

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: December 30, 2010

A. Knauf  
G. Hege  
T C. Schmidt  
HAW Hamburg  
M. Waehlich  
link-lab & FU Berlin  
June 28, 2010

A RELOAD Usage for Distributed Conference Control (DisCo)  
draft-knauf-p2psip-disco-00

Abstract

This document defines a RELOAD Usage for Distributed Conference Control (DisCo) with SIP. DisCo splits the semantic of identifier and locator of a SIP conference URI using a new Kind data structure. Conference members are enabled to select conference controllers based on proximity awareness. DisCo proposes call delegation to balance load at focus peers. The document addresses also aspects of security and trust, as well as compatibility for conference unaware clients.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on December 30, 2010.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	4
3. Overview of DisCo . . . . .	5
3.1. Reference Scenario . . . . .	5
3.2. Initiating a Distributed Conference . . . . .	6
3.3. Joining a Conference . . . . .	7
3.4. Conference State Synchronization . . . . .	8
3.5. Call delegation . . . . .	9
3.6. Resilience . . . . .	9
3.7. Topology Awareness . . . . .	9
4. RELOAD Usage for Distributed Conference Control . . . . .	11
4.1. Kind Data Structure . . . . .	11
4.2. Determining Coordinates . . . . .	12
4.3. Conference Creation . . . . .	12
4.4. Proximity-aware Conference Participation . . . . .	14
4.5. Advertising Focus Ability . . . . .	16
4.6. Resilience in a Distributed Conference . . . . .	17
5. Focus Call Control Operations . . . . .	19
5.1. Becoming an active Focus . . . . .	19
5.2. Delegating Calls . . . . .	21
5.3. Synchronizing the Conference State . . . . .	22
6. DISCO Kind Definition . . . . .	24
7. Conference State Event Package Extension . . . . .	25
7.1. The <focus-states> and <focus> elements . . . . .	25
7.2. XML Schema . . . . .	26
8. Configuration Document Extension . . . . .	29
8.1. The <landmark> and <Landmark-host> elements . . . . .	29
8.2. Relax NG Grammar . . . . .	29
9. Example . . . . .	30
10. Security Considerations . . . . .	31
10.1. Layered Security . . . . .	31
10.2. Trust Aspects . . . . .	31
11. IANA Considerations . . . . .	32
12. References . . . . .	33
12.1. Normative References . . . . .	33
12.2. Informative References . . . . .	33
Authors' Addresses . . . . .	34

## 1. Introduction

This document describes a RELOAD Usage for distributed conference control (DisCo) in a tightly coupled model with SIP [RFC3261]. The Usage provides self-organizing and scalable signaling that allows RELOAD peers and plain SIP user agents to participate in a managed P2P conference. DisCo defines the following functions:

- o A protocol scheme for distributed conference control
- o RELOAD Usage and definition of conferencing Kind
- o Mechanisms for conference synchronization and call delegation
- o Mechanisms for proximity-aware routing for conference participants
- o XML extension for the event package for conference state
- o A graduated trust delegation system

In this document, the term distributed conferencing refers to a multiparty conversation in a tightly coupled model in which the point of control (i.e., the focus) is identified by unique URI, but the focus service is located at many independent entities. Multiple SIP [RFC3261] user agents uniformly control and manage a multiparty session. This document defines a new Usage for RELOAD including an additional Kind code point with a corresponding data structure that complies the demands for distributed conferences. The data structure stores the mapping of a single conference to multiple conference controllers and thereby separates the conference URI from focus instantiations.

Delay and jitter are critical issues in multimedia communications. The proposed conferencing scheme supports mechanisms to build an optimized interconnecting graph between conference participants and their responsible conference controllers. Conference members will be enabled to select the closest focus with respect to delay or jitter.

DisCo extends conference control mechanisms to provide a consistent and reliable conferencing environment. Controlling peers maintain a consistent view of the entire conference state. The multiparty system can be re-structured based on call delegation operations.

To provide secure mechanisms, which allows users to join or even control a distributed conference, this document describes a graduated trust delegation system. The proposed system guides implementors how to maintain privacy and trust to other peers in a distributed multiparty system.

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The terminology and definitions from der RELOAD base [I-D.ietf-p2psip-base], the peer-to-peer SIP concepts draft [I-D.ietf-p2psip-concepts] and the terminology formed by the framework for conferencing with SIP [RFC4353]. Additionally the following terms are used:

**Coordinate Value:** An opaque string that describes a host's relative position in the network topology.

**Focus peer:** A RELOAD peer that provides SIP conferencing functions and implements the Usage for distributed conferencing. It can be 'active' if is already in signaling relations to conference participants. Otherwise it is 'potential' if it is only registered in a distributed conference data structure but not maintaining signaling relations yet.

### 3. Overview of DisCo

#### 3.1. Reference Scenario

The reference scenario for the Distributed Conference Control (DisCo) is shown in Figure 1. Peers are connected via a RELOAD [I-D.ietf-p2psip-base] instance, in which peers A and B are managing a single multiparty conference. The conference is identified by a unique conference URI, but located at peers A and B fulfilling the role of focus. The mapping of the conference URI to one or more responsible focus peers is stored in a new RELOAD Resource for distributed conferencing within a data structure denoted as DisCo-Registration. The owner O of the distributed conference resource holds this data.

The focus peers A and B maintain SIP signaling relations to conference participants, which may have different conference protocol capabilities. In this example, peer A is the multiparty manager for the RELOAD peer C and the plain SIP user agent E whereas focus peer B serves for RELOAD peer D and the RELOAD client F.

RELOAD peers and clients obtain the contact information for the conference from the owner O. In contrast, the user agent E receives the conference URI not by RELOAD mechanisms, but resolves the ID and joins the conference by plain SIP negotiation.

Focus peers establish a SIP signaling relation among each other used for notification messages that synchronize the conference focus peers' knowledge about the entire conference state. Additionally, focus peers can transfer calls to each other by a call delegation mechanism.

