

UNIVERSITY OF THE WESTERN CAPE

**Browser-based and mobile video
communication alternatives for
Deaf people**



by

UNIVERSITY of the
WESTERN CAPE

Yuanyuan Wang

Supervisor: Dr William D Tucker

A thesis submitted in fulfillment of the
degree of Masters of Computer Science

in the
Department of Computer Science

February 2011

Declaration of Authorship

I, Yuanyuan Wang, declare that *Browser-based and mobile video communication alternatives for Deaf people* is my own work, that it has not been submitted for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged by complete references.



Signed

Date

Abstract

This thesis offers some prototypes to provide browser-based and mobile video communication services for Deaf people and evaluates these prototypes. The aim of this research is to identify an acceptable video communication technology for Deaf people by designing and evaluating several prototypes. The goal is to find one that Deaf people would like to use in their day-to-day life. The thesis focuses on two technologies — browser-based systems and mobile applications. Several challenges emerged, for example, specific Deaf user requirements are difficult to obtain, the technical details must be hidden from end users, and evaluation of prototypes includes both technical and social aspects. This thesis describes work to provide South African Sign Language communication for Deaf users in a disadvantaged Deaf community in Cape Town. We posit an experimental design to evaluate browser-based and mobile technologies in order to learn what constitutes acceptable video communication for Deaf users. Two browser-based prototypes and two mobile prototypes were built to this effect. Both qualitative data and quantitative data are collected with user tests to evaluate the prototypes. The video quality of Android satisfies Deaf people, and the portable asynchronous communication is convenient for Deaf users. The server performance is low on bandwidth, and will therefore cost less than other alternatives, although Deaf people feel the handset is costly.

Acknowledgements

This thesis is a result of a compilation of efforts from many people, who helped me and guided me throughout my research years. Without their assistance, all this would not have happened. At first, I would like to thank my supervisor, Dr. W. D. Tucker. It is he who guided me, inspired me, helped me, encouraged me, and supported me throughout the duration of my postgraduate study. Without his help, this thesis would not have appeared. I am forever grateful to him for taking me in and assisting me to explore the world of computer science research. I have learned invaluable lessons from him. I would also like to extend my special thanks to Prof. I. Venter and Mr. M. Norman. Thank you for your help during my postgraduate years. Secondly, I would like to thank the Deaf Community of Cape Town (DCCT) and their staff. Without their help, my observations would not have been possible. Thank you for your participation and warm hospitality. Your willingness to accommodate us in your tight schedules is greatly appreciated. I also would like to thank Sign Language Education and Development (SLED) and their staff, who helped me to learn South African Sign Language (SASL). Thirdly, I would like to thank some of my colleagues: Muyowa Mutemwa, Ashley Kerchoff and Long Yee. Thank you for all the support and wonderful suggestions on how to improve the prototypes. Fourthly, I would like to thank my research sponsors, the THRIP Centre of Excellence at the University of the Western Cape. Thank you very much for the financial support during the tenure of this research. Finally, I would like to give respect and love to my family—my father, my mother and my husband. Without their unwavering support and upbringing, nothing would ever have been achievable. You are what motivated me to do my best.

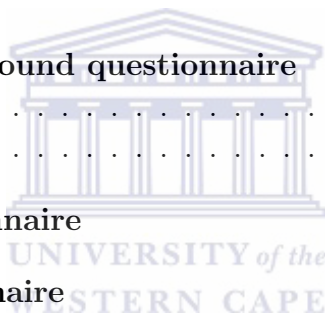
Contents

Declaration of Authorship	i
Abstract	iii
Acknowledgements	iv
Contents	v
List of Figures	ix
List of Tables	xi
Abbreviations	xiii
1 Introduction	1
1.1 Deaf with a capital ‘D’	1
1.2 South African Deaf culture	2
1.3 Motivation	3
1.4 Approach	4
1.5 Thesis outline	6
2 Related Work	7
2.1 Deaf communication	7
2.1.1 Text relay	8
2.1.2 Video relay	9
2.1.3 Sign language video	11
2.1.4 BANG projects	15
2.2 Video technologies	16
2.2.1 Off the shelf video systems	16
2.2.2 Browser-based video	17
2.2.3 Mobile video	22
2.3 Video evaluation	25
2.3.1 Evaluation of video for hearing people	26
2.3.2 Evaluation of video for Deaf people	27
2.4 Summary	28



3	Methodology	31
3.1	Research questions	32
3.2	Research methods	34
3.2.1	Quantitative research method	34
3.2.2	Qualitative research method	35
3.2.3	Software engineering method	35
3.2.4	Method integration	36
3.3	Experimental design	37
3.3.1	Disclosure of the role of the researcher as a member of a research group	39
3.3.2	Target community	39
3.3.3	Experimentation arrangement	40
3.3.4	User behaviour and background data	41
3.3.5	User tests	42
3.3.6	Performance testing	43
3.3.7	Ethical considerations	43
3.4	Summary	44
4	Prototype Design	47
4.1	Prototype-Flash	48
4.1.1	System design	49
4.1.2	User interface	51
4.2	Prototype-HTML5	53
4.2.1	System design	54
4.2.2	User interface	56
4.3	Prototype-J2ME	56
4.3.1	System design	56
4.3.2	User interface	58
4.4	Prototype-Android	59
4.4.1	System design	61
4.4.2	User interface	62
4.5	Summary	64
5	Results	65
5.1	User behaviour and technology background	65
5.1.1	Technology background	66
5.1.2	Internet usage	67
5.2	User tests	71
5.2.1	Preparation	71
5.2.2	Pilot trial	75
5.2.3	Full trial	77
5.2.4	Analysis of user tests	81
5.3	Performance tests	82
5.3.1	Flash server	83

5.3.2	HTTP server	84
5.3.3	Analysis of performance data	84
5.4	Synthesized results	85
5.5	Summary	87
6	Conclusion	89
6.1	Communication alternatives for Deaf people	89
6.1.1	Introduction	90
6.1.2	Related work	90
6.1.3	Methodology	90
6.1.4	Prototype design	91
6.1.5	Results	91
6.1.6	Conclusion	92
6.2	Suggestions for conducting trials with Deaf people	93
6.3	Limitations of this research	94
6.4	Future work	95
A	Technology background questionnaire	97
A.1	Questionnaire	97
A.2	Results	100
B	Pilot trial questionnaire	103
C	Full trial questionnaire	105
C.1	Questionnaire	105
C.2	Results	106
D	User Guide (Pilot Trial)	109
E	User Guide (Full Trial)	115
	Bibliography	119



List of Figures

2.1	Deaf communication projects	8
2.2	Video relay	9
2.3	The user interface of VP-100	10
2.4	The user interface of Omnitor products	12
2.5	The user interface of Open Video Chat	13
2.6	A MobileASL video conversation	13
2.7	Skin detection algorithms in MobileASL	14
2.8	An overview of SIMBA	15
2.9	A Flash-based application model	18
2.10	Client-server communication via RTMP	20
2.11	A HTML5 application model	22
2.12	MMAPI model	24
2.13	Android video model	25
3.1	A software engineering model	36
3.2	Methods integration	37
3.3	Spiral research model	38
3.4	Prototypes and technologies	40
3.5	Experimental arrangement	41
4.1	Prototype-Flash architecture	49
4.2	Synchronous video work flow	50
4.3	The user interface of prototype-Flash	52
4.4	Prototype-HTML5 architecture	53
4.5	Asynchronous video in prototype-HTML5	55
4.6	The user interface of prototype-HTML5	57
4.7	Work flow chart for prototype-J2ME	59
4.8	The user interface of prototype-J2ME	60
4.9	Prototype-Android architecture	61
4.10	The user interface of prototype-Android	63
5.1	Internet usage at the Bastion by year	69
5.2	Comparison of Internet usage of key items	70
5.3	Internet usage of the top five users	70
5.4	Comparison of two alternatives	82
5.5	Processor and memory usage of prototype-Flash	84

5.6	Processor and memory usage of prototype-HTML5	85
5.7	Integration of qualitative and quantitative data	86
D.1	User guide for prototype-Flash (administrator)	109
D.2	User guide for prototype-Flash (DCCT1)	110
D.3	User guide for prototype-HTML5	111
D.4	User guide for prototype-J2ME	112
D.5	User guide for prototype-Android	113
E.1	User guide for prototype-Flash	115
E.2	User guide for prototype-HTML5	116
E.3	User guide for prototype-J2ME	117
E.4	User guide for prototype-Android	118



List of Tables

2.1	Video element support	21
2.2	Video format support	21
2.3	Adobe Flash vs HTML5	22
5.1	Internet usage counts	71
5.2	Prototype evaluation results in the pilot trial	76
5.3	Best choice evaluation in full trial	77
5.3	Best choice evaluation in full trial	78
5.4	Prototype evaluation results in the full trial	79
A.1	Technology background data Part 1	100
A.2	Technology background data Part 2	101
A.3	Technology background data Part 3	101
B.1	Pilot trial result Part 1	103
B.2	Pilot trial result Part 2	104
C.1	Full trial results	107

Abbreviations

3GPP	the 3rd Generation Partnership Project
ADB	Android Debug Bridge
Ajax	Asynchronous JavaScript And XML
API	Application Programming Interface
ASL	American Sign Language
BANG	Bridging Applications and Networks Group
CDF	Cumulative Distribution Function
CLDC	Connected Limited Device Configuration
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DCCT	Deaf Community of Cape Town
DOM	Document Object Model
FLV	Flash Video
GPRS	General Packet Radio Service
HTML	Hypertext Markup Language
HTML5	Hypertext Markup Language 5
HTTP	Hypertext Transfer Protocol
IE	Internet Explorer
IP	Internet Protocol
IM	Instant Messaging
ISDN	Integrated Services Digital Network
ISP	Internet Service Provider
J2ME	Java 2 Micro Edition
JDK	Java Development Kit
JRE	Java Runtime Environment
JVM	Java Virtual Machine
MIDP	Mobile Information Device Profile
MMAPI	Mobile Media Application Programming Interface

MOS	Mean Opinion Score
MRTG	Multi Router Traffic Grapher
MSE	Mean Squared Error
MPEG	Moving Picture Experts Group
NGO	Nongovernmental Organization
OS	Operation System
PAL	Performance Analysis of Log
PC	Personal Computer
PEVQ	Perceptual Evaluation of Video Quality
PSNR	Peak Signal to Noise Ratio
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RIA	Rich Internet Application
ROI	Region Of Interest
RTMP	Real Time Messaging Protocol
SASL	South African Sign Language
SDK	Software Development Kit
SIMBA	Softbridge Instant Messaging Bridging Architecture
SIP	Session Initiation Protocol
SMS	Short Message Service
SLED	Sign Language Education and Development
SNR	Signal to Noise Ratio
TCP	Transmission Control Protocol
TISSA	Telephone Interpreting Service of South Africa
TTS	Text To Speech
UI	User Interface
USB	Universal Serial Bus
VoIP	Voice over Internet Protocol
WMV	Windows Media Video
XML	Extensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

Chapter 1

Introduction

This thesis describes an evaluation of four alternatives for sign language video communication. These prototypes are measured and compared to identify a viable video communication alternative for Deaf people, in order to help them use network technology comfortably. There are many communication systems for Deaf and hearing people. These applications provide text and video chat services. However, Deaf users have special needs. For example, some Deaf people cannot use spoken language and only communicate in sign language. This thesis aims to evaluate alternatives for Deaf users who prefer sign language communication.

Section 1.1 introduces a definition for Deaf versus deaf. Section 1.2 describes South African Deaf culture. Section 1.3 presents the motivation for this research. Section 1.4 introduces the approach. Section 1.5 outlines the structure of this thesis.

1.1 Deaf with a capital ‘D’

Deaf with a capital ‘D’ denotes people who use sign language as a preferred language. The distinctions between “deaf” and “Deaf” are based principally on the individual’s preferred language (spoken or sign) rather than on the actual degree of hearing loss. For example, a hearing child of Deaf parents is Deaf, for the child uses sign language as his mother tongue to communicate with other people from birth. Deaf people have a different culture from hearing people [43], and they usually find it hard to communicate with hearing people for both language and cultural reasons [60].

Deaf people are unable to hear and often can not speak either. They communicate with others relying on the visual and tactile, not on hearing. Deaf people prefer to use sign language to communicate. This is why they have a different culture from the majority [43]. In the traditional telephone era, Deaf people always went to visit others for face-to-face communication because they were unable to use the traditional telephone devices and technologies [43].

