ns-2 Implementation of the OMA PoC Control Plane

Jong Min Lee
Dept. of Computer Software Engineering, Dong-Eui University, Busan, Republic of Korea

ABSTRACT : *Recently technologies such as VoIP, VoLTE and VoWLAN have been widely used for the purpose of voice conversation since the proliferation of smart phones. A half-duplex group communication or the push-to-talk (PTT) has been standardized under the Open Mobile Alliance to replace the existing analog/digital TRS or a walkie-talkie service. In this paper, we design and implement the ns-2 module of the OMA PoC control plane which is a signaling protocol for the PTT service. Based on the SIP implementation of Rui Prior, we extend it to simulate the ad hoc PoC session establishment using on-demand session, which is a signaling protocol according to the rules and procedures of RFC 3261 with extended headers including PoC feature tags. Some simulation results have been shown for the verification purpose using the proposed implementation. With this implementation, we expect to perform the extensive simulation study of group communication in various network configuration.*

Keywords: *Control Plane, Group Communication, OMA PoC, Push-To-Talk, SIP*

I. INTRODUCTION

Recently voice over IP (VoIP) has been used widely in lots of Internet applications. Among applications which support VoIP, there are several smart phone applications such as Voxer and TiKL which support the group communication as well as one-to-one communication, which is also known as the push-to-talk (PTT or P2T) [1, 2, 3]. Major communication and computer companies such as Nokia, Samsung Electronics, Qualcomm, Intel and Microsoft have standardized the Push-to-talk over Cellular (PoC) to support group communication under the Open Mobile Alliance (OMA) [4].

The OMA PoC standard is based on the SIP standard, which is an application-level network protocol to support call registration, session invitation and termination *etc.*[5] To study the performance of the OMA PoC standard, we need to develop the network simulator to satisfy its signaling protocol. Rui Prior [6] implemented the SIP signaling protocol based on ns-2.27 [7]. We extend the Rui Prior's work to support the ad-hoc PoC group session with unconfirmed indication which uses on-demand session. It is simpler than other session initiation methods defined in the OMA PoC standard and easy to understand intuitively, which provides a basic measure of the OMA PoC standard consequently. By using the implementation of this paper, we evaluate the network performance of the group session initiation of users in different networks and the same network.

In Section 2, we describe the basic architecture of the Rui Prior's work and the OMA PoC standard. The extended PoC architecture for the ns-2 network simulator will be presented in Section 3 and the performance study using the proposed scheme will be given in Section 4. Finally, we give a conclusion in Section 5.

II. RELATED WORKS

2.1 SIP Implementation of Rui Prior

Rui Prior implemented the SIP signaling protocol based on ns-2.27 [6]. Main functional components are the classes SIPUA and SIPProxy. SIPUA is a logical entity that makes a new SIP request for call setup and responds for the request. SIPProxy is a logical entity that manages the session information between SIPUA's. In the beginning, SIPUA sends a registration request to SIPProxy and then SIPProxy manages the registration information for later call setup.

Fig. 1 shows the class diagram of SIPUA. Both SIPUA and SIPProxy are subclasses of SIPTU, which performs the transaction user (TU) functionality in RFC 3261 [5]. Whenever a TU wants to send a request, it generates a client transaction instance (SIPTransaction), which is passed to the transaction layer, or SIPTransLayer. The class SIPTransLayer manages a list of SIPTransaction's, which are categorized into client non-invite transactions (CltNonINVITETrans) and client invite transactions (CltINVITETrans). Main functionalities of SIPUA is to register itself to SIPProxy and make an INVITE request for call setup according to the SIP call setup procedure.

Fig. 2 shows the class diagram of SIPProxy, which is also a subclass of SIPTU. SIPProxy handles a list of registered entry (RegEntry) as a registered DB for processing call setup requests between two SIPUA's. All SIPUA's should be registered into their own SIPProxy, which can be either only one in the network or all different, before session initiation. To initiate a session, one SIPUA sends an INVITE message to its SIPProxy and then the sender's SIPProxy forwards the INVITE message to the receiver's SIPProxy, which finally forwards the INVITE message to the receiving SIPUA. If the receiving SIPUA accepts the INVITE message, it sends 200 OK message to the sender following the reverse message path through both SIPProxy's. Upon receiving the 200 OK message, the sender generates an ACK message to the receiver to confirm the reception of the 200 OK message. This is called the INVITE/200/ACK three-way handshake. After this, end-point SIPUA's start a media session or a talk burst in the half-duplex mode.

