Models for Multimedia Conference between Browsers based on WebRTC

Wajdi Elleuch
Dep. Computer Science and Applied Mathematics
National Engineering School of Sfax
University of Sfax – Sfax - Tunisia
Wajdi.Elleuch@ieee.org

Abstract— **Several models and architectures are proposed to support multiparty communication over IP. Even if some of these solutions support large number of participants using a conference server, solutions that base their media distribution on peers are adapted to support only small conferencing groups. Deploying multimedia communication based on WebRTC API for more than browser-to-browser connection is not inherently supported and creates new challenge in terms of media stream distribution as well as membership and conferencing operation control overhead. Many conference models presented in this paper can support multi-party communication between end-points over an IP networks. Otherwise, the use of WebRTC technology require direct P2P media negotiation between participants as well as signaling interaction with the Server to manage the conference. This paper introduces two specific conference models adapted to support WebRTC communication between browsers for both small scale and large scale conference. The proposed abstract protocols included in this work implements conference creation and browser joining/leaving and can be mapped using SIP/DSP primitives.**

Keywords— VoIP; WebRTC; streaming application; multi-party communication; media distribution; conference models; tree-based distribution; fully-coupled; conference control

I. INTRODUCTION

The last few years have seen new platform rise for deployment of real life communication services: The browser-embedded application, or "Web application". Among these applications, the WebRTC [1] received a great interest since this API is inherently supported by a lot of new versions of common browsers i.e. *Google Chrome* and *Mozilla Firefox*. WebRTC, that is based on HTML5 web communication, holds *PeerConnection*, *Media Stream* and *DataChannels* components API that can be combined to establish direct P2P multimedia communication between browsing peers. Unfortunately, the current version of the WebRTC API have been designed to support only browser-to-browser communication. WebRTC for "multi-browser" communication is not inherently supported especially for conference models that spread media load over participating peers/browsers.

Current research [2], interested by using WebRTC technology within multi-party conference, describes system that enables centralized media distribution based on media server that performs mixing/distribution operations. Unfortunately, this system presents some limitations to support large number of simultaneous conferences since conferencing activities is limited by the available bandwidth and computational power from the media server side.

On the first part of this paper we explore some existing conference models and analyze different media topologies that enable multi-party communication between participants within IP based networks. The second part of this paper discuss the integration of WebRTC technology on the proposed models and describe media and signaling topologies that can be used to create small and large conferencing groups. The next sections detail aspects related to our abstract message protocol used by WebRTC Browers to support both Voice (VoIP) and Video (MMoIP) communication services. The last section describes the implementation of membership operations as well as conference management mechanism and explains how to map our protocol primitive's using the standardized SIP/SDP protocols.

II. EXPLORING SOME EXISTING CONFERENCE MODELS

A. Centralized model

In this model, dialog signaling and media mixing are both carried out by only one user agent i.e. the *Conference Focus* (CF). This CF takes media from users who participate on the conference, mixes them, and sends out separately the appropriately mixed stream to each participant. Two different conference models can be created depending on the resource that takes the CF vocation: The CF could be one of the N participants or a dedicated conference server. This "End-System Mixing" is considered as the basic model that support small group of conferencing participant. It is specially characterized by the facility and the popularity of its implementation. Some VoIP provider like Skype [3] implements this model. Otherwise, the second model (Media-server Mixing) is proposed to support larger conference group and run media mixing load on the Server side since this server offer best computational power and bandwidth than a single "basic" participant. Actual large scale MVoIP solutions use this approach and some commercial MVoIP products, like IVISIT or WEBEX are ready to support up to one hundred of simultaneous users within the same conference. In [2], the conference server is called MCU (Multipoint Control Unit) and designed to support WebRTC based browsers. In both models,

978-1-4799-0428-0/13/$31.00 ©2013 IEEE

the CF of such centralized approach plays the role of a media links with every participant and sends separately the mixed flow generated by mixer instances that are loaded locally.

B. Fully-meshed model

To make the conference independent from the departure of the CF, distributed signaling and administration model can be used to support small group of multi-party communication. In this approach, every endpoint directly communicates with every other one. All the parties in the conference are "equal" and no participant is topologically special or has any additional rights or abilities beyond those of the others. Any conference member can, at any time, invite another user to participate in conference. Also, any user can ask to join the conference by sending request to one of the participants. Similarly, any member of the conference can drop out at any time, without affecting the remaining conference participants.

