ID TLV, by a USER-ID TLV.

The floor control server copies (if present) the BENEFICIARY-ID TLV from the FloorRequestInfoWanted message into the FloorRequestInfo message. Additionally, the floor control server may provide extra information about the participant by adding a USER-DISPLAY-NAME TLV, a USER-URI TLV, or both to the FloorRequestInfo message.

The floor control server adds a FLOOR-REQUEST-ID TLV for each floor request associated to the participant. Each FLOOR-REQUEST-ID TLV is followed by a number of TLVs which provide information about the floor request. The floor control server generates the TLVs that follow each FLOOR-REQUEST-ID following the rules in Section 13.2.1

13.3 Reception of a FloorRelease Message

On reception of a FloorRelease message, the floor control server follows the rules in Section 9.2 which relate to client authentication and authorization. If while processing the FloorRelease message, the floor control server encounters an error, it SHOULD generate an Error response following the procedures described in Section 13.7

The successful processing of a FloorRelease message by a floor control server involves generating a FloorRequestInfo message, which SHOULD be generated as soon as possible.

The floor control server copies the Conference ID, the TRANSACTION-ID, and the USER-ID TLVs from the FloorRelease message into the FloorRequestInfo message, as described in Section 8.2.

The FloorRelease message identifies the floor request it applies to using a FLOOR-REQUEST-ID. If the beneficiary of the floor request is not the participant identified by the USER-ID TLV in the FloorRelease message, the floor control server adds a BENEFICIARY-ID TLV to the FloorRequestInfo message identifying the beneficiary of the floor request. Additionally, the floor control server may provide extra information about the beneficiary of the floor request by adding a USER-DISPLAY-NAME TLV, a USER-URI TLV, or both to the FloorRequestInfo message.

The floor control server copies the FLOOR-REQUEST-ID TLV from the FloorRelease message into the FloorRequestInfo message.

The floor control server adds FLOOR-ID TLVs to the FloorRequestInfo message identifying the floors being requested (i.e., the floors associated with the floor request identified by the FLOOR-REQUEST-ID TLV).

The floor control server may also add a PRIORITY TLV with the Priority value requested for the floor request and a HUMAN-READABLE-INFO TLV with extra information about the floor request.

The floor control server adds a REQUEST-STATUS TLV to the FloorRequestInfo message. The Request Status value SHOULD be Released, if the floor (or floors) had been previously granted, or of Cancelled, if the floor (or floors) had not been previously granted.

13.4 Reception of a FloorInfoWanted Message

On reception of a FloorInfoWanted message, the floor control server

follows the rules in Section 9.2 which relate to client authentication. If while processing the FloorRelease message, the floor control server encounters an error, it SHOULD generate an Error response following the procedures described in Section 13.7

A floor control server receiving a FloorInfoWanted message from a client SHOULD keep this client informed about the status of the floors identified by FLOOR-ID TLVs in the FloorInfoWanted message. Floor Control Servers keep clients informed by using FloorInfo messages.

An individual FloorInfo message carries information about a single floor. So, when a FloorInfoWanted message requests information about more than one floors, the floor control server needs to send separate FloorInfo messages for different floors.

The information FloorInfoWanted messages carry may depend on the user requesting the information. For example, a chair may be able to receive information about pending requests while a regular user may not be authorized to do so.

13.4.1 Generation of the First FloorInfo Message

The successful processing of a FloorInfoWanted message by a floor control server involves generating one or several FloorInfo messages, the first of which SHOULD be generated as soon as possible.

The floor control server copies the Conference ID, the TRANSACTION-ID, and the USER-ID TLVs from the FloorInfoWanted message into the FloorInfo message, as described in Section 8.2.

If the FloorInfoWanted message did not contain any FLOOR-ID TLV, the floor control server sends the FloorInfo message without adding any additional TLV and does not send any subsequent FloorInfo message to the floor participant.

If the FloorInfoWanted message contained one or more FLOOR-ID TLVs, the floor control server chooses one among them and adds this FLOOR-ID TLV to the FloorInfo message. The floor control server adds a FLOOR-REQUEST-ID TLV for each floor request associated to the floor. Each FLOOR-REQUEST-ID TLV is followed by a number of TLVs which provide information about the floor request.

For each FLOOR-REQUEST-ID TLV, the floor control server may add a BENEFICIARY-ID TLV identifying the requester of the floor and a USER-DISPLAY-NAME TLV, a USER-URI TLV, or both providing information about the requester. Additionally, the floor control server adds FLOOR-ID TLVs to the FloorInfo message identifying the floors being

requested (i.e., the floors associated with the floor request identified by the FLOOR-REQUEST-ID TLV).

The floor control server may also add a PRIORITY TLV with the Priority value requested for the floor request and a HUMAN-READABLE-INFO TLV with extra information about the floor request.

The floor control server adds a REQUEST-STATUS TLV with the current status of the floor request.

13.4.2 **Generation of Subsequent** FloorInfo Messages

If the FloorInfoWanted message carried more than one FLOOR-ID TLV, the floor control server SHOULD generate a FloorInfo message for each of them (except for the FLOOR-ID TLV chosen for the first FloorInfo message) as soon as possible. These FloorInfo messages are generated following the same rules as for the first FloorInfo message (see Section 13.4.1, but without adding a TRANSACTION TLV.