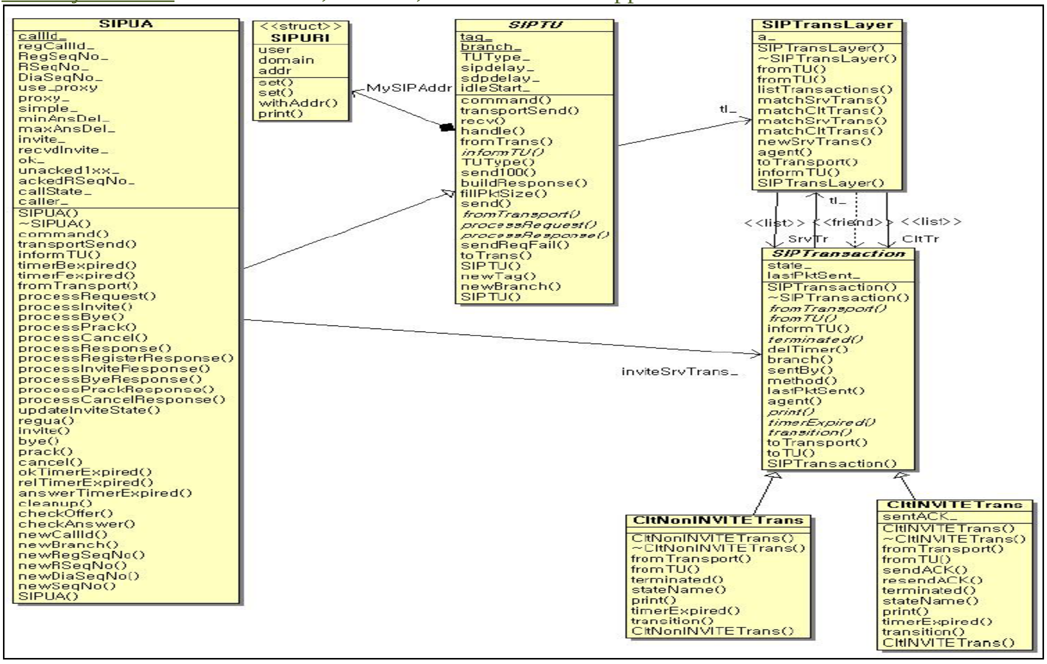


Figure 1. The class diagram of a SIP user agent

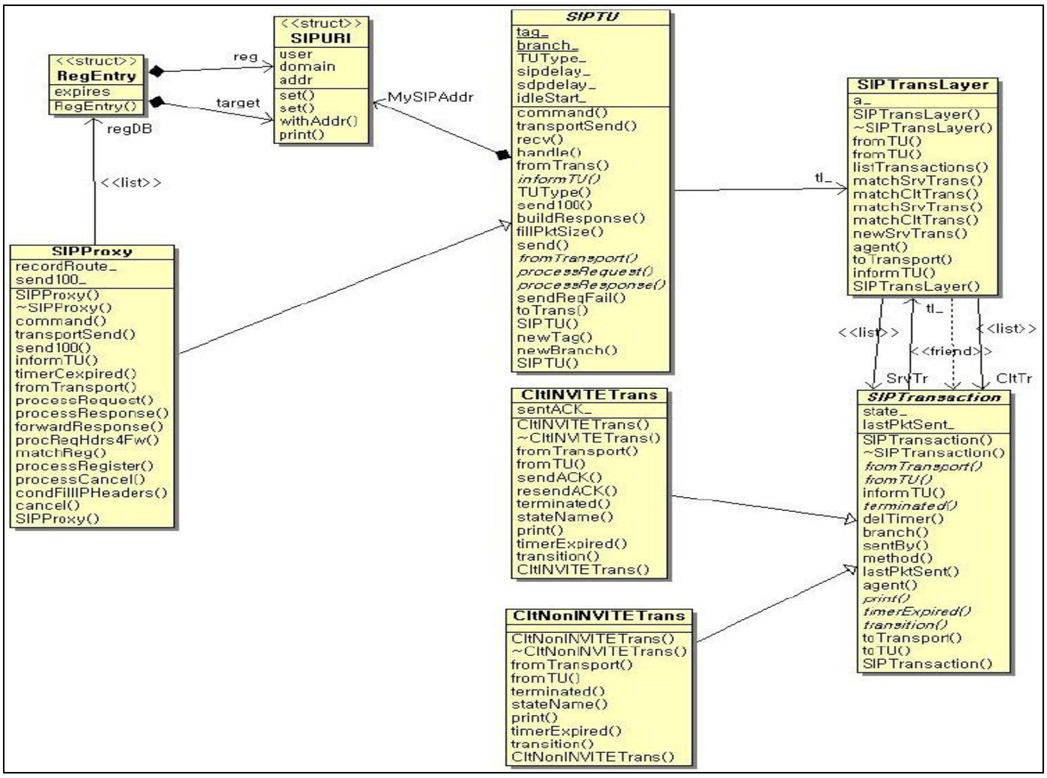


Figure 2. The class diagram of a SIP proxy

Fig. 3 shows the class diagram of SIP messages used in the Rui Prior's implementation. SIPMessage consists of lots of SIPHeader classes and at most one SIPBody. SIPMessage manages SIP header information as a linked list of several SIPHeader's, which defines a logical request recipient SIPHeaderTO, an identification information of a request sender SIPHeaderFrom, an address information of SIPUA SIPHeaderContact, a message grouping information SIPHeaderCallId, a transaction identification and ordering information SIPHeaderCSeq, a transport information for the transaction and location information for the response to be sent SIPHeaderVia and others.

