

A Distributed Multimedia Conferencing Based on SIP

Junchao Li*, Ruifeng Guo**, Xiuwu Zhang*

**School of Computer Science and Technology, University of Science and Technology of China, Hefei, China*

*** Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang, China*

***Shenyang Golding NC Tech.Co., Ltd., Shenyang, China*

lijch@mail.ustc.edu.cn, grf@sict.ac.cn, zhangxiuwu@sict.ac.cn

Abstract— Technology of multimedia communications on network is rapidly developing today. SIP (Session Initiation Protocol), which plays an important role in 3G standards, is well applied in multimedia conferencing systems. Traditionally, most SIP-based conferencing systems have been limited in scalability due to their tightly coupled centralized structures. However, distributed conferencing based on SIP is still in the theory phase. The goal of this paper is to take out a practicable solution that can meet the requirement of large scale multi-party communications. We analyze the drawbacks of existing conferencing models, and design a more practical distributed architecture for a large scale conferencing service. For high scalability, coordinated processors are settled in different areas to maintain conferences, and associated mixers are located for media treatment. There are some high speed corresponding channels among processors in different networks for communication and synchronization. We also propose a backup system to keep up meetings when an abrupt breakdown occurs on some server.

Keywords— multimedia conferencing; distributed architecture; backup system; SIP

I. INTRODUCTION

The fast development of communications and computing technology has provided the basis for multimedia conferencing on the Internet. With the function of transmitting audio, video, text, graphics and other examples of information, multimedia conferencing on network breaks the limit of human communications on the earth.

Multimedia conferencing products based on H.323 [1] are more mature in market today, with regard to balance workload, manage services and express session description language. A series of compensatory standards are also provided to meet the various requirements of conference. These standards advised by ITU-T follow the traditional telephony signal model. They are complicated and hard to extend. SIP [2] plays a core role in IMS (IP Multimedia Subsystem) [3], which is a key component in 3G architecture standards. With the issuance of 3G licenses in worldwide, people gradually realize the excellent performance of SIP:

lightweight, flexibility, extension and many more. SIP is certainly become a better choice to be the basic call signalling protocol for multimedia conferencing. It should be extended to keep up with the distinguished functionality in H.323.

SIPPING working group and XCON working group, which both belong to IETF, are doing research of multimedia conferencing frameworks separately. Although they take out distinct solutions, these two working groups both adopt a tightly coupled centralized architecture. The most prominent characteristic of this structure is that all functional parts are close integrated in the only conference server. We are able to discover that the advantages of this tightly structure are powerful control, effective management and efficient security. However, when the number of participants increases, the burden of a single server would become too heavy to keep running. Besides, the limited bandwidth is another kind of bondage. If the total reception content of media packets from participants exceeds the server's processing capability or network's traffic capacity, a block of media treatment or a loss of media packet would happen.

A solution that can enhance the extensive function of conferencing system to accommodate very large numbers of participants is imperative. We separate the media mixer from the original server to treat media independently. The remainder includes a focus and some functional modules. It takes charge of conference management and resource assignment. We here call the remainder server a conferencing processor. It sends commands to mixer by a MCP (Media Control Protocol), which can exploit the XML mode, instead of a policy strategy [4] advised by SIPPING working group. Another choice of MCP is that we can use H.248, which is adopted by the Mp reference point [3] in IMS. We can add more mixers into this system to reduce the workload of single mixer. Then a selecting policy would be added in media control function to decide which mixer to use. This policy mainly base on the load ratio of mixers.

There are many ISPs operating various internet services in worldwide. The dissimilar quality of services provided by these ISPs causes the internet speed uneven. The flow speed between two networks is slow and unstable. To improve the sound quality, a proposition is to locate mixers in different ISPs'

different networks separately. Every mixer connects with local participants using RTP (Real-time Transport Protocol) [5]. Communications among distinct areas can be achieved by high speed media stream channels among mixers, which can make media traffic more smooth, fast and stable.

The only conferencing processor which takes charge of management task would become a limitation of scalability, when participants increase to a huge number. We also exploit a distribution structure for processor. The location tactic is the same as mixers.

