SIPPING   J. Rosenberg
Internet-Draft   Cisco Systems
Expires: May 28, 2005   H. Schulzrinne
                                                     Columbia University
                                                     O. Levin, Ed.
                                                     Microsoft Corporation
                                                     November 27, 2004

A Session Initiation Protocol (SIP) Event Package for Conference State
draft-ietf-sipping-conference-package-07

Status of this Memo

Copyright Notice

Abstract

   This document defines a conference event package for the Session
   Initiation Protocol (SIP) Events framework, along with a data format
   used in notifications for this package. The conference package

allows users to subscribe to a conference URI. Notifications are
sent about changes in the membership of this conference and
optionally about changes in the state of additional conference
components.

Table of Contents

1. **Introduction**

   The Session Initiation Protocol (SIP) [6] Events framework Events
   Framework [7] defines general mechanisms for subscribing to, and
   receiving notifications of, events within SIP networks. It
   introduces the notion of a package, which is a specific
   "instantiation" of the events framework for a well-defined set of
   events. Here, we define an event package for SIP conferences. This
   package provides the conference notification service as outlined in
   the SIP conferencing framework [16]. As described there,
   subscriptions to a conference URI are routed to the focus that is
   handling the conference. It acts as the notifier, and provides
   clients with updates on conference state.

   The information provided by this package is comprised of conference
   identifier(s), conference participants (optionally with their
   statuses and media description), conference sidebars, conference
   service URIs, etc.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED",
"SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY",
and "OPTIONAL" are to be interpreted as described in RFC 2119 [1] and
indicate requirement levels for compliant implementations.

3. **Conference Event Package**

   The conference event package allows a user to subscribe to a
   conference. In SIP, conferences are represented by URIs. These URIs
   route to a SIP user agent, called a focus, that is responsible for
   ensuring that all users in the conference can communicate with each
   other, as described in Conferencing Framework [16]. The focus has
   sufficient information about the state of the conference to inform
   subscribers about it.

   It is possible a participant in the conference may in fact be another
   focus. In order to provide a more complete participant list, the
   focus MAY subscribe to the conference package of the other focus to
   discover the participant list in the cascaded conference. This
   information can then be included in notifications by using of the
   <cascaded-focus> element as specified by this package.

   This section provides the details for defining a SIP Events package,
   as specified by RFC 3265 [7].

3.1 **Event Package Name**

   The name of this event package is "conference". This package name is
   carried in the Event and Allow-Events header, as defined in RFC 3265
   [7].

3.2 **SUBSCRIBE Bodies**

   A SUBSCRIBE for a conference package MAY contain a body. This body
   defines a filter to apply to the subscription. Filter documents are
   not specified in this document, and at the time of writing, are
   expected to be the subject of future standardization activity.

   A SUBSCRIBE for a conference package MAY be sent without a body.
   This implies the default subscription filtering policy. The default
   policy is:
   o Notifications are generated every time there is any change in the
      state of the conference.
   o Notifications do not normally contain full state; rather, they
      only indicate the state that has changed. The exception is a
      NOTIFY sent in response to a SUBSCRIBE. These NOTIFYs contain the
      full state of the information requested by the subscriber.

3.3 **Subscription Duration**

   The default expiration time for a subscription to a conference is one
   hour. Once the conference ends, all subscriptions to that particular
   conference are terminated, with a reason of "noresource" RFC 3265

[7].

3.4 NOTIFY Bodies

As described in RFC 3265 [7], the NOTIFY message will contain bodies
that describe the state of the subscribed resource. This body is in
a format listed in the Accept header field of the SUBSCRIBE, or a
package-specific default if the Accept header field was omitted from
the SUBSCRIBE.

In this event package, the body of the notification contains a
conference information document. This document describes the state
of a conference. All subscribers and notifiers MUST support the
"application/conference-info+xml" data format described in Section 5.
The subscribe request MAY contain an Accept header field. If no such
header field is present, it has a default value of
"application/conference-info+xml". If the header field is present,
it MUST include "application/conference-info+xml", and MAY include
any other types.

Of course, the notifications generated by the server MUST be in one
of the formats specified in the Accept header field in the SUBSCRIBE
request.

3.5 Notifier Processing of SUBSCRIBE Requests

The conference information contains very sensitive information.
Therefore, all subscriptions SHOULD be authenticated and then
authorized before approval. Authorization policy is at the
discretion of the administrator, as always. However, a few
recommendations can be made.

It is RECOMMENDED that all users in the conference be allowed to
subscribe to the conference.

3.6 Notifier Generation of NOTIFY Requests

Notifications MUST be generated for the conference state when a new
participant joins (i.e. gets "connected" to) or a participant leaves
(i.e. gets "disconnected" from) the conference.

Subject to a local focus policy, additional changes in participants'
status, changes in their media types, and other optional information
MAY be reported by the focus.

Changes in sidebar rosters SHOULD be reported by the focus to their
participants and MAY be reported to others, subject to local policy.

Changes in conference identifiers and service URIs SHOULD be reported
by the focus to the Conference package subscribers.

Changes in other conference state information MAY be reported by the
focus to the Conference package subscribers.

3.7 Subscriber Processing of NOTIFY Requests

The SIP Events framework expects packages to specify how a subscriber
processes NOTIFY requests in any package specific ways, and in
particular, how it uses the NOTIFY requests to construct a coherent
view of the state of the subscribed resource.

Typically, the NOTIFY for the conference package will only contain
information about those users whose state in the conference has
changed. To construct a coherent view of the total state of all
users, a subscriber to the conference package will need to combine
NOTIFYs received over time.

Notifications within this package can convey partial information;
that is, they can indicate information about a subset of the state
associated with the subscription. This means that an explicit
algorithm needs to be defined in order to construct coherent and
consistent state. The details of this mechanism are specific to the
particular document type. See Section 4.6 for information on
constructing coherent information from an
application/conference-info+xml document.

3.8 Handling of Forked Requests

By their nature, the conferences supported by this package are
centralized. Therefore, SUBSCRIBE requests for a conference should
not generally fork. Users of this package MUST NOT install more than
a single subscription as a result of a single SUBSCRIBE request.

3.9 Rate of Notifications

For reasons of congestion control, it is important that the rate of
notifications not become excessive. As a result, it is RECOMMENDED
that the server not generate notifications for a single subscriber at
a rate faster than once every 5 seconds.

3.10 State Agents

Conference state is ideally maintained in the element in which the
conference resides. Therefore, the elements that maintain the
conference are the ones best suited to handle subscriptions to it.
Therefore, the usage of state agents is NOT RECOMMENDED for this

package.