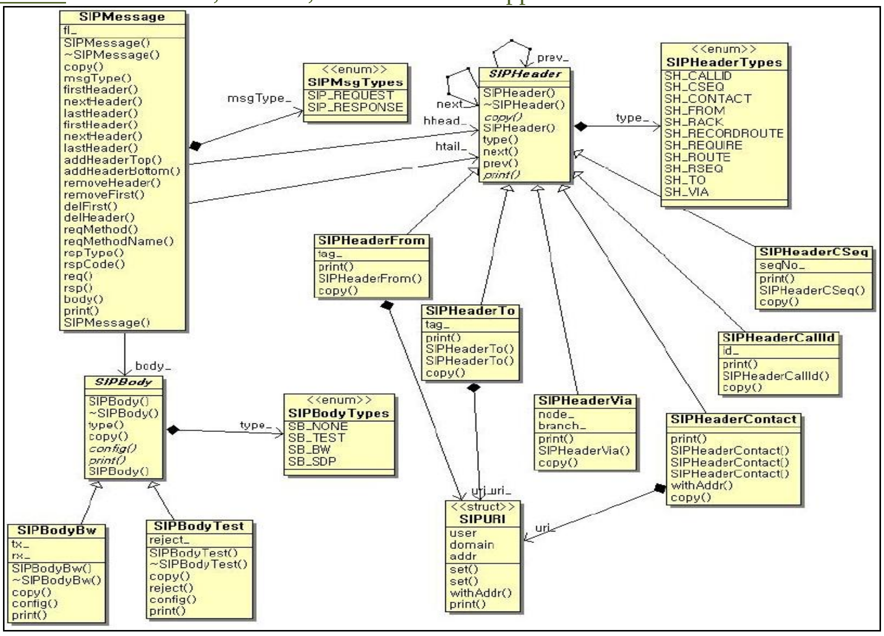


Figure 3. The class diagram of SIP messages

2.2 OMA PoC

The OMA PoC standard is mainly divided into the control plane protocol [8] which is a signaling protocol similar the SIP [5] and the user plane protocol [9] which carries user's media traffic based on RTP [10]. Fig. 4 shows the brief OMA PoC architecture. The service logic for SIP sessions are implemented in the application server using SIP/UDP/IP. The application server functionality is implemented by the PoC server when the SIP/IP Core for the PoC service is according to 3GPP/3GPP2 IP Multimedia Subsystem (IMS) [11]. Thus the SIP/IP Core and PoC Server functionalities may be in one physical entity. Media packets carrying users' voice data and the talk/media burst control for managing the talk right are transferred between PoC Clients and a PoC Server using RTP/UDP/IP [9].

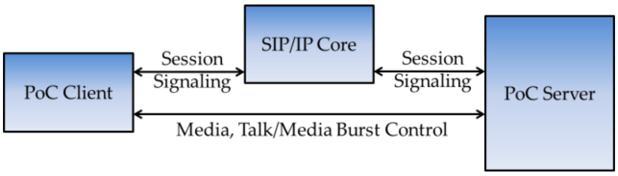


Figure 4. A brief OMA PoC architecture

The PoC Server performs either the Controlling PoC Function or the Participating PoC Function. In this paper, we call the PoC Server with the Controlling PoC Function and the Participating PoC Function as the Controlling PoC Server and the Participating PoC Server in short. The Controlling PoC Server mainly performs the management of PoC sessions such as the session establishment and the media burst control [8].The Participating PoC Server performs relays the Talk Burst and Media Burst Control messages between the PoC Client and the Controlling PoC Server and may relay RTP media packets from the Controlling PoC Server.

Each PoC Client should register to their Participating PoC Server prior to participating in the PoC session according to rules and procedures of RFC 3261 [5] with extended headers including PoC feature tags [8]. Fig. 5 shows the registration procedure of the PoC Client. In the SIP REGISTER request of the PoC Client, information such as the SIP URI and IP address of the PoC Client can be found. This information is used in the proposed scheme to keep the location information of PoC Clients.

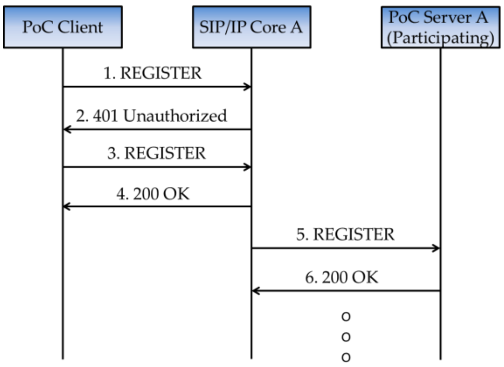


Figure 5. Registration procedure

PoC Session establishment is also made according to rules and procedures of RFC 3261 with extended headers including PoC feature tags as shown in Fig. 6 [8]. For simplicity, messages from/to the SIP Core are excluded from Fig. 6 and only messages from/to OMA PoC entities are shown. Dotted arrows represent MBCP (media burst control protocol) messages, which manages the talk right of PoC Clients. There are four kinds of PoC Sessions: 1-1, ad-hoc, pre-arranged, and chat [8, 11]. There are two session modes: the ad-hoc PoC Session and the pre-arranged PoC Session. In the ad-hoc PoC Session, the group information can be found from the recipient list in the SIP INVITE request. In the pre-arranged PoC Session, the group information is maintained by the Controlling PoC Server. A PoC Session can also be classified into the on-demand session and the pre-established session according to the time of the session establishment. The on-demand session is started when a user initiates the PoC Session with his/her recipient list [8]. The pre-established PoC Session is another method for the session establishment, which first makes a parameter negotiation to establish a PoC Session and RTP packet transmission is performed if required [8]. Fig. 6 shows the message flow of an ad hoc PoC Session establishment using on-demand signaling, in which we are interested for implementation.