Let us image the situation that a conference server suddenly breaks down during a meeting. We suppose that the maintenance men find out the reason at the first time. Then they resolve the problem and restart the server immediately. A new meeting begins to continue the last meeting. These steps may take too long time. The participants may think poorly of this conferencing product. So a backup system is proposed to cope with emergency. It should continue the meeting as fast as possible.

According to the steps described above, a broad profile of a distributed conferencing system emerges. The rest of this paper is structured as follows. In Section II, we introduce related work with regard to existing frameworks. In Section III, we present a brief architecture for practical distributed conferencing. In Section IV, we analyze the detail structure of server, the flow procedures, and compare with previous frameworks. Finally, the conclusion and outline of future research are presented in Section V.

II. RELATED WORK

Although SIPPING and XCON working groups both exploit tightly coupled model, they put forward different architectures. The SIPPING group provides a framework [4] using extended SIP protocol as call signalling protocol for whole conference management. The highlight of this design is conference policy, which contains the rules that guide the decision-making process of the focus for the management of various conference requests from the participants. An open source software called Asterisk [6] implements a rough model to achieve this framework.

The XCON conferencing framework [7] is independent of any signalling protocol, with a more particular structure. SIP is also well recommended here. This structure advises a set of standard protocols, which are complementary to the call signalling protocol, for building advanced conferencing applications. Confiance [8] is an open source software which follows the design of XCON.

The concept of conference mixer network was firstly proposed in a remarkable research work [15]. The authors took out a policy-based distributed conferencing structure, which is a development and extension of the framework of SIPPING group. The innovative conformation and methods inspire us a lot. However, the weak control functions, the use of non-standard protocols, poor compatibility and robustness are its short points.

III. DISTRIBUTED CONFERENCING FRAMEWORK

A. Limitation of Centralized Conferencing

The centralized conferencing framework does not suit the management of large-scale multimedia conferencing service, because of the capability of single server and the capacity of bandwidth. Efficient solution for enlarging scalability is urgently required. For a traditional centralized conferencing, we define that,

- n is the number of participants in a single mixer,
- R_i is the generation rate of media packets from the i^{th} participant,
- S_i is the size of a media packet from the i^{th} participant,
- $B_{network}$ is the network bandwidth of the mixer,
- P_{mixer} is the processing capability of a mixer.

We can roughly get equations as follows,

$$\sum_{i=0}^n R_i S_i \leq B_{network} \quad (1)$$

$$\sum_{i=0}^n R_i S_i \leq P_{mixer} \quad (2)$$

Equation (1) shows the limitation of bandwidth. The size of total media packets from participants per second, which flow into a signal mixer, would not exceed the bandwidth. Equation (2) depicts the capability of a mixer's processing. So the maximum number of participants in one mixer is limited.

B. Reformative Way

To accommodate large numbers of participants, a traditional centralized conferencing server is divided to a conferencing processor and a media mixer. With the separation of server, we can exploit plural mixers to share the workload and bandwidth of the single server. The conference size can become times larger.

A useful method to enhance media quality is that every participant communicates with his adjacent mixer. The word "adjacent" means that this chosen mixer has the least transmission delay from the participant. The mixer may be in the same domain or same ISP's network as the participant. For example, a roaming user can connect to the server in the visited network. Connections among mixers are through the S2S (server to server) way. As servers have more stable network circumstance and wider bandwidth than common participants, and have special high speed channel among them, the S2S method makes remote conversation more fluent and stable. As a result, mixers connect with each other to form a media mesh, which treats media streams in a holistic way. We also exploit the above method for conferencing processor. Thus all processors establish a processor mesh to manage conferences.

