

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

RECOGNITION AND SEARCH IN SKYPE CALLS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

PAVEL TOMÁŠEK

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZPOZNÁVÁNÍ A VYHLEDÁVÁNÍ VE SKYPE HOVORECH

RECOGNITION AND SEARCH IN SKYPE CALLS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

PAVEL TOMÁŠEK

VEDOUcí PRÁCE

SUPERVISOR

Doc. Dr. Ing. JAN ČERNOCKÝ

BRNO 2008

Abstrakt

Cílem projektu je rozpoznávání Skype hovorů. Jedním ze záměrů projektu je nalezení kvalitního nahrávače Skype hovorů. Následuje zpracování signálu (segmentace řeč/ticho) a samotné rozpoznávání řeči. Dalším cílem je výběr takového přehrávače, jež splňuje veškeré potřeby projektu. Podstatnou částí práce je i implementace indexace nahrávek umožňující vyhledávání. Tento projekt je tzv. 'proof-of-concept' a přibližuje tak jeden ze způsobů možného využití rozpoznávání řeči.

Klíčová slova

zaznamenávání Skype hovorů, zpracování signálu, segmentace, rozpoznávání plynulé řeči s velkým slovníkem, indexování, vyhledávání, multimediální prohlížeč

Abstract

The project aims at a system for recognition of Skype calls. One of the necessary project focuses is finding out a high-quality Skype recorder. Signal preprocessing (segmentation of speech/silence) and speech recognition follows. Another project interest is in choosing a presentation software, which meets the project needs. Essential part of this work is also indexing and search implementation. This project is a proof-of-concept and shows one of the ways of speech recognition utilization.

Keywords

Skype Call Acquisition, Signal Processing, Segmentation, Large Vocabulary Continuous Speech Recognition, Indexing, Searching, Multi-media Browser

Citace

Pavel Tomášek: Recognition and Search in Skype Calls, bakalářská práce, Brno, FIT VUT v Brně, 2008

Recognition and Search in Skype Calls

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením docenta Černockého a že jsem uvedl všechny literární prameny a publikace, ze kterých jsem čerpal

.....

Pavel Tomášek

May 6, 2008

Poděkování

Poděkování patří především docentu Černockému a všem dalším členům skupiny Speech@FIT za vedení a pomoc při studiu.

© Pavel Tomášek, 2008.

Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 3 |
| 2 | Project Analysis | 4 |
| 2.1 | Project Scheme | 4 |
| 3 | Description of Particular Parts of Project | 6 |
| 3.1 | Skype Program | 6 |
| 3.1.1 | Skype History | 6 |
| 3.1.2 | Skype Protocol | 7 |
| 3.1.3 | Skype Pros and Cons | 8 |
| 3.1.4 | Skype Alternatives | 9 |
| 3.2 | Skype Calls Recording | 9 |
| 3.2.1 | Question of Illegality | 9 |
| 3.2.2 | Recording on MS Windows Platforms | 9 |
| 3.2.3 | Recording on UNIX-derived Platforms | 11 |
| 3.2.4 | Overview | 12 |
| 3.2.5 | Decision | 13 |
| 3.3 | Signal Preprocessing | 13 |
| 3.3.1 | Resampling and Audio Format Change | 13 |
| 3.3.2 | Phoneme Recognition | 14 |
| 3.3.3 | Signal Segmentation | 14 |
| 3.4 | Speech Recognition | 15 |
| 3.4.1 | Briefly about LVCSR | 15 |
| 3.4.2 | Tested LVCSRs | 15 |
| 3.4.3 | Technical Details | 16 |
| 3.4.4 | Output of LVCSR | 17 |
| 3.5 | Indexing and Search | 18 |
| 3.5.1 | Indexing | 18 |
| 3.5.2 | Search | 18 |
| 3.6 | Presentation Software | 18 |
| 3.6.1 | Multi-media Browser | 19 |
| 3.6.2 | Alternative Presentation Applications | 20 |
| 4 | Tests and Results | 22 |
| 4.1 | CPU needs of the Simplest LVCSR | 22 |
| 4.2 | Examples of Recognition Results | 23 |
| 4.2.1 | Recognition with the Simplest LVCSR | 24 |
| 4.2.2 | Recognition with the full-featured recognizer | 25 |

| | | |
|----------|------------------------------------|-----------|
| 4.3 | Examples of Search Results | 26 |
| 5 | Conclusions and Future Work | 28 |
| 5.1 | Future Work | 29 |
| 5.1.1 | Short Term | 29 |
| 5.1.2 | Long Term | 29 |
| 6 | Cookbook | 31 |
| 6.1 | For Users | 31 |
| 6.1.1 | Recording | 31 |
| 6.1.2 | Recognition | 31 |
| 6.1.3 | Presentation of Results | 32 |
| 6.2 | For Developers | 32 |
| 6.2.1 | Recognition System Structure | 32 |
| 6.2.2 | Software Versions | 32 |
| 6.2.3 | Compilation | 32 |
| 6.2.4 | Licensing | 33 |
| 6.2.5 | Hotline | 33 |
| | Bibliography | 34 |
| | Glossary | 36 |

Chapter 1

Introduction

We live in time when information may have very high value. If there is a way to quickly and efficiently dispose of the useful information, success may not be far.

This bachelor's thesis may outline future work with audio information. Calls recognition simplifies browsing of numerous records in general. One of the greatest features is possibility of searching for a word in the recognized recordings. In this thesis the reader will be informed about the implementation idea and about particular project parts. This project is a proof-of-concept and shows one of the ways of speech recognition utilization.

This work aims at a system for recognition of Skype calls and it was supported by IBM under project "*Improving business performance by increasing the productivity of conference calls*". The system was also presented in the *STUDENT EEICT 2008* (a competition of students' projects).

In the chapters I introduce possible ways of building project like this and what stands behind. In the following chapter, *Project Analysis*, **2**, is placed submission, project scheme and other general information about this project.

The third chapter, *Description of Particular Parts of Project*, **3**, contains descriptions of particular important parts of this project such as Skype program and calls acquisition, signal processing, indexing and search and the presentation software. The next chapter, *Tests-and-Results*, **4**, contains information about the processing speed, error rates of the system and examples of recognition including results of search. Chapter *Conclusions and Future Work*, **5**, contains summary, conclusions and also informs about a possible future of this project and enumerates realizable improvements. The last chapter, *Cookbook*, **6**, informs a reader how to start recording of Skype calls, run the system and how to display results. This is also a place for developers. Information about folder structure, versions of used software, licensing and hotline is written there.

This bachelor's work is based on research on Skype call recorder and recognizer, that was the subject matter of my semestral project.

Chapter 2

Project Analysis

The goal of this bachelor's thesis is to design and to develop complete system capable of Skype call processing, which includes all steps described below in more detail.

2.1 Project Scheme

As shown in figure 2.1, the system has the following important parts:

- Skype call recorder
- Skype call preprocessing
 - undersampling
 - phoneme recognition
 - logical segmentation
- continuous speech recognition
- indexing module (enables searching in the recording)
- searcher module
- presentation software

Particular descriptions are written in the following chapter *Description of Particular Parts of Project*, 3. Shortened information about this whole project can be found also in Czech in [Tomasek2008].

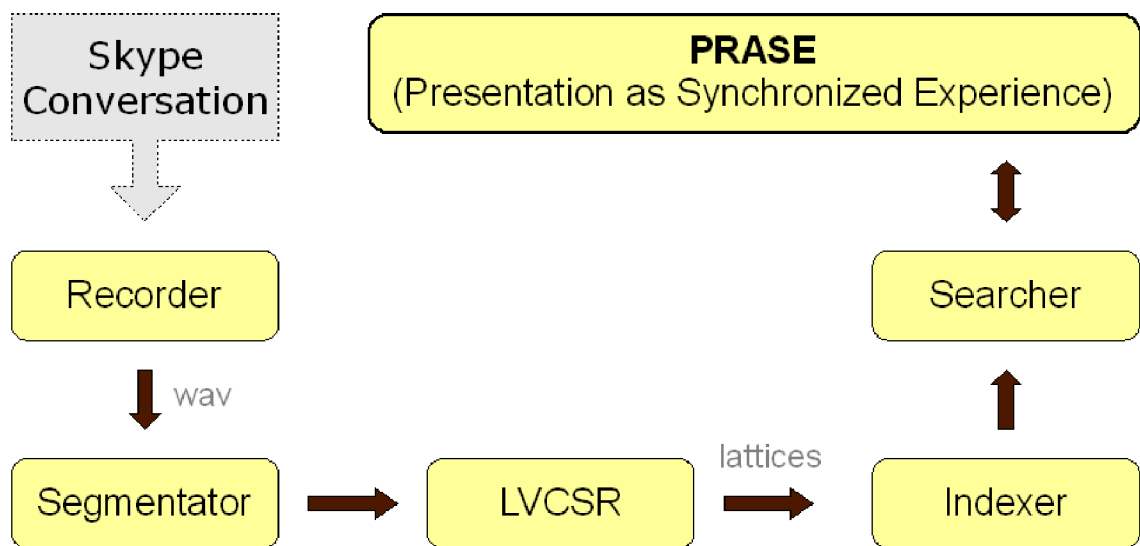


Figure 2.1: Project scheme

Chapter 3

Description of Particular Parts of Project

The particular parts of this project are *Skype Program* (section 3.1), *Skype Calls Recording* (section 3.2), *Signal Preprocessing* (section 3.3), *Speech Recognition* (section 3.4), *Indexing and Search* (section 3.5) and *Presentation Software* (section 3.6). All of them are described below. These sections are related to *Project Scheme* (section 2.1).