4. Conference Document

4.1 Format

Conference information is an XML document that MUST be well-formed
and SHOULD be valid. It MUST be based on Extensible Markup Language
(XML) 1.0 and MUST be encoded using UTF-8 [11].

4.2 Namespace

This specification makes use of XML namespaces for identifying
conference information documents and document fragments. The
namespace URI for elements defined by this specification is a URN
[2], using the namespace identifier 'ietf' defined by [4] and
extended by RFC 3688 [12]. This URN is:

   urn:ietf:params:xml:ns:conference-info

4.3 Versioning

The conference information is described by a hierarchal XML structure
with the root element <conference-info>. The root element is the
only element in the schema that carries meaningful version number for
all the elements in the document. The whole conference information
is associated with this version number.

The optional 'version' attribute MUST be included in the root
<conference-info> element.

4.4 State and Partial Notifications

All sub-elements in the <conference-info> hierarchal XML structure
can be classified in two groups: those that carry relatively small
amount of data and those that can potentially carry a lot of data.
During partial notifications, the light elements are updated as
atomic pieces of data. On the other hand, elements that can carry a
substantial amount of data have the general 'state' attribute
attached to them. That is in order to support partial notifications
for their content.

The 'state' attribute indicates whether the reported information
about the element is "full", "partial" or the element is "deleted"
from the conference state document. The default value of any 'state'
attribute is "full".

A 'state' attribute of a child element in the document MUST adhere to
its parent 'state'. It means that if the parent's 'state' is "full",
the state of its children MUST be "full". If the parent's 'state' is

"partial", the state of its children MAY be either "partial", "full",
or "deleted".

4.5 Element Keys

In the context of this specification, the element key is the set of
mandatory attributes or sub-elements of the element. The key value
MUST be unique for the element among its siblings of the same type.

In a partial notification event it must be possible to uniquely
identify each sub-element among others of the same type under a
common parent element. In order to achieve this property, all
sub-elements, with possible multiple appearances under a common
parent (which has the attribute 'state') have keys defined to them.

Below is the list of the elements with their keys as defined by this
specification:

o   Elements <conference-info>, <user>, and <endpoint> use as the key
    'entity'
o   Element <media> uses as the key 'id'
o   Sub-element <entry> of uris-type contained in elements
    <conf-uris>, <service-uris>, and <recording> uses as the key
    <uri>
o   Elements <available-media> and <active-media> of
    conference-medias-type use as the key <proto>
o   Elements <maximum-user-count> and <user-count> of count-type use
    as the key <role>
o   Element <role> of user-roles-type uses as the key <entry>
o   Sub-element <entry> of conference-type contained in element
    <sidebars-by-val> uses as the key 'entity'
o   Elements <associated-uris> and <sidebars-by-ref> of uris-type use
    as the key <uri>

4.6 Constructing Coherent State Procedure

A Conference package subscriber MUST initialize the 'version'
attribute from the <conference-info> element with the value in the
first document received.

The conference package subscriber locally maintains a local element
for each element in the schema and a table for each element with
key(s) in the schema and indexed by these key(s).

Each time a new NOTIFY is received, the value of the local version
number and the value of the 'version' attribute in the new received
document are compared. If the value in the document is less than the
local version, the document is discarded without processing. If the

value in the document is higher than the local version number, the
local version number is set to the value in the new document and the
document is processed. If the value in the received document is more
than one higher than the previous local version number and the
document contained a partial state, the subscriber SHOULD generate a
refresh request to trigger a full state notification.

Further processing of the conference information depends on the state
contained in the received conference document and indicated by the
value of the 'state' attribute in the <conference-info> element. If
it contains "full" state, the whole local content is flushed and
repopulated from the document. If it contains "deleted" state, it
means that the conference ceased to exist and the subscriber SHOULD
terminate the subscription by sending the SUBSCRIBE with Expires = 0.

If the document contains "partial" state, the document is used to
update the local content as described below.

Starting from outer elements in the received document,

1. If the parent element contains "full" state, the whole local
element content is flushed and repopulated from the document.

2. Otherwise, if the parent element contains "deleted" state, the
whole element MUST be removed from the local content.

3. Otherwise, if the parent element contains "partial" state:

3.1 For elements with keys, the subscriber compares the keys received
in the update with the keys in the local tables.

3.1.1 If a key does not exist in the local table, a row is added, and
its content is set to the element information from the update.

3.1.2 Otherwise, if a key of the same value does exist, for each
sub-element in the row the algorithm is applied from step 2.2.

3.2 For each atomic element received in the schema, the element is
replaced with the new information as a whole. Also, for each
non-atomic element received in the schema with either no 'state'
attribute included or the state attribute is set to "full", the
element is replaced with the new information as a whole.

3.3 For each non-atomic element with the state attribute set to
"partial", the algorithm is applied recursively starting from step 3.

5. Conference Data

   A conference information document begins with the root element tag
   <conference-info> of conference-type. Sections below describe the
   complex types composing the hierarchal conference-type. The full XML
   schema is defined in Section 6.

5.1 conference-type

   This type defines the following attributes:

   entity: This attribute contains the conference URI that identifies
      the conference being described in the document.

   state: This attribute indicates whether the document contains the
      whole conference information ("full"), only the information that
      has changed since the previous document ("partial"), or the
      conference ceased to exist ("deleted"). For more details see
      Section 4.
   version: This attribute allows the recipient of conference
      information documents to properly order them and it MUST be
      included when used in the root <conference-info> element.
      Versions start at 0 and increment by one for each new document
      sent to a subscriber. Versions are scoped within a subscription.
      Versions are represented using a 32 bit integer.

   The conference-type defines an extendable sequence of child elements.
   A "full" conference document MUST at least include the following
   sub-elements: <conference-description>, <conference-state>, and
   <users>.

   The child elements are described in details below:

5.1.1 conference-description of conference-description-type

   This element contains conference information that is derived from
   system conference policies, is set before the conference activation,
   and is rarely changed during the conference lifetime.

5.1.2 host-info of host-type

   This element contains information about the entity that hosts the
   conference. This information is set before the conference
   activation, and is rarely changed during the conference lifetime,
   unless the whole conference is moved to be hosted by another entity.

[5.1.3](#) conference-state of conference-state-type

   This element contains the dynamic information about the current state
   of the conference.

[5.1.4](#) users of users-type

   This element can contain an unbounded number of <user> sub-elements
   of user-type each containing the information about a participant in
   the conference.

[5.1.5](#) sidebars-by-ref of uris-type

   This element contains <entry> sub-elements of uri-type which provide
   pointers to sidebar information through sidebar URIs. The recipient
   of the information can then subscribe to sidebar information
   independently from the main conference package subscription.

[5.1.6](#) sidebar-by-val of conference-type

   This element provides sidebar information as a part of the main
   conference package information.