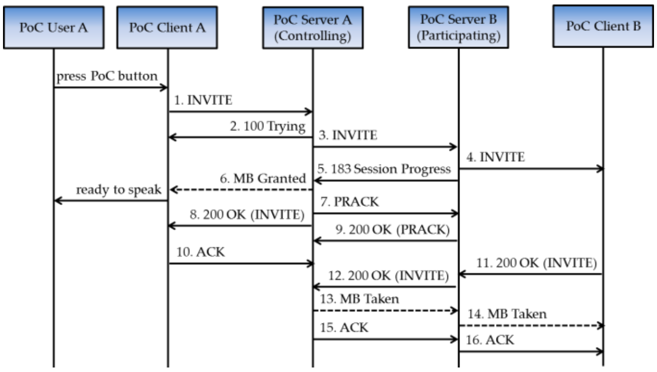


Figure 6. Ad hoc PoC Session establishment using on-demand signaling

It takes place during the PoC Session setup to determine if a PoC server performs either the Controlling PoC Function or the Participating PoC Function and lasts for the duration of the whole PoC Session. In ad hoc PoC group sessions, the Controlling PoC Server is the PoC server of the inviting user. In pre-arranged PoC group sessions, the controlling PoC server is the PoC server hosting the pre-arranged PoC group.

III. EXTENDED ARCHITECTURE FOR THE OMA POC

To implement protocols defined in the OMA PoC standard, we extend the existing ns-2 SIP implementation or the Rui Prior's work, where main components are the class *SIPUA* for a client, the class *SIPProxy* for a server and the class *SIPMessage* representing SIP messages. There are some protocol difference between SIP and the OMA PoC standard as shown in Fig. 6. Thus not only do we extend the existing classes but also we modify them to support the OMA PoC standard.

A talker uses a user agent functionality to communicate with other talkers, which is implemented by the class PoCClient of which the base class is SIPUA. Table 1 gives a brief description of PoCClient and Fig. 7 is the class diagram related to PoCClient. PoCClient deals with the OMA PoC registration and the initiation/termination of an ad hoc PoC group session. To support an ad hoc PoC group session, it also has to get a function to add invited users to a certain group session. To support PoCClient, SIPUA adds a publishing capability according to RFC 3903 [12] and PoCClient uses it to generate a SIP PUBLISH request according to rules and procedures of RFC3903 [12] and RFC4354 [13].

Table 1. A brief description of PoCClient

Class	*PoCClient*
Base Class	SIPUA
Functionalities	• OMA PoC registration • Addition of invited users • Initiation/termination of an ad hoc PoC group session • Processing of a SIP 200 OK response from a PoC server
Modified Classes	Added Functions
SIPUA	Send a PUBLISH message with a PoC service setting information to a PoC server after a successful registration

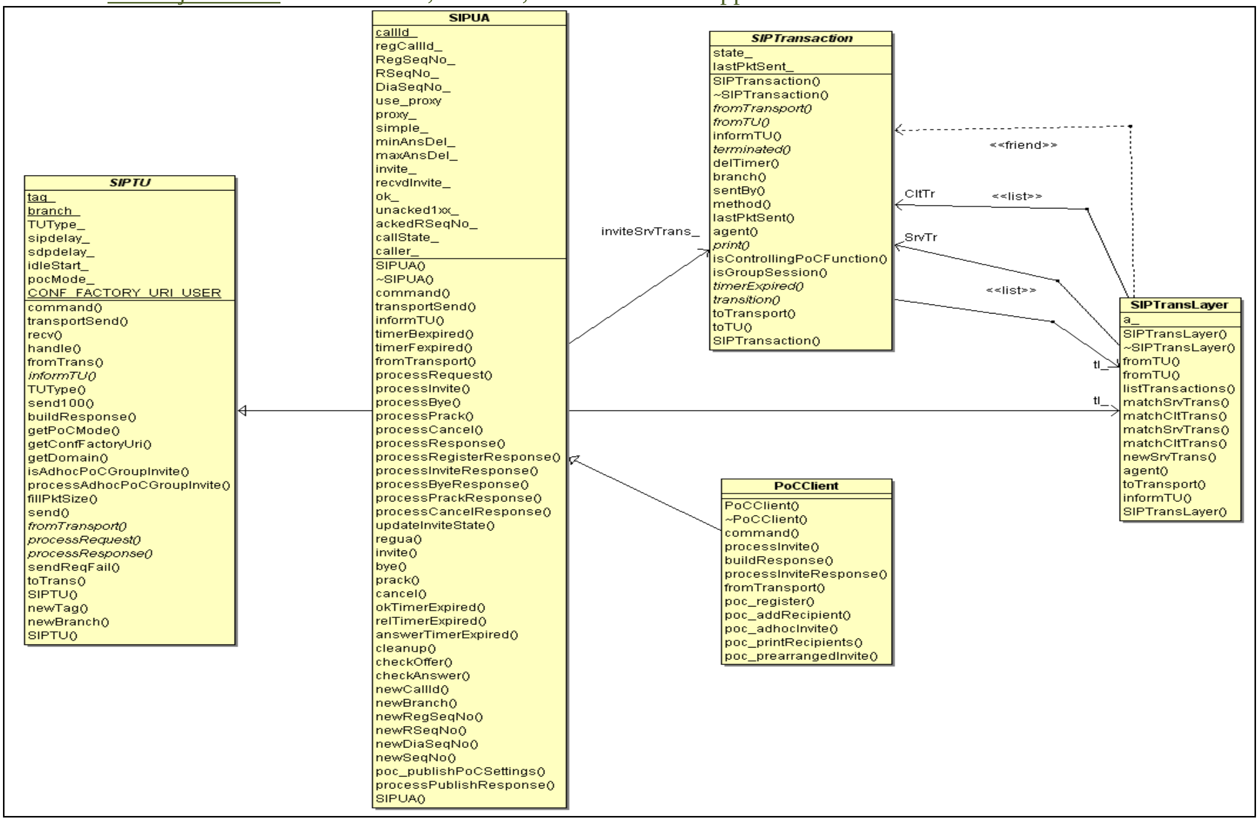


Figure 7. The class diagram related to PoCClient

A PoC server, which performs a Controlling PoC Function and/or a Participating PoC Function, deals with requests from a PoC client. Table 2 shows a brief description of the class PoCServer, of which the class diagram is shown in Fig. 8. PoCServer has main functionalities to process a PUBLISH message, a group session INVITE request, and other response messages and manage group session information. A Controlling PoC Server deals with request and/or response messages from a PoC client and one or more corresponding Participating PoC Servers according to the message exchange protocol as shown in Fig. 6. In a Controlling PoC Server, the information of an ad hoc PoC group session is maintained using the classes PoCGroupSession and PoCClientSession. PoCGroupSession has the information for the INVITE request to make a PoC group session. Whenever a Controlling PoC Server sends an INVTE request to each PoC client in the ad hoc PoC group, an instance of PoCClientSession is generated for that session. To support a PoC server, its base class SIPProxy should be modified to process an SIP PUBLISH request and an ad hoc PoC INVITE request. SIPTransaction deals with a transaction which consists of a request and responses relative to the request. SIPTransaction should be also modified to support the ad hoc PoC group session by adding functionalities to determine if a PoC server performs a Controlling PoC Function and if an INVITE request is a group session.

Table 3 shows a brief description of the class PoCSIPMessage, which extends the class SIPMessage to support messages defined in the OMC PoC standard. PoCSIPMessage, as shown in Fig. 9(a), allows multiple SIP bodies, which is useful to exchange media parameters between a PoC client and a PoC server for parameter negotiation. SIPMessage already allows multiple SIP headers, which is defined by the class SIPHeader, but it should be extended by adding subclasses PoCSIPHeaderContact, SIPHeaderAcceptContact, SIPHeaderAllow, SIPHeaderUserAgent *etc.* to support the OMA PoC standard. The added subclasses of the class SIPHeader are shown in Fig. 9(b).

Table 2. A brief description of PoCServer

Class	*PoCServer*
Base Class	SIPProxy
Functionalities	• Processing of a PUBLISH request message • Processing of an ad hoc PoC group session • Processing of 183 session progress response from a participating PoC server • Processing of a 200 OK message to an inviting user • Management of session information
Added Classes	Description
PoCGroupSession	Deals with the information of PoC group sessions
PoCClientSession	Deals with the information of PoC client session for each PoC group session
Modified Classes	Added Functions
SIPProxy	Processes an SIP PUBLISH request and an ad hoc PoC INVITE request
SIPTransaction	Determines a PoC server's PoC function and a group session

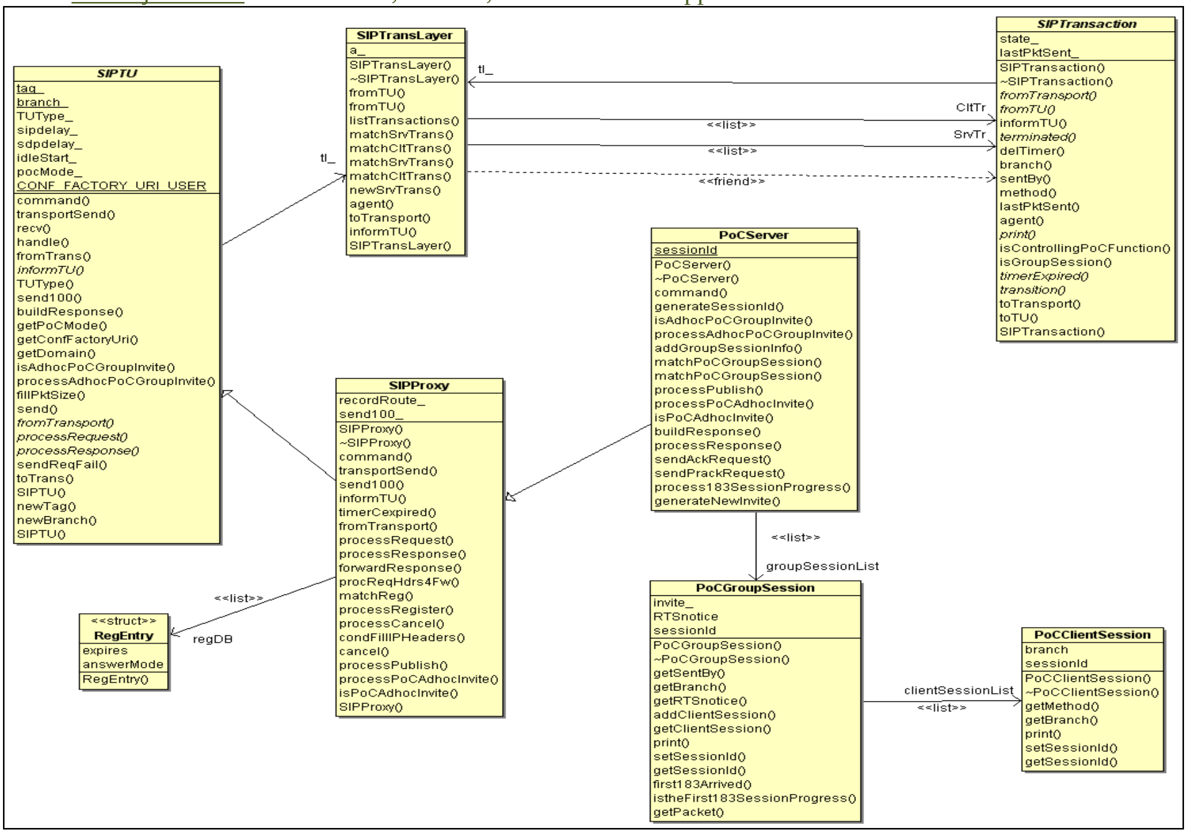
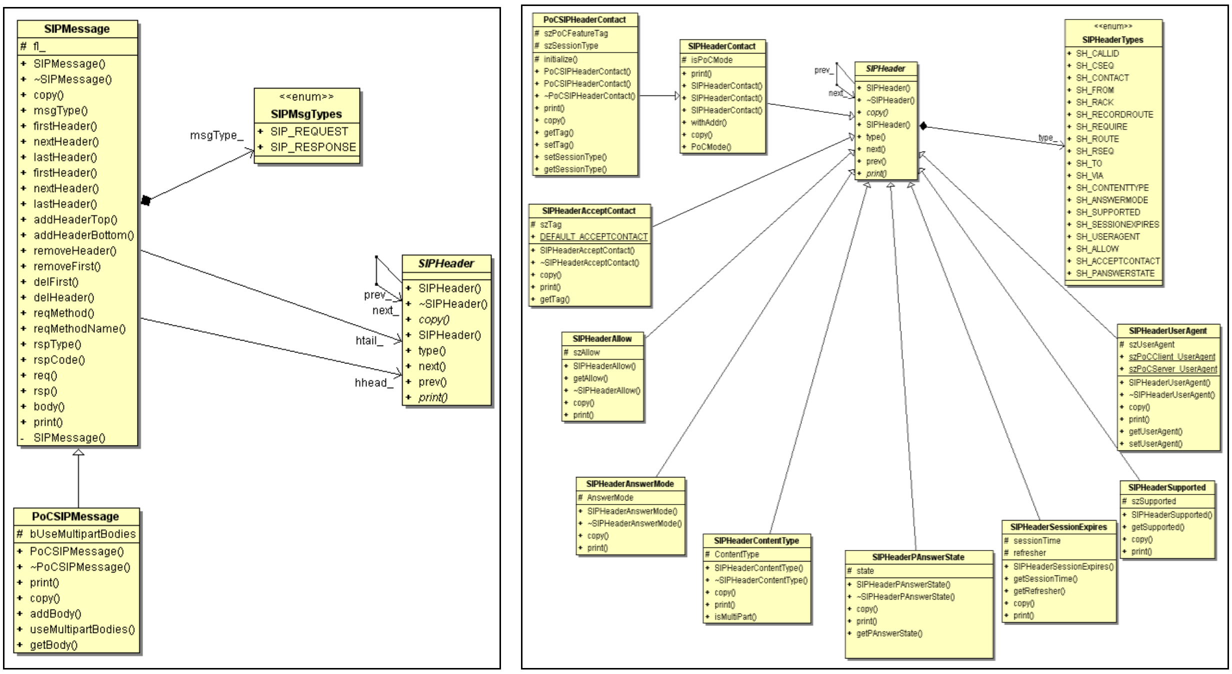