3.1 Skype Program

Skype is a cross-platform, multilingual, peer-to-peer communication program. Through this program users are able to communicate in three ways: instant messaging, calls and video-conference (with ability to circumvent firewalls¹). Communication is encrypted. Additional features are voicemail, file transfer and short message service. Skype program is capable of running on MS Windows (2000, XP, Vista and Mobile), Linux, OS X, 3SkypePhone, Nokia N800/N810 and other platforms. One of the reasons for the enormous expansion worldwide is also a fact, that Skype is available in 28 languages.

The GUI of Skype program is displayed in figure 3.1.

Skype allows calls among Skype users over the Internet free of charge. Skype also allows users to realize a communication from the Skype network to landlines and mobile phones for a fee. This service is called *SkypeOut*. *SkypeIn* is the other service, the opposite one. SkypeIn enables the Skype users to receive calls from landlines and mobile phones.

The authors of this software are the Swedish and Danish entrepreneurs *Niklas Zennström* and *Janus Friis* and a team of developers, for example Ahti Heinla, Priit Kasesalu and Jaan Tallinn. Skype Limited is a possession of *eBay*² since September, 2005.

To learn more visit Skype web site [Skype.com].

3.1.1 Skype History

Skype Limited was founded in 2003 (these information bases on [Skype.com, about pages]). First beta version for Windows was released in August, 2003 (first beta version for Linux released in 2004).

¹Firewall works like a network traffic inspector, permits traffic based on rules.

²eBay is an on-line marketplace, enabling local, national and international trade.



Figure 3.1: Snapshot of the Skype program (version 3.6.0.248) running under MS Windows XP

Another important year for Skype was 2005 when eBay purchased Skype. In the same year Skype introduces video-telephony.

Event of 100 billion registered users was celebrated in 2006. Nowadays (first quarter of the year 2008) there is over quarter of a milliard registered users.

During the history had Skype many problems. The biggest one happened on 16th and 17th of August, 2007 when the biggest drop-out happened. Skype users were unable to connect to Skype network in many countries. The high number of restarts (caused by Windows Update on *Patch Tuesday*³) affected Skype's network resources. This caused a flood of login requests combined with the lack of peer-to-peer network resources, which prompted a chain reaction that had a critical impact.

3.1.2 Skype Protocol

Skype uses a proprietary Internet telephony (VoIP⁴) network. The protocol has been made as a closed-protocol. Skype uses a peer-to-peer model rather than the more usual client-server model, what is the main difference between Skype and other typical VoIP programs.

³Patch Tuesday by microsoft.com, "When necessary, Microsoft provides a new security update on the second Tuesday of each month and publishes a bulletin to announce the update. Occasionally, updates are released more often."

⁴VoIP – Voice over Internet Protocol. It is the technology for transmitting voice conversations over dial-up or broadband connection to the Internet.

Another difference is that Skype creates decentralized infrastructure, which is less expensive than the centralised version of user directories. This also makes the Skype network scalable. By this way Skype uses the processing capacity of users' computers for network operation. According to Guha (2006) [Guha2006] some users' computers are used as *supernodes* (selection is made by Skype program and is based on some criteria – accessibility, a table of other supernodes – to make a well spread net and other conditions). This means that a portion of the user's Internet bandwidth is reserved and used by Skype program. Skype uses approximately hundreds of thousands of supernodes, program also uses special codecs to spare network and computing resources so the load is spread out.

A Skype client uses techniques similar to *passive waiting*. It listens on particular ports for incoming calls. The program maintains a table of other Skype nodes called *host cache*, uses wideband codecs, maintains a buddy list, encrypts messages end-to-end, and determines if it is behind a NAT⁵ or a firewall [Baset2004].

3.1.3 Skype Pros and Cons

Pros

- Easy to install and works without much configuration [Baset2004]
- Works behind typical network security devices, e.g. firewalls and NAT [Baset2004]
- Large community, popularity

Cons

- Skype's supernode activity
- Closed community (standard protocols such as SIP⁶ or H.323 are not used)
- Closed protocol and source – no possibility to create new Skype clients for example for using on special minor platforms
- Traffic is routed through unknown supernodes [Newton2006]
- Rapid spread of malicious files in Skype network [Newton2006]
- Lack of end-to-end service quality (Quality of Service by transit through the unknown supernodes can not be guaranteed)
- Inconvenient license – allows Skype Limited to change the license content without a notification and to cancel service to anyone. By agreeing to the license, user also “*grant permission for the Skype Software to utilize the processor and bandwidth of user's computer for the limited purpose of facilitating the communication between Skype Software users*” [Skypewww, about pages], [Newton2006]
- Variable call quality – based on changing supernodes quality in time
- Network security bypass firewalls and NATs – advantage for hackers [Newton2006]
- Difficult Skype communication limitation – this is a problem of network administrators world-wide, blockage is very difficult

⁵NAT is *Network Address Translation*, this is a network router function.

⁶SIP is *Session Initiation Protocol* used in many VoIP systems.

3.1.4 Skype Alternatives

Skype is not the only one VoIP-based communication application. There are lots of programs like Skype, which also enables text messaging, audio conversation and some of them with the possibility of videoconferencing.

The alternative applications are for example *AIM*, *Gizmo Project*, *Google Talk*, *I Hear U*, *iVisit*, *OoVoo*, *SightSpeed*, *WengoPhone*, *VZOchat* and *Yahoo! Voice*.

3.2 Skype Calls Recording

A great effort was spent on search for a sufficient Skype calls recorder.

To be able to demonstrate the system we need a software that produces two high quality mono WAV⁷ audio files or one stereo file with separable channels. One with local and the other with remote side of conversation. We also need the recordings in non compressed audio format. The recorder is preferred to be a freeware or even open-source project, which is better.

Number of MS Windows operating system users is greater than number of UNIX-derived operating system users. The focus is on ordinary user. This means that the recording part of this project should be and is preferred to be applicable in MS Windows operating system.

I have experimented with many applications capable of recording Skype calls. The experiments took place in the second half of the year 2007 and at the beginning of the year 2008. Therefore the tested programs are related to the versions of mentioned interval.

A few programs are introduced below. Complete overview of tested recorders is in table 3.1 and the final choice of a recorder is presented in the subsection *Decision* (3.2.5).

3.2.1 Question of Illegality

At this place is good to mention the possibility of law break.

To avoid breaking the law, users who want to record the conversations should check the actual law of user's country. It may or may not be illegal to record a conversation without an express consent of both involved parties. It just depends on country where the user and the called person live.

Attention: In the Czech Republic the conversation recording is legal only in case of a fore-going agreement of both communicating parties. In the other case this is considered to be a break of the law.

According to the Czech Civil Code 40/1964 [[CivilCode](#), § 12]:

“(1) Papers of personal character, portraits, visual shots and visual and audio records related to a physical person or his/her manifestations of a personal nature can be acquired or used only with his/her compliance.”

3.2.2 Recording on MS Windows Platforms

MX Skype Recorder

MX Skype Recorder is a specialised tool that cooperates with Skype and is easy to use. The recorder starts recording in the time of beginning of a Skype call. It is able to record

⁷WAV is a *Waveform* audio file format (a subset of Microsoft's RIFF) without compression with metadata in header (number of channels, sample rate, ...).

all Skype calls, and save them as mp3 or wav audio files in a separate channel.

Disadvantage is that this program is not a freeware. For testing is trial version (30 days and other limitations) downloadable, but full version costs 24.95 USD (at the beginning of the year 2008).

More information is on the project web site <http://www.skyperec.com/>.

Pamela Call Recorder

Also this software cooperates with the Skype program very well. The recorder starts recording in the time of beginning of a Skype call. It simply records Skype calls and also has other functions. With some limitations it can be used for free.

The only problem which prevented me from using this application at the time of testing (the end of the year 2007: version 3.1.0.30) was impossibility to record calls with separate or separable channels (local/remote).

More information is on the web site <http://www.pamcorder.com/>.

PrettyMay Call Center for Skype

PrettyMay cooperates very well with the Skype program. It can simply record Skype calls and has other useful functions (call center, answering machine). The recorder starts recording in the time of beginning of a Skype call.

On contrary to Pamela Call Recorder, this application is able to record all Skype calls, and save them as mp3 or wav audio files with a possibility to separate channels.

Disadvantage is that this program is not a freeware. For testing is trial version downloadable, but full version costs 24.95 USD (at the beginning of the year 2008), that is the same price as price of MX Skype Recorder.

For more information visit project web site <http://www.prettymay.net/>.

Skype Capture

This is a project of *Jiří Šimáček*, student of Faculty of Information Technology, Brno University of Technology [[Simacek2007](#)]. It is a specialised tool that cooperates with Skype and is easy to use. The recorder starts recording in the time of beginning of a Skype call. It is able to record all Skype calls, and save them as two separate mono wav audio files.

This is an open-source project. It is a simple console application which is able to communicate with Skype program where the communication is based on listening to Skype signals.

Skype Recorder

Skype Recorder is another easy to use tool for recording Skype audio conversations. The recorder starts recording in the time of beginning of a Skype call.

Also this software is not for free. For testing is trial version downloadable, but full version costs 13.96 USD (at the beginning of the year 2008).