C. Distributed Framework

The integrated panorama of our distributed conferencing system is shown in Fig.1. UA, UB, ..., UG are user clients in different networks. They connect to their adjacent networks, which must have conference service, via proxies. The proxy is the access point to a network, which accepts requests, chooses services and forwards messages. P means processor and M means mixer in the picture. A processor and a mixer compose a

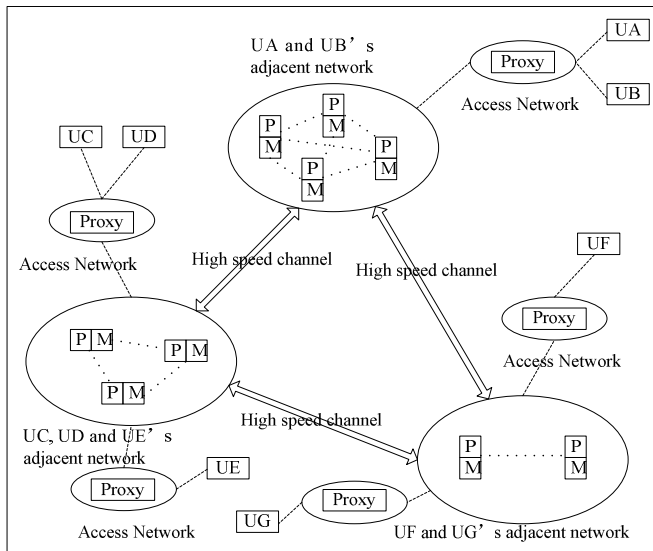


Figure 1. Distributed conferencing framework

local conference pair. There can be one or plural pairs in one ISP network. The number is decided by the business situation. Servers in different networks correspond with each other through special high speed channels.

This model has strong compatibility. Servers can be settled in kinds of networks. Especially, they can be located in IMS structure. Client is implemented in UE (user equipment), which can be a normal SIP terminal, a soft client which support the conference control and floor function, or just a normal Phone. Proxy can be a P-CSCF (proxy-call session control function), which is the first contact point within the IMS core network. Proxy can be a SGW (signalling gateway function), which connects clients from PSTN or CS domain. Proxy also transmits and transcodes media streams like a MGW (media gateway function). Processor can be split to many parts: S-CSCF (serving-call session control function), AS (application server) and MRFC (media resource function control) in IMS core network. Mixer acts the role of MRFP (media resource function processing).

Our framework is a development based on XCON conferencing. Participants communicate with servers through a set of protocols: SIP is used as call signalling protocol to maintain conference sessions; CCMP (Centralized

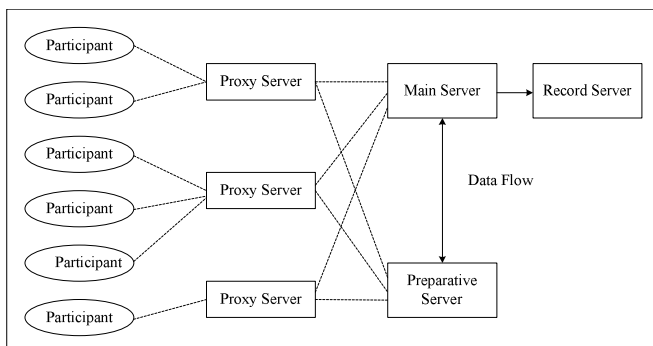


Figure 2. Backup system

Conferencing Manipulation Protocol) [9] is a protocol for conference control; BFCP (Binary Floor Control Protocol) [10] deals with source assignment; Notify [11] method exposes conference states to specific participants. Processors contact with each other by an inter-processor protocol, which can be achieved by XMPP (Extensible Messaging and Presence Protocol [12]). A local processor controls its corresponding local mixer via MCP.

D. Backup System

Traditional backup systems copy data from main server into an additional server periodically. If the main server breaks down or losses data, the data stored in the additional server would be copied back to the main one. It takes some time to resume the functionality of main server. However, participants may not have enough patience to wait for the conferencing server recovering. It is terrible that an important conference stop too long time.

A preparative server should parallel exist with a main conferencing server. When an accident of main server occurs, the connection should be altered to the preparative server. Fig. 2 shows the predigest structure of backup system. A main conferencing server backup records of finished conferences to the corresponding record server. It copies all information of current meetings to preparative server. When conference state changes, it sends up-to-date information to the preparative server immediately.