1.2 South African Deaf culture

The intended beneficiaries of this research are South African Deaf users. Therefore, it is necessary to understand Deaf culture in South Africa. According to statistics collected in 1994, there were 4 million people, or about 10% of the general population, with hearing impairment in South Africa. Of those 4 million, 10% are profoundly Deaf. These are the main users of South African Sign Language (SASL) [32]. The demographics of the South African Deaf community paint an interesting picture, and are closely related to the general South African socio-economic structure. 30% of Deaf adults are illiterate because of unsuccessful educational practices [2]. The vast majority of Deaf children never went to school or went to school at a very late age [32]. Education for Deaf children only came to be compulsory in 1996 [1]. Although the government stated that SASL is to be the medium of instruction in Deaf schools, Deaf pupils could not be educated through the medium of a signed language for many reasons [1]. The education situation created other problems. About 65% of all Deaf adults are unemployed, and many workers are underemployed [32]. The socio-economic status of the Deaf community remains compromised because of these reasons.

A Deaf non-governmental organization (NGO) in Cape Town — Deaf Community of Cape Town (DCCT), is the main target of this research. DCCT is a Deaf community where Deaf people help Deaf people. It provides the Deaf community with a wide range of beneficial programmes, such as group work and community development. DCCT has supported a community of nearly one thousand Deaf people in the Cape Town area since 1987 [2]. Many members of DCCT have poor levels of spoken, written and reading literacy in any of the eleven official South African languages. They use SASL as their primary language for daily communication [2]. They are typical South African Deaf with typical cultural characteristics such as pride, illiteracy, physiological impairment and underemployment [32]. Many

DCCT members have their own mobile phones. Only a few of them have computer experience. They often use Short Message Service (SMS) to connect with others remotely. Deaf people can understand these SMSes even if there are some grammar and spelling mistakes in them. DCCT staff always use SMS to gather members when there is an event. Most of them have no computer or Internet experience. The vast majority of DCCT members want to improve their Personal Computer (PC) skills for job hunting.

Bridging Applications and Networks Group (BANG) is a research group at the University of Western Cape. It conducts research on Voice over Internet Protocol (VoIP) and mobile phone software, and applies these technologies to Deaf communication. BANG started network projects for Deaf people from 2002 [74]. BANG provided a PC lab, consisting of six PCs with web-cams at the Bastion of the Deaf (the building where DCCT is based). Much user data about Deaf culture and Deaf behaviour with respect to technology use has been collected in collaboration with DCCT. The details will be introduced later.

1.3 Motivation

The same rights of communication should be given to both Deaf people and hearing people. Some social projects have been implemented to improve communication between Deaf people and hearing people, including sign language training courses and Internet facilities for Deaf users. There are also many technologies to help Deaf people access public services. There are some projects for Deaf users, such as the e-learning for Deaf students [25], and the videophone for Deaf people [82]. A text relay service is a way to help Deaf people talk to hearing people. However, we have noticed that Deaf people choose to use sign language communication systems, because sign language is their preferred language. VoIP technology is easily adapted for video. The Telephone Interpreting Service of South Africa (TISSA) [57] included a video-based communication system in South Africa. TISSA is a remote interpreting system which incorporates all the eleven official South African languages. For a short time, TISSA provided access to government services in sign language for Deaf people [57]. The most advanced mobile video project for Deaf people is MobileASL [11, 12]. It is a Deaf-to-Deaf video communication project. MobileASL provides video call services on mobile phones, using the existing American mobile network. BANG built a semi-automated text relay system

called Softbridge Instant Messaging Bridging Architecture (SIMBA), to provide Deaf-to-hearing communication through a human relay operator [74]. A H.264 codec video for Deaf video was also done. Some Deaf people used the system and have given feedback [48]. The results of that project showed that a video communication system with high video quality might match the requirements of Deaf people in our target community.

Although all these systems provide video communication service for Deaf users, none of them are widely used by South African Deaf people in their daily communication. There are some questions to consider, such as:

- We know that Deaf people want video communication rather than a text service. Do they prefer synchronous video or asynchronous video? And why?
- Some projects have had good evaluation results. Why do South African Deaf people not use these products in their real-life?
- These projects have built their prototypes on a variety of platforms with different technologies. Is there any technology which best matches the special needs of Deaf people?

To answer these questions, we decided to build and evaluate some video alternatives for Deaf people.

1.4 Approach

We want to implement and experiment with video communication prototypes for Deaf users to provide video services with acceptable quality. **How can we choose a promising video communication alternative to provide acceptable sign language communication service, and make sure it is easy to use for Deaf people's daily conversations?** This project, therefore, designs and compares prototypes to evaluate acceptable sign language video quality. This thesis describes the process of identifying a video communication technology that provides acceptable sign language video for Deaf people. Some Deaf related projects such as MobileASL and BANG's H.264 asynchronous video system are introduced. Two kinds of browser-based technologies — Flash and HyperText Markup Language 5

(HTML5), and two kinds of mobile technologies — Java 2 Micro Edition (J2ME) and Android are chosen. The challenges are also detailed. Firstly, the researcher gathers the specific user requirements from Deaf people in a design phase. Secondly, the User Interface (UI) of the prototypes should be simple and clear to let Deaf people use all the features easily. Thirdly, in the experiment phase, the researcher needs to find a reasonable way to measure the Quality of Service (QoS). A user test should be done to gather user feedback from Deaf people. The overall approach to explore this problem comprises three phases.

The first is the design phase. Before the prototypes are built, we should clarify the user requirements. Since Deaf people have a distinct culture, it is not suitable to use common user requirements for hearing people for the project design. Both technical details and social problems should be considered during the design phase.

The second is the implement phase. We try to present a simple and clear UI to Deaf people, hiding complex technical details. At the same time, Deaf users should take advantage of these technical features to get a comfortable user experience. Video quality needs to be well considered for it is the crux of sign language communication.

The last is the experiment phase. Traditional measurement of video quality focuses on video codecs [78]. However, the parameters affecting on-line video communication for Deaf people are different from the parameters in normal video quality evaluation. We have noticed that there is no way to evaluate Deaf video communication, especially asynchronous, using the traditional QoS methods. In this project, we need to find different ways to measure the QoS. User feedback from Deaf people is very important. User trials should be done to collect feedback data.

This research intends to help Deaf users to access the network technology in the real world naturally. This project focuses on both technology and practical aspects. Deaf culture and user behaviour will have a large effect on the research results. This thesis presents all of the technical details, the collection and analysis of user behaviour data.

1.5 Thesis outline

Chapter 2 presents related work. Firstly, research on Deaf communication is introduced. Text relay, video relay and sign language video systems are mentioned. BANG projects are also described in this chapter. Secondly, two kinds of video alternatives for hearing people are explored. The technologies used in those projects are introduced. Then the evaluation of video quality is described.

Chapter 3 discusses research methods and experimental design. The research questions are detailed and explored. We posit some research methods to answer the research questions, including qualitative, quantitative and software engineering methods. Then the experimental design is described. The arrangement of user tests is presented, and each part of the experimentation is detailed. We also discuss ethical considerations.

Chapter 4 describes the system design for the four prototypes prepared for the research process. The requirements of the prototypes are specified. Both system and UI design are presented.

Chapter 5 discusses data collection and analysis. The results of background data, user tests and performance tests are presented and discussed. The details of software installation are also mentioned. Finally, all the results are combined and analyzed.

Chapter 6 concludes the thesis. The contents of the thesis are summarized. A conclusion is drawn based on the results in Chapter 5. Suggestions are made for technologists working with Deaf users. The limitations of the research are pointed out. Future work is also suggested.

Appendices include the questionnaires used in the experimentation phase, with the actual results. Three questionnaires are listed: technology background, pilot trial and full trial. User guides used in the full trial are also presented there.

Chapter 2

Related Work

This chapter introduces work related to our project. Work related to communication applications for both Deaf and hearing people are presented. The former shows that the mainstream develops directions for Deaf communication system. The latter contains possible architectures that can be adopted for sign language communication. Technology details are also mentioned in this chapter. Section 2.1 introduces the projects for Deaf people. Section 2.2 links the applications for hearing people. Section 2.3 discusses the evaluation of video communication for both hearing and Deaf people.

2.1 Deaf communication

Relay services [64] and Deaf-to-Deaf video communication [12] are the mainstream research directions for Deaf communication projects. Figure 2.1 shows a classification of Deaf communication projects introduced in this section. Three directions are discussed: text relay, video relay and sign language video. Text relay and video relay are relay services. A relay service is a service that allows a Deaf person to communicate with a hearing person in real-time via an interpreter. A relay service is often used to make the communication between Deaf people and hearing people easier, while sign language video helps Deaf people communicate with other Deaf remotely.

Section 2.1.1 describes text relay and some projects. Section 2.1.2 describes video relay and some projects. Section 2.1.3 addresses sign language video projects.

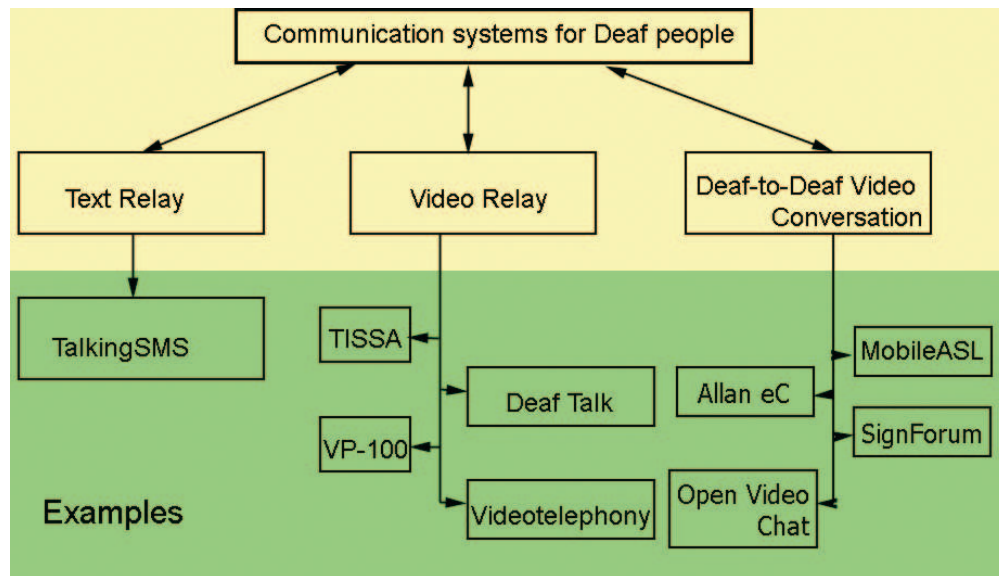


FIGURE 2.1: Deaf communication projects: The top of the figure lists three main branches in Deaf communication development. They are text relay, video relay and Deaf-to-Deaf video conversation. Below the branches some examples are presented. These are detailed in Section 2.1.

Section 2.1.4 summarizes the Deaf communication research agenda of our research group in South Africa.



2.1.1 Text relay

Text relay connects people using a text device to people using a telephone via an interpreter. A Deaf user inputs text, that is translated by an interpreter. The interpreter communicates with a hearing person in voice, then uses text to give the Deaf user a reply. Text relay makes it possible for Deaf people to communicate with hearing people via a telephone. Actually, the audience of a text relay system is not Deaf, but deaf, people. Many Deaf people are not accustomed to using spoken language to communicate because of the reasons mentioned in section 1.2.

There is research which uses automatic text-to-speech translation in a message relay service for deaf people[67]. For example, talkingSMS [59] is a text message to voice service in South Africa. This service is also being tested in the United Kingdom [45].