The only problem which prevented me from using this application at the time of testing (the end of the year 2007: version 3.6.0.38) was impossibility to record calls with separate or separable channels (local/remote).

More information is on the web site <http://www.extralabs.net/skype-recorder.htm>.

Skype Recorder +

This software cooperates very well with the Skype program. It simply records Skype calls with high audio quality. The recorder starts recording in the time of beginning of a Skype call.

Disadvantage is that this program is not a freeware. For testing is trial version downloadable, but full professional version costs 34.95 USD (at the beginning of the year 2008).

Main problem which prevented me from using this application at the time of testing (the end of the year 2007: version 3.6.0.38) was impossibility to record calls with separate or separable channels (local/remote).

More information is on the web site <http://www.powergramo.com/>.

3.2.3 Recording on UNIX-derived Platforms

ALSA Plug-in

Advanced Linux Sound Architecture (ALSA) is an open-source Linux audio system. It supports all audio interfaces and uses user space library to simplify application programming. It also supports the Open Sound System (OSS) application interface and compatibility for OSS programs.

Another feature of ALSA system is a possibility of using plug-ins⁸. The recording is enabled by ALSA *arecord* program and ALSA plug-in written in *.asoundrc* file in user's home directory (or */usr/share/alsa/asound.conf*). The plug-ins written in those files enables users to work with Linux audio in higher functionality layer.

Information was provided by ALSA project homepage <http://www.alsa-project.org/>.

I successfully tested a few plug-ins so I was able to record the audio data that went through PC audio card.

I was inspired for example by Edward Coffey's pipe based recording script in mail archive on <http://www.mail-archive.com/alsa-user@lists.sourceforge.net/msg20262.html> and by Kapil Hari Paranjape's web page on http://www.imsc.res.in/~kapil/goodies/record_live.html (both accessed on 21st August, 2007).

However, when I started Skype and recording in this way, Skype-to-Skype communication crashed. Skype always reported, "*Problem with sound device!*"

Ecasound

Ecasound (by <http://ecasound.seul.org/ecasound/>) is a software package designed for multitrack audio processing. It can be used for simple tasks like audio playback, recording and format conversions, as well as for multitrack effect processing, mixing, recording and signal recycling. Ecasound supports a wide range of audio inputs, outputs and effect algorithms. Effects and audio objects can be combined in various ways, and their parameters can be controlled by operator objects like oscillators and MIDI-CCs.

Primary platform for running Ecasound is GNU/Linux. Ecasound can also be run on many UNIX-derived systems such as FreeBSD, Mac OS X and Solaris. Limited support for Windows is available through Cygwin⁹.

⁸Plug-in is a computer program add-on (a program or a text file that provides a very specific function), which interacts with a host application and extends it.

⁹Cygwin provides a Linux-like environment and a basic functionality for MS Windows operating system.

jack_capture

jack_capture is one of *JACK*¹⁰ based programs. This software allows users to record audio files. It is capable of capturing sound that is going out of audio card. The author is Kjetil S. Matheussen.

The program can be found on the web site <http://www.notam02.no/arkiv/src/>.

Skype-rec

This open-source software was made by *Nathan Poznick*. Skype-rec is a library and wrapper script to allow a user to record conversations using Skype Internet telephony software on a Linux system. Conversion to OGG or MP3 audio format is also supported.

It is able to record all Skype calls, and save them as two separate mono wav audio files. The recorder starts recording in the time of beginning of a Skype call.

The recorder disadvantage is in the recordings quality. The program was implemented to use OSS (Open Sound System), an older Linux sound system. Recording is enabled only when Skype uses the OSS (not ALSA). The bad recordings quality is a consequence of using OSS. Recordings content creaking and a little noise.

More information is on the web site <http://sourceforge.net/projects/skype-rec/>.

Virtual Audio Socket (VAS)

Virtual Audio Socket is a very simple virtual sound card which allows to sniff and forge Skype audio data. It emulates /dev/dsp functionality just enough to be useful for Skype 1.2.0.18. It is a work in progress since only capturing Skype audio is not the main goal. Because it is a work in progress (and it is a part of a larger project, which is a work in progress itself), it lacks full documentation.

The socket-like nature comes from the fact, that it sits in between Skype and another process (that can sniff or forge audio data). But it is not a socket, because it is tolerant to internal buffer underflows and overflows. Socket blocks writer if the internal buffer is full (overflow), and the reader is blocked if there are no data (underflow). Hence it behaves as a sound card driver. VAS returns zeros when underflow occurs and overwrites existing data if overflow occurs. This should imitate a real /dev/dsp functionality as expected by programs that use it [VASwww].

VAS is a successor of VAD¹¹. The recordings quality is nearly the same as if Skype was using real /dev/dsp. But as I mentioned above (about *Skype-rec* recorder 3.2.3), the OSS based recording does not fulfill the quality conditions, even newer Skype versions do not support OSS but only the ALSA sound system.

3.2.4 Overview

In the following table 3.1 is shown list of Skype call recorders described above (*Recording on MS Windows Platforms* 3.2.2, *Recording on UNIX-derived Platforms* 3.2.3).

¹⁰JACK, *JACK Audio Connection Kit*, by <http://www.jackaudio.org/>, “*JACK is a low-latency audio server, written for POSIX conformant operating systems such as GNU/Linux and Apple’s OS X. It can connect a number of different applications to an audio device, as well as allowing them to share audio between themselves. Its clients can run in their own processes (ie. as normal applications), or they can run within the JACK server (ie. as a ‘plug-in’.*”

¹¹VAD, *Virtual Audio Device*, is a proof-of-concept demo. It requires running multiple Skype instances with the same Skype ID at the same time and on the same machine.

First column contains shortened name of application, the second column informs whether application is executable in MS Windows or Linux operating system (in case of a Linux operating system, it also contains information about used sound system: ALSA/OSS). The third column indicates whether the application is able to create output audio file without compression (simple wav / raw file type), the fourth column indicates whether there is possibility to separate channels (or whether the channels are already separated) of the application output file.

And the last column informs about program licensing: shareware / freeware / open-source (GNU GPL¹²).

| Application | Platform | No compression | Separ. ch. | License |
|-------------------------|--------------|----------------|------------|-----------|
| <i>MX</i> | MS Windows | yes | yes | shareware |
| <i>Pamela</i> | MS Windows | yes | no | freeware |
| <i>PrettyMay</i> | MS Windows | yes | yes | shareware |
| <i>Skype Capture</i> | MS Windows | yes | yes | GNU GPL 2 |
| <i>Skype Recorder</i> | MS Windows | yes | no | shareware |
| <i>Skype Recorder +</i> | MS Windows | yes | no | shareware |
| <i>ALSA Plug-in</i> | Linux (ALSA) | yes | yes | GNU GPL 2 |
| <i>Ecasound</i> | Linux (ALSA) | yes | yes | GNU GPL 2 |
| <i>jack_capture</i> | Linux (ALSA) | yes | no | GNU GPL 3 |
| <i>Skype-rec</i> | Linux (OSS) | yes | yes | GNU GPL 3 |
| <i>VAS</i> | Linux (OSS) | yes | yes | GNU GPL 3 |

Table 3.1: Overview of Skype recorders

3.2.5 Decision

The need to have trouble-free software which is able to automatically cooperate with Skype program on MS Windows platform selected three products: *MX Skype Recorder*, *Pretty-May Call Center for Skype* and *Skype Capture*. From this selection only Skype Capture (described in subsection 3.2.2) is a small free of charge noncommercial open-source application, which were the reasons to choose the Skype Capture application.

3.3 Signal Preprocessing

The signal is anti-alias filtered and resampled to be used with the following recognizer. Before the recognition a segmentation into speech chunks has to take place. A voice activity detector (VAD) available in the Speech@FIT group based on reliable phoneme recognizer was used.

Detailed description is in the following subsections.

3.3.1 Resampling and Audio Format Change

The input recording is required to be in a specific sampling rate. The rate depends on the phoneme recognizer (8 kHz) and the speech recognizer (16 kHz).

¹²GNU GPL, *GNU General Public License*, is used by most GNU programs, and by more than half of all free software. <http://www.gnu.org/licenses/licenses.html#GPL>

The phoneme and the speech recognizer also requires RAW¹³ input audio files, therefore the recording must be saved in RAW audio format.

Therefore the input signal must be resampled to the desirable rates and saved as RAW audio type.

For these purposes I used the *SoX*¹⁴ tool in Linux.

SoX is applied to split channels of a stereo recording and is configured by parameters where is defined the input stereo wav audio file, output 16 bit signed raw audio audio file, change to 8 kHz sampling rate. The signals generation for phoneme recognition using SoX is shown in the following box (to produce 16 kHz signal for speech recognition we only change the -r parameter to 16000):

```
sox -c 2 -t .wav ./recordings/${record_name}.wav -c 1 -t .raw -r 8000 \  
-s -w ./recognizer/input/${record_name}_L.raw resample avg -1  
sox -c 2 -t .wav ./recordings/${record_name}.wav -c 1 -t .raw -r 8000 \  
-s -w ./recognizer/input/${record_name}_R.raw resample avg -2
```

3.3.2 Phoneme Recognition

Phoneme recognition is a very important process of signal processing. The more phoneme¹⁵ recognizer is accurate, the better results of speech recognition we have.

There are two types of phoneme recognizers:

- HMM (Hidden Markov Model) with GMM (Gaussian Mixture Models) probability distribution
- HMM (Hidden Markov Model) with ANN (Artificial Neural Network) hybrids

In this project the HMM with ANN option was chosen. All phoneme classes except silence are mapped to the 'speech' class [Schwarz2004, Schwarz2006]. Used Speech@FIT's phoneme recognizer is in version 2.14 (contains three neuron nets).