Figure 8. The class diagram related to PoCServer

Table 3. A brief description of PoCSIPMessage

Class	*PoCSIPMessage*
Base Class	SIPMessage
Functionalities	Defines the OMA PoC message to allow multiple SIP bodies
Added Classes	PoCSIPHeaderContact, SIPHeaderAcceptContact, SIPHeaderAllow, SIPHeaderAnswerMode, SIPHeaderPAnswerState, SIPHeaderSessionExpires, SIPHeaderSupported, SIPHeaderUserAgent
Modified Classes	SIPHeaderTypes

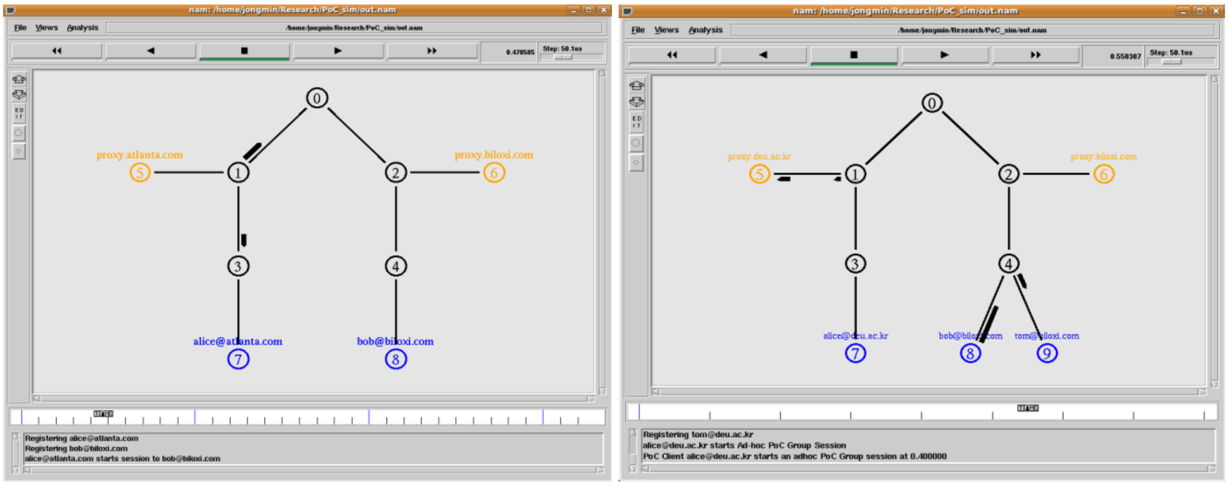


(a) (b)

Figure 9. The class diagram related to PoCSIPMessage: (a) the representation of a PoC message; (b) SIP headers added for the OMA PoC.