After generating these messages, the floor control server sends FloorInfo messages periodically keeping the client informed about all the floors the client requested information about. These messages MUST NOT carry a TRANSACTION-ID TLV.

> The rate at which the floor control server sends FloorInfo messages is a matter of local policy. A floor control server may choose to send a new FloorInfo message every time a new floor request arrives while another may choose to only send a new FloorInfo message when a new floor request is Granted.

13.5 Reception of a ChairAction Message

On reception of a ChairAction message, the floor control server follows the rules in Section 9.2 which relate to client authentication and authorization. If while processing the ChairAction message, the floor control server encounters an error, it SHOULD generate an Error response following the procedures described in Section 13.7

The successful processing of a ChairAction message by a floor control server involves generating a ChairActionAck message, which SHOULD be generated as soon as possible.

The floor control server copies the Conference ID, the TRANSACTION-ID, and the USER-ID TLVs from the ChairAction message into the ChairActionAck message, as described in Section 8.2.

The floor control server needs to take into consideration the operation requested in the ChairAction message (e.g., granting a floor), but does not necessarily need to perform it as requested by the floor chair. The operation that the floor control server performs depends on the ChairAction message and on the internal state of the floor control server.

For example, a floor chair may send a ChairAction message granting a floor which was requested as part of an atomic floor request operation that involved several floors. Even if the chair responsible for one of the floors instructs the floor control server to grant the floor, the floor control server will not grant it until the chairs responsible for the other floors agree to grant them as well.

So, the floor control server is ultimately responsible to keep a coherent floor state using instructions from floor chairs as input to this state.

If the new Status in the ChairAction message is Accepted and all the bits of the Queue Position field are zero, the floor participant is requesting the floor control server to assign a queue position (e.g., the last in the queue) to the floor request based on the local policy of the floor control server. (Of course, such as request only applies in case the floor control server implements a queue.)

13.6 Reception of a Hello Message

On reception of a Hello message, the floor control server follows the rules in Section 9.2 which relate to client authentication. If while processing the Hello message, the floor control server encounters an error, it SHOULD generate an Error response following the procedures described in Section 13.7

The successful processing of a Hello message by a floor control server involves generating a HelloAck message, which SHOULD be generated as soon as possible. The floor control server copies the Conference ID, the TRANSACTION-ID, and the USER-ID TLVs from the Hello into the HelloAck, as described in Section 8.2.

The floor control server adds a SUPPORTED-TLVS TLV to the HelloAck message listing all the TLVs supported by the floor control server.

13.7 Error Message Generation

Error messages are always sent in response to a previous message from the client as part of a client-initiated transaction. The ABNF in Section 5.3.11 describes the TLVs that an Error message can contain.

In addition, the ABNF specifies normatively which of these TLVs are mandatory, and which ones are optional.

The floor control server copies the Conference ID, the TRANSACTION-ID, and the USER-ID TLVs from the message from the client into the Error message, as described in Section 8.2.

The floor control server adds an ERROR-CODE TLV to the Error message. The ERROR-CODE TLV contains an Error Code from Table 4. Additionally, the floor control server may add a HUMAN-READABLE-INFO TLV with extra information about the error.

14. Security Considerations

TBD.

15. IANA Considerations

TBD

16. Acknowledgments

The XCON WG chairs, Adam Roach and Alan Johnston, provided useful ideas for this document. Additionally, Xiaotao Wu, Paul Kyzivat, Jonathan Rosenberg, and Miguel A. Garcia-Martin provided useful comments.

17. References

17.1 Normative References

[1] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

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

[3] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[4] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[5] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

17.2 Informational References

[6] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[7] 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.

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

[9] Schulzrinne, H., "Requirements for Floor Control Protocol", draft-ietf-xcon-floor-control-req-01 (work in progress), July 2004.

[10] Koskelainen, P. and H. Khartabil, "An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Usage for Conference Policy Manipulation", draft-koskelainen-xcon-xcap-cpcp-usage-02 (work in progress), February 2004.

[11] Rosenberg, J. and H. Schulzrinne, "A Session Initiation Protocol (SIP) Event Package for Conference State", draft-ietf-sipping-conference-package-05 (work in progress), July 2004.

[12] Arkko, J., "MIKEY: Multimedia Internet KEYing", draft-ietf-msec-mikey-08 (work in progress), December 2003.

[13] Rosenberg, J., "An Extensible Markup Language (XML) Document Format for Indicating Changes in XML Configuration Access Protocol (XCAP) Resources", draft-ietf-simple-xcap-package-02 (work in progress), July 2004.

[14] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol", draft-ietf-sipping-conferencing-framework-02 (work in progress), June 2004.

[15] Camarillo, G., "Session Description Protocol (SDP) Format for Binary Floor Control Protocol (BFCP) Streams", draft-camarillo-mmusic-sdp-bfcp-00 (work in progress), April 2005.

Authors' Addresses

Gonzalo Camarillo
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: Gonzalo.Camarillo@ericsson.com

Joerg Ott
Universitaet Bremen
MZH 5180
Bibliothekstr. 1
Bremen D-28359
Germany

EMail: jo@tzi.org

Keith Drage
Lucent Technologies
Windmill Hill Business Park
Swindon
Wiltshire SN5 6PP
UK

EMail: drage@lucent.com

Acknowledgment

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