In this "fully-coupled" model presented in [4], abstract protocol messages are proposed to enable conference creation and membership management. This same work shows that introduced abstract messages can be mapped to SIP primitives using additional optional headers.

C. Centralised signaling and fully-coupled media

On the same time where fully-coupled model is more robust and can resist to any participant departure or failure, the use of decentralized signaling approach should make the conference administration more complex. In many cases, it's important to have one or many "super" conference members that hold the conference moderator privileges. These privileges will allow conference creation, modification and termination as well as conference administration by applying conference policy when it comes to manage participant adding/removing operations for example. Conference policy rules can include a black list, pre-authorized participant, conference creation in ad-hoc mode, etc. In all cases, having a single view of these rules is very important and should make conference management more trivial while enforcing conference access security.

We distinguish two models that implement a fully-coupled media distribution where the signaling management is kept centralized on the CF end-point (that can be one of the participant or based on dedicated conference server).

D. Centralised signaling and tightly-coupled media

Fully distributing media require from each participant within N-party conference to mix N-1 media flow and to send N-1 media flow. This topology is not usually adapted for handheld devices with limited computation power and bandwidth. Another model that distribute media in tree-based model can be proposed while the signaling part of the communication is centralized around the CF. This model take in consideration the fact that conference participant are not equal in terms of network characteristics (bandwidth) and hardware capabilities (computational power). Therefore, participants should not support the same media load within the same N-party conference. This tightly model is using the tree-based distribution and affect media mixing/distribution to a set of selected end-points while the other participant will be connected to the conference in "heavy" way without having to mix or to distribute media flow.

III. CONFERENCE MODEL FOR WEBRTC BASED BROWSERS

A. Signaling and conference administration using WebRTC

WebRTC API is inherently integrated on lot of Web Browsers and strongly coupled with HTML 5 technology to enable signaling exchange between communicating participants over Internet using mainly WebSocket technology [5]. The HTML 5 WebSocket specification defines a single-socket full-duplex connection for pushing and pulling information between the browser and Web Server. Servers are deployed to play an essential role on the communication setup and should be contacted to handle media offer/answer exchange between participants. Moreover, even if WebRTC based browsers can communicate directly within a conference using the PeerConnection function, it remains necessary to use Web Server that support conference required operations like conference creation/destroying, participant joining/leaving or even removal. In addition, different scenarios for joining the conference should also be supported as the dial-in and dial-out mode.

Conference models based on fully distributed approach, i.e. Fully-Meshed peers and self mixing model require from WebRTC browser to create and manage different HTTP sessions over WebSockets. In this model, N-1 different Web sessions should be created for every participant. Since each HTTP session requires new browser instance, e.g. new window or new tab, this will make user-interface more complex to manage the N-1 participants specially for conferences with N>3. We can conclude that conference administration and signaling should be kept centralized around only one machine for the case of WebRTC browsers.

While choosing the appropriate conference model, it is important to conserve the WebRTC advantage of keeping participant browsers independent from any other plug-in. User should be able to participate on the conference and enjoy all features without having to install any package or deploying supplementary signaling protocol like SIP, Jingle or other. Centralized signaling based models, presented on previous section, require from the CF to be implemented on separated Web Server that support an extended software application adapted to handle media offer/answer between participating browsers. So that, Conference Focus role should be assumed by a dedicated server instead of one of the participating end-system.

B. Media flow distribution using WebRTC

The most actual researches on providing conference service for WebRTC browser are focalized on the centralized model that deploy a conference media server for mixing and distributing media to all participant. Such system presents a major drawback since its deployment and maintenance 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 [6,7]. Such solution will introduce additional cost to be deployed and to maintain large number of media servers. These requirements

limit the use of the conference media server for a free use by Internet community. As alternative solution, media distribution topology can be used respectively to support small-group and large group for WebRTC based browsers.