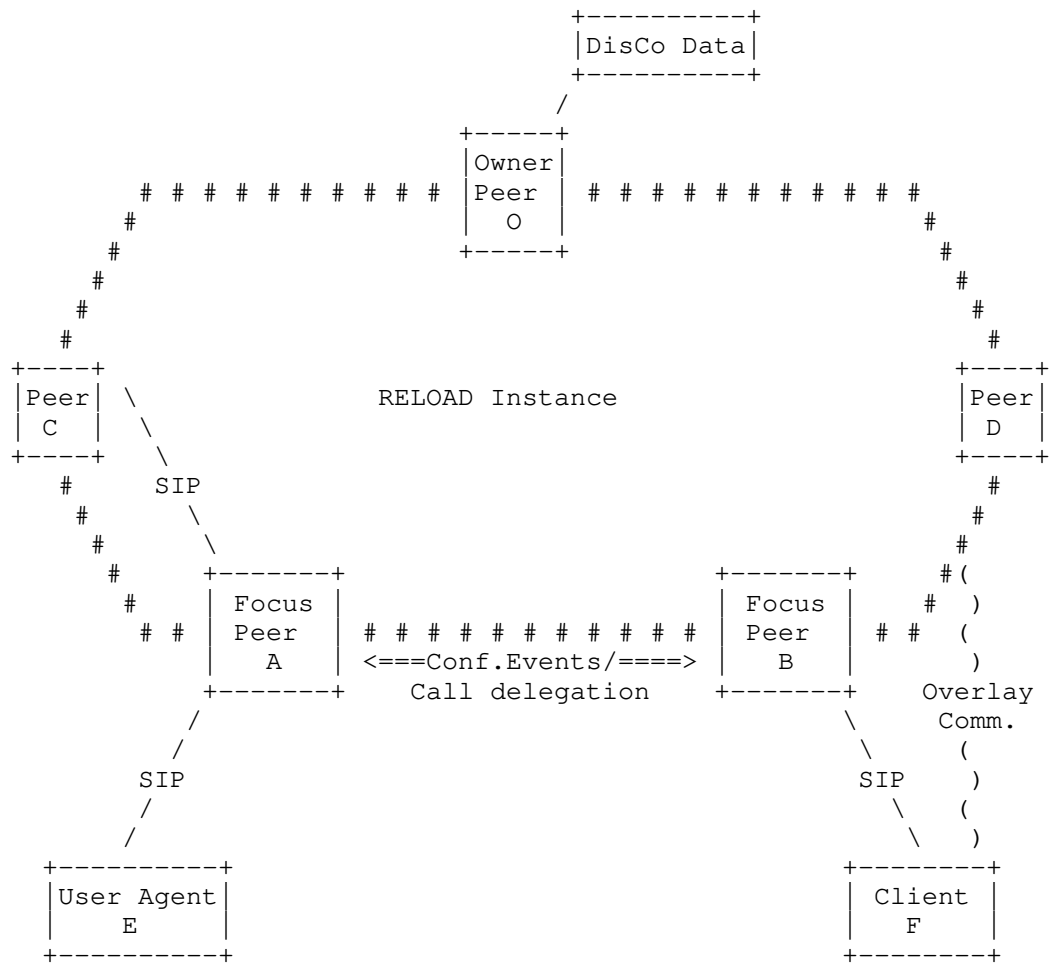


Figure 1: Reference Scenario: Focus peers A,B maintain a distributed conference

### 3.2. Initiating a Distributed Conference

To create a conference the initiating user agent announces itself as a focus for the conference. It stores its own contact information (Address-of-Record or Destination List) in the RELOAD overlay under DisCo-Registration Kind (cf., Figure Figure 2). The hashed conference URI is used as the Resource-ID. This data structure will later contain the contact IDs of all potential focus peers including optionally topological descriptors.

### 3.3. Joining a Conference

A RELOAD-aware node (cf., Bob in Figure Figure 2) intending to join an existing conference retrieves the list of potential focus peers stored in the DisCo-Registration under the conference's Resource-ID. To join the conference it selects any of the focus peers (e.g., Alice) and establishes a connection using AppAttach. This transport is then used to send an INVITE to the conference applying the chosen focus as the contact. The selection of the focus peer to contact can optionally be based on proximity information if available.

A node that is not aware of RELOAD uses common SIP signaling to retrieve the conference URI.

A conference member proposes as a focus for subsequent participants by storing a mapping of the conference URI to his Address-of-Record or Destination List in the RELOAD overlay using the conference Resource-ID. This decision should incorporate bandwidth, power, and other constraints, but details are beyond the scope of this document.