IV. PERFORMANCE EVALUATION

In order to evaluate the network performance of the OMA PoC implementation proposed in this paper, we use ns-2.33 as a base network simulator, which is a discrete event simulator used for networking research widely since mid-90's [14]. Fig. 10 shows snapshots of simulation to compare the Rui Prior's implementation and the OMA PoC implementation. The call setup scenario used for the simulation is adopted from RFC 3261 [5]. Rui Prior's implementation, as shown in Fig. 10(a), can make only a 1-to-1 session for call setup. On the other hand, the OMA PoC implementation can make 1-to-many ad hoc group session. Fig. 10(b) shows the packet flow of the OMA PoC implementation for two recipients, from which we can find that the group session progresses successfully from one PoC client to two PoC clients.



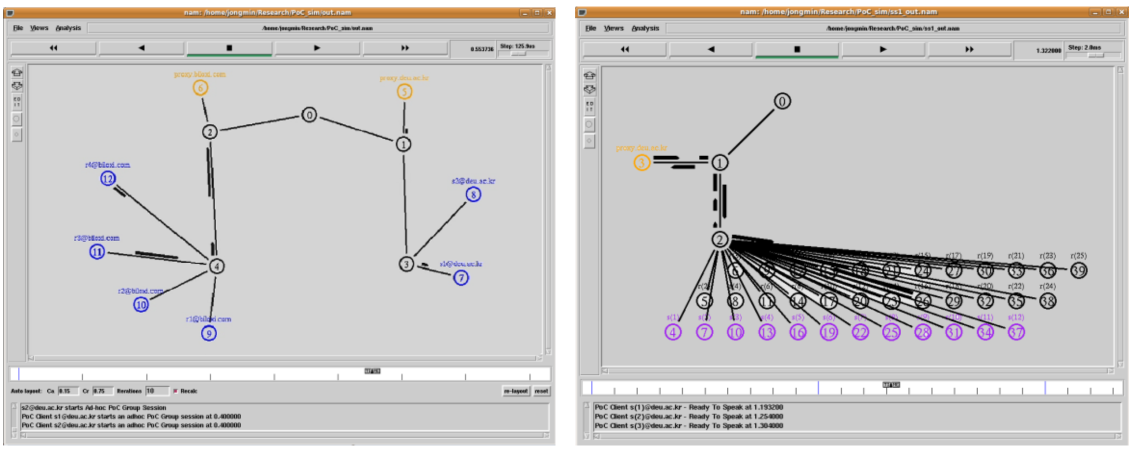
(a) (b)

Figure 10. Snapshots of simulation to compare two implementations: (a) Rui Prior's implementation, (b) OMA PoC implementation.

To see the network performance in terms of a call setup time, we carry out two experiments for different wired network configuration as shown in Fig. 11. Fig. 11(a) is to simulate packet flows between two 1-to-2 group sessions, where senders and receivers are located in different networks. Two initiating PoC clients belong to the domain "deu.ac.kr" and all PoC recipients belong to the domain "biloxi.com." In the experiment, we evaluate the network performance as the link delay and the bandwidth vary between a PoC client and its neighbor node. Link parameters in Fig. 11(a) are as follows:

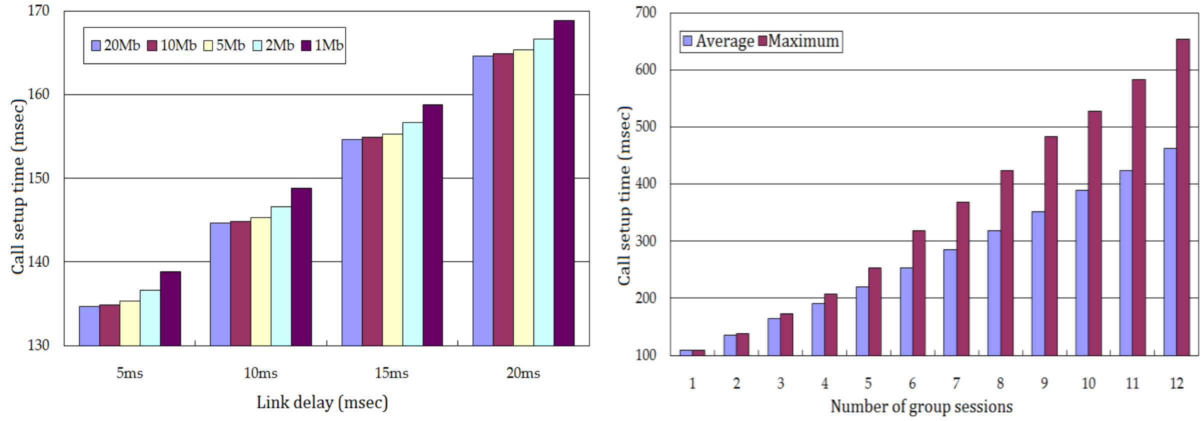
- Bandwidth between a PoC client and its neighbor node : 1~20Mbps
- Link delay between a PoC client and its neighbor node : 5~20msec
- Bandwidth of all other links : 100Mbps
- Link delay of all other links : 10msec

Fig. 11(b) is to simulate packet flows of many 1-to-2 group session where a sender and all other receivers are in the same network in order to see the network performance for the in-bound traffic congestion to a PoC server. All PoC clients belong to the same domain and are located in the same network. Link parameters Fig. 11(b) are fixed as follows: the bandwidth of all links is 100Mbps and the link delay of all links is 10msec. PoC clients are represented by *s(i)*, *r(2*i)* and *r(2*i+1)* for *i* = 1~12. PoC clients in the bottom row are senders, *s(i)*. Recipients *r(2*i)* and *r(2*i+1)* for *s(i)* are in the middle and the top rows respectively. Twelve one-to-two group sessions exist in Fig. 11(b).



