OPEN ACCESS

International
Journal
Of Modern Engineering Research (IJMER)

Performance Evaluation of Push-To-Talk Group Communication

Jong Min Lee[1]

[1]*Department of Computer Software Engineering, Dong-Eui University, Republic of Korea*

Abstract: *Recently the VoIP technology has been used for the purpose of voice conversation through the Internet since the proliferation of smart phones. In this paper, we are interested in the network performance of the group communication a.k.a. the push-to-talk (PTT) service. As the media traffic to a VoIP server increases, the VoIP server can be a bottleneck and the voice quality can be declined, which results from the asymmetry between the in-bound traffic and the out-bound traffic in a VoIP server. To simulate the voice packet transfer of the group communication, we use ns-2.34 and the ns2VoIP++ package to calculate the end-to-end mean opinion score. Simulation results show that both the packet error rate and the packet delay have a significant effect on the performance in terms of the mean opinion score.*

Keywords: *OMA PoC, User Plane, Group Communication, Push-To-Talk, MOS*

I. INTRODUCTION

Recently many Internet applications have been used widely in our lives for the proliferation of smart phones. Especially the concept of the VoIP has been used in many applications such as Skype, Viber, Voxer, TiKL etc. Among applications which support VoIP, there are several VoIP 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]. This push-to-talk technology supports the half duplex communication among end-to-end users like the trunked radio system (TRS). Nextel communications was the first company to provide the push-to-talk service commercially using iDEN.

The problem of the group communication results from the asymmetry between the in-bound traffic and the out-bound traffic in a VoIP server. As the out-bound traffic increases, the packet delay and the packet loss can be occurred. There are no problem in terms of sending and receiving packets, but there can be the decrease of the voice quality in terms of listeners due to the characteristic of voice traffic. In this paper, we analyze how these parameters results in the voice quality. We use ns-2.34 [4] for the simulation study and ns2Voip++ package [5] for the analysis of the voice quality.

In Section 2, we describe the media transfer in the group communication and ns2Voip++ package briefly. In Section 3, we describe the VoIP server added to the legacy ns-2.34 code to simulate the group communication. The performance study due to the packet delay and the packet loss is given in Section 4. Finally, we give a conclusion in Section 5.

II. RELATED WORKS

In this section, we describe the media transfer in the PoC architecture [6] briefly in order to explain the flow of VoIP packets. Then we describe the ns2Voip++ Package [5] in short to explain the simulation environment.

2.1 Media Transfer

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) [7]. We use the PoC standard to explain the group communication. After the PoC Session [8] is established, media packets including voice data are transferred between the PoC Server and PoC Clients. These media packets are transported by RTP/UDP/IP [6]. The PoC Server forwards received RTP Media packets towards all other PoC Clients that are not on hold [6], that is, the received packet is copied and forwarded to other PoC Clients in a group. The basic RTP header of media packets is described in RFC 3550 [9] and additional RTP payload formats for various audio/video codecs such as G.729A, GSM, EVRC and AMR are described in several RFC documents [10, 11, 12]. From the SSRC field of the RTP header, the PoC Server and PoC Clients can find who sends the RTP packet.

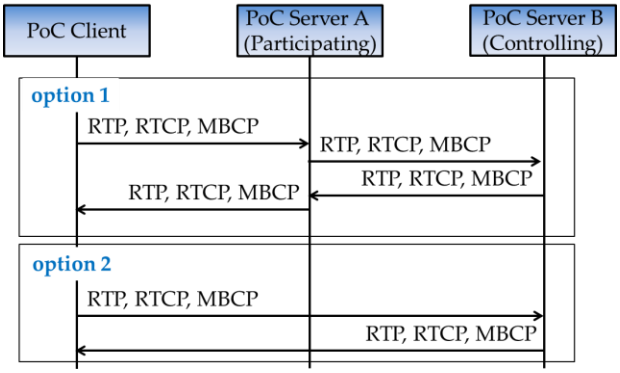


Figure 1. Media transfer options

Fig.1 shows two options that transfer voice data between PoC clients. In Option 1, the Participating PoC Server is in charge of transferring voice packets and also provides the filtering function and the transcoding function among different codecs. However the performance may be degraded due to the increased packet delay compared to Option 2 in which voice packets and other related packets are transmitted directly through the Controlling PoC Server. In this paper, we only consider Option 2 to support group communication. The incoming traffic and the outgoing traffic of a PoC Server are unbalanced because of its one-to-many communication pattern. Thus problems such as the traffic handling at a PoC Server and packet transmission to recipient PoC Clients occur especially when network congestion happens.