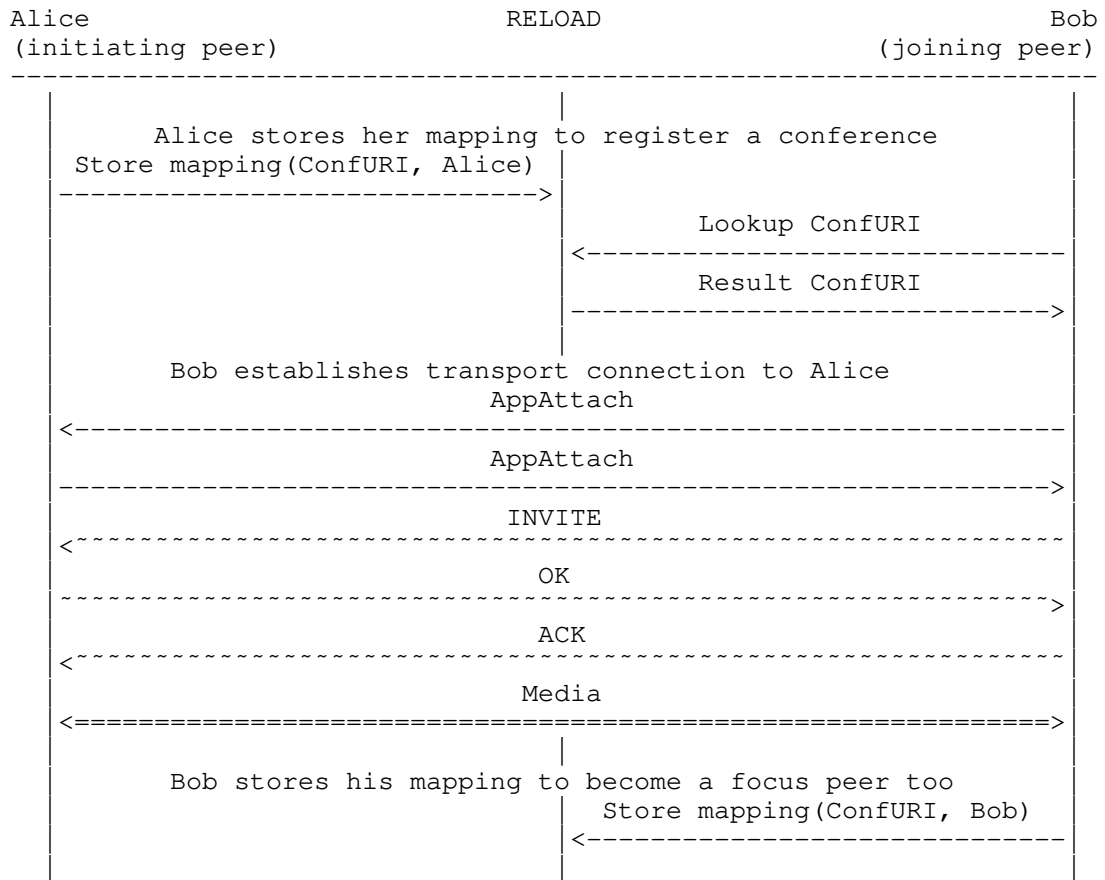


Figure 2: DisCo Usage generic Call Flow

### 3.4. Conference State Synchronization

Each focus of a conference maintains signaling connections to its related participants independently from other conference controllers. This distributed conference design effects that the entire SIP conference state is jointly held by all conference focus peers. In DisCo, state synchronization is based on SIP specific event notifications [RFC3265].

Each focus peer can complete its view of the entire conference state among the focus peers by subscribing all other focus peers for an XML event package for distributed conferences. This is defined in this document and based on the event package for conference state [RFC4575]. Receivers of event notifications update their local conference state document to regain a valid view of current



conference state.

The event notification package for distributed conferences enables focus peers to synchronize the entire conference state. It is designed as an extension to the XML event package for conference state, which provides signaling and media parameters for each peer participating in the multiparty session. The extension defines additional XML elements and complex types (see Section 7 for more details), which allow views of the responsibilities of any focus peer in the conference. By providing these views each focus peer is enabled to perform additional load balancing operations and enhances the robustness against departures of focus peers.

### 3.5. Call delegation

The call delegation (see Section 5.2.) is a feature used to transfer an incoming participation request to another focus peer. It can be applied to prevent an overloading of focus peers reaching its limit of serving new clients. Call delegation is realized through SIP REFER requests, which carry signaling and session description information of the callee to be transferred. A focus peer can decide to refer an incoming call to a less loaded remote focus. This feature is achieved transparently to the transferred user agent by using a source routing mechanism at SIP dialog establishment. Descriptions of overload detection are beyond the scope of this document.

### 3.6. Resilience

A focus peer can decide to leave the conference or may ungracefully fail. In a traditional conferencing scenario, a loss of the conference controller or the media distributor would cause a complete fail of the multiparty conversation. Distributed conferencing uses the redundancy by multiple focus peers to reconfigure a running multiparty. Participants that lost their entry point to the conference re-invite itself via the remaining focus peers or will be re-invited by the controllers. This option is based on the conference state and call delegation functions.

### 3.7. Topology Awareness

DisCo supports landmarking approaches based on an extension for the RELOAD XML configuration document (see Section 8) to construct topology-aware connections between focus and peers. Each peer intending to create or participate in a distributed conference SHOULD determine a topological descriptor that describes its relative position in the n-dimensional Cartesian space. Focus peers store these coordinate values as additional data field in the DisCo-

Registration data structure. This enables peers joining the conference to select the closest focus with respect to its coordinate values.

## 4. RELOAD Usage for Distributed Conference Control

### 4.1. Kind Data Structure

Each DisCo-Registration data structure stores the mappings for one conference to many focus peers and for each focus peer the related coordinates value. The data structure uses the RELOAD dictionary type whereas the DictionaryKey value is the Node-ID of the focus peer behind the dictionary entry. This allows a focus peer to update its mappings. The DisCo data structure of type DisCoRegistration is shown as follows:

```
enum {
    sip_focus_uri (1),
    sip_focus_node_id (2), (255)
} DisCoRegistrationType;

struct {
    opaque coordinate<0..2^16-1>

    select (DisCoRegistrationType.type) {
        case sip_focus_uri: opaque uri<0..2^16-1>

        case sip_focus_node_id: Destination destination_list<0..2^16-1>

        /* This type can be extended */
    }

} DisCoRegistrationData;

struct {
    DisCoRegistrationType type;
    uint16 length;
    DisCoRegistrationData data;
} DisCoRegistration;
```

The content of the DisCoRegistrationData structure are as follows:

- type  
type of the registration
  - length  
the length of the registration PDU
  - data  
the conference registration data
- o If the DisCoRegistration is set to "sip\_focus\_uri", then it contains an Address-of-Record (AOR) as an opaque string and opaque "coordinates" string, that describes the relative network position. See more in section 4.4.

- o If registration type is set to "sip\_focus\_node" then it contains the Destination list for the peer and an opaque string "coordinates" describing the focus' relative network position.

The structure is designed for enabling a peer to contact a focus of the conference that is the nearest to itself. A joining peer **MUST** select the focus peer, which coordinate value matches at most (see section Section 4.4) to its own. In this manner it reduces the problem of triangle inequality as without this feature a joining peer could choose an inadequate remote conference controller causing large signaling and may streaming delays.

#### 4.2. Determining Coordinates

Each RELOAD peer within the context of a distributed conference **SHOULD** be aware of it's relative position in the network topology. Those position information can support a topology-aware conference construction avoiding long signaling and media delays. Providing this the Usage for distributed conference foresees the coordinates value within the DisCo-Registration data structure that allows focus peers to store a topological descriptor. It is a generic field that describes a peer's relative position in the network as an N value long position vector in the N-dimensional Cartesian space. Focus peers store this coordinate value together with their announcement as conference focus. Joining peers likewise **SHOULD** determine their coordinates value and then select a focus peer whose relative position matches at most (see section Section 4.4).

Many algorithms determine topology information by measuring Round-Trip Times (RTT) towards a set on hosts serving as so called landmarks. To support such algorithms this document describes an extension to the RELOAD XML configuration document that allows to configure the set of Landmark hosts that peer must use for position estimation (see section Section 8). Once a focus peer has registered its mapping in the DisCo data structure, it also stores the according coordinates in the same mapping. These <Node-ID,coordinates> vectors are used by peers at conference join to select the focus peer that is relatively closest to itself.

Because topology-awareness can be obtained by many differnt approaches a concrete algorithms is out of scope of this document.

