IK2213 - Network Services and Internet-based Applications 2013-04-26
Fredrik Nordlund <fnordl@kth.se> Prakash Rajagopalan<prakashr@kth.se>

# IK2213 Sipspeaker Report

## Introduction

In recent years there has been a global process of IP-alization; making traditional systems work over IP (through the Internet). Examples of this are: vending machines connected to the Internet, security cameras, and IP-telephony. When global telephone operators decided to try telephony over IP, they started a partnership project called *third generation partnership project* (3GPP) to try to solve the problems involving telephony over IP. In particular several protocols for telephony session signaling were created. A major protocol for session signaling is the *session initiation protocol* (SIP). To describe the media stream, e.g. the media encodings, the *session description protocol* (SDP) was designed.

The goal of this small project has been to get a greater understanding of how these above mentioned protocols are used to set up a media session. Furthermore how they can be used in collaboration with a real-time media protocol, such as the *real time protocol* (RTP), to create a powerful IP-telephony answering machine application that will play back a sound file to a calling application when called.

## Problem

The problem of this project was, how do you use the above mentioned protocols and how do you integrate them together to create a powerful application. Several problems that had to be solved was for example, what are the necessary headers that has to be used when dealing with SIP and SDP, and how should you parse text to fit the different protocols.

Another problem was, since the protocols will run over UDP, how do you create an application that can handle multiple clients when using UDP sockets. After this has been successful, how do you use a predefined RTP library to send encoded audio (the sound file that should be sent back to the caller) over to a client, in the case of this project SJphone was used as the client application.

Furthermore a web server should be integrated with the above mentioned application so that a user can see the current audio message that is to be played to a caller. A user should also be able to change the message to be played by writing the new message in text and by the use of a text-to-speech library convert the text message into a playable sound file.

## Solution

The Sipspeaker is designed as three subsystems. The first one is a SIP server, the second sub system is a RTP/Session handling system, and lastly the third subsystem is a HTTP server that makes it possible for a user to view the current message and edit the current message to be played to a caller. The functionalities are as follows.

The SIP server is implemented by creating a datagramsocket (UDP) on a specified port; the port is specified either in a config file, in the command line, or by a default value which is 5060. It waits on the specified port for SIP packets to arrive. Once a SIP packet arrives at the server, it looks at the SIP session call ID to see if there is such a session in place. If there is no such session but the SIP message is a INVITE message, then the session will be prepared and appropriate SIP messages (Trying, Ringing, OK) will be sent back to the caller with appropriate attributes that is parsed from the INVITE message. In the case where the session does not exist and the SIP message is not of the type INVITE, the message will be dropped. The session can be successfully (and gracefully) taken down either by the caller or our application.

Upon receiving the ACK from a caller, a new RTP session is created which will send the audio file (.wav) to the caller's application (sjphone). Once the audio file has been successfully sent, the RTP handler will issue a command to send a SIP BYE message to the caller and call (SIP session and RTP session) is terminated. A BYE message can also be issued by the caller at any point in time, during the playing of the audio message, to terminate the call. This project also provides an advanced level feature of implementing text-to-speech conversion , where a user can access a local website and view/edit the current message. The current message is the message sent by the RTP session to the caller. The current message is specified in the config file. If a current message does not exist, then the default audio file is used.