[5.2](#) conference-description-type

   This type defines the 'state' attribute which can contain the values
   "full", "partial", or "deleted".

   This type defines an extendable sequence of the following child
   elements:

[5.2.1](#) display-text of string type

   This element contains text description of the conference.

[5.2.2](#) subject of string type

   This element contains the subject of the conference.

[5.2.3](#) free-text of string type

   This element contains free form text about the conference.

[5.2.4](#) keywords of keywords-type

   This element contains a list of words that can be used by automatic
   search engines to better classify the conference.

5.2.5 conf-uris of uris-type

This element contains a set of <entry> sub-elements - each containing
the information about an additional conference URI that this
conference can be accessed by. The value of the URI is included in
the <uri> sub-element and its description MAY be included in the
<display-text> sub-element.

Examples of such URIs include h323: [15] and tel: [14] URIs.

5.2.6 service-uris of uris-type

This element contains a set of <entry> sub-elements - each containing
the URI to be used in order to access different services available
for the particular conference. The value of the URI is included in
the <uri> sub-element and its description MAY be included in the
<display-text> sub-element. The purpose of the URI SHOULD be
included in the <purpose> sub-element. The only currently defined
<purpose> value is "web-page" which indicates a web page that
contains additional information about the conference. Future
extensions to this schema may define new values and establish an IANA
registry for the new values.

5.2.7 maximum-user-count of user-count-type

This element is used to specify the maximum number of users permitted
in the conference. The number SHOULD be provided for all
participants in total by populating the <role> sub-element with value
"any". Additionally counters for users with certain roles in the
conference MAY be separately provided.

5.2.8 available-media of conference-medias-type

This element contains information about the media types available in
the conference. The <entry> sub-element MUST contain one of the
values registered for "proto" of SDP [3] and its later revision(s).

5.3 host-type

This type defines the 'state' attribute which can contain the values
"full", "partial", or "deleted".

This type defines an extendable sequence of the following child
elements:

5.3.1 display-text of string type

This element contains display text information about the entity

hosting the conference.

5.3.2 web-page of anyURI type

This element contains a web page URI about the user hosting the
conference.

5.3.3 uris of uris-type

The <entity> sub-element contains additional URIs pointing to the
conference host.

5.4 conference-state-type

This type defines the 'state' attribute which can contain the values
"full", "partial", or "deleted".

This type defines an extendable sequence of the following child
elements.

5.4.1 user-count of user-count-type

This element is used to specify the current number of users in the
conference. The number SHOULD be provided for all participants in
total by populating the <role> sub-element with value "any".
Additionally counters for users with certain roles in the conference
MAY be separately provided.

5.4.2 active of Boolean type

This element says whether the conference is currently active or not.
For example, a conference can be scheduled for a certain start time
and its conference URI reserved and published. Still, the conference
will not be "active" till its actual start time.

5.4.3 locked of Boolean type

This element contains information about whether the conference is
currently locked. In this context, "locked" means that the
conference roster can not be added to (although participants may
leave or be removed from the conference).

5.4.4 recording of uris-type

The <entry> sub-element contains URIs related to the recording of the
conference.

5.4.5 **active-media of conference-medias-type**

   This element contains information about the media types currently
   active in the conference, which is a subset of those listed in the
   <available-media> element.

5.5 **user-type**

   This type defines the following attributes:

   entity: This attribute contains the URI for the user in the
      conference. This is a logical identifier, which corresponds to
      the authenticated identity of the participant. The 'entity'
      attribute MUST be unique in the user element list because it is
      used as the key in partial notifications about users' state. An
      anonymous participant in a conference SHOULD be represented by an
      anonymous URI generated by the focus. For multiple anonymous
      participants, the focus must ensure that each anonymous URI is
      unique. The guidelines for generating anonymous URIs in RFC 3323
      [8] should be followed. For example,

   "Anonymous1" <sip:anonymous1@anonymous.invalid>

       could be used for a participant requesting privacy.

   state: This attribute indicates whether the document contains the
      whole conference information ("full"), only the information that
      has changed since the previous document ("partial"), or the
      conference ceased to exist ("deleted").

   This type defines an extendable sequence of the following child
   elements.

5.5.1 **display-text of string type**

   This element contains the display text for the user.

5.5.2 **associated-aors of anyURI type**

   This element contains associated URIs of the user. Usually this
   information will be manually provided by a system administrator
   showing the logical association between signaling entities otherwise
   independent.

5.5.3 **roles of user-roles-type**

   This element contains the roles of the user.

5.5.4 language of language type

This element contains the language preference of the user. This
information can be automatically learned via call signaling or be
manually set per participant.

5.5.5 cascaded-focus of anyURI type

This element contains a conference URI (different from the main
conference URI) for users that are connected to the main conference
as a result of focus cascading. In accordance with the SIP
conferencing framework [16], this package allows for representation
of peer-to-peer (i.e. "flat") focus cascading only. The actual
cascading graph can not be deduced from the information provided in
the package alone. Advanced applications can construct the graph by
subscribing to both this package and the Dialog Package [17] of the
cascaded foci and correlating the relevant information.

5.5.6 endpoint of endpoint-type

This element contains information about an endpoint of the user. The
element of the endpoint-type can have unbounded number of appearance
in the user-type for each endpoint of the user participating in the
conference. In a case when authentication is performed per endpoint
(rather than per user) in a system, a focus can be not aware of the
logical association among endpoints being used by the same user. In
this case the focus MAY present the endpoints as belonging to
separate users in the conference schema.

In a different case, due to privacy concerns for a user, the focus
may want to shield the information about multiple endpoints from the
recipients of the Conference document. To do so the focus MAY
aggregate the multiple endpoint information into a single endpoint
element under this user.

5.6 endpoint-type

This type defines the following attributes:

entity: The attribute contains the endpoint URI for the user in the
   conference. In SIP terms, this is the Contact URI or GRUU. The
   'entity' attribute MUST be unique in the endpoint element list
   because it is used as the key in partial notifications about
   users' endpoints. An endpoint belonging to an anonymous
   participant in a conference SHOULD be represented by an anonymous
   URI generated by the focus. For multiple anonymous endpoints, the
   focus must ensure that each anonymous URI is unique. The
   guidelines for generating anonymous URIs in RFC 3323 [8] should be

followed.

state: This attribute indicates whether the element contains the
    whole endpoint information ("full"), only the information that has
    changed since the previous document ("partial"), or the endpoint
    has been deleted from the conference ("deleted").

This type defines an extendable sequence of the following child
elements.

5.6.1 display-text of string type

This element contains the display text for the endpoint.