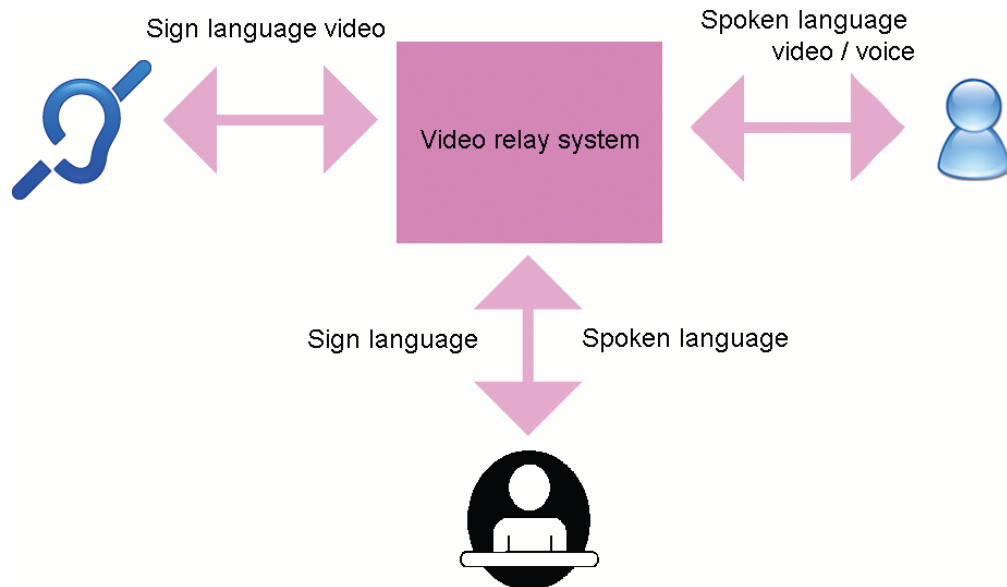


FIGURE 2.2: Video relay: A Deaf user could communicate with the interpreter using sign language, while the interpreter communicates with a hearing user using spoken language.

2.1.2 Video relay

Video relay [19] allows Deaf people to communicate with hearing people in real-time via an interpreter, albeit with sign language instead of text. Figure 2.2 shows an architecture for video relay. Video relay is more acceptable to Deaf people than text relay. Text relay asks Deaf people to use spoken language (text) to communicate. Since sign language is the preferred language of Deaf people, video communication is more natural for Deaf users.

Videotelephony is a video relay project running on both mobile phones and PCs, in the Netherlands [28]. It provides a semi-synchronous video relay service for Deaf people, via a sign language interpreter. The research demonstrates the relationship between network bandwidth and a Deaf user's experience. A conclusion is drawn that video relay can satisfy many Deaf users. A suitable network speed — 128 Kbps was established. It is clear that Deaf people are glad to accept this communication technology to help them communicate with hearing people [28].

VP-100 [19] is another video relay application for Deaf people running in America. VP-100 is a system including both software and specific hardware. The software allows a Deaf user to reach telephone users through a video relay service. The



FIGURE 2.3: The user interface of VP-100: The user interface for dial is presented in this figure [19]. A Deaf user selects the menu ‘Dial’ to connect with the relay interpreter. Then the interpreter helps this Deaf user to communicate with a hearing telephone user which whom the Deaf user wants to chat.

hardware was developed to run the system. Figure 2.3 is a screen-shot of the interface of VP-100.

Deaf Talk is a video relay system used widely in American hospitals [39]. A specific device is needed for this system. The device is placed on a rolling cart, and is able to be moved to any room with a phone line. The hospital only needs to install Integrated Services Digital Network (ISDN) lines in all the areas where the system would be used. The cost of the system and the ISDN fees are cheaper than a full-time interpreter, and Deaf Talk decreases the waiting time of the Deaf patients. In 2002, Montefiore Medical Center of America started to use Deaf Talk system [39]. Now, more than 150 American hospitals are using this system.

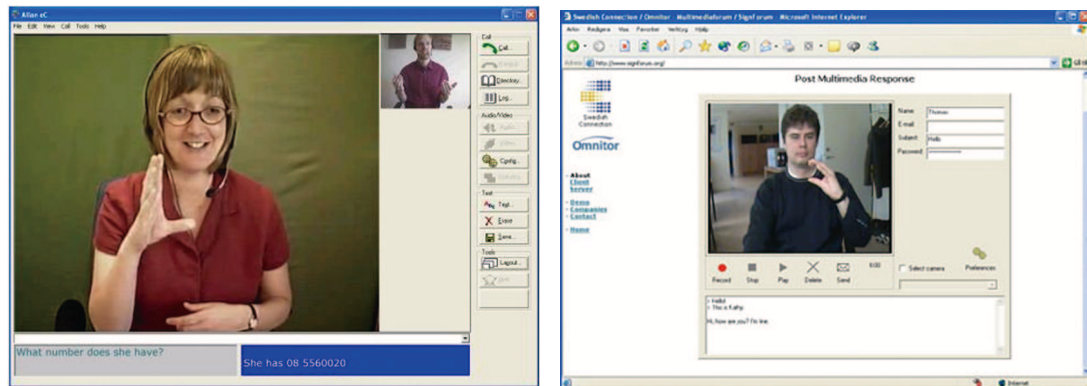
TISSA is a video relay project in South Africa [57]. It is a system to help Deaf people access public services. TISSA was the first project to provide such a service in South Africa. TISSA was meant for all eleven official languages in South Africa. TISSA wants to raise the quality of life of all South Africa people by

giving all the people an “equal voice”. This service is funded by the South Africa government. It is provided at selected government sites on weekdays. Since most of the government staff do not understand SASL and many Deaf people do not know spoken languages, it has been difficult for Deaf people to access government resources. TISSA provided SASL translation for Deaf users during a trial period in 2002 and was then cancelled. For a brief time, it gave South African Deaf people an opportunity to access public services.

2.1.3 Sign language video

Sign language video systems focus on Deaf-to-Deaf communication. Such a system provides sign language video services for Deaf people. Omnitor AB is a communication-based knowledge company which provides accessible communication services for people with disabilities. There are two sign language products developed by Omnitor: Allan eC and SignForum. The name Allan eC is derived from ‘All languages electronic Conversation’. This product offers the possibility to use text, voice and video simultaneously in a conversation. Allan eC contains a camera, a video capture board, a headset, an alerting system adapter and software. It runs on a PC, using Internet Protocol (IP). Figure 2.4(a) shows the UI of Allan eC. Deaf users can achieve sign language communication using this product. SignForum is a web discussion forum supporting sign language through video contributions. Deaf people can participate in SignForum discussions by using the client software to record a sign language video. The video is sent to the Server in the same way as conventional text messages. When a user views a previous message the video file starts automatically. The client software of SignForum provides video recording and updating features for Deaf people. It runs as an embedded program in Internet Explorer. Figure 2.4(b) shows the UI of the client software of SignForum.

Open Video Chat is a standard video chat program that enables a Deaf user to communicate with other Deaf users through connected PCs. This system was developed by the students of Rochester Institute of Technology. The program provides clear sign language video (tested using American Sign Language (ASL)) for Deaf people, with text chat, an on-line white board and video phone services on a local network. Open Video Chat runs on a specific PC Operation System (OS). It is designed for One Laptop Per Child computers. One Laptop Per Child



(a) Allan eC

(b) SignForum

FIGURE 2.4: The user interface of Omnitor products: Figure 2.4(a) shows how two Deaf people can use synchronous video services to communicate in Allan eC. Figure 2.4(b) shows how a Deaf user can record a video message and post it to SignForum. (www.omnitor.se/index_eng.html)

is a global initiative that seeks to provide low cost, rugged, connected laptops to children across the world. Figure 2.5 shows the UI of video communication using Open Video Chat [23].

The above sign language systems are based on PC platforms. To use these systems Deaf people need to prepare external devices like a web-cam or camera. However, the PC platform is not the only choice for sign language video communication systems. MobileASL is a portable video communication system for Deaf people, running on mobile devices with ASL [11, 12]. MobileASL is a real-time video communication system designed and built for Deaf people. The MobileASL system provides a synchronous sign language video service via the American General Packet Radio Service (GPRS) network. Figure 2.6 shows a video conversation in MobileASL.

To balance the video quality and bandwidth issues, MobileASL uses skin detection algorithms to find important areas in the video, called Region of Interest (ROI), and focuses on the movement within these areas only. These are areas of the body that are most used to communicate in sign language. These areas are targeted by MobileASL, for the video quality of these areas is more important than other body parts. Figure 2.7 shows how skin detection works in the MobileASL project.

MobileASL uses an H.264 standard-compliant video encoder. The setting of the encoder is optimized for ROI based coding [76]. MobileASL applies this H.264

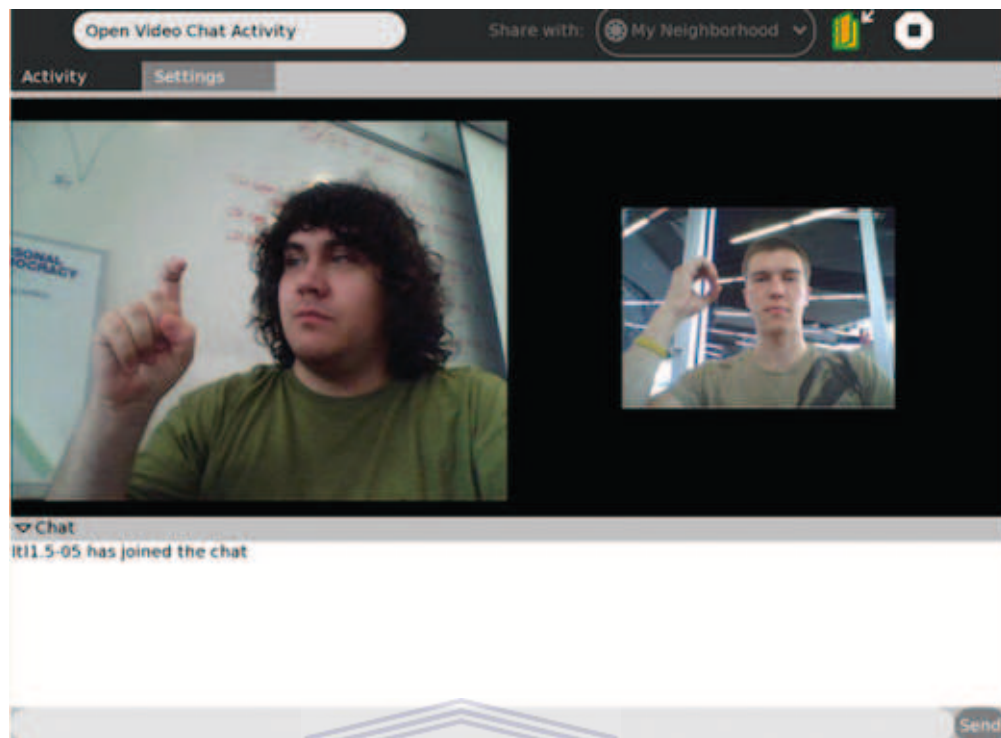


FIGURE 2.5: The user interface of Open Video Chat: This figure shows how two Deaf users of Open Video Chat can achieve sign language communication using this system. The system only runs on One Laptop Per Child computers [23].

UNIVERSITY of the
WESTERN CAPE



FIGURE 2.6: A MobileASL video conversation: This figure shows synchronous video communication between two sign language users [13].

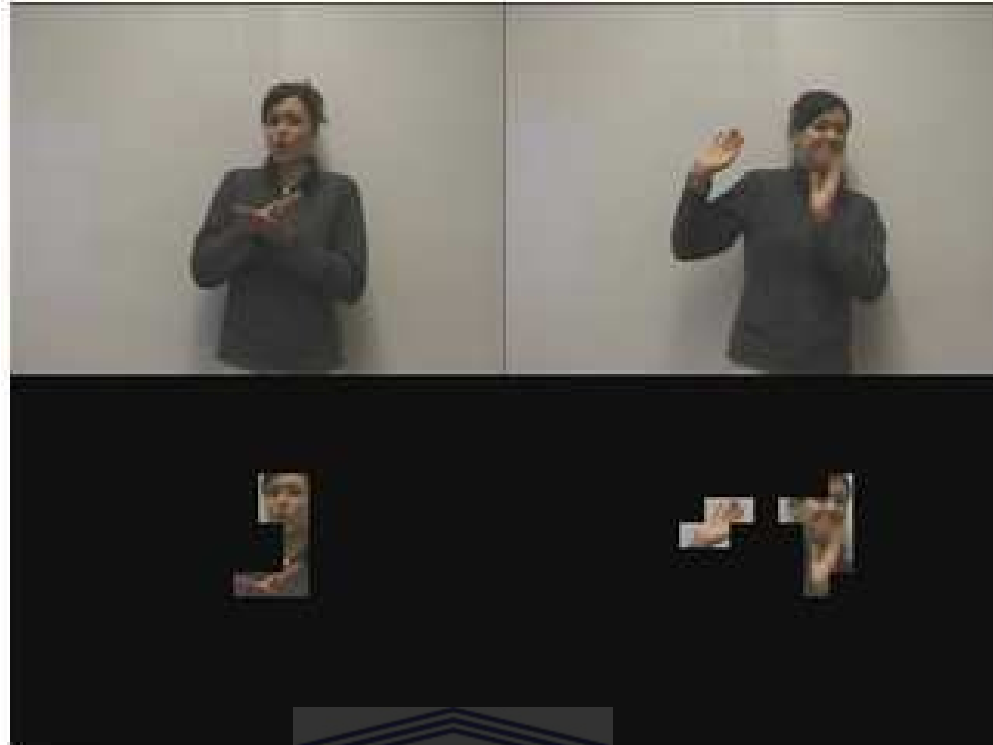


FIGURE 2.7: Skin detection algorithms in MobileASL: The left bottom figure shows the skin detection of the left top figure. The right bottom figure shows the skin detection of the right top figure. The parts of detection are face and hands, for these parts present most of the expressions in ASL. (mobileasl.cs.washington.edu)

encoder to ASL video on a mobile device. Encoding parameters are developed and tested to evaluate the system. The parameters specify the level of sub-pixel motion estimation in ROI, the partition size restrictions for non-ROI macro-blocks, and the ROI-based motion search complexity [76]. A fast off-line algorithm is used to choose parameter settings in MobileASL. User tests are done to gather qualitative data. The details are presented in section 2.3.2, where we discuss video evaluation.