2.2 ns2Voip++ Package

In general, the voice communication has two states, Talkspurt and Silence. Fig. 2 shows the basic concept of ns2Voip++ package. The package generates VoIP voice traffic according to the designated codec such as G.711, G.723, G.729A, AMR *etc*. The transmission of voice packets are performed using an ns-2 UDP agent. The package also has the functionality to calculate the voice quality using the E-model [13], with which we can finally get the mean opinion score (MOS) [14].

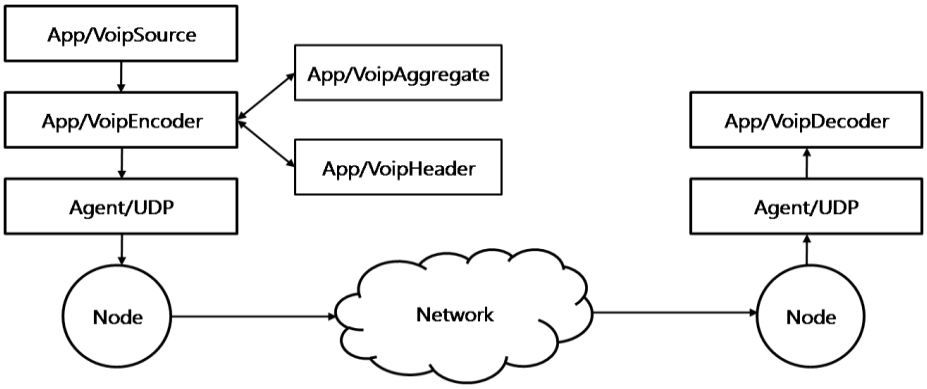


Figure 2. Basic concept of ns2Voip++ package

The toll quality has been standardized by ITU-T, in which typical methods are the MOS and the E-model. The E-model and the MOS are an objective method and a subjective method, respectively. The E-model can be expressed by the packet delay and the packet loss according the used codec, so it is used frequently to evaluate the toll quality. The package also can model the voice using Talkspurt and Silence states using the Weibull distribution. It also may generate voice packets using the exponential distribution. The average Talkspurt time and the average Silence time are assumed to be 1 seconds and 1.5 seconds, respectively.

Fig. 3 shows class diagrams of the ns2Voip++ package. Fig. 3(a) illustrates the talker-side class diagram. VoipSource initiates and terminates the voice communication according to Talkspurt and Silence states. VoipEncoder generate packets according to the designated codec when it is in Talkspurt state. Fig. 3(b) is a listener-side class diagram. VoipDecoder receives voice packets from a talker and then calculates the MOS value using the E-model.