5.6.2 referred of execution-type

This element contains information about the user who's action
resulted in this endpoint being brought into the conference (e.g.
the SIP user identified by this URI sent a REFER to the focus). It
can contain the following sub-elements:

when: This element contains the date and time that the endpoint was
    referred to the conference.

reason: This element contains the reason the endpoint was referred to
    the conference.

by: This element contains the URI of the entity who caused the
    endpoint to be referred to the conference.

5.6.3 status of endpoint-status-type

This element contains the status of the endpoint, and can assume the
following values:

connected: The endpoint is a participant in the conference.
    Depending on the media policies, he/she can send and receive media
    to and from other participants.

disconnected: The endpoint is not a participant in the conference and
    no active dialog exists between the endpoint and the focus.

on-hold: Active SIP dialog exists between an endpoint and a focus,
    but endpoint is "on-hold" for this conference, i.e. neither
    he/she is "hearing" the conference mix, nor is his/her media being
    mixed in the conference. As an example, the endpoint has asked to
    join the conference using SIP, but his/her participation is
    pending based on moderator approval. In the meantime he/she is

hearing music-on-hold or some other kind of related content.

muted-via-focus: Active SIP dialog exists between an endpoint and a
focus and the endpoint can "listen" to the conference, but
endpoint's media is not being mixed into the conference. Note
that sometimes a subset of endpoint media streams can be muted by
focus (such as poor quality video) while others (such as voice or
IM) can still be active. In this case, it is RECOMMENDED that the
"aggregated" endpoint connectivity <status> reflects the status of
the mostly active media.

pending: Endpoint is not yet in the session, but it is anticipated
that he/she will join in the near future.

alerting: A PSTN ALERTING or SIP 180 Ringing was returned for the
outbound call, endpoint is being alerted.

dialing-in: Endpoint is dialing into the conference, not yet in the
roster (probably being authenticated).

dialing-out: Focus has dialed out to connect the endpoint to the
conference, but the endpoint is not yet in the roster (probably
being authenticated).

disconnecting: Focus is in the process of disconnecting endpoint
(either DISCONNECT or BYE was sent to the endpoint).

Note that the defined transient statuss (e.g., disconnecting,
alerting, etc.) could generate a lot of notifications.
Implementations MAY choose not to generate notifications on these to
all participants if it will generate too much traffic.

[5.6.4](#) joining-method of joining-type

This element contains method by which the endpoint joined the
conference, and can assume the following values:

dialed-in: The endpoint dialed into the conference, i.e. sent INVITE
to the focus, which resulted in successful dialog establishment.

dialed-out: The focus has brought the endpoint into the conference by
sending a successful INVITE to the endpoint.

focus-owner: The endpoint is the focus for this conference. This
status is used only when a participantÆs UA acts as a conference
focus.

5.6.5 **joining-info of execution-type**

   This element contains information about how the endpoint joined and
   can contain the following sub-elements:

   when: This element contains the date and time that the endpoint
      joined the conference.

   reason: This element contains the reason the endpoint joined the
      conference.

   by: This element contains the URI of the entity who caused the
      endpoint to join the conference.

5.6.6 **disconnection-method of disconnection-type**

   This element contains method by which the endpoint departed the
   conference, and can assume the following values:

   departed: The endpoint sent a BYE, thus leaving the conference.

   booted: The endpoint was sent a BYE by the focus, booting him/her out
      of the conference. Alternatively, the endpoint tried to dial into
      to conference without success because was rejected by the focus
      according to local policy decisions.

   failed: The server tried to bring the endpoint into the conference,
      but its attempt to contact the specific endpoint resulted in a
      non-200 class final response. Alternatively, the endpoint tried
      to dial into the conference without success due to technical
      reasons.

5.6.7 **disconnection-info of execution-type**

   This element contains information about the endpoint's departure from
   the conference and can contain the following sub-elements:

   when: This element contains the date and time that the endpoint
      departed the conference.

   reason: This element contains the reason the endpoint departed the
      conference. When known, it is RECOMMENDED to include the numeric
      reason followed by the descriptive text as conveyed/reported by
      the call signaling.

by: This element contains the URI of the entity who caused the
   endpoint to depart the conference.

5.6.8 media of media-type

This element contains information about a media stream of this
endpoint. The element of the media-type can have unbounded number of
appearance in the endpoint-type for each media stream of the
endpoint. Note that it is possible that media streams listed under a
common endpoint MAY be established by separate signaling means and
consequently belong to different signaling "calls".

5.7 media-type

This type defines the following attributes:

id: The attribute is a unique identifier of a media stream on a per
   endpoint basis. This attribute is used as a key to identify media
   streams which may be added and deleted on a dynamic basis during
   the conference. If the SDP "mid" (as defined in Grouping of Media
   Lines in the SDP [9]) is used for establishing the media stream,
   the 'id' SHOULD contain the same "mid" value, otherwise the
   notification service MUST generate an 'id' value which is unique
   in the endpoint context.

state: This attribute indicates whether the element contains the
   whole media information ("full"), only the information that has
   changed since the previous notification ("partial"), or that the
   media element has been deleted from the conference document
   ("deleted").

This type defines an extendable sequence of the following child
elements.

5.7.1 display-text of string type

This element contains the display text for the media stream.

5.7.2 proto of string type

This element contains the media type for the media stream. The value
of this element MUST be one of the values registered for "proto" of
SDP [3] and its later revision(s).

5.7.3 src-id of string type

The <src-id> element, if applicable, carries the information about
the actual source of the media. For example, for the RTP/RTCP [10]

media streams the value MUST contain the SSRC value generated by the
endpoint for the stream it sends.

When an RTP mixer generates a CSRC list according to RTP/RTCP [10],
it inserts a list of the SSRC identifiers of the sources that
contributed to the generation of a particular packet into the RTP
header of that packet. A quote from RFC 3550: "An example
application is audio conferencing where a mixer indicates all the
talkers whose speech was combined to produce the outgoing packet,
allowing the receiver to indicate the current talker, even though all
the audio packets contain the same SSRC identifier (that of the
mixer)."

5.7.4 label of string type

The element <label> carries a unique identifier for this stream among
all streams in the conference and is assigned by the focus. The
value of this element corresponds to the SDP "label" media attribute
defined in [19].

5.7.5 status of media-status-type

The element <status> indicates the status in both directions of the
media stream and has the values "sendrecv", "sendonly", "recvonly",
or "inactive" as defined in SDP [3] and its later revision(s). Note
that value specifies the direction from the participant's point of
view. For example, a muted participant's stream will have the value
of "recvonly".

5.7.6 call of call-type

The <call> element is a general container for providing call
signaling detailed information. Note that privacy policies MUST be
consulted before revealing this information to third-party
participants.

Specifically, the <sip> sub-element contains the SIP dialog
identifier of the endpoint's dialog with the focus. The element
includes sub-elements <display-text>, <call-id>, <to-tag>,
<from-tag>.