Another sign language video project for Deaf people is an emergency medical scenario. A mobile system in Italy [8] allows a medical responder to tell Deaf patients some details about the patient's disease using sign language videos. This system runs on a Windows Mobile phone. The doctor chooses a sentence from a menu, and a static sign language video is played on the phone for the Deaf patient. This project is a Deaf-to-hearing communication project. Different from MobileASL, this is not a remote communication service for Deaf people.

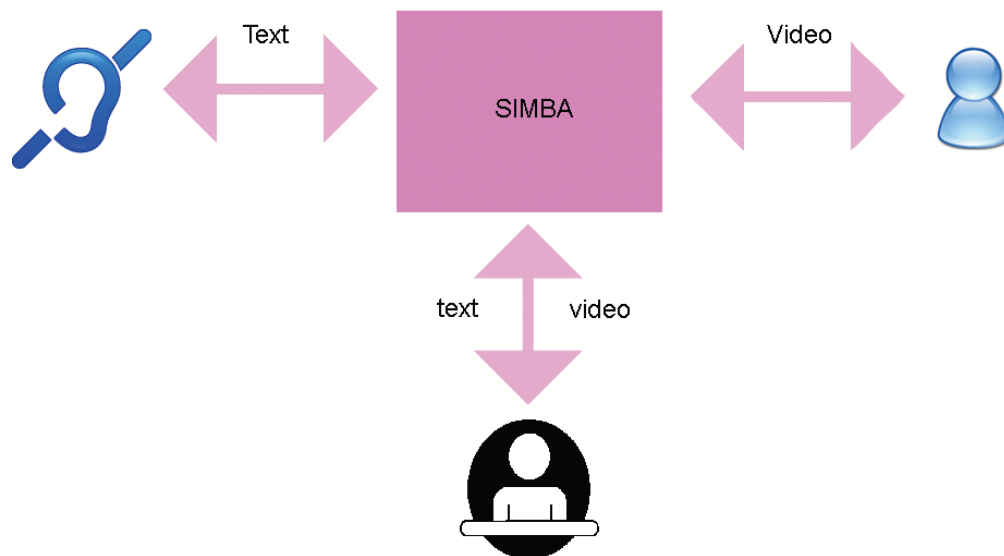


FIGURE 2.8: An overview of SIMBA [74]: There are three roles in a SIMBA conversation. A Deaf user types on a keyboard, and the text message is sent to the SIMBA server. The server sends the text to a relay operator. The interpreter translates this text message into spoken language and talks to a hearing user. Once the hearing user speaks, the interpreter translates that into text and sends it to the Deaf user via instant messaging.

2.1.4 BANG projects

BANG has rich experience of network research for Deaf people, cooperating with DCCT. The staff and members of DCCT have been invited to assist in BANG research projects since 2001 [74]. The earliest BANG project for Deaf people was Telgo323 [62], which presented a semi-automated text relay service from a Deaf user through a Teldem device to a phone of a hearing user using Text To Speech (TTS) [62].

In the next phase, a semi-automated TTS relay system called SIMBA [74] was done. SIMBA is a locally developed system to provide text relay in South Africa, just like the systems introduced in section 2.1.1. SIMBA provided way Deaf-to-hearing communication to the Public Switched Telephone Network (PSTN) with a human sign language interpreter and TTS. Figure 2.8 presents the work flow of the SIMBA system. Session Initiation Protocol (SIP) was used in this project to call traditional phone devices from the Internet. A Deaf user can also use a mobile phone as the client device for typing.

In another project, an asynchronous video was built and tested [48]. The project developed a semi-synchronous video communication system with high video quality and minimal latency services [48]. This system is for Deaf-to-Deaf communication. Some DCCT members assisted in this test. The project showed that H.264 codec can be adapted for the video quality requirements of Deaf users. At the same time, other phases of the SIMBA research continued. Kiara [83] is work-in-progress research to provide a relay service based on SIP. Automatic speech recognition is intended to be used in this system to provide full automatic relay.

2.2 Video technologies

In Section 2.1 some projects for Deaf people were introduced. In this section, some applications for hearing people are presented. Although these applications are not for Deaf users specially, they could also be used by Deaf people, albeit with some limitations.

Section 2.2.1 introduces off the shelf video systems. Section 2.2.2 discusses browser-based video application. Architecture and protocols are also indicated. Section 2.2.3 addresses mobile video applications. The technical details about video communication are introduced here.

2.2.1 Off the shelf video systems

There are many commercial and free video communication products available on the Internet. Some of them are desktop applications running on specific OSs, such as Skype (www.skype.com), Camfrog (www.camfrog.com), MSN (www.msn.com) and Google talk (www.google.com/talk). These applications provide text messaging, voice and video communication services to their users. On-line users can also leave a text message to their off-line friends using these applications. Some communication systems are browser-based systems which are cross-platform. Tokbox (www.tokbox.com), Facebook (www.facebook.com) and Winkball (www.winkball.com) are examples of browser-based communication systems. The features of browser-based video systems are similar to standard video applications. However, users of the former do not need to install client software on their PCs. Dimdim (www.dimdim.com) is an on-line video conferencing system. It aims to provide remote conferencing

service for the users. Document sharing and an on-line white board are provided by Dimdim besides text messaging and video chatting.

There are two kinds of video transmission: asynchronous and synchronous. Asynchronous video does not require the message receiver to answer the video immediately. Synchronous video, on the other hand, is similar to face-to-face communication. Facebook supports asynchronous video communication, Winkball supports both asynchronous video and synchronous video, and all the other introduced applications provide synchronous video service. For commercial reasons, we do not know exactly how the video transmission works in the above systems. Skype, Camfrog and MSN use their own protocols which are closed-source. Google talk uses a open protocol called Extensible Messaging and Presence Protocol (XMPP). Tokbox and Dimdim use the proprietary protocol Real Time Messaging Protocol (RTMP). We cannot find any information about the protocols used by Facebook and Winkball. However, from their presentation we infer that Facebook supports Hypertext Transfer Protocol (HTTP), while Winkball uses RTMP.

2.2.2 Browser-based video

A browser-based application refers to an application that runs in a web browser. Common examples of browser-based applications include web mailboxes, web shopping sites, on-line directories and on-line games. A browser-based video system provides a video communication service inside a web browser. A standard video system requires the user to install a client application on the local computer before using the service. Browser-based video communication just needs the user to open a specific web page to connect to the system.

Tokbox is a browser-based communication system. It supports live video using Adobe Flash. Users of Tokbox need not install special plug-ins on their PC. Tokbox is like a web version of Skype without PSTN breakout. Dimdim is another browser-based video conferencing system based on Adobe Flash. It is open-source and supports multiuser meetings. Dimdim users need to install custom plug-ins to use advanced features such as desktop sharing.

There are many examples providing browser-based video with Adobe Flash, or just Flash. Flash is the name of an Adobe multimedia file format, and the name of the Adobe Flash software series. The original version of Flash was proposed in 1996

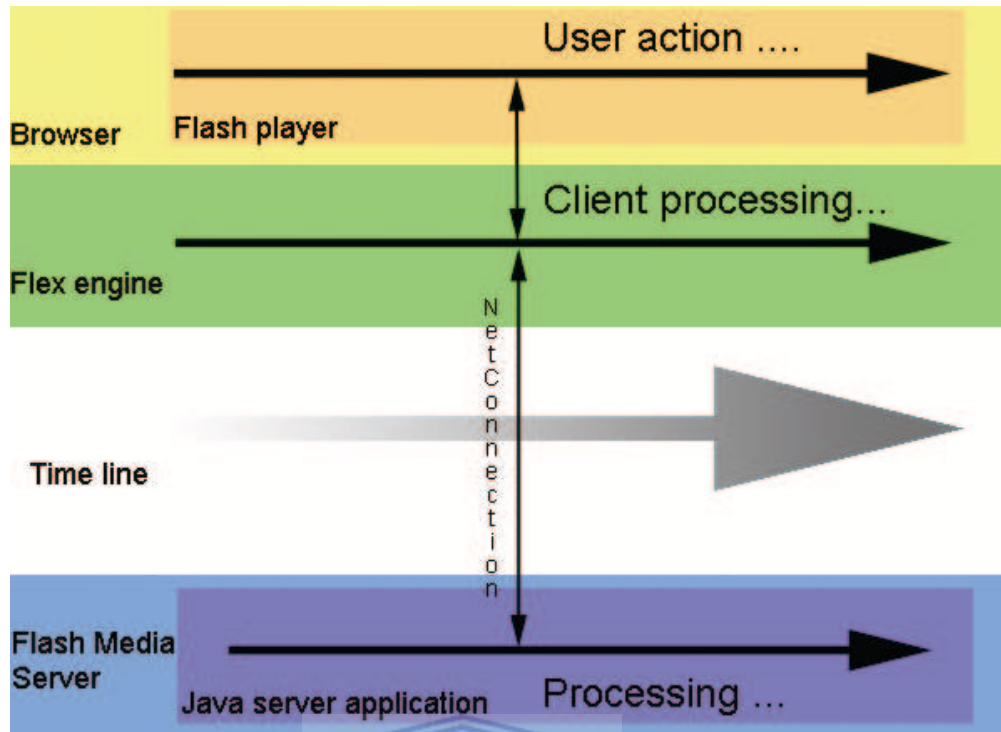


FIGURE 2.9: A Flash-based application model: The architecture of a Flash-based application is shown. There are two parts to the architecture: a Flex client and a Flash server. The two parts communicate via RTMP. The Flex client runs inside a web browser, packaged in a single file. The Flash player extension is used to play the packaged file. The Flex engine responds to some user actions and communicates with the server using a NetConnection object via RTMP. The server manages video streaming and handles user requests.

by Macromedia. The main use of Flash was to develop vector animations [42]. However, this animation technology has been enhanced by Adobe. Today, Flash is the abbreviation of a powerful technology series, with the full name Adobe Flash Platform. Adobe Flash Platform technologies include Flash Player, Adobe AIR, Flex, and Flash Media Server [58]. A cross-platform browser-based synchronous streaming system can be created using Adobe Flash Platform technologies. Figure 2.9 shows a Flash-based model. A Flash-based system often has the following components: **Flex client** to run user actions, **Flash media server** for processing requirements and **Flash player** to play **Flash Video (FLV) files** inside a browser, via a special prototype. These components are detailed below:

Flex is similar to Asynchronous JavaScript and XML (Ajax), and is used to develop Rich Internet Application (RIA)s with runtime Flash. It runs on the client side and always appears with a Flash server. The programming language used in Flex is ActionScript [66]. ActionScript was originally used to control animation,

making a figure appear or disappear. Today, it can access some local devices and communicate with a Flash server. A new version of ActionScript [7, 65] was released with the Flash Player 9 alpha.

Flash media server [44] is a Flash server application container. Flash server application runs in the Flash media server, communicating with a Flex client via RTMP. The server application is coded in Java, using specific Flash Application Programming Interface (API)s. A Flash server application handles multimedia streams. Two Flex clients can publish a net stream, and each user can get the net stream from the other client through the Flash media server. Figure 2.10 shows how Flash media server and Flex client work together. The published video is broadcast. Users can receive the video streaming with a unique identity for the video.