#### 4.3. Conference Creation

Before a peer registers to a new distributed conference, it is **RECOMMENDED** to ensure the initiating peer has a most up to date copy of the configuration document. In this way, the conference creator assures that all joining peers will equally determine their

coordinates value if such a algorithm is used. The first peer that creates a distributed conference registers it in the RELOAD overlay following the steps as described in Figure 3:

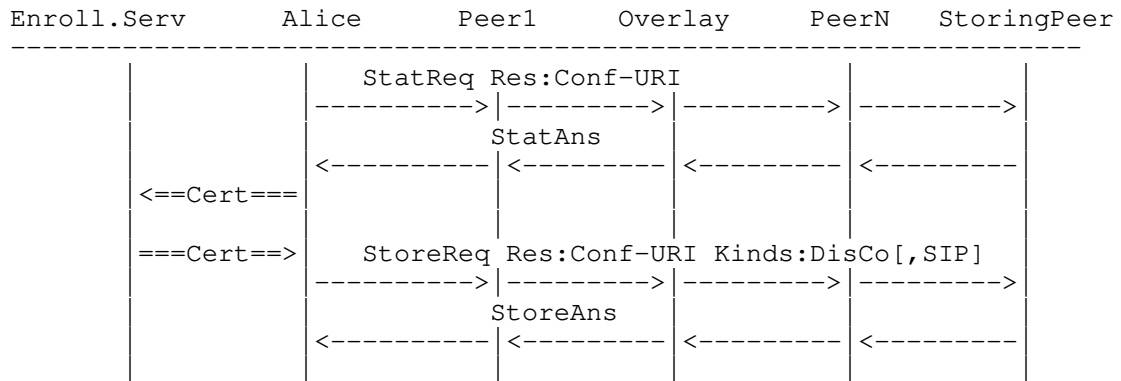


Figure 3: Creation of a Distributed Conference

1. The peer MUST determine its own coordinate value (if used).
2. The peer MUST probe whether the desired conference URI is available. It therefore generates the Resource-ID of the conference URI with the overlay hash function and sends a RELOAD StatReq towards this address. By the corresponding StatAns response the peer knows whether the desired URI is occupied by another a DisCo Kind or even a SIP-Registration Kind [I-D.ietf-p2psip-sip]. If it is, the user MUST choose another URI and repeat the availability checks. If no other DisCo or SIP-Registration Kind are stored at this Resource-ID it proceeds the registration.
3. Storing a conference registration is like to register a new virtual user that has the conference URI as its Address-of-Record. Therefore, the conference initiator MUST request the enrollment server for a new overlay certificate that contains the conference URI as user name. A sample certificate is shown below:

```
User name: conference@dht.example.com
Node-ID: 013456789abcdef
Serial: 0815
```

4. The peer finally registers the DisCo data structure signed with the above certificate by a Store request towards the storing peer (the owner of the address space for the Resource-ID of the conference URI).

The additional certificate is needed for 2 major purposes:

- o It separates the conference creator from the multiparty instance.
- o It ensures the conference initiator's privacy. Because the DisCo data structure will be accessed by many peers using the same conference certificate. If they were using the conference creators' certificate, they were permitted to write non-shared Resources of the creator.

The conference creator MAY registers the conference URI as SIP-Registration Kind as well. In this case, it also MUST sign the Store request with the private key that matches to the certificate obtained for the conference URI. This is necessary because in the case of the departure of the conference creator, the other focus peers are permitted to redirect the mapping to another focus peer still serving the conference. The SIP-Registration SHOULD be sent in the same StoreReq as the DisCo registration.

The creator of a distributed conference MUST select on the access models as described in section Section 10.1 to define the desired privacy level of the multiparty conference.

TODO: a description how a new certificate is generated in the RELOAD instance without enrollment server

#### 4.4. Proximity-aware Conference Participation

A RELOAD peer intending to join a distributed conference follows the steps showed in Figure 5 :

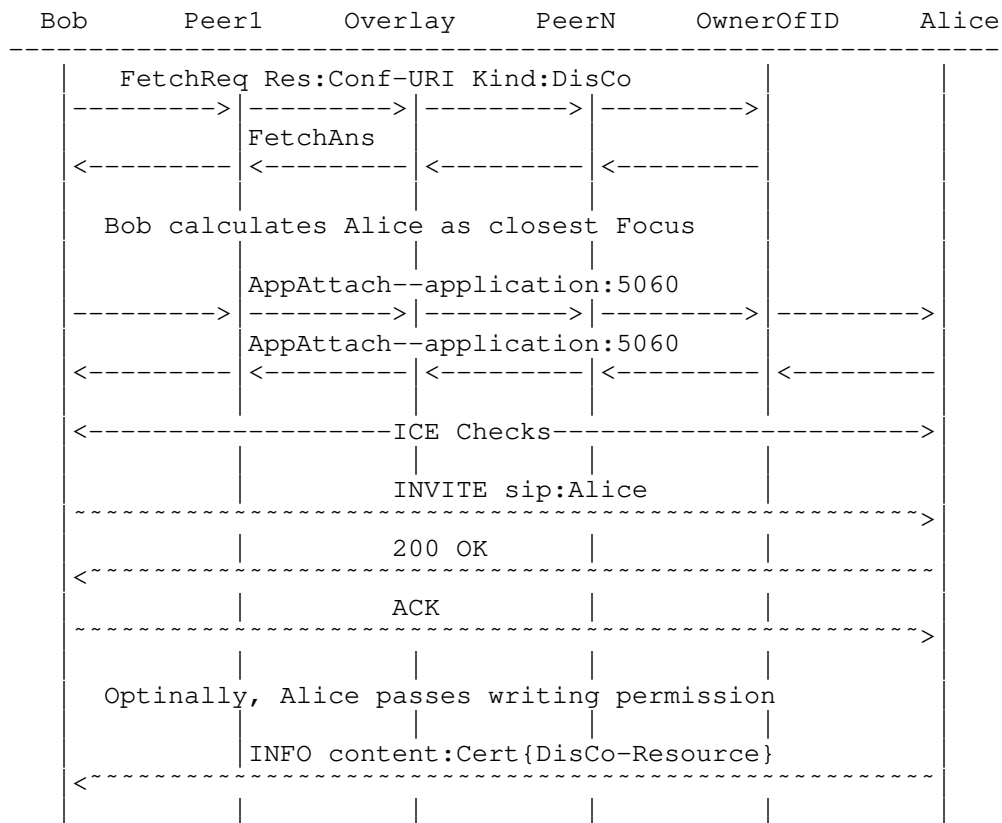


Figure 5: Participation of a Distributed Conference

1. The joining peer MUST determine its own coordinate value (if used).
2. The joining peer sends a FetchReq message for the DisCo Kind to the Resource-ID that corresponds to the hash over the conference URI using the overlays hash-function. The FetchReq SHOULD NOT include any specific dictionary keys thus it will receive all potential -and active focus peers of the conference.
3. Once the joining peer received the Fetch results, it calculates which of the focus peers is the relatively closest to itself by making the following calculation for each dictionary entry:
  - \* For each coordinate entry, calculate the each difference  $D_i = F_i - P_i$ , with  $F_i$  is the coordinates vector of the Focus peer and  $P_i$  the coordinates vector of the joining peer.

- \* For each  $D_i$ , calculate the scalar product of  $D_i$
- 4. The focus with the smallest scalar product SHOULD be chosen for establishing a SIP signaling relation.

Depending on which DisCo-Registration type the selected focus has stored its mapping, the joining Peer has the following 2 possibilities:

1. If the `DisCoRegistrationType` is `sip_focus_node_id`, the joining peer uses RELOADs AppAttach request to establish a direct transport connection to the selected focus peer. The application field of the request MUST be set to 5060 indicating for SIP. This transport connection SHOULD be used to form an ordinary SIP dialog. Further media session establishment is achieved by usual SIP mechanisms.
2. If the `DisCoRegistrationType` is `sip_focus_uri`, the joining peer MUST use the SIP-Registration [I-D.ietf-p2psip-sip] Usage to resolve the URI and form connectivity to the selected focus.

Note that in the second case a focus peer can have multiple locations for its SIP-registration. Therefore a focus MUST assure that its coordinate value corresponds to its current mapping AoR to location.

Regardless of how the focus peer has registered its mapping in the overlay a joining peer MUST add its coordinate value base64 encoded as URI-parameter in the contact-header field of the SIP INVITE request. An example contact URI is `"sip:alice@example.com;coord=YWxpY2VAZXhhbXBsZS5jb20="`. The additional parameter is used by the requested focus peer as it is not capable of serving additional conference participants. It then it MUST delegate the call (see section Section 5.2) to the focus peer whose coordinate value matches next best to the coordinates of the joining peer. The focus peer therefore uses the same calculation as described in the joining process.

After the final SIP ACK request completes the signaling relation, a conference focus MAY pass the writing permission to the new participant. It therefore sends a SIP INFO request carrying the certificate for the DisCo Resource. The decision whether to pass writing permission depends on the selected security model for the distributed conference as described in section Section 10.1.

#### 4.5. Advertising Focus Ability

All participants of a distributed conference can become a focus peer for their multiparty. The decision can depend on the capacities of



the joining peer like sufficient processing power (CPU, Memory) for the desired media type and quality of the network connectivity. Additionally, a peer intending to become focus of a conference SHOULD NOT be located behind NAT or its IP SHOULD NOT belong to the private address range. The information whether a participant is behind NAT can be obtained by ICE connectivity checks during the conference joining process.

If a participant is a candidate to become a focus of the conference it stores its mapping (Destination List or AoR) and coordinate value into the DisCo data structure. Because the DisCo Kind uses the USER-MATCH access control policy, the shared certificate passed by the participant's focus peer is sufficient to permit this peer to write the DisCo Resource. By storing the mapping into the data structure a participant becomes a potential focus.

TODO: What to do if the set of Landmark hosts changes during conference?

#### 4.6. Resilience in a Distributed Conference

The decentralized character of distributed conferences provide abilities to prevent the breakdown of the entire multiparty session in the case that a focus peer disappears. Two possibilities of a focus departure must be distinguished:

Friendly leave: A user whose peer is acting as conference focus decides to quit coconference participation.

Unexpected leave: Any case in which a peer serving as conference focus fails.

In the friendly case the leaving peer (lp) MUST accomplish the following procedure:

- o Lp deletes its mapping in the DisCo data structure by storing the "nonexisting" value as described in the RELOAD base document [I-D.ietf-p2psip-base].
- o Lp searches the conference state XML document (see section Section 7) for 'active' focus peers that have free capacities to serve further participants. Additionally, it fetches the latest DisCo data structure for this conference to obtain all 'potential' focus peers.
- o The lp then calculates for all its related participants the closest focus peer using the algorithm described in Section 4.2.

- o Based on the results from the previews step lp transfers all it's participants to their ascertained focus peers using the call delegation described in Section 5.2

If an unexpected leave is detected by a participant (e.g. missing signaling and/or media packets) it MUST repeat the joining procedure as described in Section 4.4.

Assuming unfavorably circumstances it can happen that the available capacities over all potential and active focus peers are insufficient to reassemble all lost participants. In this case it RECOMMENDED to reassemble as many participants as possible in a first come first serve algorithm and to fail the rest.

## 5. Focus Call Control Operations

This section describes SIP call flows for third party call control for distributed conferences. Those operation comprise the call delegation and state synchronization mechanisms.

### 5.1. Becoming an active Focus

A conference participant that stored its mapping to the distributed DisCo data structure serves as potential focus for further participation requests by other peers. On incoming participation request a potential focus becomes an active focus and is then responsible to grant the joining peer access to the conference. Two different scenarios for participation requests must be distinguished:

- o A joining peer requests the potential focus
- o An already active focus peer transfers a participation request to the potential focus

The second case will be discussed as part of the call delegation in Section 5.2.

For the case that a RELOAD peer directly requests a potential focus for participation the call flow in Figure 6 describes the necessary procedure. The joining peer (JP) has already established a transport connection and sends a SIP INVITE request (1) to the contact address (IP) to its selected potential focus (PF). Note that JP is thereby unaware that PF does not serve any other participants yet. PF is participating the conference through its own active focus (AF) and is aware of the offered media types for the multiparty session. This PF offers the available media parameter to JP in (2). After finalizing the signaling in (3) and establishment of the media streams the PF in charge to synchronize the distributed conference state. As the first step of the pairwise subscription (4) PF MUST send a SIP SUBSCRIBE [RFC3265] request to the AF that is the focus peer responsible for signaling with PF. It subscribes for the conference for the event package for conference state[RFC4575] with 'multi-focus' extension Section 7. This document therefore defines a new content type "application/distributed-conference-info+xml" for a MIME entity that contains conference state information for distributed conferences. After confirming the subscription (5) AF informs PF about the entire conference state by sending a 'full' XML document. It includes the list of all participants, active focus peers and used media types. In the second step for in the subscription procedure (8) AF MUST subscribe PF for distributed conference. After confirmation in (9), PF MUST inform AF about the arrival of JP and also MUST advertise its own capacities. PF therefore sends a NOTIFY request containing a

'partial' conference state XML document that describes PF's local state (e.g. capabilities, responsibilities for JP). This step finalizes the promotion of PF to an active focus peer.

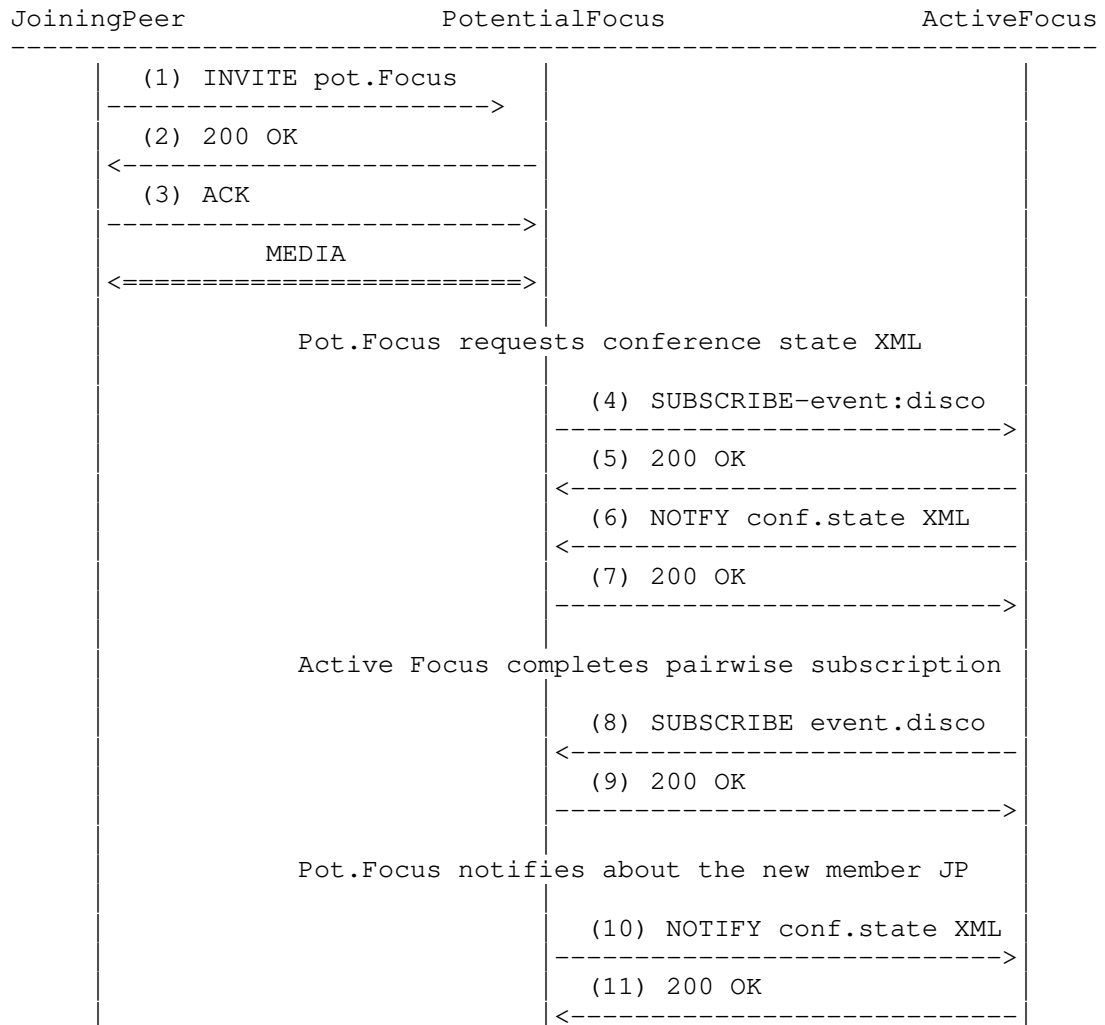


Figure 6: Pairwise subscription for conference state synchronization

Depending on whether AF has more subscriptions for the distributed conference event package it will synchronize all other focus peers sending notifications containing partial conference state XML document received from PF.

Since PF is aware of the entire conference and MAY establishes more

subscriptions to other active focus peers. This can reduce signaling delays and could serve as guideline for new routes for the media streams. A specification of how those new subscription should be done is a TODO in this document.

## 5.2. Delegating Calls

The call delegation feature described in this document provides focus peers the possibility to transfer an incoming participation request to another focus peer. A focus peer SHOULD delegate incoming participation requests if the number of participants it currently maintains is equal to the 'max\_participants' value the focus advertised in the distributed conference XML document.

A sample scenario for call delegation is shown in Figure 7. A joining peer (JP) requests the active focus (AF) for conference participation (1). AF has no more capacity to serve JP as focus peer and has to transfer the call. AF firstly temporally accepts the call (2-3) and then selects an adequate focus peer for call delegation based on JP's coordinate value (sent base64 encoded as URI parameter in (1) see section Section 4.4). AF fetches the latest DisCo data structure (not shown in Figure 7) to obtain all available potential and active focus peers. As shown in the example, AF determines the potential focus (PF) as best candidate to become JP's focus and transfers the participation request to PF sending a SIP REFER request (4). The REFER request MUST contain the session identifier from JP as payload in the request body and the call-ID of (1) as parameter in the URI of the refer-to header field, e.g., 'Refer-To: <sip:bob@dht.example;call-id=1234>'

Triggered by the REFER request PF is in charge to become JP's focus peer and to enter the conference state synchronization process. Because the call delegation operation should not interrupt JP's participation request PF MUST use the signaling and session information of (4). PF sends a re-INVITE request to JP that appears as it were originated by AF with an additional Record-Route header field set to PF's contact address. By using this technique a distributed conference appears as one single entity. The additional Record-Route header thereby ensures that further SIP signaling will be routed to PF. After the signaling process the negotiated media session can be established.

PF then is in charge to enter the conference state synchronization mechanism by pairwise subscribing PF->AF, AF-->PF for the event package for conference with multi-focus extension (9-14) as described Section 5.1. Note that PF could subscribe another active focus peer than AF since this is not necessarily the conference controller responsible for PF.

