Lund University
Lund institute of technology

Sound Quality Analysis System

Thesis at

Sony Ericsson Mobile Communications

Elvis Barudzija & Hani Fakhouri

LTH School of Engineering
Lund University
Box 882
SE-251 08 Helsingborg
Sweden

LTH Ingenjörshögskolan vid Campus Helsingborg
Lunds universitet
Box 882
251 08 Helsingborg

Printed in Sweden
Media-Tryck
Biblioteksdirektionen
Lunds universitet
Lund 2011

Written in LaTeX

Preface

This Bachelor thesis has been performed during the spring of 2011 at SEMC, Sony Ericsson Mobile Communications in Lund. It is the final part of Bachelor degree of Science in Computer Science and Engineering at Lund University.

It has been a very rewarding and exciting experience during our time at SEMC. We were fortunate to find our place in an environment surrounded by pleasant and professional people. We would like to express our gratitude to the test-team for helping us in different aspects concerning our work. We would specially express our gratitude to the Audio Acoustics team from whom we obtained the privilege of acquiring great knowledge and ideas.

"This work is dedicated to my dear dad who passed away just before starting this thesis. I wish you were here, I wish I could have done you proud once again. Without your fine and good upbringing, this would never had been possible. You dad will always be in my heart and I hope you are in a better place."

Elvis Barudzija

"I would like to dedicate this work to my dear family and specially to my parents. You have always been giving me motivation, inspiration and support."

Hani Fakhouri

Lund, Juni, 2011

Abstract

Purpose

Audio quality in mobile devices has always been an essential factor in mobile communications. The integrity insurance of voice quality between an original audio signal and a degraded audio signal during a mobile conversation is of big interest for all mobile communication companies. In this thesis work a sound quality analysis system called SQAS is to be developed for SEMC.

Problem statement

This thesis has been initiated when one of the test departments at SEMC was in need of a system that could analyze sound quality. Similar systems exist at the company in other departments but these systems are large and can not be used by other departments. Therefore a system that is light, flexible and can be shared with other departments was needed. This thesis work shows the possibility of developing such a system.

Method

SQAS aims to compare two audio files and give a final measurement value that describes the audio quality difference between these two files. The final value is calculated by taking several measurements into account. Furthermore a public algorithm called PESQ is used to present a MOS value that considers the human auditory system and uses psychoacoustic models. In order to assure the righteousness of the SQAS and the results presented by it, a suitable (test-rig) has been developed.

Supervisor
Mats Lilja - Associate Professor at Department of Automatic Control at the Lund Technical University

Examiner
Christian Nyberg - Associate Professor at the Department of Communication Systems at Lund Institute of Technology

---

Keywords: PESQ, Test-rig, Android, POLQA, SEMC, ITU, MOS, Audio, Audio-Quality, Mobile Conversation.

Sammanfattning

Syfte

Ljudkvaliten i mobila enheter är en viktig faktor för mobil kommunikation. För att säkra integriteten av kvalitèn mellan en original ljudsignal och en överförd ljudsignal under ett mobilsamtal är av stort intresse för alla mobil tillverkare. Det här examensarbetet handlar om att utveckla ett komplett ljudmätningssystem vid namnet SQAS.

Problemformulering

Det här examensarbetet har inletts då en av test-avdelningarna på SEMC var i behov av ett system som kunde analysera ljudkvalitén. Liknande system finns redan inom företaget på andra avdelningar men dessa är stora och kan inte användas av andra avdelningar. Därför behövdes ett system som är lätt, flexibelt och kan användas av andra avdelningar. I den här rapporten beskrivs hur man utvecklar ett sådant system.

Metod

SQAS syftar till att jämföra två ljudsignaler och ge ett slutligt mätvärde som presenterar skillnaden mellan dessa två signaler. För att göra jämförelsen används en publik algoritm vid namnet PESQ som presenterar ett MOS värde. Den tar hänsyn till hur människan uppfattar ljud där också psykoakustiska modeller används. För att presentera ett tillförlitligt resultat har en lämplig (test-rigg) utvecklats

Handledare
Mats Lilja - Associate Professor at Department of Automatic Control at the Lund Technical University

Eximinator
Christian Nyberg - Associate Professor at the Department of Communication Systems at Lund Institute of Technology

Nyckelord: PESQ, Test-rig, Android, POLQA, SEMC, ITU, MOS, Audio, Audio-Quality, Mobile Conversation.

Terminology

- **SQAS** - Sound Quality Analysis System.
- **SCI** - SQAS Controlling Interface, the computer application that has been developed. The application is responsible for the control of the mobile phones and for the audio file processing.
- **RCA** - Radio Corporation Of America, the cable we use to make connection between computer and phones.
- **TRS** - Tip Ring Sleeve, 3.5 mm Line-in.
- **ADB** Android Debug Bridge is a command line tool which is used for communication with Android mobile phones.
- **MOS** - Mean Opinion Score. MOS is a numerical indication of the perceived quality of received media after compression and/or transmission.
- **ALSA** - Advanced Linux Sound Architecture, The Advanced Linux Sound Architecture (ALSA) provides audio functionality to the Linux Operating System.
- **PCM** - Pulse-code modulation, is a method used to digitally represent sampled analog signals.
- **PESQ** - Perceptual Evaluation of Speech Quality, is a family of standards comprising a test methodology for automated assessment of the speech quality as experienced by a user of a telephony system.
- **PEAQ** - Perceptual Evaluation of Audio Quality, used in objectively measuring in perceived audio quality.
- **ITU** - The International Telecommunication Union is a specialized agency of the UN which is responsible for information and communication technologies.
- **TEST-RIG** - In SQAS there is a set of hardware which acts as "Test-rig", it is referred to the sound-card, cables, transformers and phones.

Contents

1 Introduction **1**
1.1 Overview 1
1.2 Sony Ericsson Mobile Communications 1
1.3 Background 2
1.4 Project stakeholders 3
1.5 Problem description 3
1.6 Purposes and goals 4

2 Work Plan **5**
2.1 Time plan 6
2.2 Pre-study 7
2.3 Source criticism 7

3 Perceptual Evaluation of Speech Quality - PESQ **9**
3.1 Background 9
3.2 Overview 10
3.3 Recommendations 11
3.4 Algorithm limitations 11
3.5 Algorithm source code 12

4 SQAS - Sound Quality Analysis System **13**
4.1 System Description 13
4.2 Software 14
4.2.1 Android Debug Bridge 15
4.2.2 Android applications 17
4.2.3 SQAS Controlling Interface 19
4.2.4 Automatic process 23
4.3 Hardware 25
4.3.1 Connections 25
4.4 Summary 30
4.4.1 The signal interference 30

5 Conclusion **31**

6 Future recommendations **33**
6.1 Expand the algorithm use 34
6.1.1 POLQA 34
6.2 SCI recommendations 36
6.3 Hardware recommendations 36

7 Bibliography **37**
7.1 Web Sources 37
7.2 Scientific Reports 37
7.3 Oral Sources 38

USER GUIDELINES - *Conditions to run the system*
APPENDIX - *The SCI Graphical User Interface*

Chapter 1

Introduction

1.1 Overview

Within the last few years, the use of mobile devices has increased dramatically. To meet public demands for quality, the mobile devices need to be constantly improved in many ways. This thesis aims to develop a system called Sound Quality Analysis System (SQAS) whose purpose is to measure the speech audio quality during a mobile conversation. The system consists of both software and hardware parts. The hardware part consists of two mobile phones, external sound-card and various custom-made adjustments. The software part consists of android applications, an audio processing algorithm and a Java application.

1.2 Sony Ericsson Mobile Communications

Sony Ericsson Mobile Communications is a global provider of mobile multimedia devices, including feature-rich phones, accessories and PC cards. The products combine powerful technology with innovative applications for mobile imaging, music, communications and entertainment. The net result is that Sony Ericsson is an enticing brand that creates compelling business opportunities for mobile operators and desirable, fun products for end users.

Sony Ericsson is a 50-50 joint venture between Sony Corporation and Telefonaktiebolaget LM Ericsson. Sony Ericsson Mobile Communications was established in 2001 by telecommunications leader Ericsson and consumer electronics powerhouse Sony Corporation. The company is owned equally by Ericsson and Sony and announced its first joint products in March 2002. [ref.7].

1.3 Background

To assure the integrity of audio quality during a mobile conversation there are many approaches that can be applied. In principle there are two methods for doing that, one is objective and the other is subjective. The subjective approach is time consuming, expensive and impractical so it is not an area of interest. On the other hand, the measurements and results of the objective method are based on mathematical analyses that compare an original and a degraded signal. Measurements of such method can for example be Signal to Noise Ratio (SNR), Total Harmonic Distortion (THD), THD+Noise, Segmental SNR and Frequency response.

Nowadays, several factors interact to modify the original signal according to specific needs. The several factors in this case are, speech coders/decoders, signal masking and jitter, which can result in the change of both amplitude and phase of each frequency component. Thus, making it hard to analyze, conclude and most importantly to compare an original audio signal with a transferred one. So these traditional measurements simply do not reflect the audio quality when comparing two audio signals. As a result psychoacoustics and perceptual techniques should be applied.

Psychoacoustics is the scientific study of sound perception. More specifically, it is the branch of science studying the psychological and physiological responses associated with sound. The psychoacoustic model builds upon the fact that human hearing is a mixture of mechanical phenomenon of wave propagation and a sensory and perceptual event. Hereby, when evaluating audio quality it may be relevant to take into account not just the mechanics of the environment, but also the fact that both ear and the brain are involved in the evaluation.

The fact that the human ear has a nonlinear response to sounds of different loudness levels is used in telephone networks by nonlinearly compressing data samples before transmission and then expanding them for playback. Therefore upon audio quality quantification knowledge of the psychoacoustic model is very important to give a result that builds upon human senses. With the help of the psychoacoustic model many parts of a digital audio signal can be removed. These parts do not effect the quality of the audio signal. In fact removing them results in an even better audio quality. This is applied in almost all modern compression formats such as MP3, ACC, Dolby Digital(AC-3) and MPEG-1 Layer II also known as MP2.

1.4 Project stakeholders

The project stakeholders are found within the SEMC organization, mainly at the *test verification department* where SQAS will be used. It is at the *test verification department* where most of the high-level requirements come from, but the *test verification department* isn't the only stakeholder.