Proxy server here doesn't only act the role of access point of network, but also knows the states of servers and decides which server to connect. If there is something wrong with the main server, the proxy would forward participants to the preparative server. When the main server is repaired, data in preparative server would be copied back to main server, and the connections would also turn back. One-to-one backup is a waste of resource, since breakdown is a small probability event. We can divide all conferencing servers in a network to several groups, and add a preparative processor and a preparative mixer for one group.

If some proxy breaks down, all participants who is in the charge of it would be influenced. They need to change a proxy to re-connect to the meeting. We can add a proxy choice mechanism in the terminal. The terminal can automatically choose an active proxy which has the smallest net delay to connect. This is also an important function for a roaming participant when he changes position using a mobile terminal. Another method is adding an automatically dial out mechanism in the processor. If the processor finds the proxy break down, it should INVITE these dropped participants through another living proxy. This mechanism is useful to the normal terminals which have no proxy choice function.

IV. ARCHITECTURE ANALYSIS

In this section, we firstly analyze the flow procedures of distributed conferencing. Subsequently, we illustrate the detail architecture of server. Then we introduce the process of conference management. Finally, we compare our distributed framework to other works.

A. Flow Procedures

We omit proxy and backup system in the analysis of flow procedures to predigest flow chart.

1) Create Conference: All pairs of processor and mixer have the same status in network. User who wants to begin a new meeting should first send a CCMP request of creating conference to a processor server in the same network through his adjacent proxy. He would automatically become the moderator of this meeting. There is only one moderator in a meeting who has highest authority to control the conference and manage participants. The position of moderator can be transferred from the current moderator to a specific participant. After creation, the processor publishes conference information to the processor mesh to allow the other processor servers to get the information of the new conference. When a remote processor gets this message, it builds a data structure of this conference in its database. When the first participant who belongs to this remote network connects to the processor to join in the meeting, a conference room in this processor would be set up.

2) Add User: Fig. 3 illustrates the flow procedures of a new participant joining into the distributed conference. Firstly, the user who wants to join a specific conference sends an INVITE request to his adjacent processor via transmission of proxy. When they get connection after negotiation, the local processor sends the “Add User” message of inter-processor protocol to the moderator’s adjacent processor. This processor then publishes the changed information of conference state of “a new user applies to join this conference” to the moderator of this meeting. When the moderator gets this notification, he can agree the application or reject it by CCMP. If he rejects it, his adjacent processor would ask the local processor to send BYE to the new user. Then session finishes. Otherwise, as the figure shows, an inter-processor protocol of “User

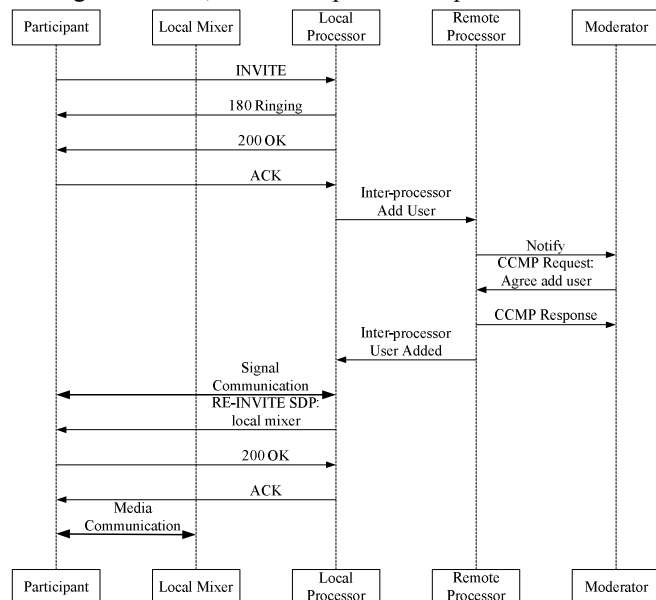


Figure 3. Flow procedures of joining conference

Added” would send back to local processor, which shows the moderator’s permission of access. At this time, a signal communication is formally established between participant and his adjacent processor server. As the mixer is separated from original server, the processor transmits a RE-INVITE request to the participant with SDP [13] containing media information. This message gives mixer’s URI and supporting media types to participants. The media communication is established after negotiation. If we add other applications, such as chat room and remote desktop, a RE-INVITE message with SDP containing new application’s information is needed. The moderate can also use CCMP to ask the processor to INVITE a new user to join the conference. It is called a “dial-out” way, which has opposite direct of flow procedures.