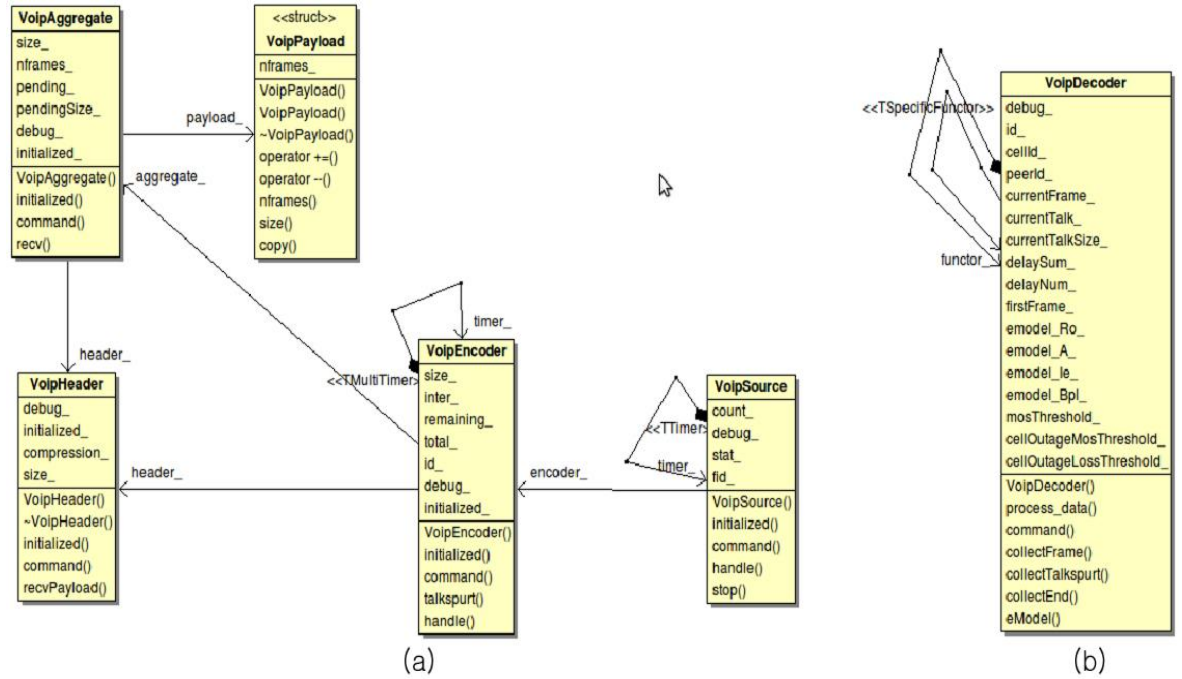


Figure 3. Class diagrams of ns2Voip++: (a) a talker-side class diagram; (b) a listener-side class diagram

III. IMPLEMENTATION OF A VOIP SERVER

To evaluate the network performance, we implement a VoIP server for group communication. In this section, we describe the basic concept and the detailed design of a VoIP server.

3.1 Basic Concept

The existing ns2VoIP++ package in Fig. 4(a) cannot model the group communication. To make the group communication possible using the ns-2 network simulator, we add an agent to mimic the functionality of a VoIP server as shown in Fig. 4(b). In the original scheme, the source node *s(0)* should make the same packet for different destination nodes *d(i)*, in which we can easily find that this scheme is not equal to the method handling the media transfer through the PoC Server, or the VoIP server.

In Fig. 4(b), we can find that the VoIP server receives packets generated by *s(0)* and replicates them to every *d(i)*. To make it possible, the VoIP server should handle the information about all groups and each group members in a group. In the PoC architecture, the PoC Server with a Controlling PoC Function performs the Talk Burst control and duplication of RTP media packets.

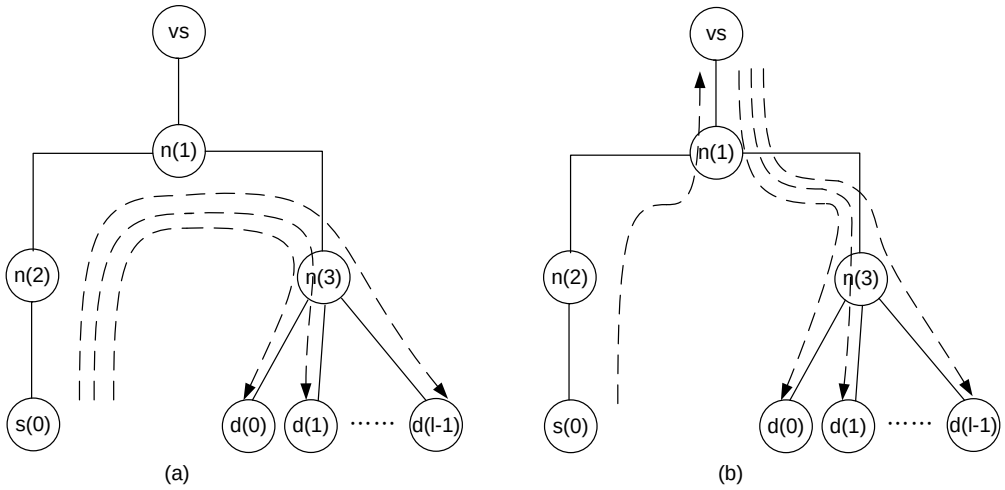


Figure 4. ns2Voip++ extension: (a) the original scheme; (b) the extended scheme

3.2 VoipServer Agent

The following functionalities of a VoIP server are required to simulate the group communication:

- The function to set the maximum number of groups
- The function to add a group member to a certain group
- The function to duplicate packets from a source node and transfer them to group members

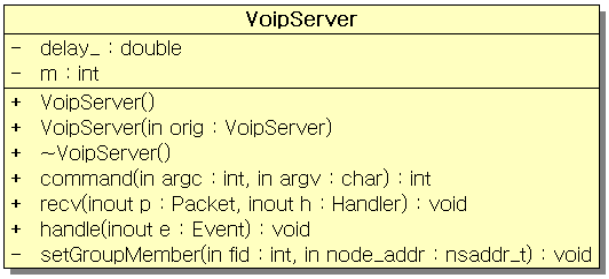


Figure 5. The VoipServer agent class

We add the VoipServer agent as shown in Fig. 5. The name of an ns-2 agent acting as a VoIP server is definded as "Agent/VoipServer" and can be declared in the ns-2 script as follows:

```
set   voipserver [$ns node]
set   agtserver    [new Agent/VoipServer]
$ns  attach-agent    $voipserver    $agtserver
```

It is easy to implement the function to set the maximum number of groups, *k*. We implement it in VoipServer::command() and it can be set as follows.

```
$agtserver    set-ksize    10
```

We assume that the number of group members, *l*, for each group is the same for simplicity. The information of group members, or destination nodes, for each group is managed using the C++ STL map class in ns-2, which is shown in Fig. 6. In the Tcl script, we can set the value *k* and *l* for arbitrary number of destination nodes.

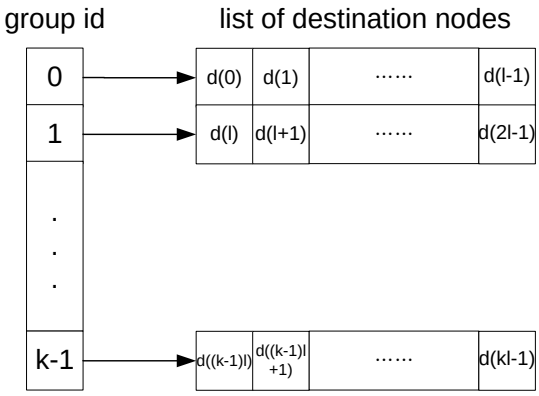


Figure 6. Data structure for managing the list of destination nodes

To implement the function to duplicate packets from a source node and transfer them to group members, we reflect the service time to make a packet for a group member and send it. The service time is fixed as 10^{-6} second. To change the service time freely without compilation, Tcl/C++ binding is used.

Except the VoipServer agent, the existing ns2Voip++ package is changed as follows:

- The configuration of RNG (random number generator) for the class 'VoipSource' to change the RNG
- The singleton class 'MosLogger' is added to record the final MOS value to get the average MOS value
- The class 'VoipDecoderOptimal' is changed to use the class 'MosLogger' to record the MOS value

IV. PERFORMANCE EVALUATION

In this section, we perform the simulation study to see the effect of voice quality due to the out-bound voice traffic of a VoIP server in the group communication.