The technical design of the system comes mainly from the *acoustics department*, it is here where excellence in signal processing is found and therefore is a natural source and a stakeholder. The *department manager* has been the one who approved most of requirements and is therefore another stakeholder. So our stakeholders are:

- Test Verification department
- Acoustics department
- Department manger

1.5 Problem description

When a mobile phone is under development it passes different stages and one of the stages is the testing stage where every test has its own purpose and a specific feature is tested in the phone. One of the essential tests is the audio quality in a mobile phone, in other words it shows how good audio quality is and how much of the quality is lost during a mobile conversation.

It is obvious that SEMC already has a robust, well developed test tools and environments for testing audio quality. But these are large and not available for all departments in the company. So they requested an audio-system that is both flexible, light and effective which this thesis work is about.

This thesis work shows the possibility of developing a test system which main objective is to compare an original audio file with a degraded audio file transferred via the mobile communication network. The system shall be developed such that it can be physically easy shared between different departments. Furthermore, phone communication, audio processing and comparing is to be done automatically with as few user interaction as possible. The system should be very easy to use, even if the system is unfamiliar to the person using it. The main importance lies in getting the whole system to function in an automatic way.

1.6 Purposes and goals

The main purposes of this thesis work is to:

- Show that it is possible in an automatic way to compare two audio files one of which passed the mobile telephone network.
- Set up a (test-rig) that simulates a mobile telephone conversation.
- Integrate software and hardware parts together and present them in a complete testing system.

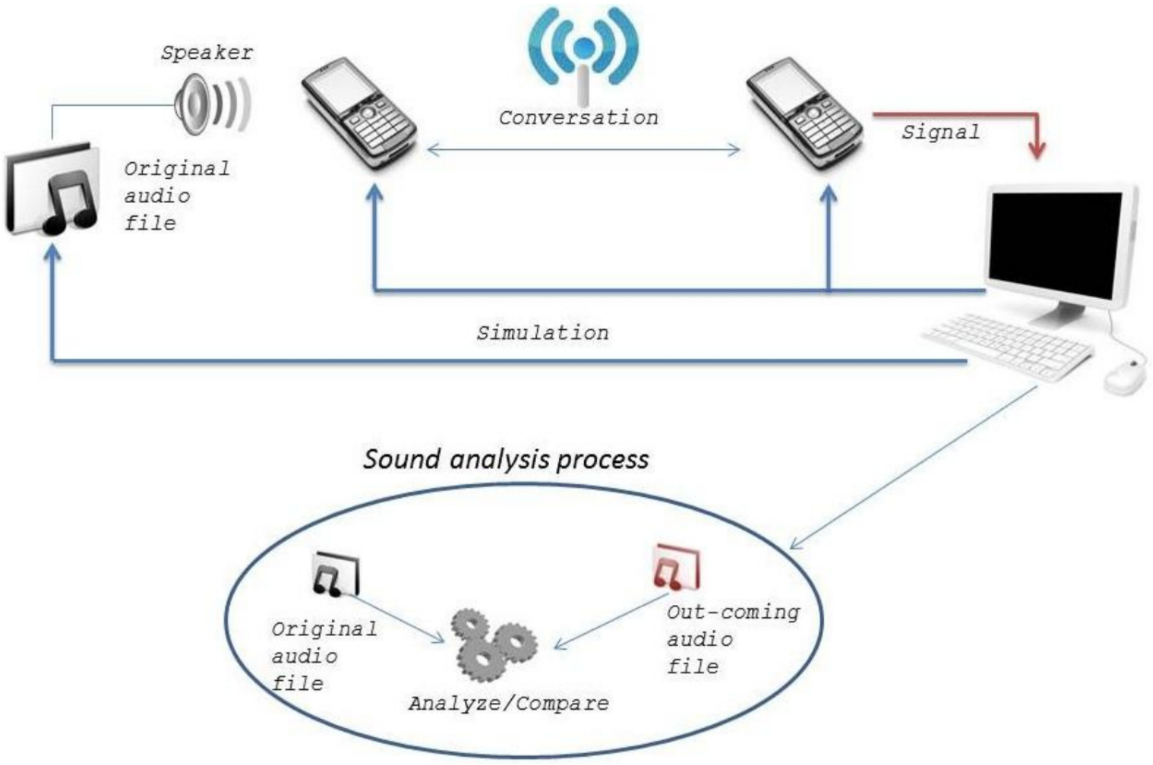


Figure 2.6: *First illustration of the system SQAS*

The figure 2.6 above is the first illustration of how the system should be developed. This was the starting point and a source of the idea how this system could look like.

Chapter 2

Work Plan

The work plan was put to complete the thesis project by starting with a pre-analysis phase where a search for relevant information was performed. The next phase was system design and development at Sony Ericsson's facilities. Having access to their facilities automatically gave access to opinions, guidance and equipment and this was very important for the project. When the implementation and development of the system started it was decided to split up the work into modules which are playback-/record, android/Java applications, hardware and the mobile phone communication. Doing this made the work more effective.

Working this way felt very natural and was very positive for the work progress. This way of work may cause that one of us was not too familiar with what the other did, therefore the work was always reflected what was done during the day and thus so was the work in sync.

Before applying solutions on the system these (solutions) were discussed between us and agreed about. Upon disagreement advices were taken from an expert just to make the right decision.

2.1 Time plan

A time plan was a very important tool for this project, where it was chosen not to avail of some digital resources, instead different "post-it" notes were used with a weekly plan. This improved both the overview and avenues to rapidly change anything. Because this thesis started a little bit late into the semester the project became a few weeks shorter so the planning was particularly important. Table below illustrates how the time plan looks like in a more abstract manner.

Week	Activity
1 2 3 4	**Orientation and research** - Define the basic concepts and aspects that shall be analyzed - Develop an algorithm that quantifies an audio file and gives quality change parameters. - Elicitation of all information about what our stakeholders really want.
5 6 7	**Implementation - Phase 1** - Develop a software that is needed to control the communication between computer and phones. Android application and a Java based application. - Build the hardware test-rig. - Make solutions how to get the process automatized.
8 9 10	**Implementation - Phase 2** - Develop application that handles playback and recording of the sound. - Merge all system parts (software and hardware) to form the system.
11 12	- Reviewing - Finish the report
13	- Presentation

2.2 Pre-study

The first three weeks were dedicated to the search for different information about audio signal processing and what parameters reflect audio quality in a signal. Furthermore a search after appropriate software tool which quantifies audio parameters was done to be able to compare two audio files.

Various advices from a number of univeristy professors [ref.8,9,11] in signal processing were taken into account. Almost every professor explained that it is difficult to compare two audio files especially because the original audio signal is effected by the mobile telephone network, various speech algorithms, audio coders and audio decoders and is therefore changed. Furthermore it seemed that writing an algorithm that quantifies audio parameters from an audio signal and compares two files would take a long time. Additionally it requires more people, skills, and tools which is a job for bigger companies.

So a different approach was taken which was reading special scientific reports and studies that already has been done in this area. All these reports pointed to the International Telecommunication Union, ITU, and their standard algorithms to measure sound quality. ITU used a special algorithm called PESQ to evaluate sound quality and that also became our choice. Before deciding to choose this algorithm there was doubts whether it was the right one and would fulfill our purposes. So further advices from different professors who confirmed that it was an appropriate algorithm to use and most importantly was getting approval from SEMC. A more detailed overview of the PESQ algorithm is given in the next chapter.

2.3 Source criticism

We have divided our sources into three categories which are web sources, scientific reports and oral sources. All the used web sources are from ITU and Opticom, these two are reliable sources as they are the ones who set the standard for sound quality. The used oral sources are university professors which have a deep topic understanding and the engineers from SEMC which are experts in the field. The scientific reports are realiable because they are simply scientific documents and have been approved prior to publication.

Chapter 3

Perceptual Evaluation of Speech Quality - PESQ

3.1 Background

The development of the PESQ algorithm was the result of an ITU competition for an objective speech-quality assessment metric to cope with 2.5G and 3G network degradations which affect quality as perceived by subscribers. The ITU competition declared PESQ the winner because it performed best for all the performance requirements and for all the applications imposed by ITU. After a comprehensive process of testing and validation against other sound quality metrics on a very large corpus of speech samples, the PESQ algorithm became the ITU-T P.862 standard (February 2001). The test databases contained speech samples provided by different speakers in different languages.

[ref.1] The PESQ Algorithm is designed to predict subjective opinion scores of a degraded audio sample, the PESQ score may be between -0.5 and 4.5. PESQ is designed to analyze specific parameters of audio, including time warping, variable delays, trans-coding, and noise. It is primarily intended for applications in codec evaluation and network testing. The idea of PESQ is very appealing because it would seem that it could provide a set of automated "golden ears" to evaluate any type of audio system and give a useful indication of the quality of the system. [ref.2]

3.2 Overview

PESQ compares an original signal X(t) with a degraded signal Y(t) that is the result of passing X(t) through a communications system. The output of PESQ is a prediction of the perceived quality that would be given to Y(t) by subjects in a subjective listening test.