Flash Player is the player for Flash. Today more than 97% of Internet users are able to view Flash files through a Flash Player [22]. Flash Player has two types, the standard Flash Player and the plug-in Flash Player. The newest version Adobe Flash Player 10 was released with some new features such as vector data type support. All mainstream PC web browsers can coordinate updating Flash Player plug-ins running inside them. However, the Flash support on mobile operating systems is not that good. Some mobile operating systems support Flash Lite, which is a lightweight version of Flash with poor graphics quality. Only a few mobiles support the full version of Flash. Some important features like local device access are not available in Flash Lite.

FLV file format includes a header and three different tags — audio tag, video tag and data tag. Each tag comprises a single stream in a FLV file. There cannot be more than one video steam or audio stream in one FLV file. The header tag contains information about the file signature, the file version, tags presented in this file and the length of the header part. The audio tag includes an audio stream. The video tag contains a video stream. The data tag constitutes meta-data information of the FLV file. The data tag can be kept at the start of the file to commence playback before the download is finished. The features of the video file are stored in the meta-data tag, such as during time, screen size, bit rate and frame rate. FLV has a good balance between compression and graphics quality. FLV is the only permitted format which can be used in synchronous video streaming between the Flash media server and a client application.

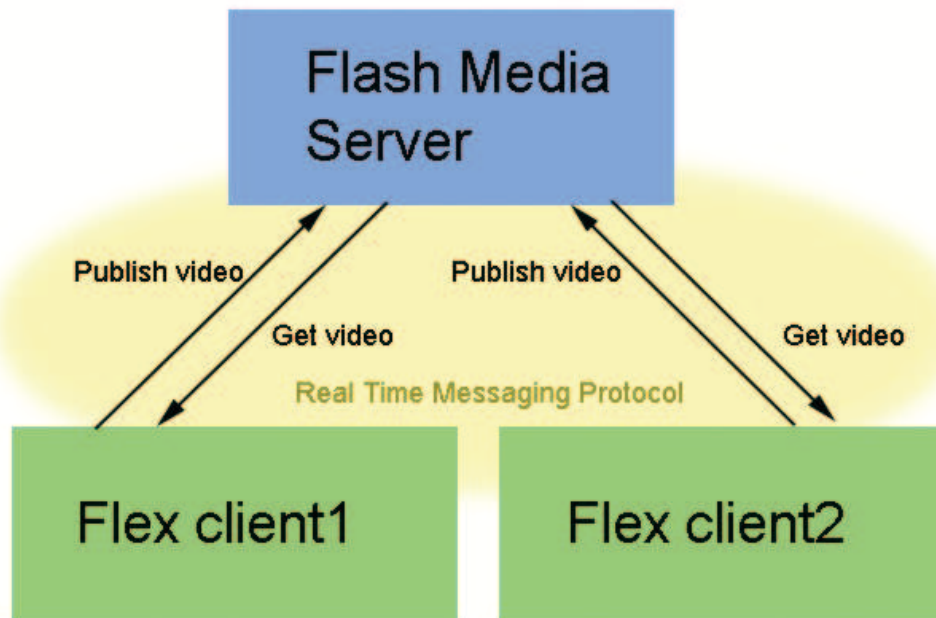


FIGURE 2.10: Client-server communication via RTMP: The video transfer in a Flash-based system is not from client1 to client2. Client1 broadcasts his video onto the network. A unique name is used as the key when receiving video. Since the unique name of client1 is not public for other clients, client2 is the only one who can receive the video.

RTMP [44] is a protocol developed by Adobe. It is used to stream audio and video data between a Flex client and a Flash media server. RTMP is based on Transmission Control Protocol (TCP). Flex technology allows the user to create Flash objects as a programmer, not an animation designer.

Another new and powerful browser-based technology for video is HTML5. Apple's website (www.apple.com) is built by HTML5. Youtube (www.youtube.com) supports Flash video playing, and also has a HTML5 version. HTML5 is the newest version of HyperText Markup Language (HTML), the core markup language of Internet web pages. HTML5 is a revision based on HTML4. HTML5 adds new tags and new API, and incorporates web forms 2.0. On-line multimedia play [63] is a basic feature of HTML5. HTML5 is an RIA development platform. It is one of the Ajax [29, 61] technologies. It can be used to build the client of an Ajax prototype. At this time, all the mainstream web browsers support HTML5 in variable degrees. HTML5 can be combined with other browser-based technologies to provide RIA service. Figure 2.11 shows the architecture of an HTML5 application.

HTML5 defines a standard way to embed video in a web page, using a ‘video’ element. The video element is a media element whose media data is ostensibly video data, possibly with associated audio data. It includes ‘src’, ‘preload’, ‘autoplay’, ‘loop’, and ‘controls’ attributes to define videos. These five attributes are the common attributes to all media elements. The ‘poster’ attribute sets the address of an image that the HTML5 web page can show while no video data is available. HTML5 web pages may provide messages to users (such as ‘buffering’, ‘no video loaded’, ‘error’, or more detailed information) by overlaying text or icons on the video or other areas of the element’s playback area.

HTML5 video is intended by its creators to become the new standard way to show video on-line, but the support for the video element is still evolving. Lack of agreement has hampered the uptake of HTML5 video. Table 2.1 shows that the mainstream browsers support HTML5 video in various versions. If a PC user uses Internet Explorer (IE)8 on his computer, he is unable to watch HTML5 videos in his browser.

TABLE 2.1: Video element support: The different browsers support HTML5 video by version.

IE	Firefox	Safari	Chrome	Opera
9.0+	3.5+	3.0+	3.0+	10.5+

HTML5 video allows developers to contain any type of video file in the web page, no matter the video codec, audio codec or container format. One <video> element can link to multiple video files, and the browser will choose the first video file it can actually play. However, some video formats are not supported by the specific browsers. Table 2.2 shows how the mainstream browsers support video formats [4, 20, 33, 34, 36, 37, 51, 53, 55, 56].

TABLE 2.2: Video format support: The different browsers support video codecs and container formats in different versions.

Browser	Ogg Theora	H.264	VP8 (WebM)	Others
IE	—	9.0+	—	—
Firefox	3.5+	—	4.0+	—
Safari	—	3.1+	—	Depends
Chrome	3.0+	3.0+	6.0+	—
Opera	10.5+	Depends	10.6+	Depends

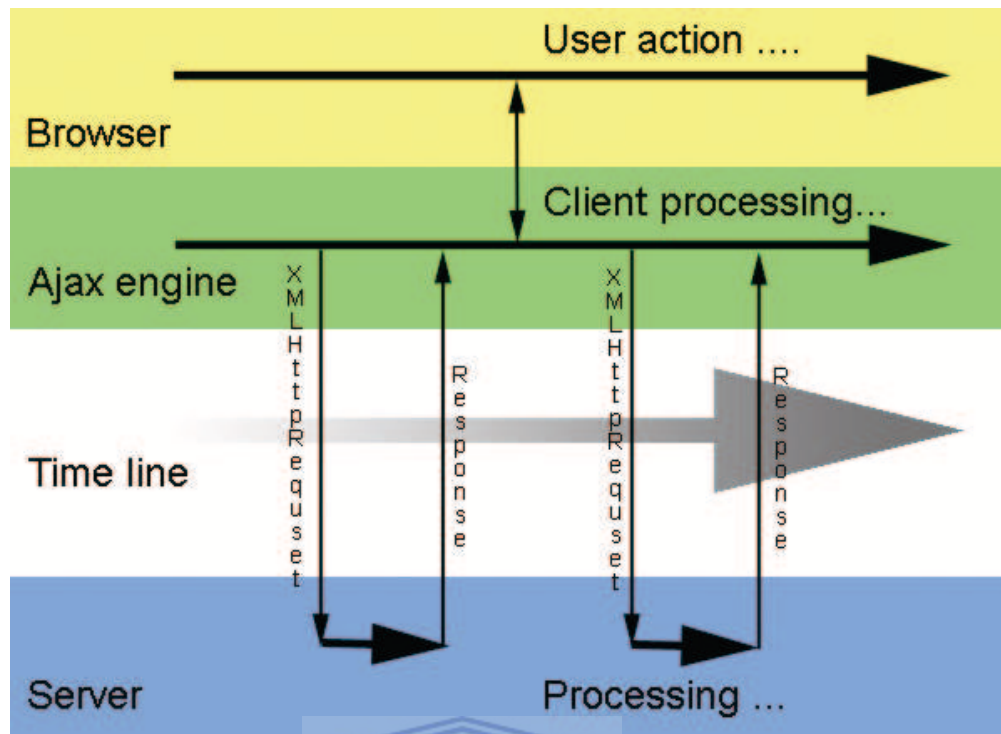


FIGURE 2.11: A HTML5 application model: The components of HTML5 applications work together like this: the HTML5 page is shown inside a browser, the client Ajax application responds to user actions, and the server responds to user requests via HTTP.

HTML5 video is not currently as widespread as Flash video. However, some HTML5-based video players are being developed by experimental companies, such as Youtube (using the H.264 and WebM formats) [9] and Vimeo (vimeo.com, using the H.264 format) [24]. These companies suggest that interest in adopting HTML5 video is increasing.

TABLE 2.3: The comparison of Adobe Flash and HTML5 about browser-based video features.

Alternatives	Video Play		Protocol	Browser Support
	Asynchronous	Synchronous		
Adobe Flash	OK	OK	RTMP	Very Good [22]
HTML5	OK	NG	TCP/IP	Varied (Table 2.1, 2.2)

Table 2.3 presents the features of Adobe Flash and HTML5. There are other lesser-known browser-based technologies that provide video services, such as Microsoft Silverlight. Silverlight is an RIA development platform. It can play multimedia resources such as animations and vector graphics. Microsoft Silverlight runs on the Microsoft Windows series and Mac OS X operating systems, and it could also run on Xbox 360 in the future. Internet Explorer, Firefox and Safari support this technology. Silverlight provides many advantages in multimedia play. Silverlight handles multimedia data with the Windows Media Video (WMV) and

running on a mobile phone. A user is able to make free Skype-to-Skype video calls using a Symbian phone. There are also some other features such as sending and receiving instant messages, sharing files and voice calls. Kyte (www.kyte.com) is another video system running on a Symbian OS. Kyte provides video services for live and on-demand content. Fring (www.fring.com) is another VoIP application that also does video calls on mobile phones. Users of Fring are able to make free mobile calls, video calls, synchronous chat and file sharing. Fring has many versions for different mobile OSs, including iPhone, Symbian and Android (developer.android.com). The Symbian OS is based on a hand-held PC OS. It is designed for those mobile devices which work continually for several months with only a few resources. Symbian has five layers in its architecture. The application service layer provides support independent of the user interface for the application [54]. Most applications and mobile software run in this layer. J2ME applications and Java Virtual Machine (JVM) are handled in this layer. J2ME is a Java platform designed for embedded systems like mobile phones. J2ME is aimed at machines with little memory and a less powerful processor than those used on typical desktop and server machines. J2ME consists of a set of profiles. Each profile is defined for a particular type of device. Mobile Information Device Profile (MIDP) is the profile for a mobile phone. MIDP includes a UI and a data storage API. Applications written for this profile are called MIDlets. Connected Limited Device Configuration (CLDC) contains a strict subset of the Java-class libraries. CLDC is the minimum amount needed for a JVM to operate. CLDC is basically used for classifying myriad devices into a fixed configuration. CLDC gives developers a solid Java platform for creating applications for mobile devices. Java support has been included in the Symbian OS from the beginning [54], and Symbian combines with Java technology closely by providing the following support [54]:

- Layer support and JVM based on the MIDP 2.0.
- Standard MIDP packages installed inside the Symbian OS.
- An implementation of the CLDC 1.1 language and utilities services.
- Low-level plug-ins to communicate between CLDC, MIDP packages and the Symbian OS.

Many applications running on Symbian are built using J2ME, as it is not device-related and can communicate with different hardware. J2ME applications handle

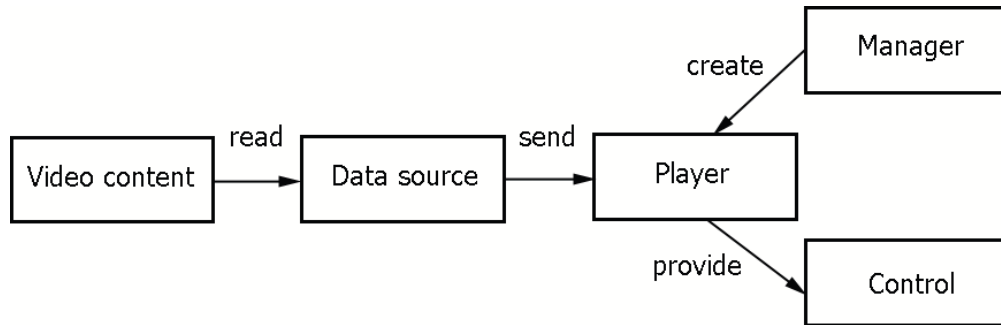


FIGURE 2.12: MMAPI model: **Data source** encapsulates protocol handling by hiding the details of how the data is read from the source. **Player** reads the data from Data source, processes it, and renders it to an output device. **Control** provides methods to control media playback. **Manager** is a factory mechanism to enable a J2ME application to create players.

multimedia files using Mobile Media Application Programming Interface (MMAPI). MMAPI is an optional package that supports multimedia applications on J2ME-enabled devices. Both the protocol and the video content need to be handled when using MMAPI. Figure 2.12 shows how MMAPI handles multimedia data [49].

Android is a new mobile operating system based on a Linux kernel, developed by Google. Android OS is similar to a standard PC OS. The default web browser in Android supports some features of HTML5 and Adobe Flash 10. Android provides some services to let users use Google's RIAs easily. It is open-source, and programmers can develop applications on this OS by using a Software Development Kit (SDK). The development of an Android application is quite similar to Java development. Android SDK includes many useful Java packages and has many new Java APIs and smart phone features. Android includes API libraries for development involving the device hardware [52], such as the camera. Android provides a standard way to communicate with devices, which means the Android applications work as expected on all Android phone devices. Android allows developers to build background services, such as checking email. The background services proceed without an interface showing on the screen. It is easy to build a client application to receive signals from a remote server. Such a client application can be executed invisibly even when another program is shown on the top. It is possible for a developer to create a program which reads and writes private user data from the phone. Inter-application communication [52] is also available for data sharing.

The Android platform offers built-in encoding/decoding for a variety of common

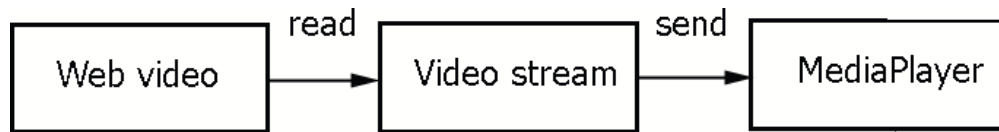


FIGURE 2.13: Android video model: The application reads a video file into a stream class, and sends the stream to the MediaPlayer class. The MediaPlayer class plays the video on the canvas, and handles playback actions.

media types. H.264 and the 3rd Generation Partnership Project (3GPP) videos are supported by Android. The MediaPlayer class is used to play video and audio files in Android applications. The data source of a video file can be from a media file stored in the application's resources, from stand alone files in the Android file system, or from a data stream arriving over a network connection. Figure 2.13 shows how an Android application plays a video stored on the Internet.

2.3 Video evaluation

Video evaluation means the evaluation of video quality. Video quality is usually measured from two directions: subjective and objective direction. Subjective video quality is concerned with how video is perceived by a subject and represents the subject's opinion on a particular video sequence. Many environment parameters can influence the results, such as room illumination, viewing distance, and the age and educational level of subjects. The main method of measuring subjective video quality is to use a Mean Opinion Score (MOS) to provide a numerical indication of the perceived quality of video files [41]. The usual work flow of the subjective video quality evaluation includes:

- Choose video sequences for testing.
- Choose settings of the target system to evaluate.
- Decide how video sequences are presented to subjects and how their opinion is collected.
- Invite a sufficient number of subjects.
- Conduct testing.
- Analyze the results based on subjects' feedback.

Objective video quality evaluation is the mathematical model based on criteria and metrics that can be measured objectively and automatically by programs. The traditional ways of digital video evaluation are Signal to Noise Ratio (SNR) and Peak Signal to Noise Ratio (PSNR). They both measure the difference between the original video signal and signal passed through the target system. PSNR is the most widely used objective video quality metric. Recently, a number of more complicated and precise metrics were developed, such as Perceptual Evaluation of Video Quality (PEVQ) (PEVQ is a standardized end-to-end measurement algorithm to score the picture quality of a video presentation by means of a 5-point MOS) [18].

Section 2.3.1 briefly presents the evaluation of video for hearing people. Section 2.3.2 presents the evaluation of video for Deaf people which has different requirements for sign language intelligibility.

2.3.1 Evaluation of video for hearing people

The evaluation of video quality for hearing people often focuses on objective video quality. Fitzek and Reisslein are interested in the relationship between video quality, bit rate and video trace [26]. A video frame size tracing library is used to analyze the QoS of Moving Picture Experts Group (MPEG)-4 [73, 80] and H.263 encodings. The evaluation of on-line video is more concerned with how bandwidth and network connection affects video quality [71, 77].

There are some elements which are always considered in Internet related video quality evaluation: frame rate [77], network delay [46], jitter [77] and bandwidth [38]. In a RealVideo performance evaluation, Wang, Claypool and Zuo prepared 388 video clips and gathered performance data from 63 International users by using a tool 'RealTracer' [77]. RealTracer can collect video statistics while the video file is playing. Wang, Claypool and Zuo compared the encoded frame rate with the frame rate when the video clip was playing on the client computer. Wang, Claypool and Zuo analyzed the frame rate of video clips across the Internet with different end-user network configurations. The data was gathered by RealTracer and analyzed using Cumulative Distribution Function (CDF). The jitter data was also gathered by RealTracer. Wang, Claypool and Zuo measured jitter by calculating the deviation of the time for playing an entire video clip. The jitter data was also analyzed using CDF [77]. In a low bit rate synchronous video evaluation,

Loguinov and Radha measured two kinds of delay: round-trip delay and one way delay jitter. Loguinov and Radha prepared a Unix video server and used three major nationwide dial-up Internet Service Provider (ISP)s to connect to the Internet. The users dialed a local access number to reach the Internet and streamed videos from the test server. The demo was a ten minute video file. It was played by a program automatically, and the delay data gathered by the client was sent to Loguinov and Radha by the program. The results were analyzed with statistics and CDF [46]. In a video evaluation via TCP, the available bandwidth was measured by using Multi Router Traffic Grapher (MRTG). MRTG is a widely used tool that displays the utilized bandwidth of a link. Jain and Dovrolis used linear analysis to handle the bandwidth data [38].

2.3.2 Evaluation of video for Deaf people

Different from projects for hearing people, both subjective and objective video quality are usually measured in Deaf-related video projects. The subjective video quality evaluation in Deaf-related video projects usually uses questionnaires with MOS. As introduced before, this evaluation is to gather subjects' feedback about a given system. This subjective video quality measurement is often omitted in projects of video systems for hearing people. It is expensive in terms of time (preparation and running trials) and human resources. However, the situation changes in video systems for Deaf people. Deaf people have a different culture from hearing people, thus the hearing researchers of video systems for Deaf people cannot so easily deduce the feedback of Deaf users. The subjective video quality evaluation is necessary for those projects that aim to provide practical video services for Deaf users.

Cavender collected both objective and subjective video quality data in project MobileASL [10–12]. Concerning objective video quality evaluation, MobileASL aimed to find an automatic intelligibility metric for sign language videos. Cavender found that the traditional video quality measures were not good enough for sign language videos. For sign language video evaluation, one must consider that the information is extracted through the face and hands. MobileASL uses skin segmentation to focus on the movements of face and hands. The intelligibility is calculated as a weighted average of Mean Squared Error (MSE) in the face and hands and is converted into a MSE-type value [10]. Subjective video quality measurement was

conducted with fifteen Deaf subjects. Two technical parameters were measured for video quality evaluation. They were frame rate and ROI. Cavender provided six setting patterns of the parameters and asked the subjects to evaluate the patterns [11]. All these users have used the system without the intervention of Cavender. Feedback was gathered by questionnaire. Cavender gathered user feedback about variable bit rate settings in another subjective test. Three different bit rates (15, 20, and 25 kilobits per second) were evaluated by eighteen adult members of the Deaf community. The Deaf subjects prefer the setting of high bit rates with high ROI [11].

The BANG asynchronous video project handled the technical details and user feedback with similar methods. Ma and Tucker tested video compression and transmission in a laboratory environment. Different codecs were examined, and some parameters of MPEG-4 codec compression were measured [48]. The relationship between video file size and transmitting speed were discussed [47]. Ma and Tucker also performed user tests by asking Deaf people to use the system and answer questionnaires. This project found a close relationship between user preferences and automated evaluation techniques [47].

2.4 Summary

This chapter introduced the related work in terms of products for Deaf people, products for hearing people and video evaluation. Relay service and Deaf-to-Deaf video communication are the mainstream research directions for Deaf communication projects. Text relay, video relay and sign language video were introduced in this chapter. BANG is a research group with rich experience of network research for Deaf people, cooperating with DCCT. The BANG projects for Deaf people were also presented. Many video systems for hearing people were also introduced in this chapter. Although these applications are not for Deaf people in particular, they could also be used by Deaf people, albeit with some limitations, such as Skype and Camfrog.

A cross-platform browser-based synchronous streaming system can be created using Adobe Flash Platform technologies. Another new and powerful browser-based technology for video is HTML5. Video on mobile phones is related to three parameters: software, mobile OS and hardware. Symbian is a popular mobile OS.

Java support has been included in the Symbian OS from the beginning, and Symbian combines with Java technology closely by providing J2ME support. Android is a new mobile operating system based on a Linux kernel, similar to a standard PC OS. The Android platform offers built-in encoding/decoding for a variety of common media types. H.264 and 3GPP videos are supported by Android.

Evaluation of video systems was also discussed. Video quality is usually measured from two directions: subjective and objective. The subjective video quality measurement is often omitted in video systems for hearing people because it is expensive in terms of time (preparation and running trials) and human resources. Deaf people have a different culture from hearing people, thus the hearing researchers of video systems for Deaf people cannot so easily interpret the feedback of Deaf users.



Chapter 3

Methodology

One reason for doing this research is that there are some gaps in the related work. For Deaf-to-Deaf communication, South African Deaf people may want to use a communication system such as MobileASL. In America this project uses GPRS to do video communication. MobileASL could potentially run on any device with a video camera situated on the same side as the screen, and was tested on some imported European phones. However, Deaf people in South Africa are unable to use MobileASL. Firstly, this is a research project of the University of Washington. There is no commercial product based on the project. Secondly, only fourteen Deaf subjects tested videos using MobileASL software [11], and only fifteen Deaf subjects tried two-way communication in another evaluation [12]. We feel that the user feedback gathered in the MobileASL research is not strong enough to show us whether this system is the best one for Deaf communication. Deaf people could also learn to use a video chat application produced for the majority, such as Skype, Dimdim and Camfrog. All these computer applications are used by some DCCT users. However, only a few Deaf people are using them frequently. As mentioned in section 1.2, Deaf people in South Africa have some unique characteristics, for example, most members of DCCT are unemployed because of their poor education. They might never have used a computer before, and it is hard for them to learn how these applications work. Many Deaf people do not know English, thus they cannot read manuals and help documents. Another reason is that these applications provide synchronous video communication for multi on-line users. Since few Deaf people have a PC, Deaf friends and family are rarely on-line for synchronous chatting. Also, the ‘hearing’ applications do not meet Deaf users’ needs very well. For example, video chat is not the core feature of Skype. The voice quality in Skype

call is prioritized, the video quality is not as good. High video quality is necessary for sign language communication. Voice can be eliminated, for Deaf people do not need a voice service at all. Mobile applications have similar problems. Although many Deaf people have learned how to use SMS, it is difficult for them to get the same user experience as hearing people. Mobile video quality also needs to be evaluated to see if it fits the requirements of sign language communication.

This chapter details the research questions and research methods. Then the experimental design is described. Section 3.1 presents the research questions. Section 3.2 proposes methods that can help to answer the research questions. Section 3.3 introduces the design of the experimentation.

3.1 Research questions

The main research question of this thesis is: **How can we choose a promising video communication alternative to provide acceptable sign language communication service, and make sure it is easy to use for Deaf people's daily conversations?** This research question can be split into three questions. The first is: what development technologies should be considered as alternatives? The second is: how to build up prototypes using the alternatives? The third is: how to evaluate prototypes to find a suitable alternative for sign language communication?

The first research question can be answered in a design phase. Two alternatives are browser-based technology and mobile technology. Browser-based systems are cross-platform and always run inside a browser. Unlike standalone applications like Skype and Camfrog, the user does not need to install software for a browser-based system. This is very important to our target audience, for they have limited computer background and might not know how to install software. Browser-based systems can also be used anywhere. Users always see the same interface, whether at the Bastion PC lab or at an Internet cafe. It is easy to use because users do not need to change the settings again and again. Browser-based systems do not use the local disk to store downloaded files. Maintenance of a browser-based system is easier than a desktop program. However, some external devices are necessary for a browser-based system. To use the video communication service, a user has to prepare a computer, the Internet connection and a web-cam. If a Deaf user

does not have these devices, he is unable to use a browser-based system to do sign language communication. Mobile applications do not need external devices. Many Deaf people have their own mobile phones already. They would be able to use such a sign language communication service with their mobile experience. A disadvantage of a mobile application is OS dependency. A user might not be able to share a mobile application with a friend if the friend is using another brand. The video quality of mobile applications is variable, depending on the specific phone model. A Deaf user might be confused if the video from him is clear while the video to him is of poor quality. Although both browser-based systems and mobile applications have some disadvantages, their strengths are attractive and they are easy to use for our target group of Deaf users.

The second question is about the prototype design and implementation. We need to build prototypes with the necessary features for sign language communication. Two browser-based technologies, Flash and HTML5, are considered in our browser-based implementations. The good point of Flash technology is that it provides synchronous video streaming. A Deaf user could ‘chat’ to a friend using Flash, just as if they are face to face. The weak point of Flash is that the file format used in streaming is limited. Video quality can only be adjusted in a narrow range. QoS of synchronous video streaming is closely linked to network speed [15]. Delay is a possible problem of a Flash video streaming system. HTML5 is also easy to blend with other browser-based technologies. The layout of a HTML5 page is able to be fully controlled by the designer. It provides on-line video playing. Different browsers support HTML5 with different levels. The effect of UI vary for this reason. Some HTML5 features might be disabled by a given browser. Two mobile technologies — J2ME and Android, are chosen for our mobile implementations. J2ME can run on the Symbian OS. Symbian has been widely used from high-end mobile models to low-end mobile models in South Africa. Deaf people definitely have some experience with Symbian OS. That experience can help them learn to use the J2ME prototype. Android runs on the Android OS. The usage of Android phone is increasing rapidly. There are several brands producing Android phone, the development on an Android OS is very easy, and the standard UI looks nice. Android also supports a touch panel, so a Deaf user does not need to press buttons again and again during an action, and can concentrate on signing.

The third question is related to experimentation and evaluation. The prototypes need to be tested and data needs to be collected. We use several kinds of tests

to gather data. Internet behaviour data is recorded, and technology background is collected. User tests are arranged to be done in a real life environment while performance is examined in an in-lab experiment [6]. All these details are presented in section 3.3.

3.2 Research methods

Three kinds of research methods are used in this project. In order to answer the first research question, the author uses qualitative methods to clarify user requirements. Software engineering methods are used to answer the second research question. Qualitative and quantitative methods are used together in experimentation for the third research question. These methods are combined to answer the main research question.

Section 3.2.1 describes quantitative research methods concerning objective data and data collection. Section 3.2.2 discusses qualitative research methods. Section 3.2.3 addresses software engineering methods for system design and evaluation. Section 3.2.4 details the integration of these research methods.



3.2.1 Quantitative research method

Quantitative research refers to those quantitative methods used to collect and analyze data. Quantitative research always focuses on quantity, and the related data appears in numbers. Quantitative research methods are often used in the natural sciences such as physics and chemistry, for such natural phenomena are easy to explain by using formulas. Quantitative research is the basic research method in these subjects. In other words, if the research aims of a subject can be explained with a quantitative description, a quantitative research method should be considered in the research. Quantitative research is generally made for scientific methods [17], which usually have four phases:

- Collection of experimental data.
- Manipulation of variables.
- Analysis of data.

- Evaluation of results.

There are two main quantitative research methods: statistics [81] and linear programming [21]. We will use statistics to analyze our test results. We collect quantitative data by performance testing. The server of our prototypes runs continually for forty eight hours. The performance data is recorded and analyzed using statistical methods. Internet usage data is also logged at the Bastion PC lab. The logs are handled using statistical methods, too. More details are given in Section 3.3 and Chapter 5.

3.2.2 Qualitative research method

Qualitative research methods [70] are used in many research areas to gather an in-depth understanding about human behaviour and the reasons that govern this behaviour [69]. These research methods include content analysis and participant observation. The main data gathering tools in qualitative research include interviews, discussions, user observations and surveys. An interview helps the researcher to access a subjects' experiences, feelings and social worlds [27]. Discussion is used to incorporate various views and draw a conclusion, helping researchers choose a relatively objective direction or architecture for the research. User observation gives researchers occasion to review research results in the subjects' social environments [27]. A survey is usually used in the design phase and evaluation phases, to help researchers to understand the requirements of participants clearly. All these data collection methods explore comprehension of the boundaries in the researched phenomenon. We gather qualitative data in several ways: surveys, questionnaires and observation. A survey is used to collect Deaf user background data. User observation is done during the user tests, along with a questionnaire. More details are discussed in Section 3.3 and Chapter 5.

3.2.3 Software engineering method

Software engineering methods are used to design and implement software prototypes. We employ a spiral model [5] in the whole software life cycle. The research starts from planning and through development, evaluation and analysis, spirals up and re-enters the planning stage. Multiple prototypes are developed for the

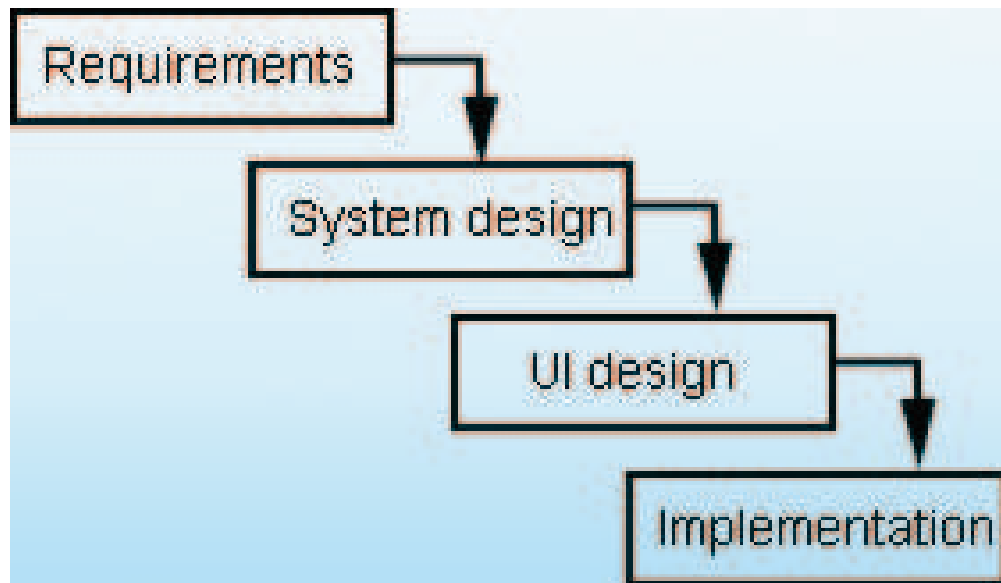


FIGURE 3.1: A software engineering model: A waterfall model is used in the development phase.

evaluation and the analysis. A simple prototype is built and evaluated quickly at first, and then the prototype is intensified based on the analysis result. This spiral model is detailed in section 3.3. From the planning to the analysis, a traditional waterfall model [40] is used. Figure 3.1 shows the waterfall model in this research. Each phase of the waterfall transforms an outcome of the previous step into the income of this step, and produces a new outcome as output finally. The results of this method are the prototypes described in Chapter 4. The experimental design uses these prototypes to gather user feedback and performance data.

3.2.4 Method integration

The technical system development method for this thesis is a prototyping approach. It is a vector triangle with three axes: qualitative research methods, quantitative research methods and software engineering methods. This approach is shown in Figure 3.2. As mentioned in section 3.2.3, this research starts from planning. A survey is done in this phase to gather Deaf user background data. This data helps us to analyze Deaf users' requirements for sign language communication. In the development phase, software engineering methods are the main research methods. Qualitative data gathered in the planning phase is used in the system design step and the UI design step in the waterfall model (see Figure 3.1).

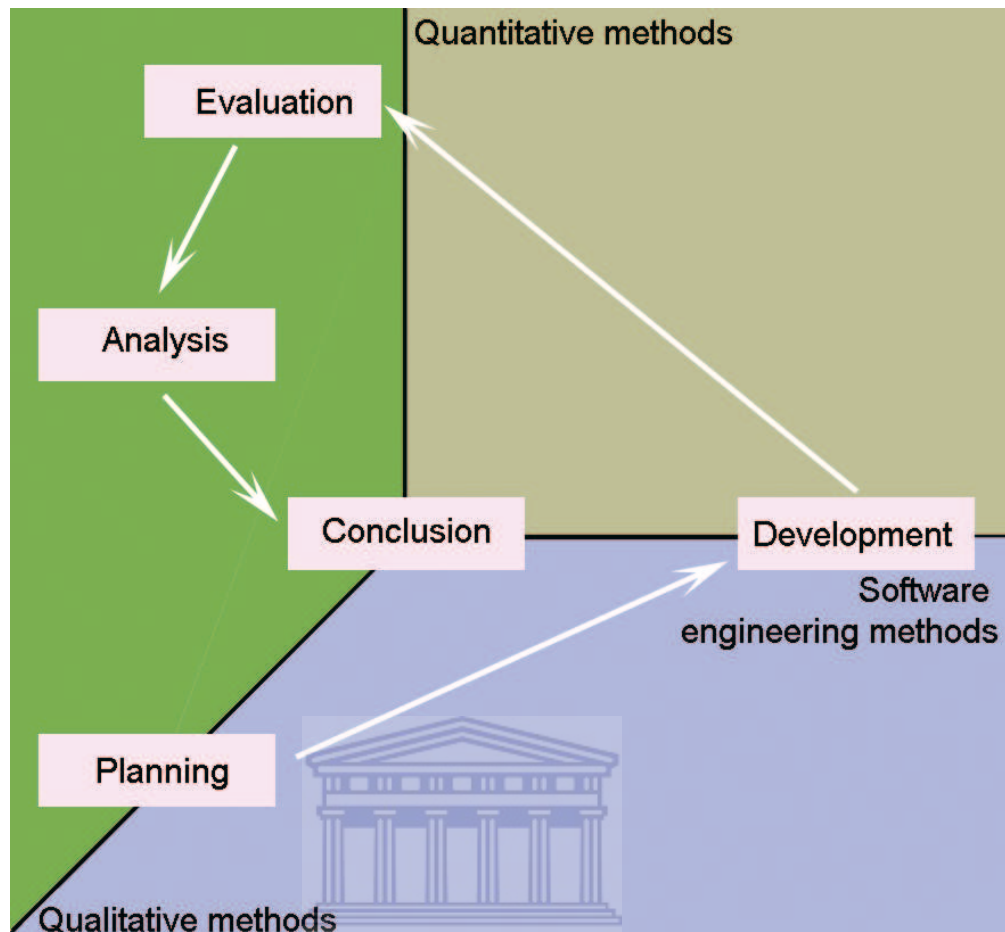


FIGURE 3.2: Methods integration: Qualitative methods are used in the planning phase, then software engineering methods are applied in the development phase. In the evaluation and analysis phases, both qualitative and quantitative methods are used to handle data.

In the evaluation and analysis phase, the data gathered in the experimentation are processed by both quantitative and qualitative methods. Finally, a conclusion is drawn, and the research re-enters the next planning phase (see Figure 3.3).