In future, the <call> element can be expanded to include other call
signaling protocol identifiers.

6.   **XML Schema**

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:conference-info"
xmlns:tns="urn:ietf:params:xml:ns:conference-info" xmlns:xs="http://www.w3.org/
2001/XMLSchema" xmlns="urn:ietf:params:xml:ns:conference-info"
elementFormDefault="qualified" attributeFormDefault="unqualified">
<!--
        This import brings in the XML language attribute xml:lang
-->
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
schemaLocation="http://www.w3.org/2001/03/xml.xsd" />

<!--
   CONFERENCE ELEMENT
-->
<xs:element name="conference-info" type="conference-type"/>

<!--
   CONFERENCE TYPE
-->
<xs:complexType name="conference-type">


   <xs:sequence>
       <xs:element name="conference-description" type="conference-
description-type" minOccurs="0"/>
      <xs:element name="host-info" type="host-type" minOccurs="0"/>
      <xs:element name="conference-state" type="conference-state-type"
minOccurs="0"/>
      <xs:element name="users" type="users-type" minOccurs="0"/>
      <xs:element name="sidebars-by-ref" type="uris-type" minOccurs="0"/>
      <xs:element name="sidebars-by-val" type="sidebars-by-val-type"
minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>

   <xs:attribute name="entity" type="xs:anyURI" use="required"/>
   <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
   <xs:attribute name="version" type="xs:unsignedInt" use="optional"/>
   <xs:anyAttribute namespace="##other"/>

</xs:complexType>

<!--
```

```
      STATE TYPE
   -->
   <xs:simpleType name="state-type">
       <xs:restriction base="xs:string">
              <xs:enumeration value="full"/>
              <xs:enumeration value="partial"/>
              <xs:enumeration value="deleted"/>
       </xs:restriction>
```

```
    </xs:simpleType>

    <!--
       CONFERENCE DESCRIPTION TYPE
    -->
    <xs:complexType name="conference-description-type">

       <xs:sequence>
         <xs:element name="display-text" type="xs:string" minOccurs="0"/>
         <xs:element name="subject" type="xs:string" minOccurs="0"/>
         <xs:element name="free-text" type="xs:string" minOccurs="0"/>
         <xs:element name="keywords" type="keywords-type" minOccurs="0"/>
         <xs:element name="conf-uris" type="uris-type" minOccurs="0"/>
         <xs:element name="service-uris" type="uris-type" minOccurs="0"/>
         <xs:element name="maximum-user-count" type="user-count-type"
minOccurs="0"/>
         <xs:element name="available-media" type="conference-medias-type"
minOccurs="0"/>
         <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
       </xs:sequence>

       <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
       <xs:anyAttribute namespace="##other"/>
    </xs:complexType>

    <!--
       HOST TYPE
    -->
    <xs:complexType name="host-type">

       <xs:sequence>
         <xs:element name="display-text" type="xs:string" minOccurs="0"/>
         <xs:element name="web-page" type="xs:anyURI" minOccurs="0"/>
         <xs:element name="uris" type="uris-type" minOccurs="0"/>
         <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
       </xs:sequence>

       <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
       <xs:anyAttribute namespace="##other"/>
    </xs:complexType>

    <!--
       CONFERENCE STATE TYPE
    -->
```

```
<xs:complexType name="conference-state-type">

   <xs:sequence>
       <xs:element name="user-count" type="user-count-type" minOccurs="0"/>
     <xs:element name="active" type="xs:boolean" minOccurs="0"/>
     <xs:element name="locked" type="xs:boolean" minOccurs="0"/>
```

```
      <xs:element name="recording" type="uris-type" minOccurs="0"/>
      <xs:element name="active-media" type="conference-medias-type"
minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>

   <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
   <xs:anyAttribute namespace="##other"/>
 </xs:complexType>

 <!--
    CONFERENCE MEDIAS TYPE
 -->
 <xs:complexType name="conference-medias-type">

   <xs:sequence>
     <xs:element name="entry" type="conference-media-type"
maxOccurs="unbounded"/>
   </xs:sequence>

   <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
 <xs:anyAttribute namespace="##other"/>

 </xs:complexType>

 <!--
    CONFERENCE MEDIA TYPE
 -->
 <xs:complexType name="conference-media-type">
   <xs:sequence>
     <xs:element name="proto" type="xs:string"/>
     <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>

   <xs:anyAttribute namespace="##other"/>
 </xs:complexType>

 <!--
    URIs TYPE
 -->
 <xs:complexType name="uris-type">
   <xs:sequence>
     <xs:element name="entry" type="uri-type" maxOccurs="unbounded"/>
   </xs:sequence>
```

```
      <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
      <xs:anyAttribute namespace="##other"/>
   </xs:complexType>

   <!--
```

```
        URI TYPE
    -->
    <xs:complexType name="uri-type">
        <xs:sequence>
          <xs:element name="uri" type="xs:anyURI"/>
          <xs:element name="display-text" type="xs:string" minOccurs="0"/>
          <xs:element name="purpose" type="xs:string" minOccurs="0"/>
          <xs:element name="modified" type="execution-type" minOccurs="0"/>
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>

        <xs:anyAttribute namespace="##other"/>
    </xs:complexType>

    <!--

        USER COUNT TYPE
    -->
    <xs:complexType name="user-count-type">
        <xs:sequence>
          <xs:element name="entry" type="count-type" maxOccurs="unbounded"/>
        </xs:sequence>

        <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
        <xs:anyAttribute namespace="##other"/>
    </xs:complexType>

    <!--
        COUNT TYPE
    -->
    <xs:complexType name="count-type">
        <xs:sequence>
          <xs:element name="role" type="xs:string"/>
          <xs:element name="count" type="xs:nonNegativeInteger"/>
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>

        <xs:anyAttribute namespace="##other"/>
    </xs:complexType>

    <!--
        KEWORDS TYPE
    -->
    <xs:simpleType name="keywords-type">
            <xs:list itemType="xs:string"/>
```

```
        </xs:simpleType>

        <!--
```

```
       USERS TYPE
    -->
    <xs:complexType name="users-type">
       <xs:sequence>
           <xs:element name="user" type="user-type" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
       </xs:sequence>

       <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
       <xs:anyAttribute namespace="##other"/>
    </xs:complexType>

    <!--
       USER TYPE
    -->
    <xs:complexType name="user-type">
       <xs:sequence>
         <xs:element name="display-text" type="xs:string" minOccurs="0"/>
         <xs:element name="associated-aors" type="uris-type" minOccurs="0"/>
         <xs:element name="roles" type="user-roles-type" minOccurs="0"/>
         <xs:element name="language" type="xs:language" minOccurs="0"/>
         <xs:element name="cascaded-focus" type="xs:anyURI" minOccurs="0"/>
         <xs:element name="endpoint" type="endpoint-type" minOccurs="0"
maxOccurs="unbounded"/>
         <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
       </xs:sequence>