The degraded signal and the reference signal are individually level aligned and filtered with the transfer characteristics of a receiving device. Basically two different filter functions – a narrow band (IRS) filter and a wide band filter – are available and may be chosen depending on the preferred application. After that the two signals are time aligned in order to compensate for small time shifts that can occur in e.g. Voice over IP networks due to delay and jitter changes and coding. In order to account for the distortions that are actually perceived by a human listener the model transforms the two aligned and filtered signals from the time-amplitude domain into a frequency-loudness domain (auditory transform). By subtracting the two signal representations an estimate of the audible differences is derived. The audible differences are accumulated over time while they are weighted differently depending on whether a distortion was added to the signal or if parts of the signal were missing after the transmission (cognitive model). Finally after the analysis a single Mean Opinion Score (MOS) is generated. The MOS is commonly used to describe the voice quality on a scale from -0.5 (bad quality) to 4.5 (excellent quality). [ref.6]

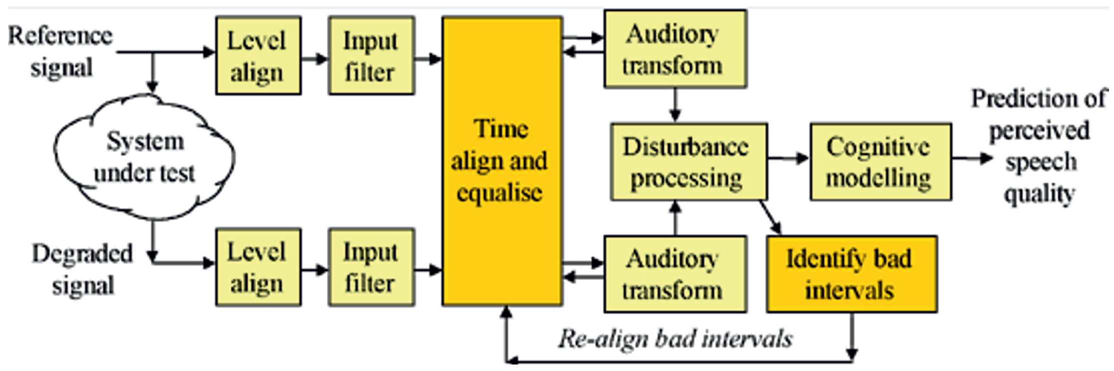


Figure 4.2 *The workflow of PESQ*

3.3 Recommendations

It is recommended that PESQ is used for speech quality assessment of 3.1 kHz (narrow-band) handset telephony and narrow-band speech codecs.

The source signal should be processed as appropriate for the system under test. It is desirable to avoid any further distortion by unnecessary quantization, amplitude clipping, or re-sampling. The preferred format for storing original and degraded signals is 8 kHz sample rate, 16-bit linear PCM. PESQ was validated on both 8 and 16 kHz sampling rate.

The speech activity in the reference speech, which can be measured based on [ref.3] should be between 40% and 80%. There should be a minimum of 3.2 s active speech in the reference and the input audio file should be 6 to 20 seconds in length.

3.4 Algorithm limitations

PESQ algorithm does not provide a comprehensive evaluation of transmission quality. It only measures the effects of one-way speech distortion and noise of speech quality. The effects of loudness loss, delay, side-tone, echo, and other impairments related to two-way interaction (e.g. center clipper) are not reflected in the PESQ scores. Therefore, it is possible to have high PESQ scores, yet poor quality of the connection overall.

3.5 Algorithm source code

The algorithm source code consists of the following three header files and five source files:

- dsp.h - Header file for dsp.c.
- pesq.h - General header file.
- pesqpar.h - PESQ perceptual model definitions.
- dsp.c - Basic DSP routines.
- pesqdsp.c - PESQ DSP routines
- pesqio.c - File input/output
- pesqmain.c - Main application
- pesqmod.c - PESQ high-level model

The source code can be downloaded from ITUs website and is available in full version. Although the code is license protected it can be downloaded and used for research and test purposes, to test whether this algorithm meets some specific demands. It is to be mentioned that even some audio test files are included inorder to perform tests on them.

Chapter 4

SQAS - Sound Quality Analysis System

4.1 System Description

The system that has been developed is divided into two main modules: a software module and a hardware module.

The software module consists of a computer application, two android mobile applications and the PESQ algorithm. The main goal of this software is to assure the communication between the mobile phones and the computer, send different triggers and ADB commands to automate the whole conversation process and most importantly to compare two audio files and give a final result. The hardware module consists of two mobile phones, a computer and an external sound-card.

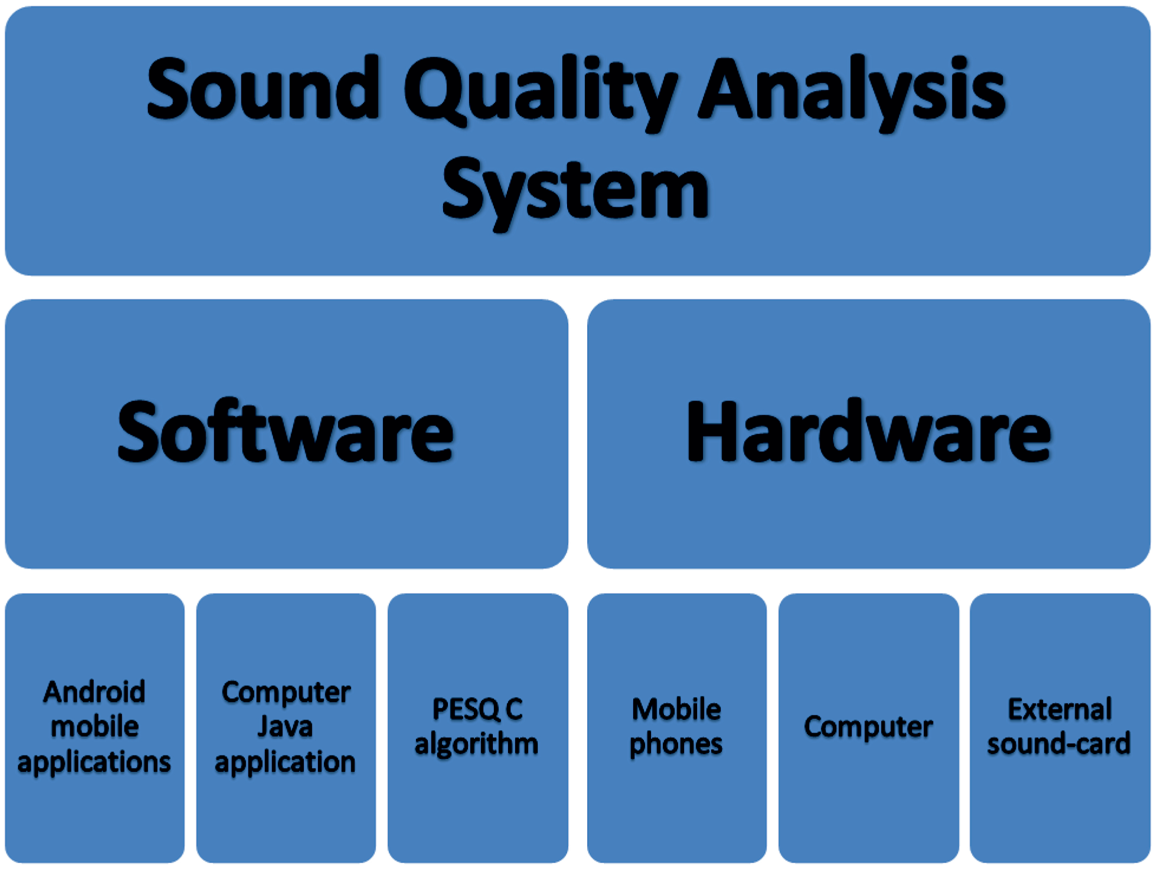

Sound Quality Analysis System
Software
Hardware
Android mobile applications
Computer Java application
PESQ C algorithm
Mobile phones
Computer
External sound-card

Figure 5.1: *SQAS overview*

4.2 Software

In this chapter the mobile phone that is being tested shall be called (the phone under test). The mobile phone that is being used as reference shall be called (the reference phone). The computer Java based application is called SCI (SQAS Controlling Interface).

4.2.1 Android Debug Bridge

To assure the synchronization and atomization of the conversation process, communication between the mobile phones and SCI is very important. In order to fulfill that communication purpose several approaches can be applied and the most natural one is to apply a server-client communication through USB. In fact there already exists a communication bridge, Android Debug Bridge (ADB), which is based on server-client communication. More specifically ADB is a versatile tool that manages the state of an emulator instance or Android-powered device. ADB includes three components:

- A client, runs on the development machine.
- A server, runs as a background process on the development machine.
- A daemon, runs as a background process on each emulator or device instance.

In other words ADB is used to control a mobile phone and send different ADB commands that change the state of the phone for specific purposes. In this context, ADB has been used in SCI to:

- Open TCP ports for communication
- Invoke a phone application

The automatic nature of the conversation process requires a two way communication. That is, if a specific ADB command is sent to the phone from the application on the PC, the application should receive a response from the phone and vice versa. This scenario is not available in the ADB. For example when sending an ADB command like Start_Some_Application to the phone, the client on the PC gets no acknowledgment that the application has been started.

For this reason SCI uses a custom server-client socket application that is actually a two way communication. As in the ADB, the mobile phone takes the role of the server and the SCI application takes the role of the client. The communication sequence between the phones and the computer is shown as a sequence diagram in the next page.

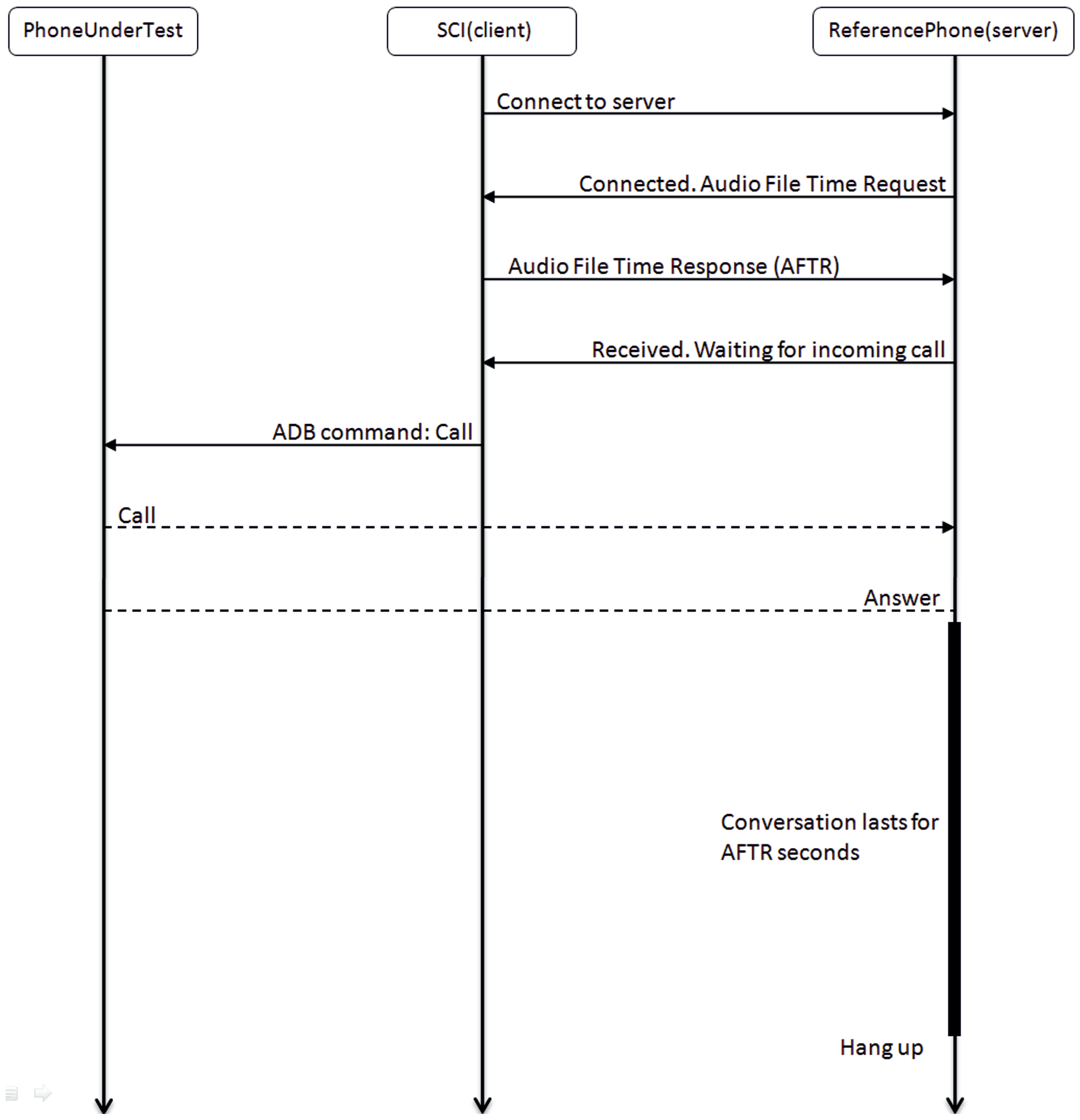

PhoneUnderTest
SCI(client)
ReferencePhone(server)
Connect to server
Connected. Audio File Time Request
Audio File Time Response (AFTR)
Received. Waiting for incoming call
ADB command: Call
Call
Answer
Conversation lasts for AFTR seconds
Hang up

Figure 5.2: *SQAS overview*

4.2.2 Android applications

The application found in the phone under test is called AutoConversation and has four classes. These classes are responsible for the automatic answer and the automatic hang up. These two are accomplished by using a virtual headset click. So when an incoming call is detected a headset click is performed to answer the call and when the conversation is over the same is done to end the conversation. One of the classes implements service intent which means that the application runs all the time in background and listens if there are any incoming calls. Furthermore, the application has a server class which is responsible for sending different phone events and states to the SCI application in order to synchronize the process. Phone events and states are predefined and can be:

- Phone ready
- Audio File Length Received
- Conversation Started
- Conversation Finished

The four classes are listed below with their respective purpose:

1. **AutoAnswerIntentService.java**
   This class extends IntentService and is the class that runs in background and waits for incoming calls. As soon as a call is received this class gets notified and a virtual headset answer simulation occurs and the call is answered. When the conversation time is finished the same headset simulation is done to end the call. Conversation time is acquired by the client in the computer.

2. **AutoAnswerReceiver.java**
   This class extends BroadcastReceiver and is the class that notifies AutoAnswer-IntentService as soon as an incoming call is detected.

3. **myServer.java**
   This class implements a server. It creates a server socket connection via the port 4487 and waits for clients to connect to it.

4. **AutoConv.java**
   This class is the main method of the application and therefore it extends Activity.

The method that simulates a headset click for answering and hanging up a phone call is shown below:

```java
private void answerPhoneHeadsethook(Context context) {
  // Simulate a press down of the headset button
  Intent buttonDown = new Intent(Intent.ACTION_MEDIA_BUTTON);
  buttonDown.putExtra(Intent.EXTRA_KEY_EVENT, new KeyEvent(KeyEvent.ACTION_DOWN, ↩
       KeyEvent.KEYCODE_HEADSETHOOK));
  context.sendOrderedBroadcast(buttonDown, "android.permission.CALL_PRIVILEGED")

  // Simulate a press up of the headset button
  Intent buttonUp = new Intent(Intent.ACTION_MEDIA_BUTTON);
  buttonUp.putExtra(Intent.EXTRA_KEY_EVENT, new KeyEvent(KeyEvent.ACTION_UP, ↩
      KeyEvent.KEYCODE_HEADSETHOOK));
  context.sendOrderedBroadcast(buttonUp, "android.permission.CALL_PRIVILEGED");
}
```

The length of an audio file is calculated and sent to the phone under test through socket communication. The code below shows how the mobile phone requests the file length and sets the length, which is used later to decide the conversation time.

```java
server.send2client("Phone ready. Audio File Length Request:");
String time = server.getFromClient();
((myServer) this.getApplication()).setConversationTime(time);
server.send2client("Received. Waiting for incoming call...");
```

When the time is set, it is then used by the class AutoAnswerIntentService.java. After a phone call is received and answered the conversation duration lasts according to the time that was set. The code below shows how conversation time is used:

```java
// Answer a phone call
answerPhoneHeadsethook(context);
try {
    // Wait a certain amount of time: conversation length
  String time = ((myServer)getApplicationContext()).getConversationTime();
  Thread.sleep(Integer.parseInt(prefs.getString("delay", time)) * 1000);
  // Hang up
  answerPhoneHeadsethook(context);
} catch (InterruptedException e) {}
```

The application found in the reference phone is called AutoCall and has only one class, AutoCall.java. This class makes an auto call to a specific phone number which is a constant number that only can be changed in the source code.

4.2.3 SQAS Controlling Interface

The application found on the computer is called SQAS Controlling Interface, SCI. This application is the core of the whole conversation and audio processing process. The purposes of this application are:

- TCP Client
- Device Detection
- Event Synchronization
- Phone Triggering
- Audio File Information
- Audio File Processing
- Audio Playing
- Audio Recording
- Audio Comparison (PESQ)
- Graphical User Interface

The application consists the following four classes:

1. **theDevice.java**
   This class is responsible for aquiring the names of the mobile telephones when they are connected to the computer.

2. **theClient.java**
   This class implements a client. It is used to communicate with the server found on the mobile phone.

3. **theGUI.java**
   This class extends javax.swing.JFrame. It contains the application graphical user interface.

4. **audioFileProc.java**
   This class is responsible for the audio files processing. Furthermore the audio file comparison using a PESQ executable is found here. A sample code from this class is shown below.

```java
public void recordRight(String duration, String audioFileName) throws ↩
    IOException{ audioFileName = audioFileName.replace(".wav", "");

 String degraded = getResultsDir() + getLatestTestNr()+ "_RC_" + audioFileName +↩
      ".wav";
        // Recording from right channel:
        Runtime.getRuntime().exec("arecord -f S24_3LE -c 1 -r 44100 -d" + ↩
            duration + " -D record_right " +  degraded);
```

recordRight: *The following method records audio from the right channel of the sound-card and saves the recorded file in a pre-defined directory which contains all degraded audio files.*

```java

    public void pesqAlgRight(String originalAudioFileName, JTextArea txtArea)
            throws IOException, InterruptedException{
        int x = new Integer(getLatestTestNr()).intValue() - 1;
        String preLastTestNr = Integer.toString(x);
        String originalAudio = getAudioDir() + originalAudioFileName;
// Location of the PESQ executable:
        String pesqAlg = getClassesDir() + "PESQ_ALG ";

        // Degraded audio right channel which name is: x_RC_y (x: test number, y: file name)
        String degradedRight = getResultsDir() + preLastTestNr + "_RC_" + originalAudioFileName;
        String cmdRight = pesqAlg + originalAudio + " " + degradedRight + " +16000";

        Runtime run = Runtime.getRuntime();
        Process pr = run.exec(cmdRight);
        pr.waitFor();

        BufferedReader buf = new BufferedReader( new InputStreamReader( pr.getInputStream() ) ) ;
        String line;
        txtArea.removeAll();
        while ( (line = buf.readLine() ) != null ){
              System.out.println(line) ;
              txtArea.append(line + "\n");
        }
    }
```

pesqAlgRight: *The following method performs the PESQ algorithm and prints out the results to the text area txtArea.*

```java
public class theClient {

private Socket clientsoc;

  public theClient(){

   }

   public boolean createSocket(int socket){
        try {
            clientsoc = new Socket("localhost", socket);
        } catch (UnknownHostException e) {
            return false;
            //e.printStackTrace();
        } catch (IOException e) {
            return false;
            //e.printStackTrace();
        }
        return true;
   }

   public void send2Server(String data){
        try {
           PrintWriter out = new PrintWriter(
               clientsoc.getOutputStream(), true);
           out.println(data);
       } catch (IOException e) {
           e.printStackTrace();
       }
  }

  public String getFromServer(){
       String data=null;
       try {
           BufferedReader in = new BufferedReader
           (new InputStreamReader(clientsoc.getInputStream()));
           data = in.readLine();
       } catch (IOException e) {
           e.printStackTrace();
       }
       return data;
  }

}
```

theClient: *This is how the client works which is used for communication between the phones and computer. This is the whole class.*

4.2.4 Automatic process

As mentioned before the test-rig consists of two mobile phones and a computer. The reference phone is only used to transport the audio signal to the phone under test and no measurements are performed on the phone it self, instead the phone under test is being measured. Therefore all audio processing is switched off in the reference phone so that the audio signal passes the phone without any significant change.

Both the reference phone and the phone under test have the applications AutoConversation and AutoCall installed on them which makes the whole process automatic. As aforementioned, the SCI application is responsible for the synchronization of all the modules. The applications found on the mobile phones are assumed to be already installed before the start of the testing process. Both applications on the reference phone and on the phone under test are triggered by SCI in an appropriate sequence. When plugging in both mobile-phones via USB and running SCI, the application automatically detects the phones and their respective names. Figure 5.1 shows how SQAS setup looks like. A more detailed sequence of how SQAS works is shown in the sequence diagram on the next page.

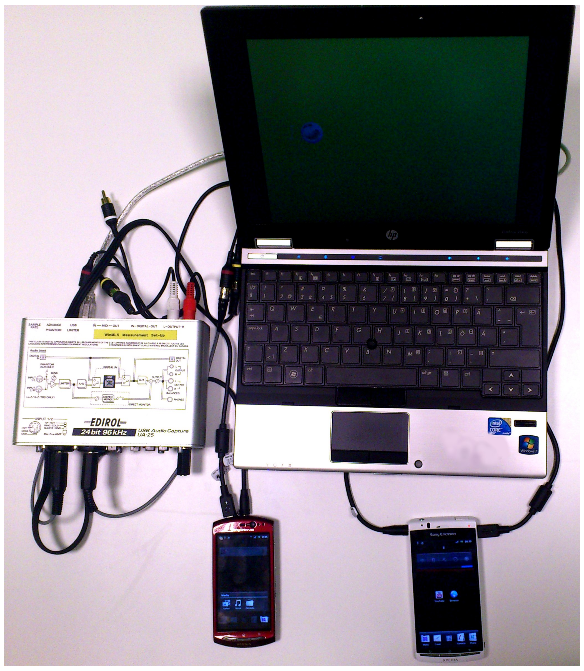
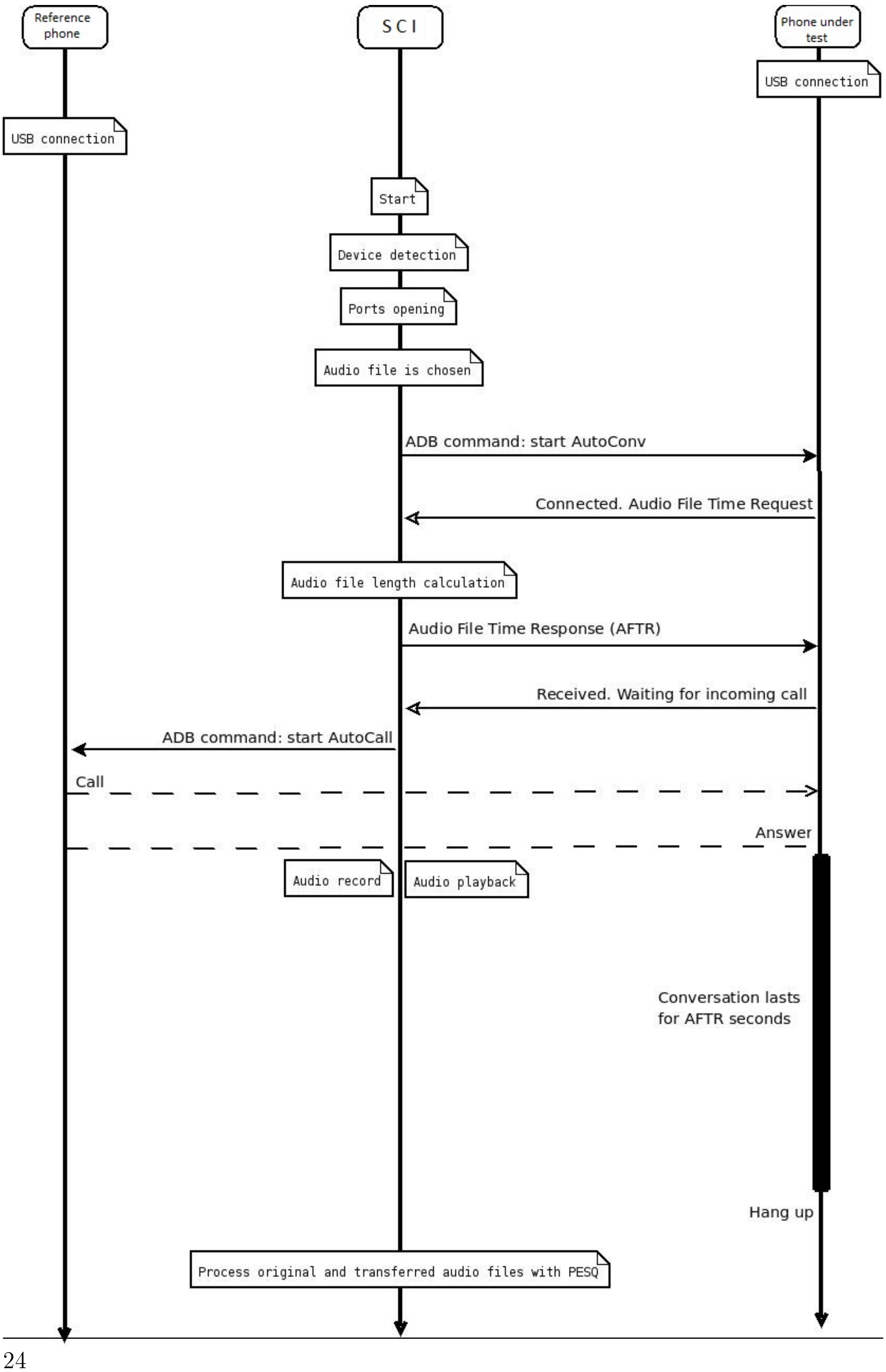


Figure 4.1: The test-rig setup


Reference phone
SCI
Phone under test
USB connection
USB connection
Start
Device detection
Ports opening
Audio file is chosen
ADB command: start AutoConv
Connected. Audio File Time Request
Audio file length calculation
Audio File Time Response (AFTR)
Received. Waiting for incoming call
ADB command: start AutoCall
Call
Answer
Audio record
Audio playback
Conversation lasts for AFTR seconds
Hang up
Process original and transferred audio files with PESQ

4.3 Hardware

After the applications on the telephone were finished and ready to use, the next step was to fix hardware connections, audio processing on the computer and audio playback and record.

4.3.1 Connections

The connection of the channels

The external sound-card that has been used is called Edirol UA-25. The reason that an external sound-card was used and not the internal one in computer is because there are much more options and settings that can be adjusted. For example gain adjustment and more output and input channels than the internal sound-card found in the computer.

Figure 4.2: Edirol UA-25

The are four channels in the sound-card two of which are used for audio playback and two for audio recording. A RCA connector has been used to connect a mobile phone to the sound-card. As known RCA connector has three plugs at one of its ends, two for stereo audio and one for composite video. The last one has been used in this case as a microphone by extending it with a special cable extension that is supplied with a resistor, so that the telephone interprets this connection as a head seat.

At the other end of the RCA connector there exists a 3.5 mm TRS connector which was connected to the mobile phone. The sound-card itself is connected to the computer using a firewire/USB cable. Furthermore electrical transformers has been connected at the the output channels to avoid ground loops that can result in unwanted noise and audio interference. Figure 5.2.0 show the used sound-card. Moreover, the figure below shows the connections between all hardware modules:

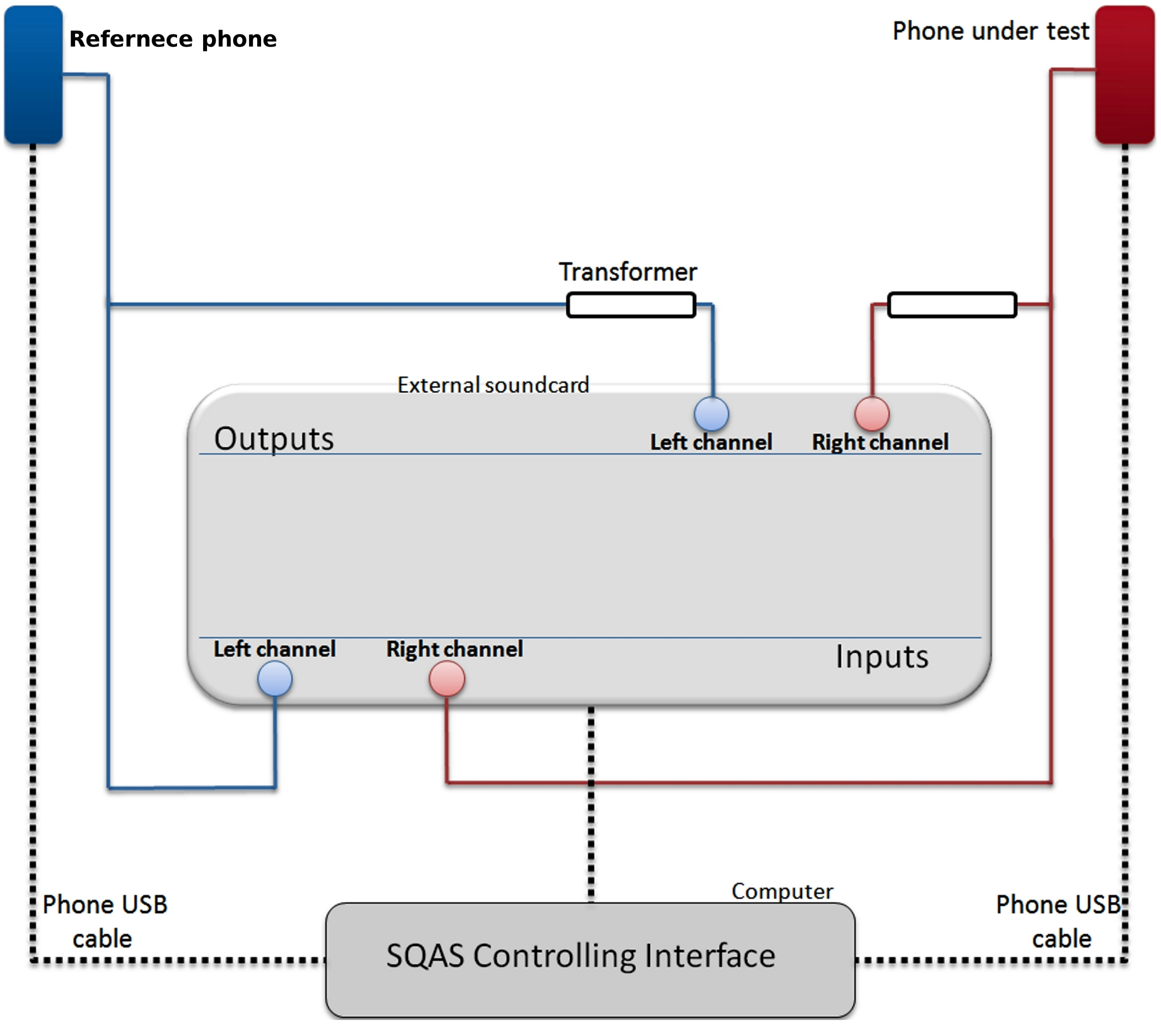


Figure 5.2.0: *The connection of hardware components in SQAS*

Simultaneous audio playback and record

Subsequent to the stage of connecting the cables to the sound-card, the next step was to apply the playback and recording of a conversation trough the sound-card. Because playback and recording is done via the same sound-card and twice, one output channel and one input channel must be used simultaneously. In other words two channels must be disabled when the other two channels are functional and vice verse. As a beginning that was done manually using a free and open source audio application called Audacity. With Audacity the playback and recording was first tested with one mobile phone. After that a more realistic scenario was performed which was connecting both mobile phones to the sound-card then starting a conversation.

In order to perform a realistic test of the audio quality delivered by a mobile phone, a two way test technique should be achieved. That is the audio quality during "up link" when the phone receives data and the audio quality during "down link" when the phone sends data. That should be tested to give a perspicuous result for the delivered audio quality by the phone under test. During up link only two channels must be listened to, and the other two channels should be disabled. The same applies during the down link but with opposite channels.

The following two pictures describe up link and down link principles. Furthermore which channels should be disabled during up link and down link are shown.

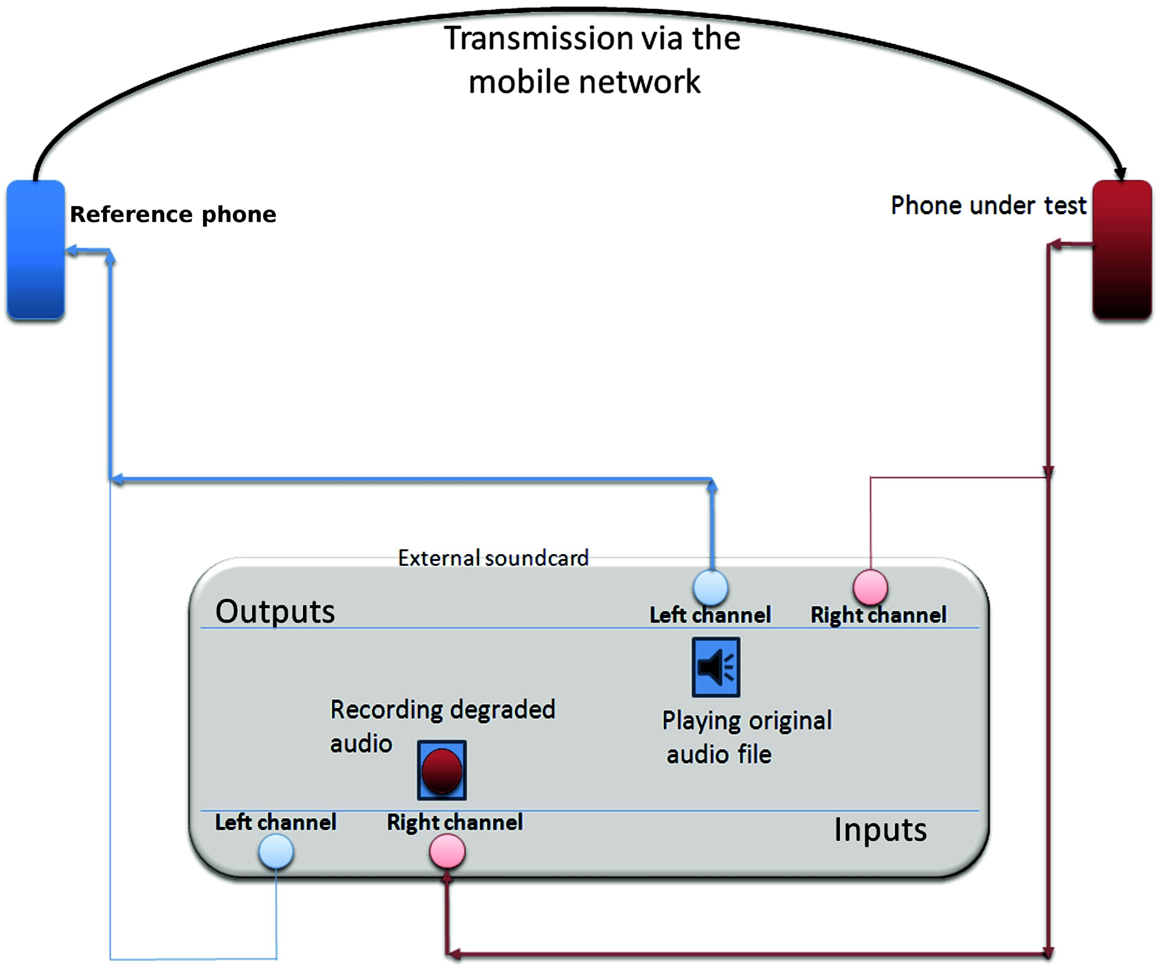


Figure 5.2.1: *Audio flow showing up link measurment*

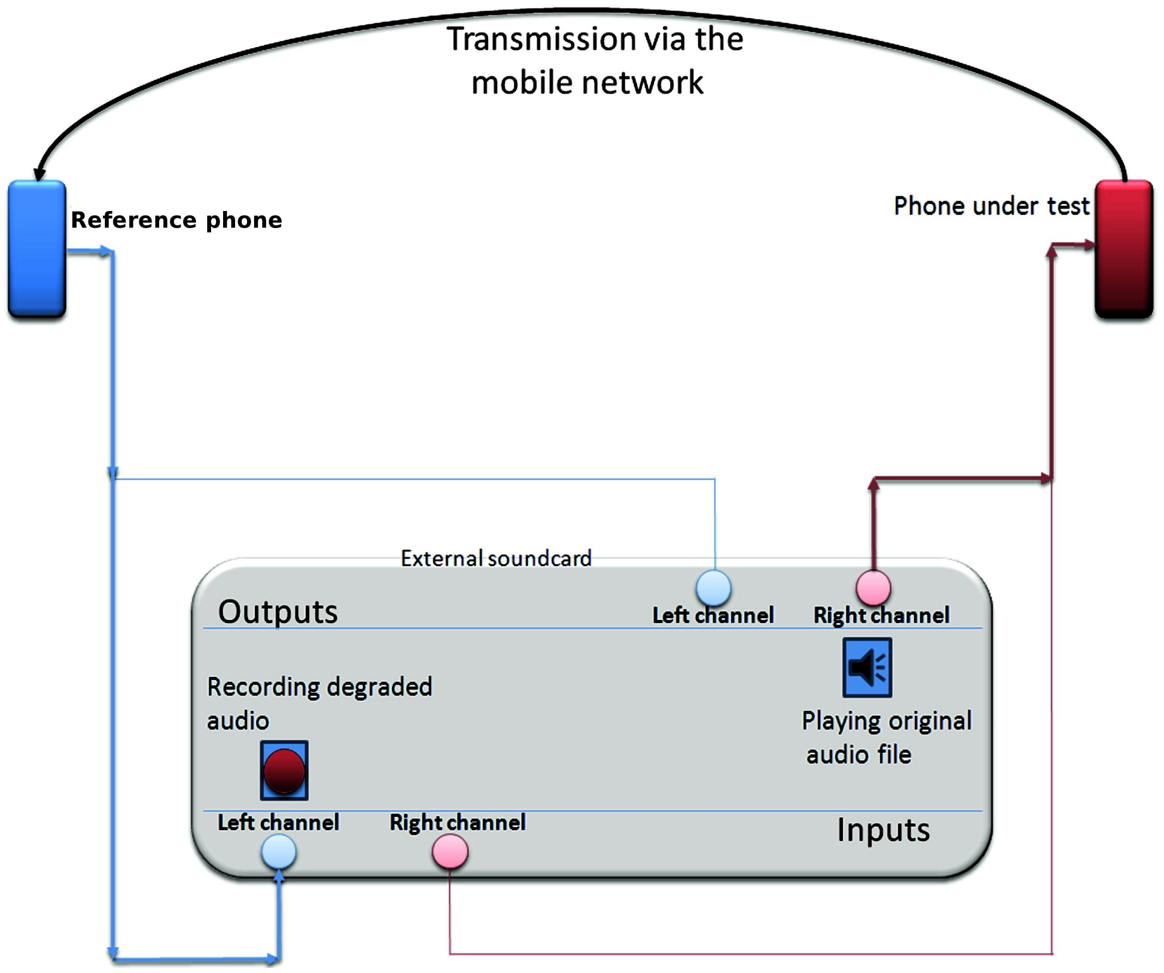


Figure 5.2.2: *Audio flow showing down link measurment*

The recording process

Recording is done in the computer using "arecord" which is a command-line sound-file recorder for the ALSA sound-card driver. The reason arecord was chosen is due to its ability to support several file formats and multiple sound-cards with multiple devices. Furthermore, this application is console based(command-line) which gives opportunity to record programmatically. The sound is captured from two input channels depending on what is being tested, up link or down link. The right channel of the sound-card has been used for up link and the left channel has been used for down link. In order for ALSA to distinguish between left and right channels and to record from the required channel (mono), a special ALSA plug-in was used. The plug-in is called "dsnoop" and is presented as virtual interfaces. These virtual interfaces are defined in a hidden file with the name ".asoundrc" which is a configuration file for ALSA drivers. Special dsnoop interfaces has been defined according to the following:

```
pcm.record_left {
 type dsnoop
 ipc_key 234884
 slave {
  pcm "hw:0,0"
  channels 1
 }
 bindings.0  0
}
pcm.record_right {
 type dsnoop
  ipc_key 2241234
 slave {
  pcm "hw:0,0"
  channels 1
 }
 bindings.0  1
}
```

dsnoop: *Dsnoop interface is used as an extra function in ALSA to enable multichannel recording.*

As a result audio is recorded from left channel of the sound-card using Linux terminal as follows:

arecord f S24_3LE c 1 r 44100 d 5 D record_left sample_1.wav

The playback process

Playback is done in the computer using "aplay" which is also a command-line sound file for the ALSA sound-card driver. Unlike arecord, aplay does not provide special plug-ins where a specific channel can be selected for playback. So in order to playback an audio file from a specific channel on the sound-card, the original audio file must be a mono left audio file or a mono right audio file. Otherwise the degraded audio file will contain the original audio and the degraded one, which is not desirable.

4.4 Summary

4.4.1 The signal interference

When the test-rig was set up and ready, some manual recording tests were performed and the results were good. The MOS value was 2.5 and would have got higher if the signal was tuned. When doing test using SQAS the result went down to 1.7. The sound was very different of course from the sound that was manually recorded were interference in the signal could be heard very clearly.

The interference occurred due to the USB connections to the computer and the telephone. The USB cables that were used for the communication between SCI and mobile applications are very critical for the whole system to work.

Figure 4.3: *XLR transformer*

Attempts were performed to remove interference in the signal, such as trying to use other cables, USB hubs, connect one USB cable to another computer, doing tests with different phones and building balanced connectors for the inputs. Additionally several solutions to overcome this problem were proposed, such as changing the way of communication to WiFi or bluetooth instead USB. Before trying to apply these changes a final attempt was made to get rid of the interference because it was known that this was caused by some sort of "ground-loop" in the connections.

It was concluded that the best solution was to use a XLR in-line transformer unit LL1584 which is designed for breaking up ground loops for balanced-to-unbalanced conversion in mobile or stationary audio systems at the output of the sound-card. Fig. 5.3 shows the transformer that has been used.

Test was applied after breaking the interferance in the signal and the MOS values went up from 1.7 to 3.9. The MOS value could even been higher if the signal power was correctly tuned to 1mv (-60dBV) on the output of the sound-card.

Chapter 5

Conclusion

To compare an audio file transferred through a mobile telephone network a robust test-rig consisting of hardware and software modules has been built. The possibility to accomplish the comparison in a flexible and automatic way has been proven with SQAS.

The hardware part of the rig included a Linux based computer, two Android mobile phones and an external sound-card. The software part included two Android mobile applications and a computer Java application. An algorithm that is based on mathematical calculations and perceptual of the human hearing system has been used for the comparison of an original and a degraded audio signals. This algorithm is an ITU standard and is called PESQ.

Results of the SQAS are presented as numerical values that vary from -0.5 to 4.5 which describe the audio quality difference between two audio files one of which effected by the mobile network and various components in between. These results may not be very accurate and do not indicate where in the components chain the quality is degraded. However, SQAS presents primary audio quality results which can give the tester a general overview of the degraded audio quality before proceeding to more advanced tests. Moreover, the test sequence is designed so that the tester can smoothly and repeatedly perform it with very short time intervals between two successive tests. Additionally, the overall process from mobile phone calling to audio comparison is done in an automatic manner with minimal tester/computer interaction.

Chapter 6

Future recommendations

The system that has been built can be furthermore developed and various functions can be added to make it more stable and effective. For example, the use of newer algorithms from the ITU can result that tests can be expanded to other areas than speech assessment or give better results. As late as 2011, a new algorithm is available for the public for measurement of voice calls and a direct successor to PESQ. Because it just arrived onto the market, it has not been possible to examine it. However, as a theoretical study made of the information available from Opticom / ITU and based on that, it seems very promising.

The main goal of this thesis has been to deliver a system that measures speech audio quality difference between two audio files. That was done with the help of the PESQ algorithm. However there are more audio fields that can be measured such as the playback quality of a phone. For this purpose the best choice is to use ITU's algorithm PEAQ which has a universal use and can be used for both playback and perhaps voice call. However, the PEAQ algorithm is not as reliable as PESQ when it comes to voice calls due its inaccurate MOS values, but for playback it should operate at an acceptable manner.

6.1 Expand the algorithm use

6.1.1 POLQA

POLQA is introducing the next-generation mobile voice quality testing standard for HD-Voice, VoIP, 3G and 4G/LTE.

The new POLQA standard was in development since 2006 by the leading experts, as the result of a competition carried out by Study Group 12 of the ITU-T, with the objective of defining a technology update for PESQ/P.862. During its development more than 100,000 fixed, mobile and VoIP test calls were gathered by experts from all over the world, thus representing the largest and most accurate database ever used in the development of a perceptual quality algorithm.

POLQA – an acronym for “Perceptual Objective Listening Quality Analysis” - will offer a new level of benchmarking capa- bility to determine the voice quality of mobile network services. [ref.6]

POLQA the PESQ successor

With more than 20.000 licensed test units, the Perceptual Evaluation of Speech Quality (PESQ/P.862) algorithm, has evolved as the most successful industry standard for voice testing since its definition in 2000. Naturally, voice coding technology and IP based transmission standards have further advanced, so the timing is appropriate for an update of the state-of-the-art benchmarking method. POLQA, the designated successor for PESQ was approved in 2011 as the new ITU-T Standard P.863. [ref.6]

Measurement Accuracy of POLQA

In the super-wideband mode, POLQA covers an extended audio frequency range up to 14 kHz, suitable to assess modern HD-Voice applications. Apart from the 2005 wideband extension, PESQ originally was limited to 'vintage' narrowband telephony applications (POTS), only. Consequently, POLQA would clearly be the choice for any modern benchmark that targets at wideband and super-wideband voice services. For "classical" narrowband use cases like mobile network benchmarking, POLQA clearly offers a huge advance in measurement accuracy. [ref.6]

POLQA As Android Application

Right from the start of POLQA, the new ITU-T Recommendation P.863, there is a library toolset in a mobile form factor for Android Smartphones. The POLQA Q-App complements PESQ/P.862 Q-App for Android. Q-Apps are available under OEM license to mobile manufacturers, walk drive test tool vendors, system integrators and network operators. [ref.6]

Figure 6.1: An illustration of POLQAS Q-app

6.2 SCI recommendations

Various improvements can be done to the SCI Java application but due to lack of time these improvements could not be applied. The following can be done to improve the usability of SCI:

- More flexible graphical user interface.
- Ability to test a whole folder containing audio test files.
- Saving mobile phone numbers.
- Auto detection of mobile phone number.

6.3 Hardware recommendations

As the operating system that was used is Linux based it is almost impossible to control the external sound-card with a virtual mixer controller. A virtual controller would make the switching of channels much easier during playback of audio. Another drawback when using an external sound-card is that the controller knobs must not be physically changed. Any change would effect the measurements and would give false results. Furthermore, it is quite hard to tune the card by selecting a nominal level of signal strength. Despite that it is fully possible to get good results if right adjustments are used.

Thus, it is recommended to use an internal sound-card that can be controlled with a virtual mixer controller. The card which is recommended to use is "ESI Juli@". Juli@ (pronounced Juliette) which is a PCI-based sound-card with two analog inputs and two outputs. Juli@ provides a sound resolution to the entire 24bit 192kHz for both recording and playback, whether the analog or digital connections are used. Julia is equipped with both RCA (unbalanced) and TRS (balanced).

Figure 6.2: *The ESI JULi@ internal sound-card*

Chapter 7

Bibliography

7.1 Web Sources

[1] OPTICOM PESQ MANUAL $http : //www.opticom.de/technology/pesq.html$

[2] The PESQ Algorithm as the Solution for Speech Quality Evaluation on 2.5G-3G Networks

[3] Opticom PESQ tech-spec,$http : //www.opticom.de/download/SpecSheetPESQ_05 - 11 - 14.pdf$

[4] *“Audio Quality Assessment Techniques – Edward Jones and Martin Glavin.“*

[5] $http : //www.itu.int/rec/R - REC - BS.1387/en$

[6] $http : //www.opticom.de/download/MR_Opticom_MesseNewsLetter_2011 - 02 - 02.pdf$

7.2 Scientific Reports

[7] Case study of PESQ performance in live wireless mobile VoIP environment, Zizhi Qiao

7.3 Oral Sources

[8] Ove Edfors *"Professo, Deparment of Electronics at Lund University"*

[9] Bertil Larsson *"Adjunct lecturer, Deparment of Electronics at Lund University"*

[10] Björn Gröhn, *"Staff Systems Engineer at SEMC"*

[11] Nedelko Grbic, *"A.Professor, Department of Signal Processing at BTH"*

[12] Peter Isberg, *"Audio Aucistics Specialist at SEMC"*

[13] Martin Eriksson, *"Audio Aucistics, Senior Engineer at SEMC"*

[14] Adis Bjelosevic, *"Audio Aucistics, Engineer at SEMC"*

User Guidelines

The phone number of the mobiles phone under test are constant and can only be changed in the source code of the Android application (AutoCall).

Conditions

- The Operating System which is used and is recommended to use is Linux Ubuntu.
- ALSA drivers and plugs-ins like dsnoop, .asourcerc must be correctly configured.
- This system only works for mobiles running on Android with at least 2.1 update 1 platform.
- The (Mobile phone under test) should have the android application AutoConv installed.
- The (Reference mobile phone) should have the android application AutoCall installed.
- The (Reference mobile phone) must be set to USB debug mode.
- The (Mobile phone under test) should be connected to the USB computer port before connecting the (Reference mobile phone).
- The test audio files should be in WAV format in a length between 6 to 20 seconds. Sampling rate should be between 8 and 16 kHz.
- Before starting a test the screen of the mobile phones should be turned off.
- Test audio files should be located in the directory Audio/Test_Audio_Files.
- Recorded (degraded) audio files are located in the directory Audio/Results.
- Right and left test audio files should be located in the directory Audio/Test_Audio_Files/Right_Channel and Audio/Test_Audio_Files/ Left_Channel respectively.

Appendix

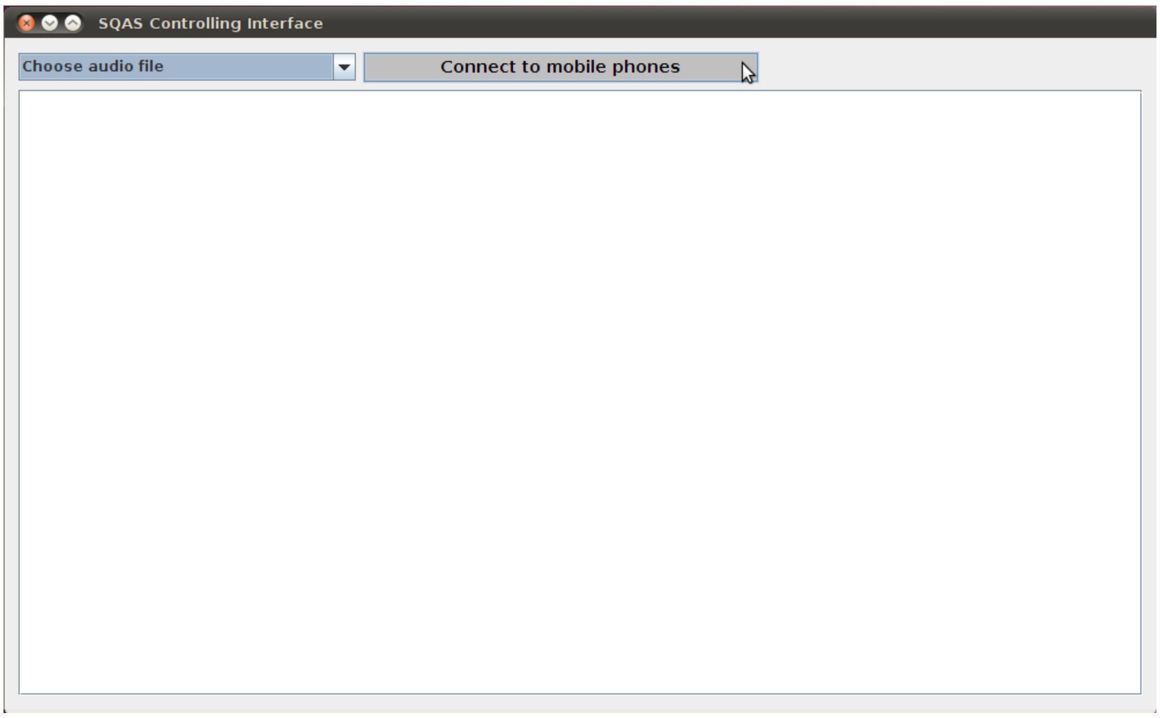
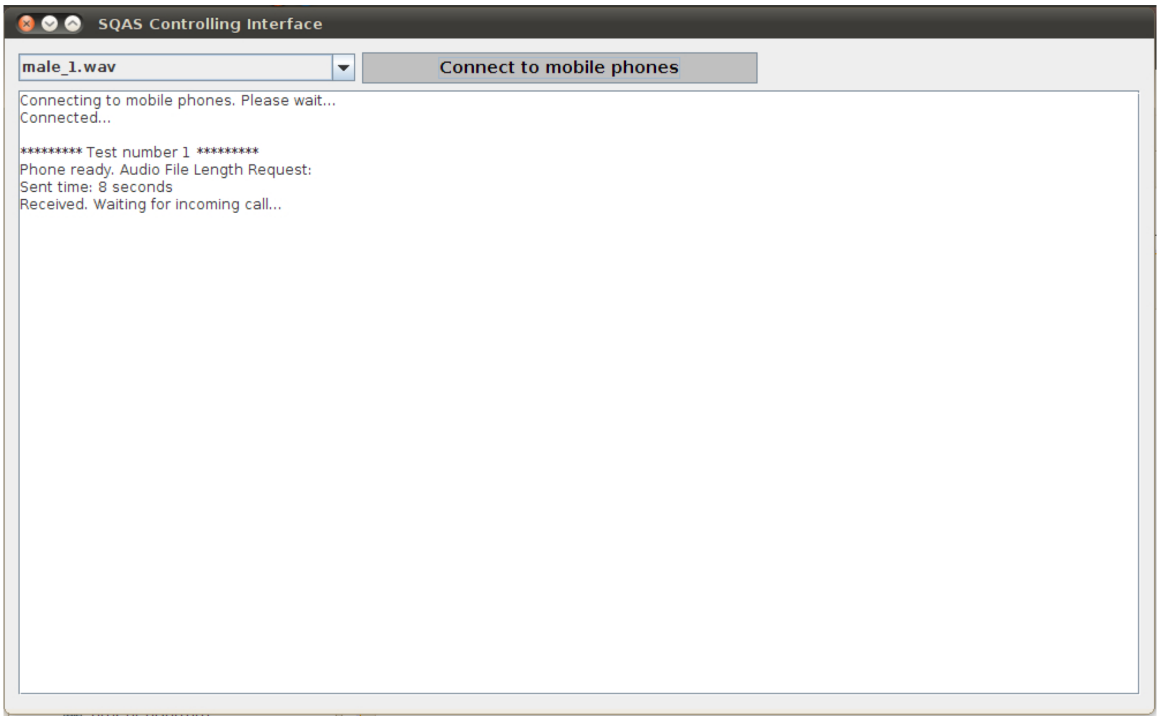


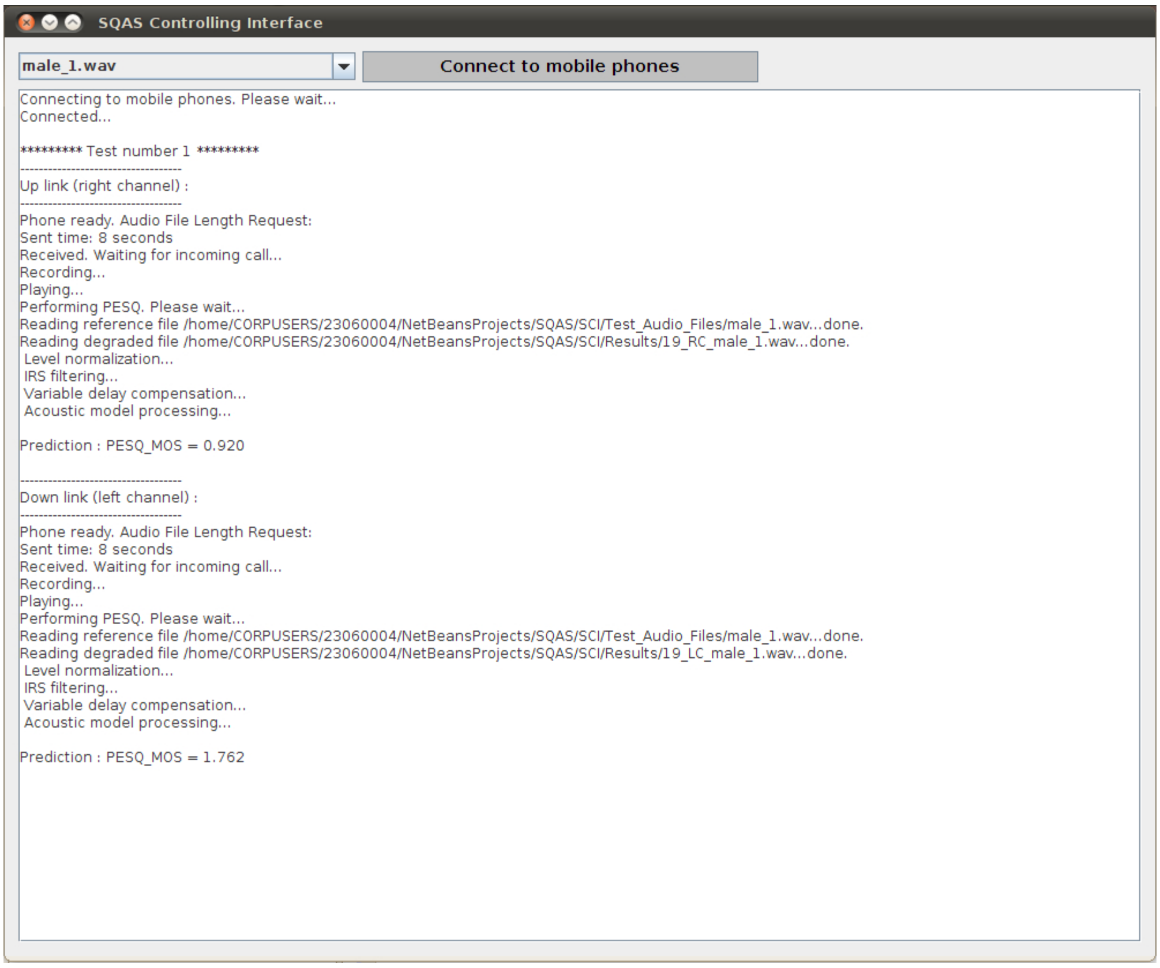
Figure: *Graphical Interface of SQAS, selecting a test-file*



```
Connecting to mobile phones. Please wait...
Connected...

********* Test number 1 *********
Phone ready. Audio File Length Request:
Sent time: 8 seconds
Received. Waiting for incoming call...
```

Figure: *Graphical Interface of SQAS, system in progress in analyzing the sound quality*



```
Connecting to mobile phones. Please wait...
Connected...

********* Test number 1 *********
----------------------------------
Up link (right channel) :
----------------------------------
Phone ready. Audio File Length Request:
Sent time: 8 seconds
Received. Waiting for incoming call...
Recording...
Playing...
Performing PESQ. Please wait...
Reading reference file /home/CORPUSERS/23060004/NetBeansProjects/SQAS/SCI/Test_Audio_Files/male_1.wav...done.
Reading degraded file /home/CORPUSERS/23060004/NetBeansProjects/SQAS/SCI/Results/19_RC_male_1.wav...done.
 Level normalization...
 IRS filtering...
 Variable delay compensation...
 Acoustic model processing...

Prediction : PESQ_MOS = 0.920

----------------------------------
Down link (left channel) :
----------------------------------
Phone ready. Audio File Length Request:
Sent time: 8 seconds
Received. Waiting for incoming call...
Recording...
Playing...
Performing PESQ. Please wait...
Reading reference file /home/CORPUSERS/23060004/NetBeansProjects/SQAS/SCI/Test_Audio_Files/male_1.wav...done.
Reading degraded file /home/CORPUSERS/23060004/NetBeansProjects/SQAS/SCI/Results/19_LC_male_1.wav...done.
 Level normalization...
 IRS filtering...
 Variable delay compensation...
 Acoustic model processing...

Prediction : PESQ_MOS = 1.762
```

Figure: *Graphical Interface of SQAS, results are being printed out to the user*