Input of such a phoneme recognizer is a raw audio file and the output is a *rec* file containing the phoneme data (starting and ending frame, phoneme). When running the recognizer, the phoneme set must be also specified.

After this phoneme recognition there is a need to make some little changes of the output *rec* files. We have to prepare them to the segmentation process. It means just an only a little modification of the *rec* files and joining speech pauses if they are in a row. After this, the output file type is *phn.lab*.

3.3.3 Signal Segmentation

To make this project capable of processing long recordings, the segmentation must be implemented. The partitioning take place in longer speech pauses. The segmentation process is based on the output of phoneme recognizer described in the previous subsection.

¹³RAW is an audio file format without compression and without any metadata (no header).

¹⁴SoX stands for *Sound eXchange* and is a command line utility that can convert between various formats of audio files.

¹⁵Phoneme is the smallest acoustic unit.

This section can be considered as a voice activity detector (VAD). It informs about signal segments with a speech and non-speech (labeled as a silence). We can imagine an utterance where a speaker stops short to take a deep breath to continue in his speech. This case is an example of two segments and a silence between.

There are two ways of segmentation. We can name the two approaches as a **hard** and a **soft** segmentation. Hard segmentation means dividing the original recording into separate files with speech chunks. The soft segmentation represents the other approach. While hard segmentation creates many files from the original one, the soft segmentation is a logical way of audio segmenting. It just creates a file containing frame intervals.

Input of such a segmentation is the name of processing raw audio file and phoneme recognition output – phn.lab file. The output is a .lab file containing division into silence and speech segments, *output.list* file, which contains the speech segments description including the starting and the ending frame with the input file location.

Other parameters set the maximum number of seconds per segment number of frames going to be appended to the beginning and to the ending as a buffer.

Example of segmentation:

```
cernoc_L_FROM_0000027_TO_0000117.raw=input/cernoc_L.raw[0000027,0000117]
cernoc_L_FROM_0000508_TO_0000904.raw=input/cernoc_L.raw[0000508,0000904]
cernoc_R_FROM_0000150_TO_0000466.raw=input/cernoc_R.raw[0000150,0000466]
cernoc_R_FROM_0001428_TO_0001503.raw=input/cernoc_R.raw[0001428,0001503]
```

3.4 Speech Recognition

Speech recognizer is the key part of this project.

3.4.1 Briefly about LVCSR

Large Vocabulary Continuous Speech Recognizer (LVCSR) is a system including an acoustic model (determines phonemes occurrence likelihood) a pronunciation dictionary and a language model (contains word sequences probabilities). The architecture is shown in figure 3.2.

The quality of recognition depends (among others) on the quality of processed signal – background noise, type of microphone, compression, sampling rate, ...

3.4.2 Tested LVCSRs

The recognition engine is derived from Speech@FIT's work in the AMI(DA)¹⁶ project, where Speech@FIT participates at the development of LVCSR for meetings [Hain2006].

First tests were done with a complex speech recognizer (typically contains fifty thousand words for English) which needed several ten times more time to process the recording than real-time. Therefore, a more compact recognizer with limited vocabulary and language model was defined and tested: the language model was a simple bi-gram and simpler

¹⁶AMIDA stands for *Augmented Multi-party Interaction with Distance Access*.

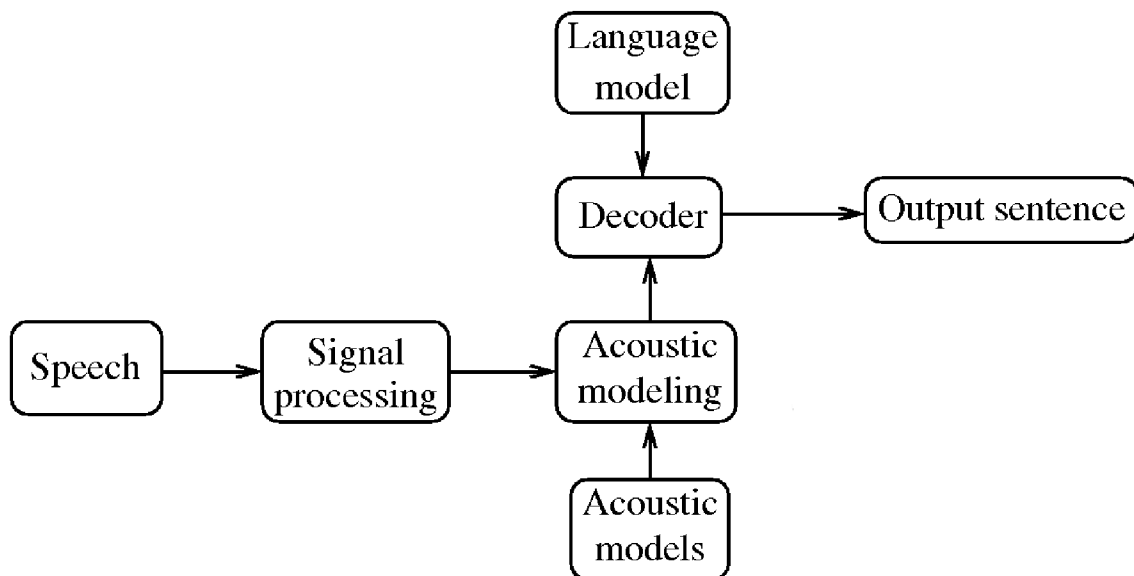


Figure 3.2: LVCSR system architecture (source: [Fapso2007]).

acoustic models than in full system are used – without adaptation, discriminative training and the recognizer does not use posterior features. The recognizer required approximately 200 MB of operating memory and was approximately six times slower than real-time on an average PC (2 GHz, 1 GB RAM).

This simplest version of Speech@FIT’s LVCSR was only for demonstration purposes. It had quick responses, an easy configuration and a smaller size. However, the results of such a derived recognizer were insufficient. Therefore, the full-featured AMI(DA) speech recognizer was finally selected for this project.

3.4.3 Technical Details

Complete description (used language models, lexicon, features, and so on) can be found in [HainRT2007], [HainICASSP2007].

The system was based on STK and the SRI LM toolkit¹⁷. The system operates in three passes (each with different HMM).

First pass: the initial pass only served for adaptation purposes.

- fea*: PLP_0 (13 coefficients) + Delta + Accerelation + Triple
Delta = 52 coefficient
Reduced by HLDA transformation to 39 coefficients
- hmm*: cross-word 16 Gaussians per state, 51 381 tied states
ML and MPE training

Second pass: lattices are generated.

- fea*: PLP_0 (13 coefficients) + Delta + Accerelation
- hmm*: cross-word 16 Gaussians per state, 7 790 tied states,
trained ML used only for VTLN warping factors estimation

¹⁷SRI language modelling toolkit web page: <http://www.speech.sri.com/projects/srilm>

Third pass: lattices are generated and later rescored with different acoustic models.

fea: PLP_0 (13 coefficients) + Delta + Acceleration + Triple

Delta = 52 coefficient

Reduced by HLDA transformation to 39 coefficients

CONCATENATED with LCRC Bottle Neck – Neural Network features.

Outdimensionality = 35, further reduced by HLDA to 30 coefficients

Total 69 dim features

hmm: cross-word 16 Gaussians per state, 4 583 tied states

ML, SAT (Speaker Adaptive Training) and then MPE,

estimation of CMLLR adaptation transformations on testing data

for lattices generation,

2gram language model,

lattices expansion with 3gram language model

and further with 4gram language model

The most important technological features are **UNISYN dictionary**, **Smoothed Heteroscedastic Linear Discriminant Analysis** (S-HLDA), adaptation from **Conversational Telephone Speech** (CTS), **Vocal Tract Length Normalisation** (VTLN), discriminative training using the **Minimum Phone Error criterion** (MPE) and **Maximum Likelihood Linear Regression** (MLLR) adaptation.

Training data: 170 hours of meeting data in total for IHM (Individual Head Microphones) system and 130 hours of data for MDM (Multiple Distant Microphones) system.

Trained models: Models were trained on meeting data only and a combination of PLP¹⁸ and LCRC features.

Language models construction: “LMs were constructed in a two-stage process. In the first instance out of more than 15 language models the nine most highly weighted LMs are selected and used as background language model for a web data search 20MW of web data are collected and used to train an additional LM component. In the second stage a new LM is constructed from the ten most highly weighted LMs but components with a weight of less than 1% are removed.” according to [HainRT2007]

3.4.4 Output of LVCSR

The recognizer can produce two types of output:

- *N-BEST word string*, which includes only the N most probable spoken words, typically 1-best string.
- *Recognition lattice* (lattice is a directed acyclic graph of word hypothesis, in this case the lattices are saved in the standard lattice format (SLF) – a sequence of node and link definitions [HTKBook]) that is used in the indexing and search phase, example of such a lattice is shown in figure 3.3. Despite N-best lattices enables more efficient storage and allows hypotheses in parallel. The structure consists of linked nodes with a likelihood assignment (example of a lattice is shown in figure 3.3). Another advantage is in possibility to gain 1-best output from lattices.