(a) (b)

Figure 11. Snapshots of simulation for different network configuration: (a) group session initiation among PoC clients in different networks, (b) group session initiation among PoC clients in the same network.



(a) (b)

Figure 12. Call setup time: (a) group session initiation among PoC clients in different networks, (b) group session initiation among PoC clients in the same network.

Fig. 12(a) shows the simulation result of the experiment in Fig. 11(a). As the link delay increases, the average call setup time also increases linearly. We can find this result intuitively since signaling messages for call setup are exchanged among PoC clients and Controlling/Participating PoC Servers. The distribution of the call setup time is in the interval of 135~170msec, which is short enough to start media transfer after the connection of a group session. However, this result is derived from a situation that the signaling traffic is not enough to become overloaded. We try the experiment in Fig. 11(b) to make the network overloaded.

Fig. 12(b) shows the simulation result of the experiment in Fig. 11(b), which initiates many 1-to-2 group sessions in the same network. In this case, the traffic bottleneck is the PoC server, which performs both the Controlling PoC Function and the Participating PoC Function. Each group session consists of one initiating PoC clients and two recipient ones. As the number of group sessions increases, the traffic in the network also increases. The average call setup time increases linearly as the number of group sessions increases. But the maximum call setup time increases rapidly compared to the average call setup time, from which we can find that some PoC recipient clients cannot join its group session within a certain amount of time and thus media traffic such as voice cannot be delivered to those PoC clients for cooperation.

V. CONCLUSION

Recent wide-spread use of Internet technologies such as VoIP, VoLTE and VoWLAN results from the proliferation of smart phones. A half-duplex group communication mechanism or PTT has been standardized to replace the existing analog/digital walkie-talkie service. In this paper, we have designed and implemented the ns-2 module of the OMA PoC control plane to deal with the ad hoc PoC session establishment using on-demand signaling. We have used the SIP implementation of Rui Prior and extended it to deal with the signaling protocol for the ad hoc PoC session establishment using on-demand signaling. We have shown that the signaling protocol operates exactly for the purpose of verification and we have also performed the simulation study for various network configuration. We expect that the ns-2 implementation of the OMA PoC control plane can be used for the effective network simulation study of group communication.

ACKNOWLEDGEMENTS

This work was supported by Dong-Eui University Foundation Grant (2012).

REFERENCES

[1] Open Mobile Alliance, *Push to talk over cellular (PoC) – architecture: approved version 2.0* (Aug. 2011).

[2] R.S. Cruz, M. Serafim, G. Varatojo, and L. Reis, Push-to-talk in IMS mobile environment, *Proc. 2009 fifth Int'l Conf. on Networking and Services*, Valencia, Spain, 2009, 389-395.

[3] L.-H. Chang, C.-H. Sung, H.-C. Chu, and J.-J. Liaw, Design and implementation of the push-to-talk service in ad hoc VoIP network, *IET Communications, 3(5)*, 2009, 740-751.

[4] Open Mobile Alliance Mobile Phone Standards and Specification, http://openmobilealliance.org/.

[5] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler, SIP: Session Initiation Protocol, RFC 3261, June 2002.

[6] Rui Prior's homepage – ns, http://www.dcc.fc.up.pt/~rprior/ns/.

[7] ns-2 homepage, http://nsnam.isi.edu/nsnam/index.php/Main_Page.

[8] Open Mobile Alliance, *OMA PoC control plane: approved version 2.0* (Aug. 2011).

[9] Open Mobile Alliance, *PoC user plane: approved version 2.0* (Aug. 2011).

[10] H. Schulzrinne, S. Casner,R. Frederick and V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, RFC3550, July 2003.

[11] G. Camarillo and M.A. Garcia-Martin, *The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds, 3rd ed.* (West Sussex, UK: Prentice-Hall, 2008).

[12] A. Niemi, Ed., Session Initiation Protocol (SIP) Extension for Event State Publication, RFC 3903, Oct. 2004.

[13] M. Garcia-Martin, A Session Initiation Protocol (SIP) Event Package and Data Format for Various Settings in Support for the Push-to-Talk over Cellular (PoC) Service, RFC 4354, Jan. 2006.
[14] ns-2 wiki homepage, http://nsnam.isi.edu/nsnam/index.php/Main_Page.

Jong Min Lee received the B.S. degree in computer engineering from Kyungpook National University, Korea, in 1992, and the M.S. and the Ph.D. degrees in Computer Science from Korea Advanced Institute of Science and Technology (KAIST) in 1994 and 2000, respectively. From Sept. 1999 to Feb. 2002, he worked for Samsung Electronics as a senior engineer. Since 2002, he has been a faculty member of the Department of Computer Software Engineering, Dong-Eui University, Busan, Korea. From Feb. 2005 to Feb. 2006, he visited the University of California, Santa Cruz as a research associate. From Feb. 2012 to Feb. 2013, he was a visiting scholar of the Department of Computer Science at The University of Alabama, Tuscaloosa, AL. His research interests include routing in ad hoc networks and sensor networks, mobile computing, and parallel computing.