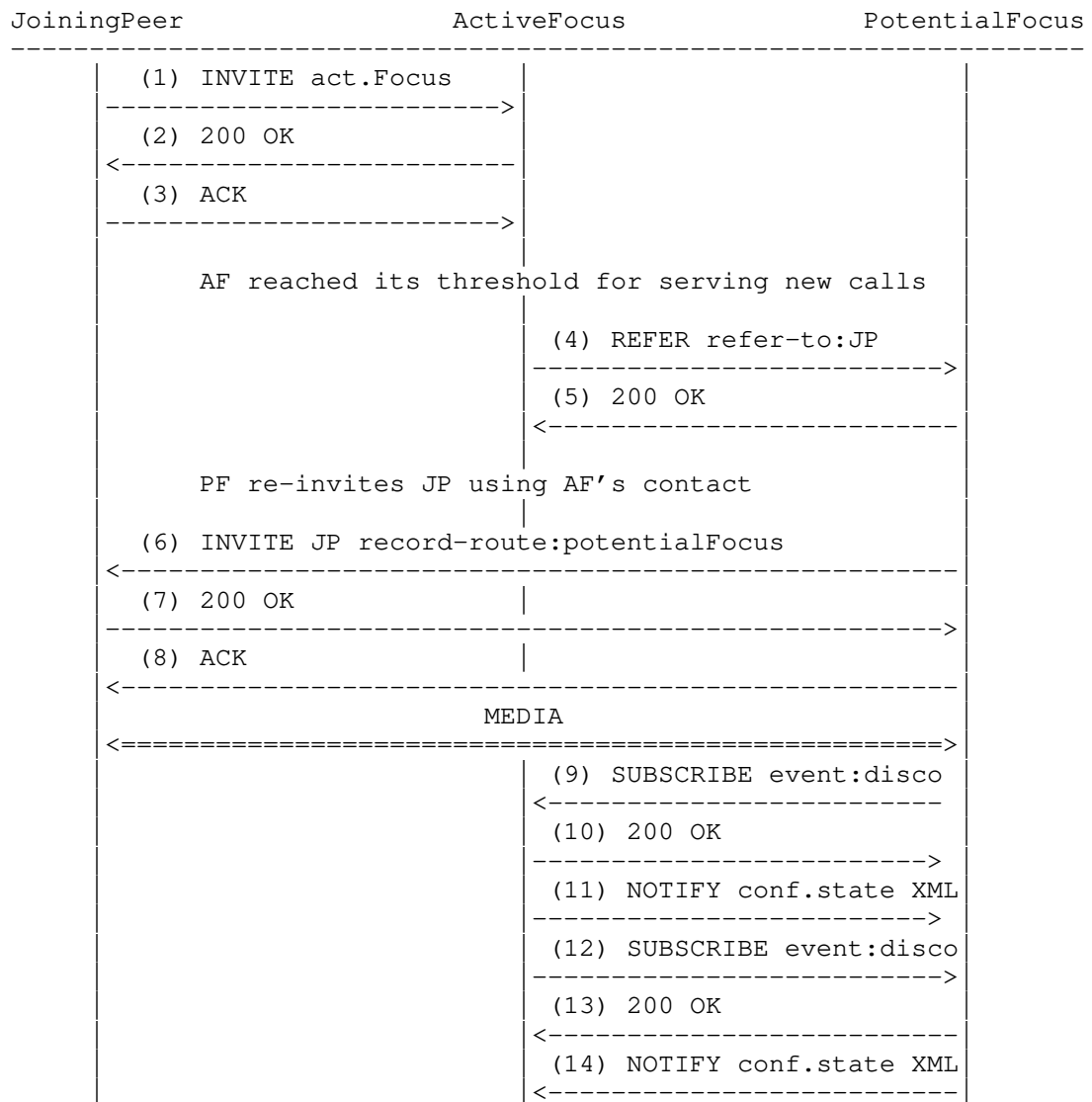


Figure 7: Call delegation to potential focus peer

### 5.3. Synchronizing the Conference State

The entire state of the distributed conference changes partly on every event that happens at an active focus peer. Most commonly those events refer to joins or lefts of conference participants. In order to maintain a coherent conference state all focus peers MUST send NOTIFY messages [RFC3265] to all subscribers (other focus peers)

to synchronize the conference state. The payload of the notifications MUST only contain a partial state information about the changes in the state from the previous conference state.

If the connection graph build by the pairwise subscriptions between the focus peers is not structured in a full mesh topology, state notifications MUST be forwarded by intermediate focus peers. The extension for the event package described in Section 7 therefore provides an XML element that allows every focus to reconstruct the connection graph among the focus peers.

If a state notification is received multiple times a focus peer MUST NOT forward the duplicate state information for prevent loops.

## 6. DISCO Kind Definition

This section formally defines the DisCo kind.

Name

DISCO-REGISTRATION

Kind IDs

The Resource name DISCO-REGISTRATION Kind-ID is the AOR of the conference. The data stored is the DisCoRegistrationData, that contains a coordinates value describing a peers relative network position acting as focus for the conference. Additionally it contains either the peers URI or a Destination list.

Data Model

The data model for the DISCO-REGISTRATION Kind-ID is dictionary.

The dictionary key is

the Node-ID of the peer action as focus.

Access Control

USER-MATCH

The data stored for the Kind-ID DISCO-REGISTRATION is of type DisCoRegistration. It contains a "coordinates" value, that describes the peers relative network position and XOR one of the two following data:

sip\_focus\_uri

the URI of the peer action as focus

sip\_focus\_node\_id

the Destination list of the peer acting as focus



## 7. Conference State Event Package Extension

This section presents the XML extension for the event package for conference state that enables a focus peer to have a view on the responsibilities of each other focus peer. The additional information by extending the XML schema defined RFC4575 [RFC4575] is can be used in the case of a focus departure and call delegations. The new <focus-states> element as shown in Figure 8 is placed as child-of the root-element <conference-info>. It's child element <focus> represents every conference participant that is in the role of an active focus peer to the conference.

```
conference-info
|
|-- conference-description
|
|-- host-info
|
|-- users
|
|-- focus-states
|   |
|   |-- focus
|
..
|-- ..
```

### 7.1. The <focus-states> and <focus> elements

The <focus-states> element serves as container of the <focus> sub-elements, each describing the responsibilities of a conference participant acting as focus peer.

The <focus> uses the following attributes:

**entity:** This attribute contains the AoR of the focus peer that is declared in the user name field in the RELOAD certificate. This AoR MUST correspond to the entity attribute defined in the focus peer's <user> element in the base conference event package. A user that wishes to lookup a focus peer's signaling information can retrieve it by looking at the corresponding <user> element with the same AoR in the entity attribute.

**state:** This attribute indicates whether the transmitted conference state for this focus peer is 'full', 'partial' or 'deleted' and have to be interpreted as defined in [RFC4575].

The <focus> element uses the complex focus-type that contains the

following child-elements:

`<focus-capacity>` : This element describes a focus peer's maximal number of participants it can serve respectively the maximal number media connections to other focus peers it can handle.