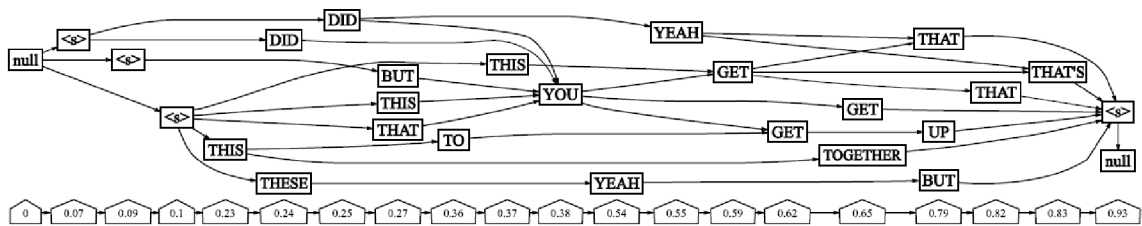


Figure 3.3: Example of a word lattice (source: [Burget2006]).

To have the possibility of searching for words, the output of LVCSR is set to be in lattices. The number of produced lattices is the same as the number of segments with speech (segments bases on voice activity detector, segmentation subsection 3.3.3).

3.5 Indexing and Search

3.5.1 Indexing

The output lattices from LVCSR is used as input to Speech@FIT’s speech indexer [Burget2006]. Words are converted to unique numeric IDs (mapped to the used lexicon), and a forward (sorted by document and within by the occurrence) index from all lattices produced by speech recognizer is created.

An inverted index (sorted by word IDs) is created to inform about a lattice in which a keyword appeared and what is its nodeID in this particular lattice.

The lattices are stored to binary form to allow fast access.

3.5.2 Search

When the user enters a query to be searched for, the keyword is looked up in the inverted index and a list of candidates is generated.

In case of a multi-word query, the searcher sorts the list of candidates according to the upper bound of confidence and a re-sorting using a true Viterbi algorithm¹⁹ follows. The result of a search is a sorted list of hypothesis [Fapso2007].

Phonetic search is not used in this project.

3.6 Presentation Software

As I mentioned at the beginning of this thesis, one of the essential parts of this project is an adequate presentation software. The fact that we have made a great tool for spoken speech recognition may not be automatically considered to be the end of our strive. If there is no acceptable way of presentation of recognition results, the project can have lower opportunity to excel.

There are several ways how to display results of recognition process and there are three important demands on the presentations software. Necessary components of such a presentations software are:

¹⁸PLP is *Perceptual Linear Prediction*.

¹⁹Viterbi algorithm finds the most likely sequence of hidden states in the context of HMM (so called *Viterbi path*).

- an audio player component, which can play the original recording
- a component for displaying recognized speech (a transcript)
- and finally a component capable of searching in the recognizer's output (lattices or 1-best output, described in the *Speech Recognition* section, 3.4)

For these purposes I selected *Multi-media Browser* developed by Petr Schwarz and Jakub Kubalík. This program fulfills the complete conditions enumeration and is described below (subsection 3.6.1). Other presentation software programs are mentioned in *Alternative Presentation Applications* subsection (3.6.2).

3.6.1 Multi-media Browser

The recognizer's output is presented by Multi-media browser (developed by Jakub Kubalík) developed at the Speech@FIT – PRASE [PRASEwww, Kubalik2007] (Presentation as Synchronized Experience). The environment, written using wxWidgets, supports multiple media streams, annotations and features an interface to the searcher.

In this project, only the audio is available, so that the user can use a player, view the 1-best result of the recognition and search in the audio using searcher engine.

This program fulfills all the project needs. On contrary to many other applications for transcription presentation mentioned in this section Multi-media Browser is capable of searching for strings in the recognized lattices.

The layout is configurable. It is defined by XML format saved in *layout.xml*. The application starts after reading this file where the component layout is defined. This approach allows users to easily modify, add or remove the components.

Example of a simplified layout.xml:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<layout name="main">
  <window name="window1" title="Multi-media Browser"
    left="25%" top="25%" width="50%" height="50%">
    <textbox name="textbox1" text="" left="25%" top="10%" width="50%"
      height="5%" onconfirm="<message name="\exec\"/>" amsg="*.*:sendText"/>
    <button name="button_example" text="Button" left="45%" top="30%"
      width="10%" height="4%" onclick="<message name="\media.play\"/>"/>
  </window>
</layout>
```

Description of basic components:

- a *Button* component
- an *Execute* component for external application execution, the standart output is afterwards transferred back to the browser
- a *MediaPlayer* component for displaying the video and/or playing the audio
- a *Slider* component for audio/video playback sliding
- a *TextBox* component for text input (used for search queries)

- a *TimeAxis* component, which shows the time progress and activity of the speakers while playing the recording
- a *Transcript* is a transcript displaying component (1-best string of LVCSR in a xml file)
- an *UrlGet* component for remote and local files addressing via HTTP²⁰

One of the possible appearances is shown in figure 3.4 and can be easily modified. It also displays a recognition output of a call, which took place on the fourth of April, 2008.

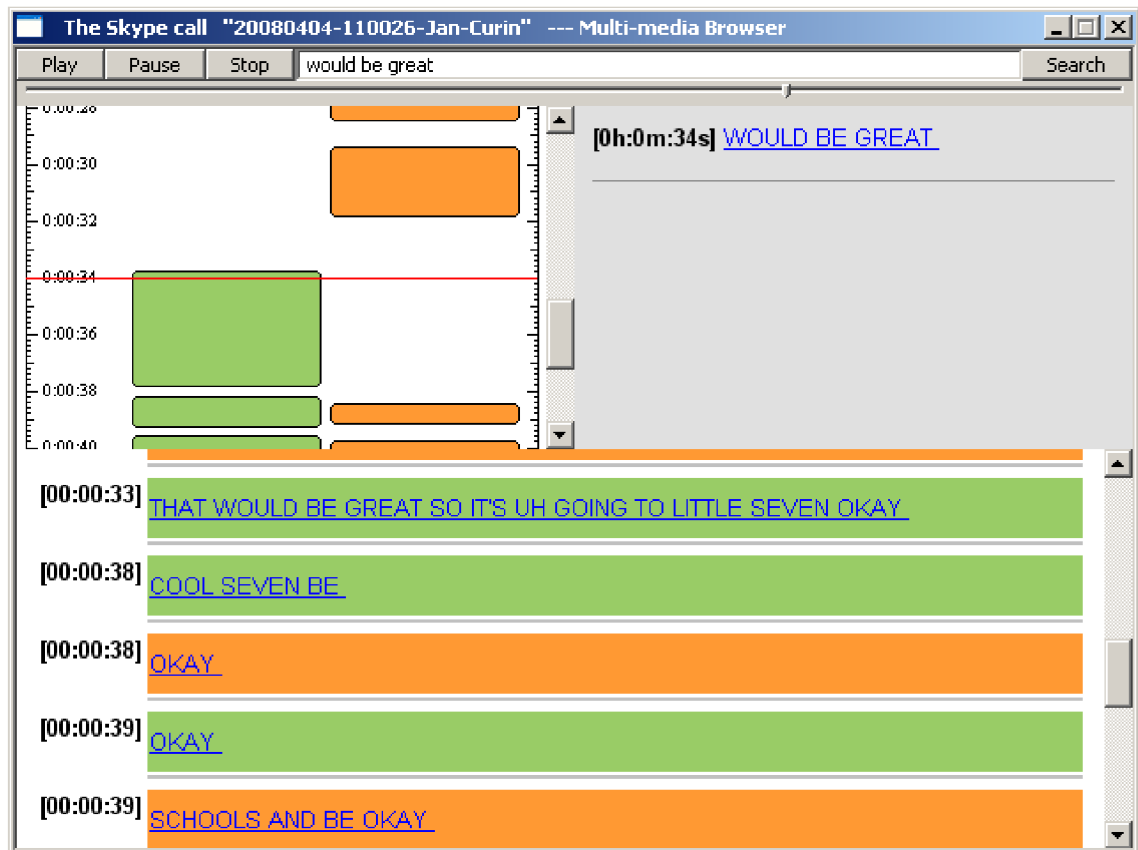


Figure 3.4: Snapshot of the Multi-media browser – PRASE displaying recognized Skype call with search results for “*would be great*”.

3.6.2 Alternative Presentation Applications

Lots of software products fulfill the first two conditions (play the recording and show the recognized text). These programs, usually used for annotation/transcription (free or commercial products), are for instance:

- *Anvil* (<http://www.anvil-software.de/>)
- *ELAN* (EUDICO Linguistic Annotator, <http://www.lat-mpi.eu/tools/elan/>)

²⁰HTTP stands for *HyperText Transfer Protocol*

- *Express Scribe Transcription Playback Software* (<http://www.nch.com.au/scribe>)
- *PitchWorks* (Scicon R&D, <http://www.sciconrd.com/pitchworks.html>)
- *Prosogram* (Mertens Piet, Department of Linguistics, KU Leuven
<http://bach.arts.kuleuven.be/pmertens/prosogram/>)
- *SoundIndex* (<http://michel.jacobson.free.fr/soundIndex/>)
- *Transana* (<http://www.transana.org/>)
- *Transcriber* (<http://trans.sourceforge.net/>)
- *WaveSurfer* (<http://www.speech.kth.se/wavesurfer/>)

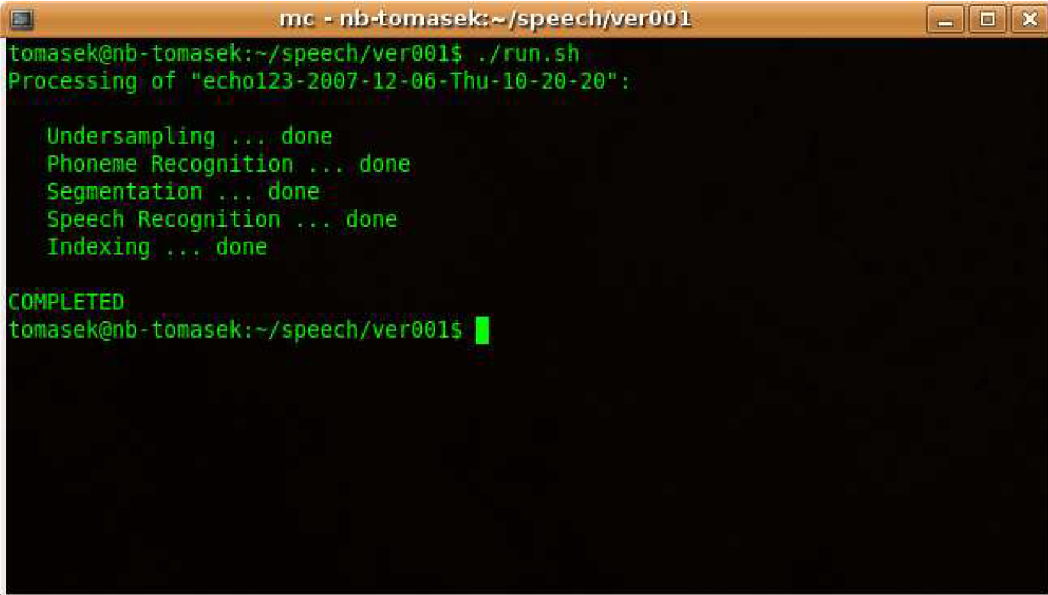
Chapter 4

Tests and Results

In this chapter are placed some results of tests made on the recognition system. Typical console output of such a recognition is shown in figure 4.1.

Subject of the testing was time consumption with recognition accuracy. These two aspects are related. Naturally, the results gets worse by making the recognition process quicker and vice-versa.

The testing took place in MS Windows XP and Ubuntu 7.10 operating systems.