B. Detail Structure

The detail processor functional structure is shown in Fig. 4. This framework eliminates mixer out from the original server, and add a dispatcher and a backup peer into the processor. In the picture, we can see the client connect with its local servers. This approach avoids the unstable remote connection. As servers have wider bandwidth and more careful maintenance, remote connections among servers are more credible.

Firstly, we introduce the connections between server and client. Focus establishes relationship with client by maintaining a SIP dialog as call signalling protocol. Conference control server connects with client by CCMP, which is used for data manipulation and state retrieval. BFCP is in charge of source assignment, which provides the interface between floor control server and client. Notification server notifies up-to-date conference’s and users’ states to client.

Then we illustrate protocols between servers. Processor, preparative and record server all have a backup peer. Processor copies data to preparative server when conference states alter, and backups records of finished conferences to the record server though FTP (File Transfer Protocol). Preparative server copies data back when processor recovers from a breakdown. Record server send records back when someone wants to see the replay of a conference. Mixer also does backup actions like processor, which is not shown in the figure. Processors communicate among mesh via XMPP. Mixer mesh treats media in a holistic way. A local mixer blends all media streams from local participants, and forms a stream *A*. It transmits a copy of stream *A* to other mixers. It receives all other mixers streams,

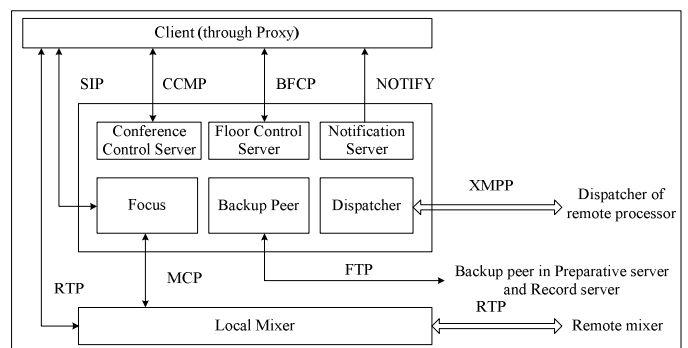


Figure 4. Detail of processor structure

and blends them with stream *A*. This step forms stream *B*. Then stream *B* subtracts the participant's stream, and send the last result to him. This is the finally stream the participant receives.

C. Conference Management

All processors have equal status in the structure, but participants don't. The moderator has the highest authority to control the conference, assign resource and manage other participants.

Normal users can use CCMP to retrieve conference information. Besides, the moderator can manage the whole conference and all users via this protocol. For example, he can add a new user, kick a current participant, and change conference mode. His command, which is a CCMP request, is firstly sent to his adjacent processor. Then the local processor forwards the message to processors with regard to this request. Corresponding reactions would happen in these associated processors just like in centralized servers. The results of these actions would feed back to the moderator. The change of conference states would be shown to the authorized participants in the meeting.

Normal participants apply action right, which is also called floor, such as speech and demo right, through BFCP. When a local processor gets a BFCP request from one of its adjacent clients, it would forward the message to the remote processor, which directly connects with the moderator of meeting. Then this processor would show the application to moderator. Moderator can grant or reject this source request. A BFCP response would send the decision back to requested processor. If it is a permission message of speech right in meeting, the processor would ask local mixer to add the participant's stream via MCP. A Notify message would notify all participants about this change. A moderate can also actively give a floor to some participant, or deprive a right from someone. Action rights consume system resource, so we say the moderator assigns resource through BFCP.