4.1 Simulation Environment

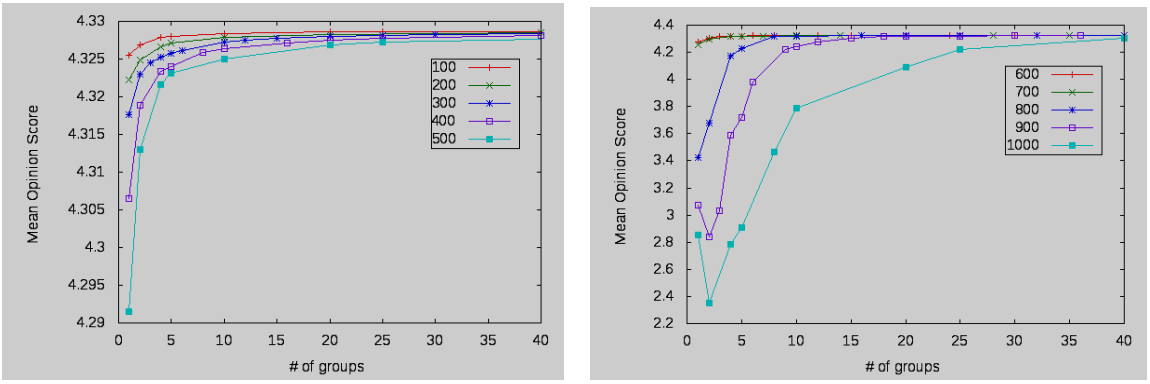
We use ns-2.34 for the simulation and simulation parameters are shown in Table 1. We use the AMR codec for voice encoding and decoding, which generates 32-byte voice packet every 0.02 second. The aggregation scheme is used to merge the voice packet because it is too small for transmitting through Internet. The aggregation size 'two' means that two continuous voice packets are merged as one and it is transmitted every 0.04 second for network efficiency. The IP header size is assumed to be 3 because we assume that the IP header compression scheme [15] is used in the simulation. A source node, destination nodes, and a VoIP server are configured as the network model in Fig. 4(b).

Table 1. Simulation parameters

Parameter	Value
Link bandwidth	10 Mbps
Link delay	10 msec
Queue length	5000
Packet loss rate	0.00 ~ 0.20
Service time	10^{-6} sec
Talkspurt p.d.f.	Weibull distribution f(x; 0.412, 0.824)
Silence p.d.f.	Weibull distribution f(x; 0.899, 1.089)
codec	AMR
aggregation size	2
IP header size	3

4.2 Simulation Results

First, we perform the simulation by fixing the number of destination nodes $k * l$. Thus we can find that the maximum number of groups, k, is inverse proportion to the number of group members in each group, l. Fig. 7(a) shows the change in the MOS value according to the number of groups for the fixed number of destination nodes=100~500. In cases with small number of groups less than 10, MOS values are not good compared to group bigger than 10. However, the MOS value of the worst case with the number of destination nodes = 500 is 4.29, which means the voice quality is in 'very satisfied' condition. In this case, the link bandwidth is big enough to handle voice packets and thus there is no performance decrease.



(a) (b)

Figure 7. The effect of the number of group members: (a) The number of destination nodes=100~500; (b) The number of destination nodes=600~1000

In Fig. 7(b), the number of all destination nodes are varied from 600 to 1000. The overall curve looks similar as in Fig. 7(a). In case that the number of destination nodes becomes bigger than 900 and the number of groups, k, in in [1, 10], the MOS values becomes less than 3, which means the voice quality becomes poor to listen. In case that the number of all destination node = 1,000, the performance in terms of the MOS value is still good

with the number of groups > 20. As the number of groups become larger, we can find easily that the number of group members become smaller and the overall performance in terms of MOS show good behavior.

To see the performance in terms of MOS better, we fix the number of group members, *l*, and then change the number of all destination nodes as shown in Fig. 8. For $l \leq 20$, the overall performance show good but it degrades as the number of group members becomes larger than 20. Thus we can find that it is important to manage the number of group members per each group to a certain degree.

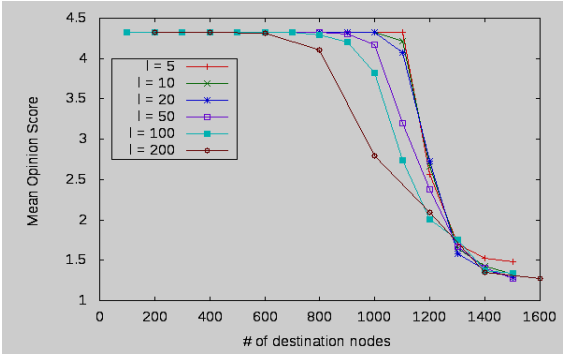


Figure 8. The effect of the number of group members

Now we change the packet error rate on the out-bound link of a VoIP server to see the overall performance, which is shown in Fig. 9. The number of group members per each group, *l*, is assumed to be 10. In Fig. 9, the MOS value decreases from 4.9% to 6.7% as the packet error rate (per) increases by 0.01. For the packet error rate greater than 0.03, the MOS values becomes less than 3.5, which means the listener gets the poor voice quality. Thus we can find that the packet error rate has a great influence on the voice quality.

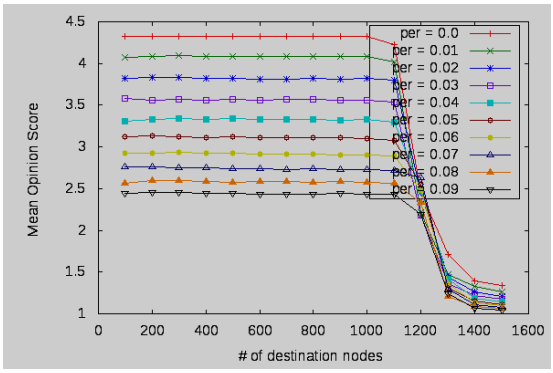


Figure 9. The effect of a packet error rate

We also change the packet delay on the network of a VoIP server to see the overall performance, which is shown in Fig. 10. The number of group members per each group, *l*, is also assumed to be 10. For the packet delay greater than 20 msec, the overall performance in terms of MOS degrades to a poor state. The packet delay can be occurred when the packet is routed on the Internet due to the congestion and the link state, which results in the poor voice quality for the group communication.

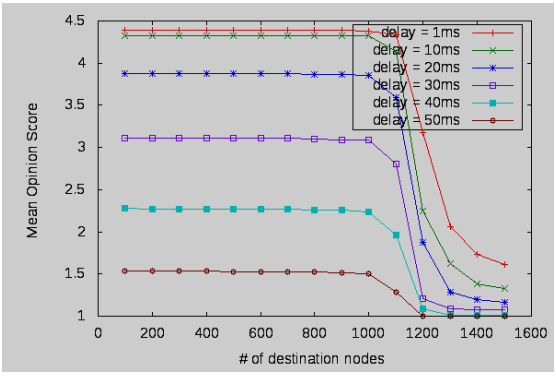


Figure10. The effect of a packet delay

V. CONCLUSION

In this paper, we have evaluated the overall performance of the group communication in terms of the mean opinion score (MOS), that is the voice quality. We have used ns-2.34 for the simulation and the ns2VoIP++ package for the analysis of the voice quality. To simulate the group communication, we have added some functionalities of the VoIP server. The large number of group members has a significant effect on the overall performance, so we have found that it should be managed equal to or less than 20. We also have found that both the packet error rate and the packet delay have a significant effect on the performance in terms of MOS. From the simulation result, we can find that the efficient scheme to manage the voice quality for the group communication is required to handle those situations.

ACKNOWLEDGEMENTS

This work was supported by Dong-Eui University Foundation Grant (2013).

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] ns-2 homepage, http://nsnam.isi.edu/nsnam/index.php/Main_Page.
[5] M.M. Andreozzi, D. Miqliorini, G. Stea and C. Vallati, Ns2Voip++, and Enhanced Module for VoIP Simulations, Proc. 3rd Intl. ICST Conf. on Simulation Tools and Techniques, Brussels, Belgium, 2010.
[6] Open Mobile Alliance, PoC user plane: approved version 2.0 (Aug. 2011).
[7] Open Mobile Alliance Mobile Phone Standards and Specification, http://openmobilealliance.org/.
[8] Open Mobile Alliance, OMA PoC control plane: approved version 2.0 (Aug. 2011).
[9] H. Schulzrinne, S. Casner,R. Frederick and V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, RFC 3550, July 2003.
[10] H. Schulzrinne and S. Casner, RTP Profile for Audio and Video Conferences with Minimal Control, RFC 3551, July 2003.
[11] A. Li, RTP Payload Format for Enhanced Variable Rate Codecs (EVRC) and Selectable Mode Vocoders (SMV), RFC 3558, July 2003.
[12] J. Sjoberg , M. Westerlund, A. Lakaniemi and Q. Xie, RTP Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs, RFC 4867, April 2007.
[13] ITU-T Recommendation G.107, "The E-Model, a Computational Model for Use in Transmission Planning," March 2005.
[14] ITU-T Recommendation P.800, "Methods for Subjective Determination of Transmission Quality," Aug. 1996.
[15] Bormann (Ed.), Robust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed, RFC 3095, July 2001.

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.