IV. Deploying WebRTC based Browsers in small conferencing group

A. General View

The model where media load is fully coupled and signaling in centralized on the web server (as shown on Fig.1) should comply with small scale conference based on WebRTC browsers since each browser in conference with N participants will create very limited number of media connections. These connections can be managed within the same HTTP session established with the Web Server and using adapted JavaScript code. Each node will receive N-1 incoming flows and send N-1 outgoing flows. Browsers based on WebRTC can use low-rate speech codec like iLBC that consume a bandwidth range of 13 to 15 kbps [1]. The conference size should be less than 10 participants to enable all internet connected devices to use the application but also to freely participate on the conference as active speaker.

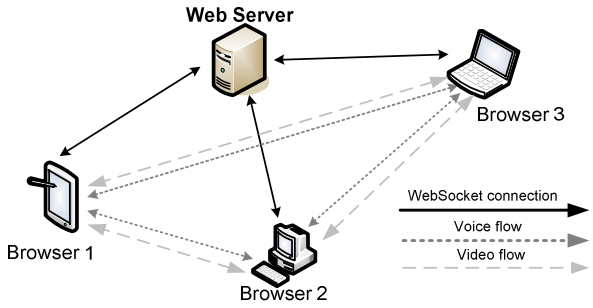

Fig. 1. WebRTC based Browser interconnection for small scale conference

B. Signaling mechanism

WebSocket based connection create P2P bi-directional communication between the Web Server and each active conference participant. To join conference, Browser should send HTTP request using the appropriate URI. This URI can be obtained using email, Instant messaging system and phone or even by publishing it publically on forums or social web sites. The URI concern the specific conference room and HTTP request will include related information about current participant membership. In general, the Web Server that administrates such conference should support some operations as:

- Initialization: Manage the first accessed participant by creating conference room within the server i.e. generate random (and unique) number to identify the conference as the case of the "apprtc.appsport.com" application Demo.
- Participant joining: By handling both media offer and answer negotiation between participants. Web Server can use the Third Party Call Control philosophy (used by SIP protocol in [8]).
- Dynamic Web content: the JavaScript code to be sent to each participant will be adapted to each conference characteristic i.e. third participant should receive and execute a Java Script code that force sending of two media offers to the server. Each offer will be redirected to the corresponding participant.
- Support participant departure/failure: Any departure or failure should be detected (using the HTML link) and in this case server should notify the remaining participant.
- Apply Conference policy: A policy can be defined by the conference creator to manage access rules (e.g. use of black list, predefined conference members, etc.), conference life period, etc.

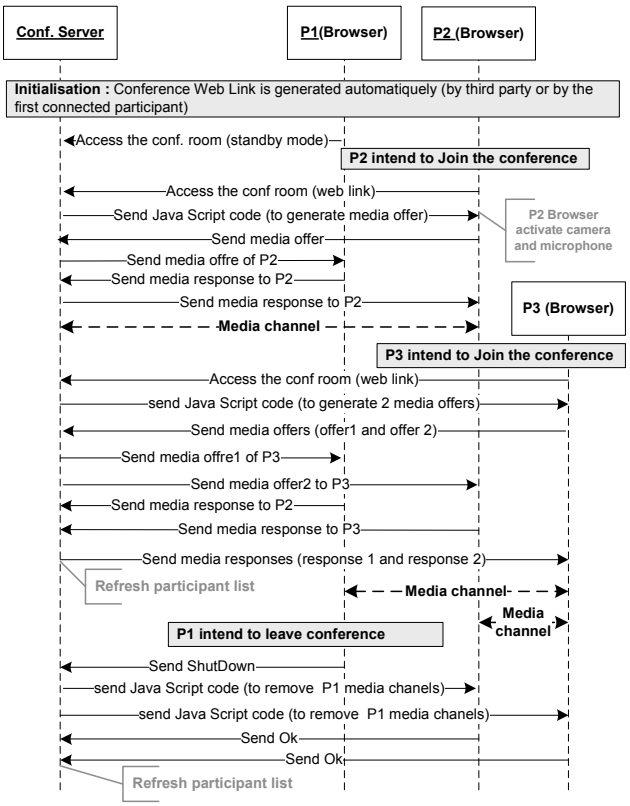

Fig. 2. Example of some signaling operations to manage fully-coupled conference

Fig.2 shows an example of some operations to be implemented by the signaling mechanism. We used simplified and comprehensive protocol denomination.

