IK2213 Network Service and Internet-based Application
Assignment 2

# SIP Speaker (Advanced)

Yuxin Cheng yuxinc@kth.se
Hio Ian Lei hilei@kth.se

## Introduction

This report corresponds to the assignment 2 of IK2213. A SIP client application and a HTTP server were built up using Java. The application can answer and terminate a call with the proper use of SIP and SDP. After answering the phone call, the application will send out a wav message file by RTP using the library of "jlibrtp". The wav message is generated from the text which users input in the web page hosted by our HTTP server using the library of "freeTTS".

## Implementation

1. Answering the SIP call

To answer the phone call, the first task is accepting the incoming packet generated by other SIP applications such as SJphone. The application listens to the port 5060 by default. When an INVITE SIP package is received, the application will send out a 200 OK message to the caller if the user accepts. The 200 OK message also contains SDP message which describes the media type used later. When the user wants to hang up the call, it will send out a BYE message to end the call.

2. Sending out voice message

After sending the 200 OK SIP message and receives the ACK with the same Ceq number, a RTP session will be set up directly between the two users. The RTP session uses the ports which are stated in the SDP part inside the SIP package sent before. When the session is ready, our application will send out the voice message which transformed from text to the caller and end up the call afterwards. We use the jlibRTP library to transmit the audio.

3. Generating sound data

The user can change the default message content through a web page. In this project, we also launch a HTTP server which allows user to submit, show and delete the message. The web page in it contains a form which will send the user's input to the HTTP server and the server will pass the message to our core application. The sound data sent via RTP later is generated with the use of the FreeTTS library. Still we need to encode the sound data or the other side will not accept our message. In this project, we convert the sound data to PCM_ALAW format from the default PCM_SIGNED format of the default generated wav file using FreeTTS. Finally, we can send out the audio messages by RTP after this proper encoding.

## Difficulties

1. Communication with other's SIP applications

As different SIP applications have some different features, we have to process packages carefully. We check every package carefully to avoid error.

2. Audio encoding

As RTP accepts limited types for transmissions, we need to encode it well first. At the beginning, we decided to encode it into GSM. However, we tested several libraries and found that it is really difficult as GSM is badly supported in Java. We turned to ALAW instead which can also be transmitted afterwards.

3. Multi-threading

To support multi-thread, we use a hashmap to match Call-ID to a Working Class. This Working class is used to send different replies in SIP and also extend Thread so that it can start RTP transmission in a different thread. When a BYE message is received from remote application, the Call-ID is checked and correlated Working class RTP thread is stopped and an ACK of this BYE is sent.

To support TCP and UDP socket listening simultaneously, we create a daemon thread for TCP HTTP handling, so that it will not interrupt the UDP socket receiving and sending the datagram packet.

## Conclusion

User can use the web page as an interface to submit their message. The application will accept the message from the HTTP server and transform it into sound audio. When the application receive a SIP call, it will accept it and have a RTP session with the other client. After sending the encoded voice message back, the application will hang up the phone.

SIP and SDP is the main part of the assignment and we learnt a lot of it. Moreover, RTP is extremely important and we spend most of our time on it. All-in-all, the assignment is challenging. It gives us a great chance to study SIP, SDP, RTP, and what's more is to practise our programming skill in Java.