       <xs:attribute name="entity" type="xs:anyURI"/>
       <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
       <xs:anyAttribute namespace="##other"/>
    </xs:complexType>

    <!--
       USER ROLES TYPE
    -->
    <xs:complexType name="user-roles-type">
       <xs:sequence>
         <xs:element name="entry" type="xs:string" maxOccurs="unbounded"/>
       </xs:sequence>

       <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
       <xs:anyAttribute namespace="##other"/>
```

```
        </xs:complexType>

        <!--
            ENDPOINT TYPE
        -->
        <xs:complexType name="endpoint-type">
            <xs:sequence>
```

```
        <xs:element name="display-text" type="xs:string" minOccurs="0"/>
        <xs:element name="referred" type="execution-type" minOccurs="0"/>
        <xs:element name="status" type="endpoint-status-type" minOccurs="0"/>
        <xs:element name="joining-method" type="joining-type" minOccurs="0"/>
        <xs:element name="joining-info" type="execution-type" minOccurs="0"/>
        <xs:element name="disconnection-method" type="disconnection-type"
minOccurs="0"/>
        <xs:element name="disconnection-info" type="execution-type"
minOccurs="0"/>
        <xs:element name="media" type="media-type" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>

      <xs:attribute name="entity" type="xs:anyURI"/>
      <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
      <xs:anyAttribute namespace="##other"/>
   </xs:complexType>

   <!--
      ENDPOINT STATUS TYPE
   -->
   <xs:simpleType name="endpoint-status-type">
       <xs:restriction base="xs:string">
               <xs:enumeration value="pending"/>
               <xs:enumeration value="dialing-out"/>
               <xs:enumeration value="dialing-in"/>
               <xs:enumeration value="alerting"/>
               <xs:enumeration value="on-hold"/>
               <xs:enumeration value="connected"/>
               <xs:enumeration value="muted-via-focus"/>
               <xs:enumeration value="disconnecting"/>
               <xs:enumeration value="disconnected"/>
       </xs:restriction>
   </xs:simpleType>

   <!--
      JOINING TYPE
   -->
   <xs:simpleType name="joining-type">
       <xs:restriction base="xs:string">
               <xs:enumeration value="dialed-in"/>
               <xs:enumeration value="dialed-out"/>
               <xs:enumeration value="focus-owner"/>
       </xs:restriction>
   </xs:simpleType>
```

```
<!--
   DISCONNECTION TYPE
-->
<xs:simpleType name="disconnection-type">
```

```
          <xs:restriction base="xs:string">
                <xs:enumeration value="departed"/>
                <xs:enumeration value="booted"/>
                <xs:enumeration value="failed"/>
          </xs:restriction>
     </xs:simpleType>

     <!--
        EXECUTION TYPE
     -->
     <xs:complexType name="execution-type">
        <xs:sequence>
          <xs:element name="when" type="xs:dateTime" minOccurs="0"/>
          <xs:element name="reason" type="xs:string" minOccurs="0"/>
          <xs:element name="by" type="xs:anyURI" minOccurs="0"/>
        </xs:sequence>

        <xs:anyAttribute namespace="##other"/>
     </xs:complexType>

     <!--
        MEDIA TYPE
     -->
     <xs:complexType name="media-type">
        <xs:sequence>
          <xs:element name="display-text" type="xs:string" minOccurs="0"/>
          <xs:element name="proto" type="xs:string" minOccurs="0"/>
          <xs:element name="src-id" type="xs:string" minOccurs="0"/>
          <xs:element name="label" type="xs:string" minOccurs="0"/>
          <xs:element name="status" type="media-status-type" minOccurs="0"/>
          <xs:element name="call" type="call-type" minOccurs="0"/>
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>


        <xs:attribute name="id" type="xs:string" use="required"/>
        <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
        <xs:anyAttribute namespace="##other"/>
     </xs:complexType>

     <!--
        MEDIA STATUS TYPE
     -->
     <xs:simpleType name="media-status-type">
          <xs:restriction base="xs:string">
                <xs:enumeration value="recvonly"/>
```

```
                <xs:enumeration value="sendonly"/>
                <xs:enumeration value="sendrecv"/>
```

```
                    <xs:enumeration value="inactive"/>
            </xs:restriction>
      </xs:simpleType>


      <!--
          CALL TYPE
      -->
      <xs:complexType name="call-type">
         <xs:choice>
            <xs:element name="sip" type="sip-dialog-id-type"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:choice>

         <xs:anyAttribute namespace="##other"/>
      </xs:complexType>


      <!--
          SIP DIALOG ID TYPE
      -->
      <xs:complexType name="sip-dialog-id-type">
         <xs:sequence>
            <xs:element name="display-text" type="xs:string" minOccurs="0"/>
            <xs:element name="call-id" type="xs:string"/>
            <xs:element name="from-tag" type="xs:string"/>
            <xs:element name="to-tag" type="xs:string"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:sequence>

         <xs:anyAttribute namespace="##other"/>
      </xs:complexType>


      <!--
          SIDEBARS BY VAL TYPE
      -->
      <xs:complexType name="sidebars-by-val-type">
         <xs:sequence>
            <xs:element name="entry" type="conference-type" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:sequence>

         <xs:attribute name="state" type="state-type" use="optional"
default="full"/>
         <xs:anyAttribute namespace="##other"/>
      </xs:complexType>

   </xs:schema>
```

[7](#). Examples

[7.1](#) Basic Example

   The following is an example conference information document:


```
   <conference-info entity="sips:conf233@example.com" state="partial"
version="5" >
   <!--
      CONFERENCE INFO
   -->
     <conference-description>
       <subject>Agenda: This month's goals</subject>
       <service-uris>
               <entry>
                       <uri> http://sharepoint/salesgroup/</uri>
                       <purpose>web-page</purpose>
               </entry>
       </service-uris>
     </conference-description>

   <!--
      CONFERENCE STATE
   -->
     <conference-state>
       <user-count>
               <entry>
                       <role>any</role>
                       <count>33</count>
               </entry>
       </user-count>
       <active-media>
               <entry>
                       <proto>audio</proto>
               </entry>
       </active-media>
     </conference-state>

     <users>
     <!--
         USER
     -->
       <user entity="sip:bob@example.com" state="full">
               <display-text>Bob Hoskins</display-text>
               <!--
               ENDPOINTS
               -->
```