```
mc - nb-tomasek:~/speech/ver001
tomasek@nb-tomasek:~/speech/ver001$ ./run.sh
Processing of "echo123-2007-12-06-Thu-10-20-20":

  Undersampling ... done
  Phoneme Recognition ... done
  Segmentation ... done
  Speech Recognition ... done
  Indexing ... done

COMPLETED
tomasek@nb-tomasek:~/speech/ver001$
```

Figure 4.1: Snapshot of a console window with text output from the whole process (preprocessing, recognition, indexing). Only one Skype call was processed (call with Skype Call Testing Centre)

4.1 CPU needs of the Simplest LVCSR

The system was examined for time consumption of some short stereo Skype calls recordings. Times needed to gain the complete output (preprocessing + recognition + indexing) are written in the table 4.1.

The first column informs about computing machine, where the recognition took place. The second column contains length of recognized recording and the consequent processing time.

Input recordings are stereo recordings (local and remote channel of a Skype call).

The testing took place on two laptop computers. Here are the description of used machines:

A. stands for laptop Compaq Evo N600c, Pentium III 1.2 GHz, 512 MB RAM

B. stands for laptop ASUS A6000, Centrino Duo T2300 1.6 GHz, 1024 MB RAM

| Recording length (seconds) | Processing time (seconds) | |
|----------------------------|---------------------------|-----------|
| | Machine A | Machine B |
| 00:00:20 | 00:06:46 | 00:03:07 |
| 00:00:40 | 00:16:35 | 00:06:43 |
| 00:00:50 | 00:13:42 | 00:05:31 |
| 00:04:06 | 01:17:44 | 00:30:02 |

Table 4.1: Examples of time costs of whole recognition process

As presented in the table 4.1, the times on machine A are approximately 2.5 times longer than the times on machine B. The processing time responds to the computing power rate of both computers.

The greatest time consumer is the speech recognizer, which needs at about 90% of the total processing time.

In the table is also one little surprise. We can be a little bit confused, if we compare the processing time of the forty seconds and the fifty seconds recording (the difference occurred on both machines). The difference consists in the recording – the processing time difference is based on the difference in the amount of spoken words in the recording. Logically, the recording which is richer in speech takes more processing time. That is why the presented time consumption results are just only for estimative purposes. Consequently another recordings of forty and fifty seconds length would take different processing time.

4.2 Examples of Recognition Results

The HTK tool *HResults* [HTKBook] was used for summary.

The correctness is given by quotient of number of correct labels (H) and total number of labels (N):

$$\%Correct = \frac{H}{N} \cdot 100\% \quad (4.1)$$

The accuracy is computed by quotient of difference between number of correct labels (H) and number of insertions (I), and total number of labels (N):

$$\%Accuracy = \frac{H - I}{N} \cdot 100\% \quad (4.2)$$

4.2.1 Recognition with the Simplest LVCSR

Speaker #1

Original spoken text: ... Israel's biggest banks he's expected to be lost about investigations as he tried to influence the process to help her friend. The Iraqi government has called on the Bush administration to end the cooperation with the private security from black water. The Iraqis accused Gods working for the company and having fight on civilians in Baghdad last month killing seventeen people. From Baghdad John Brain reports. ...

Recognized text: ISRAEL'S BIGGEST BANKS HE'S EXPECTED TO BE LOST ABOUT ON OCCASIONS THAT HE TRIED TO INFLUENCE THE TO HELP OF FRIENDS THE IRAQI GOVERNMENT TO SCHOOLS ALL THE BUSH ADMINISTRATION TO AND IS COOPERATION WITH THE PRIVATE SECURITY FROM BLACK WAS A IRAQIS ACCUSED ALSO LOOKING FOR THE COMPANY OF HAVING FIGHT ON CIVILIANS IN FACT THAT LOST MOST KILLING SEVENTEEN PEOPLE FROM BYPASS DON'T BRING THE HOLD

The differences: (the first line contains original text and the second contains the recognized speech, not well recognized words are typed bold)

Evaluation: A few problems are based on a fact, that some words are not in the recognizer dictionary. For this recognition this is typically the word *Baghdad*.

| | | |
|----------------------------|---------------------------|----|
| | Number of correct labels: | 44 |
| Correctness: 66.67% | Number of deletions: | 1 |
| | Number of substitutions: | 21 |
| Accuracy: 62.12% | Number of insertions: | 3 |
| | Total number of labels: | 66 |

Speaker #2

Original spoken text: ... Two teenagers have been held on suspicion of the attended murder of armed police officers who were round with the car. One officer was drag under a car and another with the bullet to battle seen in South London during the early hours of the morning. The on response team went to the area after reports that a guy have been seen with weapons. ...

Recognized text: TO TEENAGERS ARE BEING HELD ON SUSPICION OF THE ATTENDED THE THE ON POLICE OFFICERS WHO LIVE ROUNDS WITH THE CALL ONE OFFICE IT WAS DRAG ON HER CAR THAN OTHER WAYS OF THE ONLY TO BUSES IN SOUTH LONDON DURING THE OTHER HOUSE OF THE MORNING THE ON RESPONSE TEAM WENT TO BE AREA OF TO REPORTS THIS GUY HAVE BEEN SEEN WITH WEAPONS

Evaluation:

| | |
|----------------------------|------------------------------|
| Correctness: 62.50% | Number of correct labels: 40 |
| | Number of deletions: 2 |
| Accuracy: 57.81% | Number of substitutions: 22 |
| | Number of insertions: 3 |
| | Total number of labels: 64 |

4.2.2 Recognition with the full-featured recognizer

There are two Skype calls used as testing data. The results of recognition are written in the tables below (4.2, 4.3 and 4.4). Each table contains several segments with original spoken text (this means the really spoken words including word repetitions and stammering) and text generated by the full-featured recognizer (so called 1-best string, see section 3.4 about speech recognition). At the bottom of each table there is a summary (information about correctness, accuracy etc.).

The summary presented in the bottom of each of the following tables.

| | Original text | Recognized text |
|----------------|--|---|
| Segment 1 | <i>yeah I hope so I hope so yeah now I found the uh the headphones</i> | <i>yeah I hope so I have so we have now I found the uh the headphones</i> |
| Segment 2 | <i>so we can probably do the the recording without uh hearing each other</i> | <i>so we can probably do the the recording without uh hearing each other</i> |
| Segment 3 | <i>and the other channel</i> | <i>and the other channel</i> |
| Segment 4 | <i>yeah that's possible maybe it's uh maybe it's the type of microphone I don't know what do you have uh like uh</i> | <i>yeah that's possible maybe it's uh maybe it's the type of microphone and what do you have uh like uh</i> |
| Segment 5 | <i>a direction microphone or you have</i> | <i>a direction microphone or you have</i> |
| Segment 6 | <i>is this what you call this this microphone type I don't know</i> | <i>is this but what would you call this this microphone like I don't know</i> |
| Segment 7 | <i>yeah I can I can imagine that I hope I mean I I think we leave microphones are not very good so</i> | <i>yeah I can I can imagine that I hope I mean I I think we leave the microphones are not very good so</i> |
| Segment 8 | <i>yeah I have this one will be better than</i> | <i>yeah I have this one will be better than</i> |
| Segment 9 | <i>yes I will</i> | <i>yes I will</i> |
| Segment 10 | <i>okay uh_huh buy</i> | <i>okay uh_huh but</i> |
| Summary | Correctness: 93.58% Accuracy: 89.91% | Number of correct labels: 102 Number of deletions: 2 Number of substitutions: 5 Number of insertions: 4 Total number of labels: 109 |

Table 4.2: Results of recognition process of speaker #3

| | Original text | Recognized text |
|----------------|---|---|
| Segment 1 | <i>hi so this looks like professional audio setting</i> | <i>hi so this looks like professional audio settings</i> |
| Segment 2 | <i>okay so</i> | <i>okay so</i> |
| Segment 3 | <i>uh I think the problem is that we have the little bit of uh well actually not a little bit but lots of crosstalk uh in uh your recording</i> | <i>uh I think the problem is that we have the little bit of uh well actually not a little bit but also crosstalk uh in uh you're recording</i> |
| Segment 4 | <i>oh it's it's funny because I also had loud speakers so basically it's maybe mixed somewhere in the sound card</i> | <i>oh it's it's funny because I also that the all speakers so basically it's maybe makes somewhere evening some power</i> |
| Segment 5 | <i>it is this is the most shitty microphone that that you can get is the basic</i> | <i>it is this is the most should do microphone that the that we can get is the basic</i> |
| Segment 6 | <i>the plastic microphone for for you can get for hundred crowns</i> | <i>the plastic microphone for for you can get four hundred pounds</i> |
| Segment 7 | <i>okay so we are almost over would you put it to the same directory</i> | <i>okay so we are almost over would you put this to the same directory</i> |
| Segment 8 | <i>okay thank you very much bye bye</i> | <i>okay thank very much bye bye</i> |