D. Comparison

We firstly compare our structure with centralized conference. All functional parts are tightly coupled in a single server in centralized model. It couldn't support very large size conference service, since the process capability and bandwidth capacity of a single server is limited. In our design, a lot of processors and mixers share the burden of processing and transmission. The scalability of distributed conference can be multiple larger. If a participant from different ISP network joins a meeting in centralized server, the communication quality may not be satisfied. However, in our model, the S2S structure makes all remote connections among different areas be realized though special high speed channels among servers, which makes transmission stable and fast. This framework also has excellent dynamic extension. When one processor receives an "add-in" request of a new server, every server in mesh would get this information through XMPP. The number of server is added after permission of the request.

For evaluating the communication quality, transmission comparison between centralized and distributed frameworks is simulated. We know from the above analysis of flow procedures, that distributed structure's signalling model has extra steps. They extend signalling transmission delay in a small degree, but bring a more stable circumstance. As signalling is not very frequent in meetings, the delay is acceptable. The emphasis of our experiment is testing media transmission delay. The experiment theory is proposed by some papers [14], [15]. We set several test clients and servers in different networks, such as China Telecom Network and Unicom Network. We choose several kinds of clients, including hard and soft terminals, such as Cisco Unified SIP phone, Grandstream GXV IP phone, and Minisip. The test points are also located in different cities. The total transmission delay data includes delay in client, delay on road and delay in mixer. Delay in client includes two aspects: the prepare time before the packets send out from client, which includes the time to collect all the samples constituting a packet, and the processing delay to perform operations such as silence detection; the time taken by a participant to process a received mixed media packet before playing it out. Delay on road includes three aspects: time from participant to mixer, time between two mixers and time from mixer to participants. Delay in mixer includes decoding, mixing, coding and packing. Fig. 5 is a histogram showing the transmission delay of different architecture. The left solid bar shows media transmission delay in traditional centralized conferencing framework, and the right hollow bar means delay of distributed conferencing. The media transmission delay of distributed structure is smaller. Its increasing rate is also slower, when the number of participants enlarges. The result provides evidence for our belief that distributed structure shares the workload of media process with many mixers. The process time of every mixer is much shorter than centralized one. Although the media streams transmit through more mixers in distributed architectures, their total time of transmission is shorter.

Then we compare our framework with policy-based distributed conferencing [15]. We get inspiration from this