```
                  <endpoint entity="sip:bob@pc33.example.com">
```

```
                                <display-text>Bob's Laptop</display-text>
                                <status>disconnected</status>
                                <disconnection-method>departed</disconnection-
method>
                                <disconnection-info>
                                        <when>2005-03-04T20:00:00Z</when>
                                        <reason>bad voice quality</reason>
                                        <by>sip:mike@example.com</by>
                                </disconnection-info>
                        <!--
                                MEDIA
                        -->
                        <media id="1">
                                <display-text>main audio</display-text>
                                <proto>audio</proto>
                                <src-id>432424</src-id>
                                <label>34567</label>
                                <status>sendrecv</status>
                        </media>
                </endpoint>
        </user>

        <!--
            USER
        -->
        <user entity="sip:alice@example.net" state="full">
        <display-text>Alice</display-text>
        <!--
                        ENDPOINTS
                -->
                <endpoint entity="sip:4kfk4j392jsu@example.net;grid=433kj4j3u">
                        <status>connected</status>
                        <joining-method>dialed-out</joining-method>
                        <joining-info>
                                <when>2005-03-04T20:00:00Z</when>
                                <by>sip:mike@example.com</by>
                        </joining-info>
                        <!--
                                MEDIA
                        -->
                        <media id="1">
                                <display-text>main audio</display-text>
                                <proto>audio</proto>
                                <src-id>534232</src-id>
                                <label>34564</label>
                                <status>sendrecv</status>
                        </media>
        </endpoint>
```

```
        </user>
```

```
      </users>

   </conference-info>
```

7.2 Rich Example

   The following is an example conference information document. In this
   example of a partial state notification, there are 32 participants in
   a voice conference. The user Bob has been booted from the conference
   by Mike due to bad voice quality. Note that there are three sidebars
   in the conference, two are referenced just by their sidebar URIs and
   information about the third sidebar is included in this notification.
   Also note that while this conference offers both audio and video
   capabilities, only audio is currently in use.

```
   <conference-info entity="sips:conf233@example.com" state="partial"
version="5" >
      <!--
        CONFERENCE INFO
      -->
       <conference-description>
               <display-text>Weekly Sales Meeting</display-text>
               <subject>Agenda: This month's goals</subject>
               <free-text>We will start strict on time</free-text>
               <keywords>sales, meeting, weekly</keywords>
               <conf-uris>
                       <entry>
                               <uri>tel:+18005671234</uri>
                               <display-text>TTI Bridge</display-text>
                       </entry>
                       <entry>
                               <uri>h323:conf545@h323.example.com</uri>
                       </entry>
               </conf-uris>
               <service-uris>
                       <entry>
                               <uri>http://sharepoint/salesgroup/</uri>
                               <purpose>web-page</purpose>
                       </entry>
               </service-uris>
               <maximum-user-count>
                       <entry>
                               <role>any</role>
                               <count>52</count>
                       </entry>
```

```
                    <entry>
```

```
                          <role>participant</role>
                          <count>50</count>
                  </entry>
          </maximum-user-count>
          <available-media>
                  <entry>
                          <proto>audio</proto>
                  </entry>
                  <entry>
                          <proto>video</proto>
                  </entry>
          </available-media>
  </conference-description>
  <!--
    HOST INFO
  -->
  <host-info>
          <display-text>Sales Host</display-text>
          <web-page>http://sharepoint/salesgroup/hosts/</web-page>
          <uris>
                  <entry>
                          <uri>sip:sales@example.com</uri>
                  </entry>
          </uris>
  </host-info>
  <!--
    CONFERENCE STATE
  -->
  <conference-state>
          <user-count>
                  <entry>
                          <role>any</role>
                          <count>33</count>
                  </entry>
                  <entry>
                          <role>participant</role>
                          <count>32</count>
                  </entry>
          </user-count>
          <active>true</active>
          <locked>false</locked>

          <recording>
                  <entry>
    <uri>http://quicktime.streaming.com/54634/recording.mov</uri>
                          <display-text>Quicktime</display-text>
                  </entry>
                  <entry>
```

```
             <uri>http://real.streaming.com/54634/recording.ram</uri>
                     </entry>
             </recording>

             <active-media>
                     <entry>
                             <proto>audio</proto>
                     </entry>
             </active-media>
     </conference-state>

     <!--
         USERS
       -->
     <users>

        <user entity="sip:bob@example.com">
             <display-text>Bob Hoskins</display-text>
             <associated-aors>
                     <entry>
                             <uri>mailto:bob@example.com</uri>
                             <display-text>email</display-text>
                     </entry>
             </associated-aors>
             <roles>
                     <entry>participant</entry>
             </roles>
             <language>en</language>

             <!--
                 ENDPOINTS
             -->
             <endpoint entity="sip:bob@pc33.example.com">
                     <display-text>Bob's Laptop</display-text>
                     <referred>
                             <when>2005-03-04T20:00:00Z</when>
                             <reason>expert required</reason>
                             <by>sip:mike@example.com</by>
                     </referred>
                     <status>disconnecting</status>
                     <joining-method>dialed-out</joining-method>
                     <joining-info>
                             <when>2005-03-04T20:00:00Z</when>
                             <reason>invitation</reason>
                             <by>sip:mike@example.com</by>
                     </joining-info>
                     <disconnection-method>booted</disconnection-method>
                     <disconnection-info>
```

```
                                   <when>2005-03-04T20:00:00Z</when>
                                   <reason>bad voice quality</reason>
                                   <by>sip:mike@example.com</by>
                           </disconnection-info>

                           <!--
                                   MEDIA
                           -->
                           <media id="1" state="full">
                                   <display-text>main audio</display-text>
                                   <proto>audio</proto>
                                   <src-id>432424</src-id>
                                   <label>34567</label>
                                   <status>sendrecv</status>
                                   <call>
                                           <sip>
                                              <display-text>full info</display-
text>
                                              <call-id>hsjh8980vhsb78</call-id>
                                              <from-tag>vav738dvbs</from-tag>
                                              <to-tag>8954jgjg8432</to-tag>
                                           </sip>
                                   </call>
            </media>
            </endpoint>
            </user>
      </users>

      <!--
          SIDEBARS BY REFERENCE
      -->
      <sidebars-by-ref>
          <entry>
                  <uri>sips:conf233@example.com; grid=45</uri>
                  <display-text>sidebar with Carol</display-text>
        </entry>
        <entry>
                  <uri>sips:conf233@example.com; grid=21</uri>
                  <display-text>private sidebar with Peter</display-text>
        </entry>
      </sidebars-by-ref>

      <!--
          SIDEBARS BY VALUE
      -->
      <sidebars-by-val>
           <entry entity="sips:conf233@example.com; grid=77" state="partial">
                  <users>
```