V. Deploying WebRTC based Browsers in large conferencing group

A. General View

In substitution to centralized and fully-coupled media flows, Application Layer Multicast (ALM) end-points based solution for large scale conference has been proposed [9-11].

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 use media flows supplied by its parents and migrate within the constructed tree to minimize the redundant transmission on physical links. While media processing uses dynamic tree-shaped decentralized approach, the control network is based on a topology centralized around the Web Server. Each user that joins a conference can offer its media processing service to the administrator that controls media distribution using the established Web Sockets and HTML 5 technology by remotely establishing, updating and removing media sessions between participants. The proposed model can be dynamically constructed depending on the user activity (in talk on in only listen mode) and on the user preference to offer mixing service to other, mixing only for himself of let other participant mixing for him. This model can be used by WebRTC browsers to create both VoIP and MMoIP conferences as detailed in following sections.

B. *Case of VoIP conference*

In case of VoIP large scale conference based on WebRTC, our proposed model build two different meshed networks that enable voice audio distribution between browser and general conference control. The media network ensures audio flows delivery in tree-shaped topology as shown on Fig.3. Note that we will designate participating browser end-points as Nodes on the rest of this paper to facilitate system presentation and to comply with usual P2P systems notation. This network uses three different media roles components: the Mixer/Distributor Node (MDN), the Distributor Node (DN) and the Leaf Node (LN).

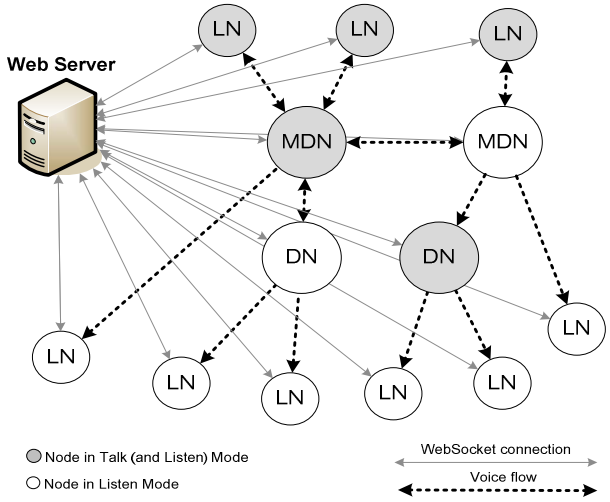

Fig. 3. WebRTC browser interconnection for large scale VoIP conference

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, DN nodes are there who receive media flow from their parents, play it locally and distribute it to their children. 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 proposed architecture will also take in consideration the activity of each user. Some users prefer to participate as speaker when some other will be just a listener.

C. *Case of MMoIP conference*

The use of the same architecture as VoIP to achieve video based content is not possible since video streams could not be mixed. Participant nodes that participate on the media distribution could be either DN (Distributer Node) or LN (Leaf Node). Different video streams should be achieved using different and parallel created channels between participant nodes as illustrated in Fig.4. By using RTCDataChannel interfaces within the Peer-to-peer Data API, the DN node will be able to act as "proxy" to send/receive media. At any time, DN can act as "presenter" in the conference (the case of DN2 in Fig 8), and in that case, video stream will be sent to the neighbor nodes. Create a topology that enables transmission of both video and audio within the same conference should be designed in manner to reproduce the same media topology and by considering VoIP MDN as simple video DN.

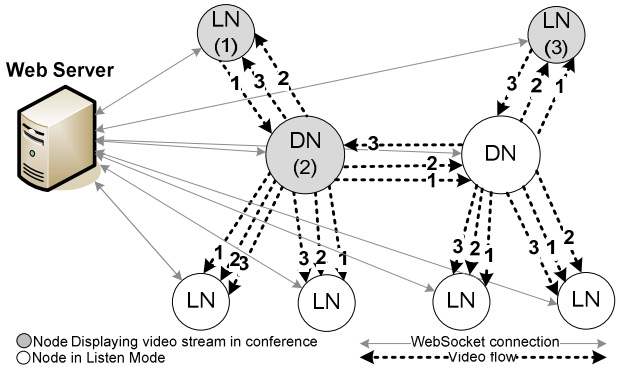

Fig. 4. Media flow distribution

VI. Supporting Membership Operations in Large Scale Conference Based on WebRTC

A. *Introduction*

In this section, we focus on the study of the large scale VoIP conference model adapted to WebRTC browsers. 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 and coherence. 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*

There's a lot of ways that can be used to establish conference i.e dial-in, Dial-out, ad-hoc, scheduled conference, etc. Since WebRTC don't supply any mechanism to register and to identify browsers, we will consider that the only way to add user in our system is the use of the dial-in mode. Thus, the new user should initiate the call by sending the JOIN_REQ

message with its media offer as shown on message (1) of the Fig.5. The Web Server redirects the media offer from the New User (NU) to the available and appropriate MDN. The media answer supplied by MDN is included on the ADD_MEDIA_Ok response message sent to the Web Server and redirected to the NU using JOIN_REP response (message (4). Since each media offer/answer contains the address of its original supplier, it become possible to establish direct media flow between NU and the selected MDN. The signaling dialogs of this communication remain completely managed and controlled by the Web Server.

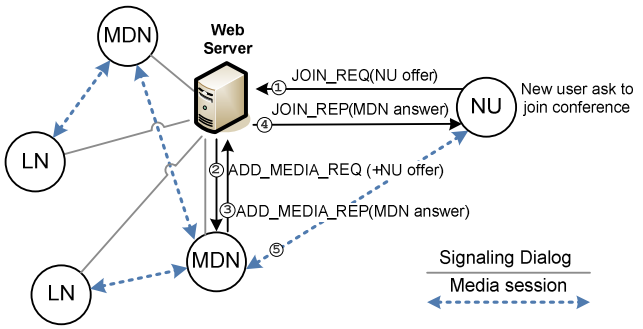

Fig. 5. Message Flow for user adding scenario

C. LN member departure or failure

LN can at any time leave conference, in dial-in mode, by sending LEAVE_REQ (message (1)) to the Web Server as illustrated in Figure 6. The dial-out mode is used when the Web Server chooses to disconnect LN from conference. On the two modes, the Web Server 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 the Web Server.

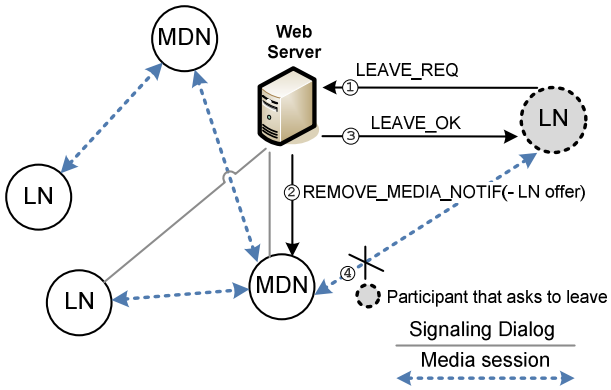

Fig. 6. Message Flow for LN departure scenario

In this case, media flow between LN and its parent will be automatically disconnected. The parent of the LN will receive OnRemoveStream event from the PeerConnectionObserver. On the same case, the Web Server, by holding HTTP connection with LN, will receives "dialog-disconnected" signaling event notification from Web Socket stack. Accordingly, it becomes easy for the Web Server and for LN's parent to detect such member fast departure or failure and to subsequently update their states. Even if messages (2) that notify MDN about the LN departure are not required in this scenario, it remains appropriate to use it to guarantee conference coherence.

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 the Web Server is aware about this departure, (the dial-in or dial-out mode is used), media tree can be reconstructed proactively. In proactive approach, the Web Server will redirect the affected children to new parents before accepting this departure. In that case, the media disruption should be minimized. If MDN leaves without notification, media reconstruction will launched in reactive mode by the Web Server as soon as "dialog-disconnected" event is received from the WebSocket stack. Fig. 7 illustrates the required message flow, based on the UPDATE_MEDIA_REQ/REP/ACK and ADD_MEDIA_REQ /REP mechanisms, used for this scenario.

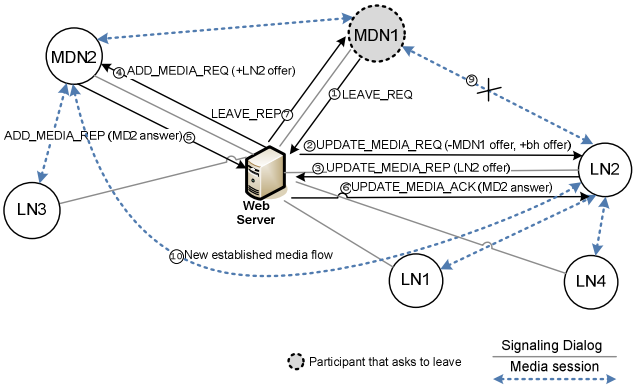

Fig. 7. Message Flow for MDN departure scenario

VII. HTML 5 Implementation

A. Mapping membership operation protocol

The message protocol defined and used on message flow diagrams of previous sections should be implemented by the Web Server using the HTML 5 technology and Java script generated code. Each conference should be uniquely identified on the server by an ID. This ID should be generated as soon as the first participant will contact the Web Server. The use of Get or Post method within HTML protocol will facilitate the use of many parameters within the conference room. The Web server should also adapt the Java script code to be sent to every participant, depending on the operation to perform. WebRTC browsers should also have an "improved" version of WebRTC stack that enable MDN participant to mix audio flow before sending them.

B. Implementing Media offer/answer

Our solution uses SDP protocol to describe media offer/answer exchange mechanism. SDP content, encapsulated in signaling control message, includes general description of the session, the creation time and the used Media. Our

messaging protocol that implements media flow connection/update follows the same concept as the Third Party Call Control Flow I as specified in [8]. 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 JOIN_REQ message already shown on Fig.5, 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 payloads 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. A participant that wants to change its values should contact the Web Server 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 Conference management

Each conference is uniquely identified by SIP URI address created by the Web Server. AN can contact a SIP registrar server to associate the conference address to its network address. Tierce mechanism ca automatically publishes registered conference URL to public web pages. Conference description can use XML based model to describe participants and related media tree. Each media node will include key information parameters about its media interconnection on the tree. At every membership alteration, the Web Server should update XML file. An example of XML structure will be:

```
<Conf_description>
 <headers>
       <Conf_URL> … </Conf_URL>
       <Conf_title> … </Conf_title>
       <Max_participants> … </Max_participants>
       <Creation_date> … </Creation_date>
       …
 </headers>
 <media_tree>
       <Node address =… , input_degree=… , outpu_degree=…>
        <Node address =… , relation=input/output/input-output>
        <Node address =… , relation=input/output/input-output>
       . . .
     </Node>

 </media_tree>
</Conf_description>
```

E. 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 other parameters 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
```

VIII. CONCLUSION

In this work, we introduced new multimedia conference models that comply with WebRTC based browsers technology. We identified the different components that should be integrated to support conference control/administration and media tree distribution among participants in both VoIP and MMoIP services depending on the conference scale. We also defined a mechanism that implement media offer/answer and speech/media assignment floors control. Member adding/removing scenarios are discussed and implemented on the proposed models using abstract message protocol to preserve media distribution coherence.

REFERENCES

[1] WebRTC, Official Web Site : www.webrtc.org, 2013.

[2] Rodríguez, P., Cerviño, J., Trajkovska, I., & Salvachúa, J. "Advanced Videoconferencing Services Based on WebRTC", In proceeding of : IADIS Multi Conference on Computer Science and Information Systems, March, 2013

[3] Baset, S. A., & Schulzrinne, H. "An analysis of the skype peer-to-peer internet telephony protocol" IEEE infocom, 2006.

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

[5] WebSocket, Web Site : http://www.websocket.org

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

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

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

[9] Elleuch, W., Houle, A.C, "Multiparty Voice over IP (MVoIP) Peer-based System for Large-scale Conference Support", 4th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, 2008.

[10] Elleuch, W., Houle, A.C, "SIP-based Protocol for P2P Large-scale Multiparty VoIP (MVoIP) Conference Support", 6th Annual IEEE Consumer Communications & Networking Conference, 2009.

[11] J.W. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed Content Delivery Across Adaptive Overlay Networks," IEEE/ACM Trans. Networking, 2004