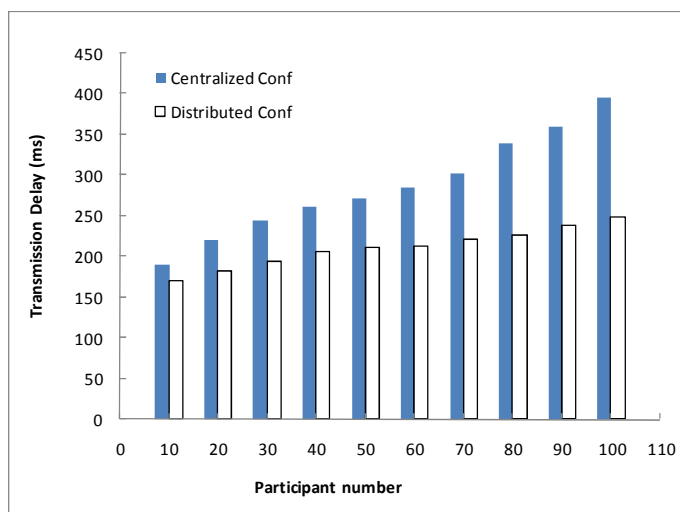


Figure 5. Media transmission delay

remarkable research work and develop a more practical model. The policy-based conference is a development on structure proposed by SIPPING group, which use extended SIP message by authors. The use of non-standard protocols results in poor compatibility. Servers would be located in normal networks. Special terminal should be designed to match the system. Our design is an improvement on XCON structure. As we mentioned in Section II, the XCON framework is more practical and sophisticated, as its functions are delicate and support standards. We adopt standard protocols in our system, and make the structure suit IMS architecture. Thus our system has wonderful compatibility, which can be settled in IMS core networks and other nets. As we use standard protocols, the selection of terminals are great range. A terminal which supports all these protocols can control conference as a moderator. A normal SIP phone or a 3G mobile phone can attend conference according to steps of IMS specification. Normal phone can be participant through the transfer of SGW and MGW. The proxy mechanism is well suitable for roaming users. Participant can switch appropriate proxy to continue meeting when his position changes. In policy-based conferencing system, one of the focuses is selected as a primary focus to manage conference, and its mixers also have layered. The topology of servers is a start or tree structure for a meeting. The policy rules in the primary focus decide the control strategy of the meeting. In our framework, statuses of all similar servers are equal. The topology of servers is full mesh for a meeting. The moderator manage the meeting manually. The advantage of manual control is that we can manage the meeting more handy and flexible. And it is easy to change the manager of conference. Another characteristic of our system is strongly robust because of the backup system. If some server suddenly breaks down, backup system would keep the meetings alive.

V. CONCLUSIONS

In this paper, we presented a novel design for a distributed multimedia conferencing. We introduced the distributed tactic, flow procedure and particular structure. We also analyzed its scalability, stability, robustness, extension, and transmission delay of this architecture. Finally, the result shows our framework is suitable for very large scale communications.

Further research is planed to develop the distributed conferencing by adding a policy of dynamic route. It would reduce the transmission delay to a much smaller degree. The judgment of shortest path and transition via best routes should occur periodically. This policy could be used to acclimatize the system to the protean network environment.

ACKNOWLEDGMENT

The authors would like to thank all the reviewers for their comments and suggestions. The authors wish to appreciate previous researchers whose work helps us a lot. The authors are indebted to friends and members of laboratory for their proofreading of this paper and useful advices.

REFERENCES

- [1] G. ITU-T Rec. H.323, "Packet based Multimedia Communications System," *Telecommunication Standardization Sector of ITU*, July 2003.
- [2] J. Rosenberg, H. Schulzrinne et al., "SIP: Session Initiation Protocol," *RFC 3261*, IETF, June 2002.
- [3] 3GPP TS 23.228 V9.1.0, "IP Multimedia Subsystem (IMS); stage 2 (Release 9)," Sep. 2009.
- [4] J. Rosenberg, "A Framework for Conferencing with the Session Initiation Protocol (SIP)," *RFC 4353*, IETF, Feb. 2006.
- [5] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *RFC 3550*, IETF, July 2003.
- [6] Asterisk: *The Open Source PBX & Telephony Platform*. [Online]. Available: <http://www.asterisk.org>.
- [7] M. Barnes, C. Boulton, and O. Levin, "A Framework for Centralized Conferencing," *RFC 5239*, IETF, June 2008.
- [8] Confiance: *an implementation of the IETF XCON framework and BFCP*. [Online]. Available: <http://sourceforge.net/projects/confiance/>
- [9] M. Barnes, C. Boulton, S P. Romano and H. Schulzrinne, "Centralized Conferencing Manipulation Protocol," *draft-ietf-xcon-ccmp-02 (work in progress)*, IETF, March 2009.
- [10] G. Camarillo, J. Ott, and K. Drage, "The Binary Floor Control Protocol (BFCP)," *RFC 4582*, IETF, Nov. 2006.
- [11] A. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification," *RFC 3265*, IETF, June 2002.
- [12] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," *RFC 3920*, Oct. 2004.
- [13] M. Handley, V. Jacobson, and C. Perkins, "SDP: Session Description Protocol," *RFC 4566*, IETF, July 2006.
- [14] H. Vin, P. Rangan, and S. Ramanathan, "Hierarchical Conferencing Architecture for Inter-Group Multimedia Collaboration," in *Proc. Conf. Org. Comput. Syst.*, vol. 12, Nov. 1991, pp. 43–54.
- [15] Y. Cho, M. Jeong, J. Nah, W. Lee and J. Park, "Policy-Based Distributed Management Architecture for Large-Scale Enterprise Conferencing Service Using SIP," *IEEE JSAC*, vol. 23, Oct. 2005, pp. 1934–49.