`<participant>` : Each participant element describes a conference member that has this focus peer as it's conference controller. It uses the 'entity' attribute that contains the AoR that this RELOAD peer uses as user name in the overlay certificate. It corresponds to the AoR in the `<user>` element in the base conference event XML document. Additionally, the `<participants>` element uses the 'state' attribute to provide the partial notification mechanism as defined in [RFC4575].

`<graph>` : Each `<graph>` element describes a conference state synchronization relation this focus peer maintains. By reference to this element each conference controller has a view of the entire synchronization topology over the focus peers. It uses the 'state' attribute as well.

## 7.2. XML Schema

The schema for XML extension is:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  targetNamespace="http://www.example.org/inet-ci-multifocus-ext"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.example.org/inet-ci-multifocus-ext"
  elementFormDefault="qualified" >
  <!--
    FOCUS STATES ELEMENT
  -->
  <xs:element name="focus-states" type="focus-states-type"/>
  <xs:complexType name="focus-states-type">
    <xs:sequence>
      <!--
        FOCUS ELEMENT
      -->
      <xs:element name="focus" type="focus-type"
        maxOccurs="unbounded" minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax"/>
    </xs:sequence>
    <xs:attribute name="state" type="state-type"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <!--
```

```

    FOCUS TYPE
-->
<xs:complexType name="focus-type">
  <xs:sequence>
    <xs:element name="focus-capacity"
      type="focus-capacity-type" maxOccurs="1" minOccurs="0"/>
    <xs:element name="participant" type="participant-type"
      maxOccurs="unbounded" minOccurs="0"/>
    <xs:element name="graph" type="graph-type"
      maxOccurs="unbounded" minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax"/>
  </xs:sequence>
  <xs:attribute name="entity" type="xs:anyURI"/>
  <xs:attribute name="state" type="state-type"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
    FOCUS-CAPACITY TYPE
-->
<xs:complexType name="focus-capacity-type">
  <xs:sequence>
    <xs:element name="max-participants" type="xs:int"
      maxOccurs="1" minOccurs="0"/>
    <xs:element name="max-focus-references" type="xs:int"
      maxOccurs="1" minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax"/>
  </xs:sequence>
  <xs:attribute name="state" type="state-type"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
    PARTICIPANT TYPE
-->
<xs:complexType name="participant-type">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax"/>
  </xs:sequence>
  <xs:attribute name="entity" type="xs:anyURI"/>
  <xs:attribute name="state" type="state-type"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
    GRAPH TYPE
-->
<xs:complexType name="graph-type">
  <xs:sequence>
    <xs:element name="ref-to-focus" type="xs:anyURI"
      maxOccurs="1" minOccurs="0"/>

```

```
        <xs:any namespace="##other" processContents="lax"/>
      </xs:sequence>
      <xs:attribute name="state" type="state-type"/>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:schema>
```

## 8. Configuration Document Extension

This section defines an additional parameter for the <configuration> element that extends the RELOAD XML configuration document. The proposed <landmarks> element allows RELOAD provider to publish a set of accessible and reliable hosts that SHOULD be used if RELOAD peers use landmarking algorithms to determine relative position in the network topology.

### 8.1. The <landmark> and <Landmark-host> elements

The <landmarks> element serves as container of the <landmark-host> sub-elements each representing a single host that serves a landmark. The <landmark-host> uses the following attributes:

address: The IP address (IPv4 or IPv6) of the landmark host.

port: The port on which the landmark host responses for distance estimation.

More than one landmark hosts SHOULD be present in the configuration document.

### 8.2. Relax NG Grammar

The grammar for the Landmark configuration document extension is:

```
<!--  
  LANDMARKS ELEMENT  
-->  
parameter &#x26;= element landmarks {  
  attribute version { xsd:int }  
  <!--  
    LANDMARK-HOST ELEMENT  
  -->  
  element landmark-host {  
    attribute address { xsd:string },  
    attribute port { xsd:int }  
  }*  
}?
```

## 9. Example

TODO: Call flow examples for joining, delegating

## 10. Security Considerations

### 10.1. Layered Security

TODO: An ad hoc conference can be set up to a layered security model. Three models: open access, focus authenticate, closed access model.

### 10.2. Trust Aspects

TODO: Describing the privacy level for a conference instance; define whether a joining user is allowed to become a member or even focus of a conference.

## 11. IANA Considerations

TODO: register Kind-ID code point at the IANA



## 12. References

### 12.1. Normative References

- [I-D.ietf-p2psip-base]  
Jennings, C., Lowekamp, B., Rescorla, E., Baset, S., and H. Schulzrinne, "REsource LOcation And Discovery (RELOAD) Base Protocol", draft-ietf-p2psip-base-08 (work in progress), March 2010.
- [I-D.ietf-p2psip-sip]  
Jennings, C., Lowekamp, B., Rescorla, E., Baset, S., and H. Schulzrinne, "A SIP Usage for RELOAD", draft-ietf-p2psip-sip-04 (work in progress), March 2010.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [RFC3265] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.
- [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, August 2006.

### 12.2. Informative References

- [I-D.ietf-p2psip-concepts]  
Bryan, D., Matthews, P., Shim, E., Willis, D., and S. Dawkins, "Concepts and Terminology for Peer to Peer SIP", draft-ietf-p2psip-concepts-03 (work in progress), July 2008.
- [RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, February 2006.

## Authors' Addresses

Alexander Knauf  
HAW Hamburg  
Berliner Tor 7  
Hamburg  
Germany

Phone: +4940428758067  
Email: [alexander.knauf@haw-hamburg.de](mailto:alexander.knauf@haw-hamburg.de)  
URI: <http://inet.cpt.haw-hamburg.de/members/knauf>

Gabriel Hege  
HAW Hamburg  
Berliner Tor 7  
Hamburg D-20099  
Germany

Phone: +4940428758067  
Email: [hege@fhtw-berlin.de](mailto:hege@fhtw-berlin.de)  
URI: <http://inet.cpt.haw-hamburg.de/members/hege>

Thomas C. Schmidt  
HAW Hamburg  
Berliner Tor 7  
Hamburg 20099  
Germany

Email: [schmidt@informatik.haw-hamburg.de](mailto:schmidt@informatik.haw-hamburg.de)  
URI: <http://inet.cpt.haw-hamburg.de/members/schmidt>

Matthias Waehlich  
link-lab & FU Berlin  
Hoenower Str. 35  
Berlin D-10318  
Germany

Email: [mw@link-lab.net](mailto:mw@link-lab.net)  
URI: <http://www.inf.fu-berlin.de/~waehl>