| Summary | Correctness: 84.11% Accuracy: 81.31% | Number of correct labels: 90 Number of deletions: 3 Number of substitutions: 14 Number of insertions: 3 Total number of labels: 107 |

Table 4.3: Results of recognition process of speaker #4

4.3 Examples of Search Results

Table 4.5 with search results contains information about searched word, it's real number of occurrences, number of all found items, number of correctly found items and correctness.

The correctness is given by quotient of number of correctly found items and number of all found occurrences. The results strongly depends on amount of similar words (or idem sonans words) spoken in the recording.

As an input data the results of recognition process of the two recordings mentioned in subsection *Recognition with the full-featured recognizer* (4.2.2) were used.

| | Original text | Recognized text |
|----------------|---|---|
| Segment 1 | <i>okay so the recording should be on now</i> | <i>okay so the recording should be oh now</i> |
| Segment 2 | <i>basically it's tells me recording duration eight seconds nine seconds great</i> | <i>basically it's tells me recording duration eight seconds nine seconds right</i> |
| Segment 3 | <i>well actually this data is for the the project that we have uh with I. B. M.</i> | <i>well actually did they taste for the the project that we have uh with I. B. M.</i> |
| Segment 4 | <i>that uh mister the Pavel Tomášek is working on</i> | <i>that uh mister the automatic is working on</i> |
| Segment 5 | <i>so basically the it it's really nice he just a showed me the project and pretty much everything is in place</i> | <i>so basically the please it's really nicely just a showed me the project and pretty much everything is in place</i> |
| Segment 6 | <i>so the segmentation recognition well uh what what else indexing search and it's interfaced with the prase browser which is really nice of course this is without video so no video</i> | <i>so the segmentation recognition well uh what what else indexing service and this interface to the press a browser which is really nice of course this is without we don't know we do</i> |
| Segment 7 | <i>so tell me something about</i> | <i>so tell me something about</i> |
| Segment 8 | <i>yeah yeah and actually we are working right hard</i> | <i>yeah yeah and actually we are doing right hard</i> |
| Summary | Correctness: 81.98% Accuracy: 80.18% | Number of correct labels: 91 Number of deletions: 2 Number of substitutions: 18 Number of insertions: 2 Total number of labels: 111 |

Table 4.4: Results of recognition process of speaker #5

| Searched word | Really spoken | Found | Correctly found | Correctness |
|----------------------|----------------------|--------------|------------------------|--------------------|
| <i>actually</i> | 3 | 3 | 2 | 66.67% |
| <i>basically</i> | 2 | 2 | 2 | 100.00% |
| <i>hope</i> | 4 | 4 | 4 | 100.00% |
| <i>just</i> | 3 | 13 | 3 | 23.08% |
| <i>microphone</i> | 6 | 7 | 6 | 85.71% |
| <i>recording</i> | 2 | 3 | 2 | 66.67% |
| <i>seconds</i> | 2 | 2 | 2 | 100.00% |
| <i>should</i> | 1 | 7 | 1 | 14.27% |
| <i>something</i> | 1 | 3 | 1 | 33.33% |
| <i>take</i> | 1 | 4 | 1 | 25.00% |
| <i>tell</i> | 2 | 6 | 2 | 33.33% |
| <i>this</i> | 7 | 17 | 7 | 41.18% |
| <i>working</i> | 3 | 4 | 3 | 75.00% |

Table 4.5: Table with correctness information of search results

Chapter 5

Conclusions and Future Work

This project is focused on a system for recognition of Skype calls. In this thesis the reader was informed about the implementation idea and particular implementation solutions.

My main work was focused on integration of all project parts.

At the beginning of my work I focused on understanding the process of signal processing and speech recognition. One of the most difficult parts of this project was search for a Skype call recorder with a high-quality recording. The easiest part was MBrowser integration, which did not require hardly any configuration on contrary to the full-featured recognition system.

The consequences of using the simplified recognizer (without adaptive techniques) at the beginning of my work are quicker recognition with higher error rates. The last implemented recognizer is full-featured, therefore the results are significantly better.

Very appealing scenario of this project usage can be determination who made a particular statement, when and where the next meeting was planned for, and if particular keywords or key-phrases (such as “budget cut”) were used during a conference.

This system was tested by several people. After reading instructions written in *Cookbook* (6), users were able to start recording of Skype calls without any problem. They were only a little bit confused by confirming approach of Skype Capture to the Skype program.

Also recognition and presentation with MBrowser was without any problem. Users needed a few seconds to get acquainted with the program nature.

The project served as a proof-of-concept of integration of different speech technologies developed at Speech@FIT. At the time of writing this bachelor’s thesis, the system acquires the data on Windows, these data need to be transferred to a Linux box for signal processing, recognition and indexing, and the results are presented using PRASE (Presentation as Synchronized Experience) again on a Windows machine. Only one Skype call recording can be processed simultaneously.

This project demonstrates wide area of possible usage of software and technologies developed by the Speech@FIT group. Recognition and search in Skype calls is just an illustrative example of the already mentioned wide area of utilization. Processing of almost any higher quality audio recordings is possible in similar way.

5.1 Future Work

By effort to improve the error rates of the recognition the following improvements could be implemented:

5.1.1 Short Term

Voice activity detection

It would be beneficial (mainly for presentation) to improve the segmentation process which should create shorter segments. And there is also a need to improve the creation of adequate segments of a recording which contains some noise.

Searching in a selected channel

This would enable searching in both, local or remote speech (local/remote speaker).

Multi-media browser

Continual work on presentation software, Multi-media Browser, will surely affect this project in the future (waveform, scrollbar, ...).

Recognition and indexing server

In the future, running a recognition and indexing server to which the user would submit the recorded audio would be beneficial.

There is also a plan to place the recognition system (Skype call preprocessing through indexing) on a Linux server which would serve the possibility to upload the recordings of Skype calls, their recognition and possibility of downloading a compressed package with results of the recognition process. Such a package provided by the server should include:

- original audio, which was uploaded for recognition (afterwards needed by a presentation software for playback)
- output of the recognition process (lattices and/or 1-best output)
- a search engine for searching in the output lattices
- a start script or program aimed at the presentation of results
- a “readme.txt” file to inform user about the results in general and about the presentation requirements and initiation

5.1.2 Long Term

Speaker identification

At the time of writing this thesis the system is capable of processing a general Skype call between two people. Another future step may be implementation of conference calls (Speaker Identification needed).

Automatic recording & recognition server connecting

This would simplify demands on users. The recording would start automatically as well as the further recognition.

Video capture

The videoconferences will be captured (sound and video) to present the call with the possibility of video playback.

Chapter 6

Cookbook

6.1 For Users

This section informs about recommendations for users who want to try to make calls and recognize them. All the software and scripts can be found on the attached DVD.

6.1.1 Recording

First of all a user needs Skype and *Skype Capture* program, which is described in [3.2.2](#) subsection.

To start the recording just simply execute the “*run.bat*” in a MS Windows operating system, which starts the capturing and saves the recordings into “*recordings*” directory placed in the directory of the *run.bat* file. After execution of this file the Skype program asks user to enable/disable approach of Skype Capture to the Skype program. The user should “*enable*” the approach. Now, the user can make as many calls as he/she wants. When the user wants to end the recording, he/she can only click to the Skype Capture command line window and hit enter. Afterwards, in the *recordings* directory there should be placed the recorded calls.

The program should work with all newer versions of Skype (version 3.0 and newer).

In case of a serious problem, contact the author of Skype Capture – Jiří Šimáček, xsimac00@stud.fit.vutbr.cz.

Another recommendation is to use headset (to reduce crosstalk and noise) and to speak clearly while calling.

6.1.2 Recognition