```
                <user entity="sip:bob@example.com" state="partial"></user>
```

```
                  <user entity="sip:mark@example.com" state="partial"></user>
                  <user entity="sip:dan@example.com" state="partial"></user>
              </users>
         </entry>
    </sidebars-by-val>

    </conference-info>
```

8. **Security Considerations**

   Subscriptions to conference state can reveal very sensitive
   information. For this reason, the document recommends authentication
   and authorization, and provides guidelines on sensible authorization
   policies.

   Since the data in notifications is sensitive as well, end-to-end SIP
   encryption mechanisms using S/MIME SHOULD be used to protect it.

   Since a focus provides participants identity information using this
   event package, participant privacy needs to be taken into account. A
   focus MUST support requests by participants for privacy. Privacy can
   be indicated by the conference policy - for every participant or
   select participants. It can also be indicated in the session
   signaling. In SIP this can be done using the Privacy header field
   described in RFC 3323 [8]. For a participant requesting privacy, no
   identity information SHOULD be revealed by the focus such as a URI
   (e.g. the Address of Record, Contact, or GRUU). For these cases,
   the anonymous URI generation method outlined in section Section 5.5
   of this document MUST be followed.

9. IANA Considerations

   This document registers a SIP event package, a new MIME type,
   application/conference-info+xml, a new XML namespace, and a new XML
   schema.

9.1 conference Event Package Registration

   This specification registers an event package, based on the
   registration procedures defined in RFC 3265 [7]. The following is
   the information required for such a registration:
   Package Name: conference
   Package or Template-Package: This is a package.
   Published Document: RFC XXXX (Note to RFC Editor: Please fill in XXXX
      with the RFC number of this specification).
   Person to Contact: Jonathan Rosenberg, jdrosen@jdrosen.net.

9.2 application/conference-info+xml MIME Registration
   MIME media type name: application
   MIME subtype name: conference-info+xml
   Mandatory parameters: none
   Optional parameters: Same as charset parameter application/xml as
      specified in RFC 3023 [5].
   Encoding considerations: Same as encoding considerations of
      application/xml as specified in RFC 3023 [5].
   Security considerations: See Section 10 of RFC 3023 [5] and Section 8
      of this specification.
   Interoperability considerations: none.
   Published specification: This document.
   Applications which use this media type: This document type has been
      used to support SIP conferencing applications.
   Additional Information:
      Magic Number: None
      File Extension: .cif or .xml
      Macintosh file type code: "TEXT"
   Personal and email address for further information: Jonathan
      Rosenberg, <jdrosen@jdrosen.net>
   Intended usage: COMMON
   Author/Change controller: The IETF.

9.3 URN Sub-Namespace Registration for
     urn:ietf:params:xml:ns:conference-info

   This section registers a new XML namespace, as per the guidelines in
   RFC 3688 [12].

   URI: The URI for this namespace is
      urn:ietf:params:xml:ns:conference-info.
   Registrant Contact: IETF, SIPPING working group, <sipping@ietf.org>,
      Jonathan Rosenberg <jdrosen@jdrosen.net>.
   XML:

   BEGIN
   <?xml version="1.0"?>
   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
             "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
   <html xmlns="http://www.w3.org/1999/xhtml">
   <head>
     <meta http-equiv="content-type"
        content="text/html;charset=iso-8859-1"/>
     <title>Conference Information Namespace</title>
   </head>
   <body>
     <h1>Namespace for Conference Information</h1>
     <h2>urn:ietf:params:xml:ns:conference-info</h2>
     <p>See RFCXXXX.</p>
   </body>
   </html>
   END

9.4 XML Schema Registration

   This specification registers a schema, as per the guidelines in RFC
   3688 [12].
      URI: please assign.
      Registrant Contact: IETF, SIPPING Working Group
      (sipping@ietf.org), Jonathan Rosenberg (jdrosen@jdrosen.net).
      XML: The XML can be found as the sole content of Section 6.

10. **Acknowledgements**

The authors would like to thank Dan Petrie, Sean Olson, Alan
Johnston, Rohan Mahy, Cullen Jennings, and Miguel Garcia for their
comments and inputs.

11. References

11.1 Normative References

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

   [2]   Moats, R., "URN Syntax", RFC 2141, May 1997.

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

   [4]   Moats, R., "A URN Namespace for IETF Documents", RFC 2648,
          August 1999.

   [5]   Murata, M., St. Laurent, S. and D. Kohn, "XML Media Types", RFC
          3023, January 2001.

   [6]   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.

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

   [8]   Peterson, J., "A Privacy Mechanism for the Session Initiation
          Protocol (SIP)", RFC 3323, November 2002.

   [9]   Camarillo, G., Eriksson, G., Holler, J. and H. Schulzrinne,
          "Grouping of Media Lines in the Session Description Protocol
          (SDP)", RFC 3388, December 2002.

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

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

   [12]   Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,
          January 2004.

11.2 Informative References

   [13]   Schulzrinne, H., Rao, A. and R. Lanphier, "Real Time Streaming
          Protocol (RTSP)", RFC 2326, April 1998.

   [14]   Vaha-Sipila, A., "URLs for Telephone Calls", RFC 2806, April

          2000.

     [15]   Levin, O., "H.323 Uniform Resource Locator (URL) Scheme
            Registration", RFC 3508, April 2003.

     [16]   Rosenberg, J., "A Framework for Conferencing with the Session
            Initiation Protocol",
            draft-ietf-sipping-conferencing-framework-03 (work in
            progress), October 2004.

     [17]   Rosenberg, J., "An INVITE Inititiated Dialog Event Package for
            the Session Initiation Protocol (SIP)",
            draft-ietf-sipping-dialog-package-05 (work in progress),
            November 2004.

     [18]   Rosenberg, J., "Obtaining and Using Globally Routable User
            Agent (UA) URIs (GRUU) in the Session Initiation Protocol
            (SIP)", draft-ietf-sip-gruu-02 (work in progress), July 2004.

     [19]   Levin, O. and G. Camarillo, "The SDP (Session Description
            Protocol) Label Attribute",
            draft-ietf-mmusic-sdp-media-label-00 (work in progress),
            September 2004.

Authors' Addresses

     Jonathan Rosenberg
     Cisco Systems
     600 Lanidex Plaza
     Parsippany, NJ 07054
     US

     Phone: +1 973 952-5000
     EMail: jdrosen@cisco.com
     URI: http://www.jdrosen.net

     Henning Schulzrinne
     Columbia University
     M/S 0401
     1214 Amsterdam Ave.
     New York, NY 10027
     US

     EMail: schulzrinne@cs.columbia.edu
     URI: http://www.cs.columbia.edu/~hgs

Orit Levin (editor)
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
US

EMail: oritl@microsoft.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment