SIP-based Protocol for P2P Large-scale Multiparty VoIP (MVoIP) Conference Support

Wajdi ELLEUCH and Alain C. HOULE

Abstract— Even if both traditional centralized and decentralized models can support small conferences, their deployment for large number of participants is less obvious. In fact, while server-based centralized conference facilitates control and administration operations, supporting centralized media processing in large scale conference causes system overhead on the mixer. On the other hand, decentralized solutions that use multi end-system media processors will introduce an overhead to control conference floors and users membership. Our solution introduces a new model that enable multi-host media process support while the conference control and management is kept simplified and centralized around the administrator. To do that, we build two different meshed networks to enable both voice audio distribution between participants and general conference control. Our conference system includes different components that enable conference creation/destruction, user addition/removal, media assignment and speech floor control. We introduce an abstract protocol message that implements membership operations. We also discuss the implementation of system components and we detail their mapping to SIP.

***Index Terms*—VoIP, SIP, streaming applications, multi-party communication, conference models, tree-based distribution, conference control, floor control, media distribution**

I. Introduction

Several models and architectures are proposed to support Multiparty VoIP (MVoIP) services. Initial solutions [1][2] are based on network infrastructure and use routers to redirect live streams for large groups of participants by using multicasting addresses. Since not all routers in the Internet are IP-multicast enabled, this creates a major deployment problem. To overcome this shortcoming, actual MVoIP solutions implement multicast services on top of the IP unicast service of end-systems. The aim is to create an overlay communication topology based on application-links known as Application Layer Multicast (ALM) systems [3][4]. ALM brings the advantage of requiring only application software installation on host peers without any added network infrastructure.

Depending on media processing assignment, two approaches are actually used by ALM systems to support MVoIP service. The first approach implements media processing on peers and distinguishes two models. The first model uses tightly-coupled media topology based on the end-system mixing (ESM) that the existing most popular VoIP System Skype [5] uses. The second model, based on peers and described in [6] and [7], implements fully-coupled media topology in a decentralized media mixing manner. While these two peers based models are easy to implement, their deployment remains adapted only to small scale conferences. In fact, the end-system mixing limits the number of participants to four or five while the fully-coupled model presents serious limitations for conference floor control. The second approach uses a dedicated conference server machine that manages conference control and processes media mixing/distribution. Actual large scale MVoIP solutions use this approach and some commercial MVoIP products, like IVISIT [8] or WEBEX [9], are actually available and able to support up to one hundred simultaneous users within the same conference. Deploying and maintaining conference server in these solutions are very expensive and require large bandwidth from the server side. To resolve problems related to the unique server deployment overload, bandwidth congestion and central point of failure, some researches introduce multi-server based solutions that spread media processing load among different servers. The servers are then geographically dispersed and each participant will be automatically attached to the nearby server [10][11]. Such solution introduces additional cost to deploy and to maintain large number of servers. On the other hand, since more and more participant devices can offer large computational power and high bandwidth access, it becomes possible to assign media processing load to some of them instead of deploying conference servers. The aim here is to create peer-based MVoIP conferences that collectively fulfill the audio stream processing task depending on the availability and capabilities of each participant.

In contrast to network and server infrastructure-based stream content delivery networks, ALM end-point based solution has recently received attention [12-14]. These ALM solutions tend to produce self-organizing, efficient and self-improving overlay meshes that can be dynamically adapted to different network variations. In these systems, a participant node can decide about its parents and migrate within the constructed tree to minimize the redundant transmission on

Manuscript received July 31, 2008

W. Elleuch B.Ing, M.Sc.A is a Ph.D candidate in the department of Electrical Engineering and Computer Engineering at Université de Sherbrooke, Sherbrooke (Québec) CANADA. (e-mail: Wajdi.Elleuch@USherbrooke.ca).

A. C. Houle, ing. Ph.D. is a professor in the department of Electrical Engineering and Computer Engineering at Université de Sherbrooke, Sherbrooke (Québec) CANADA. (e-mail: Alain.Houle@USherbrooke.ca).

physical links. Unfortunately, we found these solutions are mostly focused on the network resources aspect instead of user preferences and activity inside the MVoIP conference. For example, active participants that talk on the conference should be handled distinctively from a participant that is just in listen mode. Also, implementing complex control functionalities on all end-host might restrict access for handheld user devices with limited resources. Moreover, designer of large scale MVoIP based on media processing peers should consider different aspects related to conference creation, user membership management, control media distribution and speech floor control for each participant. Implementing these control functionalities on all peers, in a distributed manner, is excessively complex since a global and unified view of the conference floor and unique control decisions about user's membership and media distribution are required. Maintaining an updated version of such information, and sharing it between peers, for large scale conferences is also an excessive overhead.

To resolve these challenges, we introduce, in this work, an innovative peer-based system that supports large scale conferences by making separation between media network and control network. While media processing uses dynamic tree-shaped decentralized approach, the control network is based on a topology centralized around the conference administrator. The proposed system enables conference creation/destruction and user addition/removal. Moreover, each user that joins a conference can offer its media processing service to the administrator that controls media distribution using the Third Party Call Control mechanism by remotely establishing, updating and removing media sessions between participants. This system also enables speech floor control to manage participant talking privileges and consequently adapts the media mixing/distribution network topology. Users can then migrate within the media tree depending on their capabilities and activities. Media tree optimization will not be treated in this work that focuses more on the model components presentation and protocol implementation. The proposed system supports participant fast leave or failure and uses mechanisms that enable fast handoff to repair media tree in such situations. Our system implements the application session management using abstract message protocol that can be mapped to the IETF Session Initiation Protocol (SIP) standard.

In section II we introduce our conference topology and deployed components. Section III include different membership operations supported by our developed protocol and section IV explains how to implement the proposed system in SIP protocol. Section V concludes this paper.

II. System Model

A. System topology

Our solution introduces new model that enables multi-host media processing support while the control is kept centralized on the administrator side. To do that, we build two different meshed networks that enable both voice audio distribution

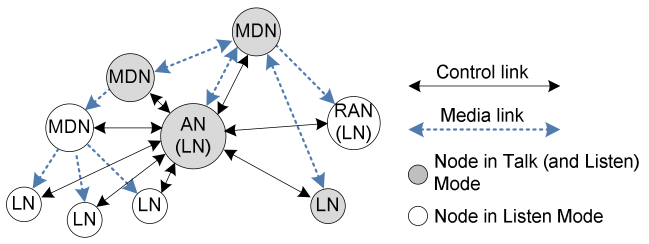

Figure 1: System control and media networks topologies

between participants and general conference control as shown on Figure 1. The first network ensures media delivery in tree-shaped topology. This network uses two different media roles components: the Mixer/Distributor Node (MDN) and the Leaf Node (LN). The MDN role is affected to nodes that maintain more than one audio session on the conference. Over and above processing media for themselves, MDN nodes can mix or even distribute media for others. On the other hand, LF role concerns nodes that hold only one audio session. Their departure should not prevent any other participant from continuing conferencing. Such role is adapted for light handheld devices with limited resources, for conventional IP phones or for participants that would not offer any part of their computational power and bandwidth to others. The second network establishes star-shaped control network constituted by connection link from each participant to the Administrator Node (AN). For each new added participant, new control link is established. Theses links are used to deliver control information between the AN and all remaining participants. As we do not wish to rely on a single non-failing entity to control conference, the AN should elect replica node, that we call Replica Administrator Node (RAN), among participants to replace it in cases of departure or failure.

B. System Components

The conference system defined in our model allows a large number of geographically dispersed users to exchange audio communication in real time. Supporting a large number of users involve new challenges for media processing distribution and conference control. Our system control includes the following components as show on Figure 2:

1) The Conference Management

This component mainly supports conference announcement, creation, modification and destruction. Each conference is uniquely identified by an URI address created by the AN. This address is communicated to public or private users by displaying it on dedicated web pages or shared between users

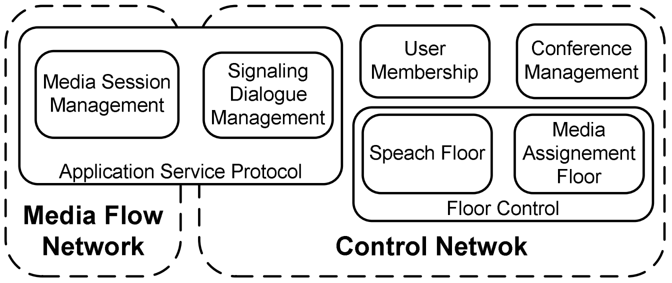

Figure 2: System components

on discussion forum, instant messaging, e-mail or other third communication system. General parameters related to the conference title/subject, maximum number of supported participants, maximum simultaneous speaker, etc. should be defined in this component.

2) Membership Management

This component is responsible of user addition and removal operations. Requests that enable user to join or to leave can be initiated whether by the users themselves (dial-in mode) or by the administrator (dial-out mode). Before accepting a user join request or before adding new users, user membership component should comply with conference management rules and restrictions. Conference access rules (pre-authorized participants, black-list, etc) can be used to facilitate membership management.

3) Floor control

Floor control manages authorizations to use conference shared resources by generally limiting the number of simultaneous access. IP-based Conference may have any number of floors, depending on the features supported by the conference i.e. audio, video, white board, mouse pointed, etc. Since we limit our actual system to MVoIP service support, we focus the conference floor on the available participant shared bandwidth (media assignment floor) and the talking rights (speech floor).

4) Application session management

This component implements communication protocol that enables both voice call establishment between users and conference control/administration. Call establishment requires from protocol to support session initialization, modification and termination. On the other hand, exchanging control and administration data between participants can be achieved by a subscription/notification mechanism. Such functionalities can be supported by the Session Initiation Protocol (SIP)[15] normalized by IETF and adopted by the 3GPP [16] as the standard that implements IMS services. SIP mechanisms can initiate such conferences, allowing users to join and leave conferences and making it possible for a participant to ask a third party to join. AN, in our model, use Third Party Call Control (3PCC) technique [17] to establish media session between two remote participants. Since the SIP message body can include textual formatted data, XML data format can be encapsulated in exchanged messages.

III. DESIGNING MEMBERSHIP OPERATIONS PROTOCOL

A. Approach overview

In this section, we present the abstract protocol that our control system uses to maintain media links between participants. In our approach, we assume that the number of participants that can mix and distribute media for others is usually sufficient to maintain conference existence. Also, since all added participants supply their input and output degree parameters, the AN remains aware about media processing availability on the conference.

B. Adding user to conference

Two different modes are offered to add new users to the conference. On the Dial-in mode, a new user initiates the call by sending the JOIN message with its media offer. The AN redirect the media offer from the New User (NU) to the available and appropriate MDN using message (1) then (2) as shown on Figure 3a. The media answer supplied by MDN is included on the ADD_MEDIA_Ok response message sent to the AN and redirected to the NU using JOIN_Ok response (message (3) and (4)). In this scenario, the NU sends the JOIN_Ack acknowledgment to confirm the reception of the JOIN_Ok. On the Dial-out scenario, the AN asks the NU to supply a media offer by sending black hold (bh) media offer [18] on the message (1). The NU includes its media offer using message (2) and receives the media answer from the message (5) as shown on Figure 3b. Since each media offer and answer contains the address of its original supplier, the two scenarios are able to establish direct media flow between NU and an MDN. The signaling dialogs of this communication remain completely managed and controlled by the AN.

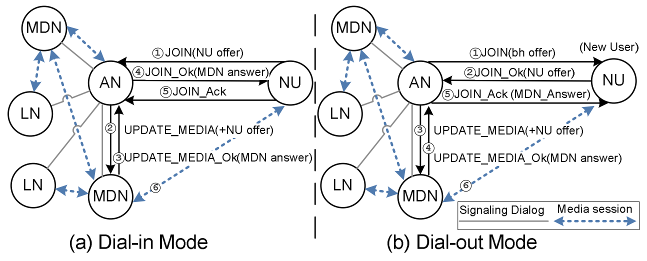

Figure 3: Message flow for user adding scenario

C. LN member departure or failure

LN can at any time leave conference, in dial-in mode, by sending LEAVE message (1) to AN as illustrated in Figure 4. The dial-out mode is used when the AN chooses to disconnect LN from conference using LEAVE message. On the two modes, AN should send message (2) to the parent of LN. The LN parent is the corresponding MDN that connects LN to the media tree. For a lot of reasons, related to hardware/software or even network problems, LN can leave conference without notifying AN. In this case, media flow between LN and its parent will be automatically disconnected. The parent of the LN will receive “media-disconnected” event notification about this failure. On the same case, AN that holds signaling dialog with LN, will receives “dialog-disconnected” signaling event notification. Accordingly, it become easy for AN and for LN’s parent to detect such member fast departure or failure and to subsequently update their states. Even if messages (2) and (3) that notify MDN about the LN departure are not required in this scenario, it remains appropriate to use them to guarantee conference coherence.

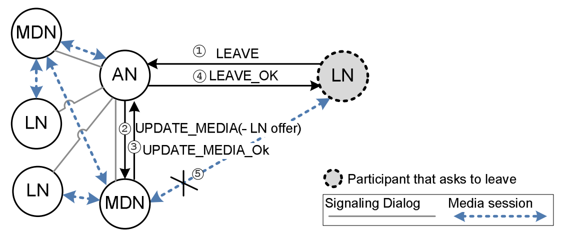

Figure 4: Message flow for LN leaving scenario

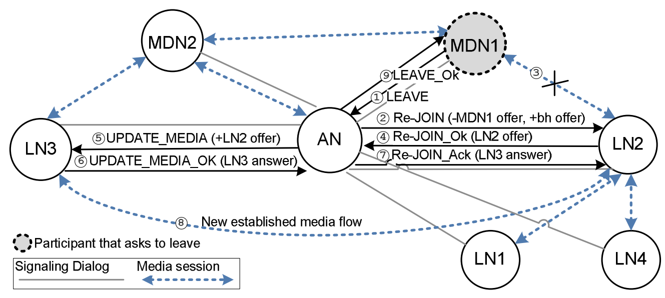
MDN2
MDN1
LN3
AN
LN2
LN1
LN4
⑨LEAVE_Ok
①LEAVE
③
② Re-JOIN (-MDN1 offer, +bh offer)
④ Re-JOIN_Ok (LN2 offer)
⑦ Re-JOIN_Ack (LN3 answer)
⑤UPDATE_MEDIA (+LN2 offer)
⑥UPDATE_MEDIA_OK (LN3 answer)
⑧ New established media flow
Participant that asks to leave
Signaling Dialog
Media session

Figure 5: Message flow for MDN departure scenario

D. *MDN member departure or failure*

MDN departure or failure is less obvious to support than LN case since MDN departure affects media distribution. MDN departure causes media tree partition in one or even many sub-trees. If AN is aware about this departure, (the dial-in or dial-out mode is used), media tree can be reconstructed proactively. In proactive approach, AN will redirect the affected children to new parents before accepting the MDN departure. In that case, the media disruption should be minimized. If MDN leaves without notification, media reconstruction will launched by AN, in reactive mode, as soon as "dialog-disconnected" event is received. Figure 5 illustrates the required message flow, based on the Re-JOIN message, used for this scenario.

E. *AN departure or failure*

Since we do not wish to rely on a single non-failing entity to control conference, the AN should elect, at last, one replica node, that we called Replica Administrator Node (RAN). AN should supply to the RAN an updated information about the conference structure. This information should contain enough information about already existent media flow trees and signaling dialogs to enable RAN to take over the conference administration at any time. Before leaving conference, AN should notify RAN about the last updated version of the conference. RAN launches the reconstruction of the media tree by using REPLACE_AN mechanism based on a request/response/acknowledgment as shown on Figure 6. Media flows between participants will be reestablished and their new dialogs will be centralized around the RAN (the new AN). Each participant that establishes a new dialog with the new AN, should notify the old AN by sending a LEAVE message (8), (10) and (14).

IV. SIP Implementation

A. *Mapping membership operation protocol*

The message protocol defined and used on message flow diagrams of section III can be mapped to Session Initiation Protocol SIP [15]. This implementation requires the introduction of some new headers. Each message needs to indicate the conference identification information by using a new *Conf-ID* header. Each conference will be then uniquely identified by an ID. This ID should be generated by the initial AN and by possibly using the same procedure as that used to generate the value of the *Call-Id* field. JOIN message used to add user to conference is mapped to SIP INVITE and RE-

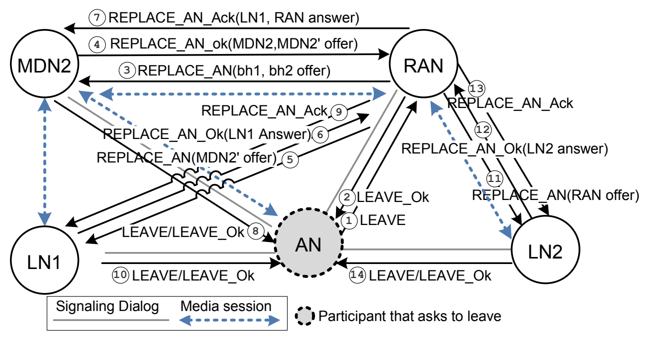

Figure 6: Message flow for AN departure scenario

JOIN is mapped to reINVITE. ReINVITE SIP message that updates an existing established dialog, use existing dialog identifiers (the same *to-tag*, *from-tag* and *Call-Id*). The two subsequent phases of the connection process of both JOIN and re-JOIN map naturally: JOIN_Ok and re-JOIN_Ok become a 2xx-class success response and JOIN_ACK and Re-JOIN_Ack are SIP ACKs.

UPDATE_MEDIA message is mapped to SIP UPDATE message based on two phases (request/response) transaction mechanism. REPLACE_AN message, mapped to INVITE SIP, should contain a *Replaces* header to identify the existing dialog that should be replaced by the new one. LEAVE message is mapped to SIP BYE request.

B. *Implementing Media offer/answer*

Our solution uses SDP protocol [19] to describe media offer/answer exchange mechanism that establishes media flow between two participants. SDP content, encapsulated in SIP messages, includes general description of the session, the creation time and the used Media. Since our messaging protocol that implements media flow connection/update follows the Third Party Call Control Flow I as specified in [17], we use that media description for each offer/answer transaction. Each media source is described in this format:

```
m=<media> <port> <transport> <fmt list>
c= <IP address>
```

As an example, the NU media offer included on the UPDATE_MEDIA message already shown on Figure 3a, will add the following lines for each media direct child on the tree:

```
m=audio 49230 RTP/AVP 0
c= IN IP4 224.5.6.7
```

In this example, *49230* is the media source port of the NU user, *0* is the type number of predefined media codec payload and *224.5.6.7* is the IP address of the NU.

C. *Implementing media assignment floor Control*

A user that can support media distribution or media mixing load can publish its capabilities within SDP. To support this extension, we add a new letter, for example "d=" in SDP line to contain the *input_degree* and the *output_degree*. SDP inherently support such extension:

```
d= IN 6 OUT 3
```

A participant that wants to change its values should contact the AN by sending UPDATE_MEDIA containing the updated SDP offer. A participant that doesn't support this extension will be handled as LF participant.

D. Implementing Speech Floor Control

SDP syntax allows the use of some parameters that specify the call flow directions (*sendrecv, sendonly, recvonly*). Since all negotiated media offers/answers are controlled by the AN, it becomes easy to add/remove them to manage the speech floor. Speech floor control uses external information related to the availability of media processors, maximum number of speakers and others to decide about speech privileges to assign to each participant. Suppose that LN1 and LN2 are connected to the same MDN. If LN1 is in listening mode and LN2 in talking mode, then the media offer supplied to MDN will be:

```
m=audio 49230 RTP/AVP 0
c= IN IP4 add_LN1
a=recvdonly
m=audio 49231 RTP/AVP 0
c= IN IP4 add_LN2
a=sendrecv
```

E. Implementing Conference management

Each conference is uniquely identified by SIP URI address created by the AN. AN can contact a SIP registrar server to associate the conference address to its network address. Tierce mechanism ca automatically publishes registered conference URI to public web pages. When AN leaves conference, the RAN that takes over conference control should change conference registration by redirecting the conference URI to the new AN. Conference management uses XML based model to describe the conference and related media tree. Each media node tag will include key information parameters about its current dialog established with the AN (*call-id*, *to-tag* and *from-tag*). An example of XML structure will be:

```
<Conf_description>
   <headers>
      <Conf_URI> … </Conf_URI>
      <Conf_title> … </Conf_title>
      <Max_participants> … </Max_participants>
      <Creation_date> … </Creation_date>
      …
   </headers>
   <media_tree>
      <Node add=… Call_Id=… to-tag=… from-tag=… >
      //children description
         <Node add=… Call_Id=… to-tag=… from-tag=… media_param/>
         <Node add=… Call_Id=… to-tag=… from-tag=… media_param>
         //children description
             <Node add=… … />
             <Node add=… … />
         </Node>
      </Node>
   </media_tree>
</Conf_description>
```

At every membership alteration, AN should notify the selected RAN by sending an updated copy of the XML conference description. Since AN and RAN have established SIP dialog, AN can use NOTIFY SIP request within the same dialog. SIP dialog identification included on the XML file will be used by the RAN on the *replaces* header within the REPLACE_AN request. *media_param* parameter is used to reconstruct the original flow that lie each node with its parent.

V. CONCLUSION

In this work, we introduced new conference model that allows media distribution for large scale conference based on peers processing. We identified the different components that should be integrated to allow both conference control/administration and media tree distribution among participants. We proposed an abstract protocol that implements operations related to membership management that we mapped to SIP. We also defined a mechanism that implement media offer/answer and speech/media assignment floors control. Member departure scenarios are provided and supported in our system to avoid the central point of failure problem.

VI. BIBLIOGRAPHY

[1] Watanabe K., "Distributed Multiparty Desktop Conferencing System: MERMAID", Proceedings of the Conference on Computer-Supported Cooperative Work, 1990.

[2] Mike Macedonia and Don Bruzman entitled "MBONE, the Multicast Backbone" IEEE Computer, 1994

[3] Chu, Y., Rao, S. G., Seshan, S., and Zhang,H., "A Case for End System Multicast", Measurement and Modeling of Computer Systems, 2000.

[4] Wong, S. H. Y., Lui, J. C. S., "An Architectural Infrastructure and Topological Optimization for End System Multicast," mascots, p. 0481, 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'02), 2002

[5] The Skype Internet Telephony System, www.skype.com, 2008

[6] Lennox,J., Schulzrinne,H., "A Protocol for Reliable Decentralized Conferencing", International Workshop on Network and Operating System Support for Digital Audio and Video, 2003

[7] M. Radenkosvic and C. GreenHalgh. Multi-party Distributed Audio Service with TCP Fairness. Proc. of ACM Multimedia 2002, Juan-les-Pins, France, Dec. 2002.

[8] iVisit : Real Time Multi-party video conferencing and collaboration tools for PC and Mac, www.ivisit.com, 2008

[9] webex : Cisco Web Meeting and Collaboration Solutions, www.webex.com, 2008

[10] Singh, K., Nair, G., Schulzrinne, H., "Centralized conferencing using SIP", In Internet Telephony Workshop, 2001

[11] Koskelainen, P., Schulzrinne, H., Wu,X., "A SIP-based Conference Control Framework", International Workshop on Network and Operating System Support for Digital Audio and Video, pp. 53-61, 2002.

[12] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "SCRIBE: A Large-Scale and Decentralized Application-Level Multicast Infrastructure" Proc. ACM SIGCOMM '01, Aug. 2001.

[13] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-Bandwidth Multicast in Cooperative Environments," Proc. 19th ACM Symp. Operating Systems Principles (SOSP '03), Oct. 2003.

[14] J.W. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed Content Delivery Across Adaptive Overlay Networks," IEEE/ACM Trans. Networking, vol. 12, no. 5, pp. 767-780, Oct. 2004.

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

[16] 3GPP: TS 23.228: IP Multimedia Subsystem (IMS) (Stage 2) Release.5,2002,

[17] Rosenberg, J., Peterson, J., Schulzrinne, H., G. Camarillo, "Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)", IETF RFC 3725, 2004.

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

[19] Handley, M., Jacobson, V.: SDP: Session Description Protocol, IETF RFC 2327 (1998)