The recognition takes place in a Linux operating system. The recognition system requires *awk*, *grep*, *perl*, *sed*, *sox* and *tcsh* to be installed on the machine.

After realization of a few calls, the user has to copy the recordings to the Linux machine to the “*skype_stt/recordings*” directory, change directory to “*skype_stt*” and execute “*./run.sh*”. This starts the complex system and the user will be shortly and clearly informed about the progress. The recognition process takes a long time.

After the recognition is done, the output is placed in the “*output*” directory. The directory should include packages with names of each processed recording.

6.1.3 Presentation of Results

The presentation take place back in a MS Windows operating system. To present the results of recognition, the user must have installed wxWidgets (<http://www.wxwidgets.org/> version 2.8.3 at least).

After unpacking and opening a package made by the recognizer, user can start the presentation simply by executing “*MBrowser.exe*”.

6.2 For Developers

6.2.1 Recognition System Structure

| | |
|--------------------------|--|
| <code>skype_stt/</code> | – This directory (Skype speech to text) contains main <i>run.sh</i> script, a readme file and a file with licensing information |
| <code>MBrowser/</code> | – This directory contains presentation system (executable file with a necessary library, a readme file and wxWidgets setup files for MS Windows operating system) |
| <code>output/</code> | – This is a place for already recognized recordings in archives – the output of recognition process with a copy of MBrowser and a search module; it is named by the original recording name |
| <code>recognizer/</code> | – This directory contains the recognition system (with a configuration file) including the phoneme recognizer, LVCSR, and other scripts for recognition and indexing, libraries and dictionaries |
| <code>recordings/</code> | – This is a place for recordings waiting for recognition |
| <code>scripts/</code> | – This is a directory for other general scripts (segmentation and output XML modifying script) |
| <code>searcher/</code> | – This directory contains indexing scripts, a lexicon and a search module with a search script for MS Windows operating system |

6.2.2 Software Versions

| | |
|---------------------------|------------------------------|
| <i>Skype Capture</i> | – version 0.1 |
| <i>Phoneme recognizer</i> | – version 2.14 |
| <i>Speech recognizer</i> | – AMIDA system, build 2007 |
| <i>Search engine</i> | – version 2.x (build 2008/3) |
| <i>MBrowser</i> | – build 159 |

6.2.3 Compilation

Skype Capture Compilation

Requirements: MinGW (<http://www.mingw.org/>)

To compile Skype Capture user have to unpack *src.zip*. Afterwards he have to use command line to change the directory to *src* and type *make*. The executable file will be created in the *src* directory.

Recognition and Indexing System Compilation

Requirements: autoconf, awk, build-essential, g++, gcc, glibc, grep, libstdc++, make, perl, sed, sox, tclsh

The compilation recognition and indexing system is very user-friendly. User can start the compilation of important parts of this project by setting only one option to the main script. User must change directory to the *skype_stt* and write *./run.sh --make*.

6.2.4 Licensing

Copyright: (C) 2008 Speech@FIT
Faculty of Information Technology,
Brno University of Technology

The program (source files and binary version) is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Licence for use for research and educational purposes is granted. For any other use, please contact Jan Černocký (cernocky@fit.vutbr.cz).

6.2.5 Hotline

More information about particular parts of this system can be found on Speech@FIT web pages <http://speech.fit.vutbr.cz/>.

Here are e-mail contacts of the authors of parts of system:

| | |
|---------------------------|---|
| Integration | – Pavel Tomášek (xtomas23@stud.fit.vutbr.cz) |
| Skype Capture | – Jiří Šimáček (xsimac00@stud.fit.vutbr.cz) |
| Phoneme recognition (VAD) | – Pavel Matějka (matejkap@fit.vutbr.cz) – Petr Schwarz (schwarzp@fit.vutbr.cz) |
| LVCSR | – Martin Karafiát (karafiat@fit.vutbr.cz) – Lukáš Burget (burget@fit.vutbr.cz) – Ondřej Glembek (glembek@fit.vutbr.cz) |
| Indexing and search | – Michal Fapšo (ifapso@fit.vutbr.cz) – Igor Szöke (szoke@fit.vutbr.cz) |
| Multi-media browser | – Petr Schwarz (schwarzp@fit.vutbr.cz) – Jakub Kubalík (xkubal05@stud.fit.vutbr.cz) |

Bibliography

- [Baset2004] Baset Salman and Schulzrinne Henning. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. Technical report, Department of Computer Science, Columbia University, September 2004.
- [Burget2006] Burget Lukáš, Černocký Jan, Fapšo Michal, Karafiát Martin, Matějka Pavel, Schwarz Petr, Smrž Pavel, and Szöke Igor. Indexing and search methods for spoken documents. In *Proceedings of the Ninth International Conference on Text, Speech and Dialogue, TSD 2006*, number 4188 in LNCS, pages 351–358. Springer Verlag, 2006.
- [CivilCode] First part – General institution, Head two: Participants of civil-law relations, Protection of personality. In *Civil Code 40/1964 digest*, 1964.
- [Fapso2007] Fapšo Michal. Search in Speech Data. Master’s thesis, Brno University of Technology, Faculty of Information Technology, 2007.
- [Guha2006] Guha Saikat, Daswani Neil, and Jain Ravi. An Experimental Study of the Skype Peer-to-Peer VoIP System. In *The 5th International Workshop on Peer-to-Peer Systems (IPTPS ’06)*, 2006.
- [HTKBook] Young Steve, Evermann Gunnar, Gales Mark, Hain Thomas, Kershaw Dan, Moore Gareth, Odell Julian, Ollason Dave, Povey Dan, Valtchev Valtcho, and Woodland Phil. *The HTK book (for HTK Version 3.3)*. Entropics Cambridge Research Lab., 2005.
- [Hain2006] Hain Thomas, Burget Lukáš, Karafiát Martin, Dines John, Vepa Jithendra, Garau Giulia, Lincoln Mike, and Wan Vincent. The AMI Meeting STT System. In *Proc. The Rich Transcription 2006 Spring Meeting Recognition Evaluation Workshop*, 2006.
- [HainICASSP2007] Thomas Hain, Vincent Wan, Lukáš Burget, Martin Karafiát, John Dines, Jithendra Vepa, Giulia Garau, and Mike Lincoln. The ami system for the transcription of speech in meetings. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, pages 357–360. IEEE Signal Processing Society, 2007.
- [HainRT2007] Hain Thomas, Burget Lukáš, Dines John, Garau Giulia, Karafiát Martin, Leeuwen David van, Lincoln Mike, and Wan Vincent. The 2007 AMI(DA) System for Meeting Transcription, 2007.
- [Kubalik2007] Kubalík Jakub. Multimedia Browser for Lectures. Bachelor’s thesis, Brno University of Technology, Faculty of Information Technology, 2007.

- [Newton2006] Newton Tom. Skype: how safe is it? *SECURE Online Magazine*, Issue 8:16–18, September 2006.
<http://www.net-security.org/dl/insecure/INSECURE-Mag-8.pdf>.
- [PRASEwww] Kubalík Jakub and Schwarz Petr. PRASE Multi-Media Presentation Tool.
http://merlin.fit.vutbr.cz/wiki/index.php/PRASE_Multi-media_presentation_tool. accessed 10th March 2008.
- [Schwarz2004] Schwarz Petr, Matějka Pavel, and Černocký Jan. Towards Lower Error Rates in Phoneme Recognition. In *Proceedings of 7th International Conference Text, Speech and Dialogue 2004*, page 8. Springer Verlag, 2004.
- [Schwarz2006] Schwarz Petr, Matějka Pavel, and Černocký Jan. Hierarchical structures of neural networks for phoneme recognition. In *Proceedings of ICASSP 2006*, pages 325–328, 2006.
- [Simacek2007] Šimáček Jiří. Záznam hovoru ze Skype. Systémy zpracování řeči – Projekt 2007, Brno University of Technology, Faculty of Information Technology.
- [Skypewww] Web pages. Skype - How do you hello? <http://www.skype.com/>. accessed 10th March 2008.
- [Tomasek2008] Tomášek Pavel. Recognition and Search in Skype Calls. In *Proceedings of the 14th Conference STUDENT EEICT 2008 Volume 1*, pages 201–203. Brno University of Technology, Faculty of Electrical Engineering and Communication, Faculty of Information Technology, 2008.
- [VASwww] Web pages. Distributed Systems Research Group, Department of Software Engineering, Faculty of Mathematics and Physics Charles University.
<http://dsrg.mff.cuni.cz/>. accessed 15th October 2007.

Glossary

| | |
|----------------|--|
| ALSA | Advanced Linux Sound Architecture |
| AMIDA | Augmented Multi-party Interaction with Distance Access |
| CTS | Conversational Telephone Speech |
| BUT FIT | Brno University of Technology, Faculty of Information Technology |
| LVCSR | Large Vocabulary Continuous Speech Recognition |
| GUI | Graphical User Interface |
| GNU | GNU's Not Unix, a recursive acronym |
| GPL | General Public License |
| HTK | Hidden Markov Models Tool Kit, also known as HMM toolkit |
| JACK | JACK Audio Connection Kit, a recursive acronym |
| LM | Language Model |
| MDM | Multiple Distant Microphones |
| MLF | Master Label File |
| NAT | Network Address Translation |
| OSS | Open Sound System |
| PLP | Perceptual Linear Prediction |
| PRASE | Presentation as Synchronized Experience |
| SIP | Session Initiation Protocol |
| SLF | Standard Lattice Format |
| VAS | Virtual Audio Socket |
| VoIP | Voice over Internet Protocol |
| VTLN | Vocal Tract Length Normalisation |
| XML | eXtensible Markup Language |