3.3 Experimental design

This section presents more details about how to use the research methods. Figure 3.3 shows the spiral research model used. This project follows on from a BANG asynchronous Deaf communication project conducted in 2008 [48]. In the related project, a desktop application was built to provide an asynchronous video service. In our planning phase, a survey was done to gather Deaf users' background information to clarify the user requirements. The user requirements show us the

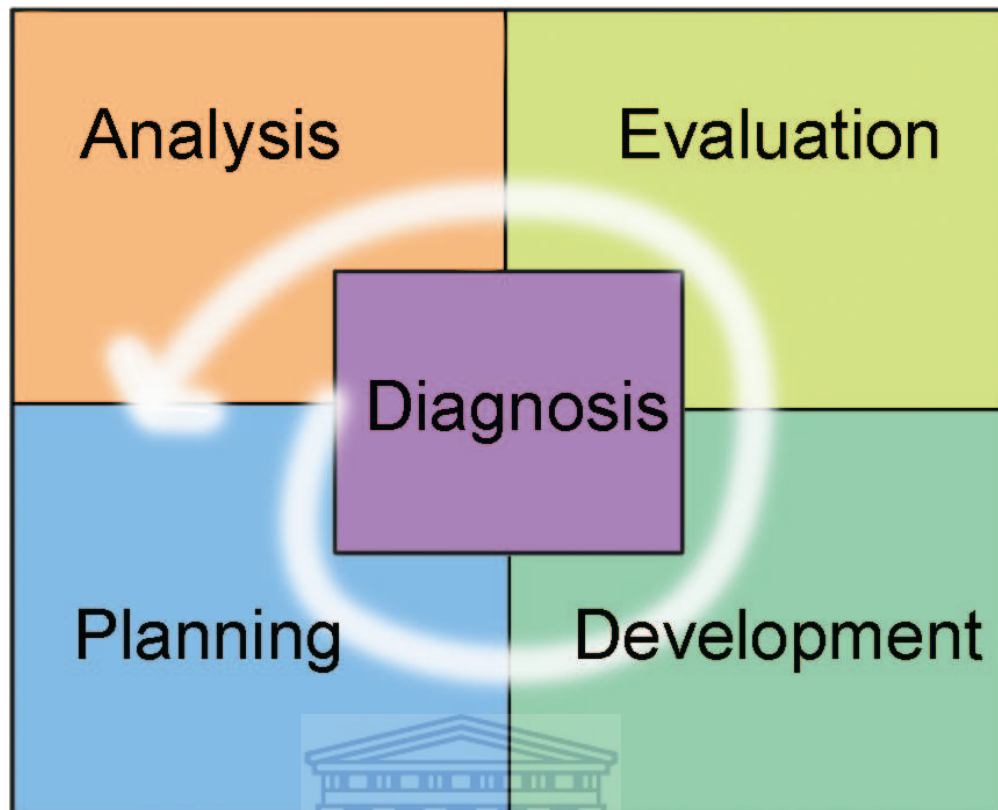


FIGURE 3.3: Spiral research model

research direction. Then we go to the next phase: development. Four prototypes are designed and built with the data collected in the planning phase. When the implementation is finished, the evaluation is started. User tests and performance tests are done to gather qualitative data and quantitative data. Both qualitative and quantitative research methods are used in this phase. Finally, we analyze the collected data and draw a conclusion. The result of analysis is an input of the next research cycle, and the next planning phase starts.

An important step in this process is to collect the requirements of Deaf users for video communication. Since Deaf culture is different from mainstream culture, and there are many difference between SASL and English, we do not want to ‘imagine’ what kind of software interface Deaf people want to use in their daily life. We need to clarify their requirements in a soft and natural way, from our target community directly.

Section 3.3.1 describes the role of the thesis author in the research group. Section 3.3.2 introduces the target community of this project. Section 3.3.3 introduces the phases of experimentation. Section 3.3.4 addresses the collection of Internet

usage and Deaf user background data. Section 3.3.5 discusses how we conduct user tests. Section 3.3.6 details how we conduct performance tests. Section 3.3.7 discusses the ethical issues for this effort.

3.3.1 Disclosure of the role of the researcher as a member of a research group

BANG aims to design, develop and evaluate network-based communications to bridge the digital divide in South Africa [74]. The author is a member of the BANG group. Only the author is responsible for the work reported in this thesis.

BANG has been associated with the DCCT group since 2001 [31]. BANG aims to expand Deaf telephony technology and to build network applications for Deaf people. Besides the related work mentioned in section 2.1.4, there are several other BANG projects for DCCT. BANG has built a PC lab at the Bastion for the Deaf. This lab provides Deaf people with Internet for free. The PC lab is the most successful BANG project, for it allows DCCT members to learn computer and Internet skills. BANG supplies PCs, network devices and software. As a member of BANG, the researcher has visited DCCT each week. If the machines in the PC lab have any hardware or software problems, BANG members fix them. A computer literacy project was started this year. A DCCT website has been built. Furthermore, the researcher and the BANG group workshop with DCCT staff every three months, to evaluate work-in-progress research projects and to fine-tune the projects. BANG and DCCT have a very good relationship, which is helpful in running this project with DCCT members.

3.3.2 Target community

DCCT has more than one thousand Deaf members. Most of them have limited education and are under employed. The DCCT office is located in a building called the Bastion of the Deaf in Newlands, Cape Town. The BANG computer lab is at the Bastion. The computer lab is open to all DCCT members for free, from Monday to Friday. Six computers in the lab are connected to the Internet. DCCT members can use this lab to do job hunting and search information. Hundreds of unemployed Deaf people use this lab each year. These users of Bastion computer

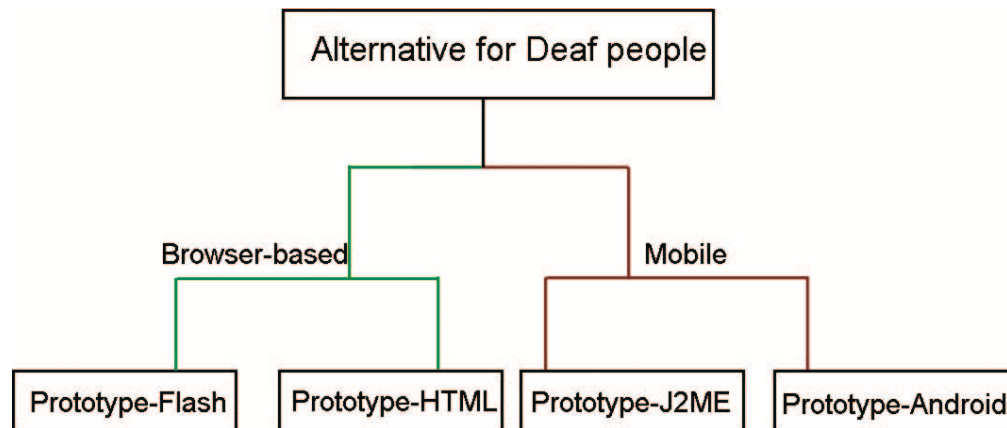


FIGURE 3.4: Prototypes and technologies: Four prototypes are presented in this research. They are compared to answer the third research question.

lab may not have PCs and the Internet connection in their home. The computer lab lets them be in touch with network technology. It is also a good place to find Deaf subjects who are interested in network technology. Many BANG projects have been tested in the computer lab, including the related work mentioned in section 2.1.4.

DCCT has a general members meeting at the Bastion on the third Sunday of every month. Many activities are held on this day, such as drama, dance and soccer. Over one hundred Deaf people appear on every third Sunday. It is rare that so many Deaf people show up together. Normally, less than twenty members visit the Bastion on weekdays. The third Sunday is also the best opportunity to meet the employed DCCT members, as they cannot visit the Bastion during weekdays.

3.3.3 Experimentation arrangement

Several prototypes are designed and implemented to answer the research questions, including a synchronous browser-based prototype, an asynchronous browser-based prototype and two asynchronous mobile prototypes on different mobile OSs. We design and test the four prototypes to learn how to choose the most promising one for sign language communication. The names of the prototypes are: prototype-Flash, prototype-HTML5, prototype-J2ME and prototype-Android. Figure 3.4 shows the four prototypes. In this section, the problems of design and implementation are discussed.

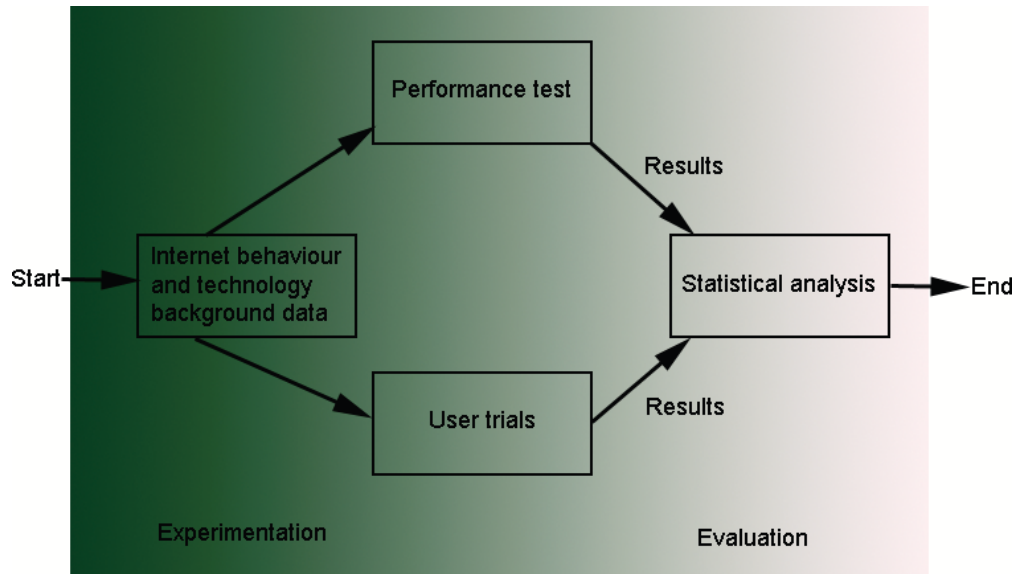


FIGURE 3.5: Experimental arrangement

To answer the third research question, experimentation is done. Both qualitative and quantitative data are collected and analyzed. The experimentation has three parts. First is the Internet behaviour and technology background data collection. This helps us specify user requirements. These qualitative results help indicate the direction of the next phase of the research. The second part is the performance test [79], and the third one is the user tests [16, 50]. Performance tests collect system performance data. User tests collect Deaf users' feedback. The results of data triangulation is used for each iterative process of the spiral model. In this way, the architecture mentioned in Figure 3.3 is implemented. The arrangement of the experimentation and evaluation is shown in Figure 3.5.

3.3.4 User behaviour and background data

To understand Deaf culture and clarify user requirements, some user background data is collected. There are two questions we want to know from the background data collection. The first question is: what do Deaf people know about communication technology? The second question is: what do Deaf people usually do on the Internet? The answers to these two questions help us analyze what kind of video communication system Deaf people like. To answer the first question, a survey with a questionnaire was conducted at the Bastion, to collect background data (see Appendix A). The survey asked Deaf volunteers to fill in a questionnaire

about computer, mobile and Internet usage. The results of the questionnaire are in Appendix A and summarized in section 5.1.1.

To answer the second question, we analyzed Internet usage data recorded at the Bastion. DCCT staff record what Deaf users do when they go to the PC lab, and we have continually aggregated the data from 2007. This data is analyzed to see what Deaf users do when they connect to the Internet. The results of Internet usage are in section 5.1.2.

3.3.5 User tests

There are two aims for the user tests: prototype evaluation and user observation [68]. Figure 3.4 shows that there are four prototypes prepared for this research. Two are browser-based and the other two are mobile-based. In the user tests, the researcher asks Deaf people to use these prototypes and give their feedback.

We conducted two rounds of user tests. A pilot trial examined the prototypes with a small number of participants in a laboratory environment [6]. Four Deaf users used the two browser-based prototypes, while another four Deaf subjects used the two mobile prototypes. These participants have experience with computer software and mobile software. A questionnaire was prepared and the experienced volunteers answer it. The results are in Appendix B, and analyzed in section 5.2.2.

A full trial is conducted to test all the prototypes: prototype-Flash, prototype-HTML5, prototype-J2ME and prototype-Android, with a larger number of Deaf people. Almost forty DCCT members used the prototypes and gave us feedback. These subjects were selected randomly. A questionnaire (see Appendix C.1) is answered by the participants. The results of the full trial are detailed in section 5.2.3.

We organized the user tests at the Bastion. A SASL interpreter is invited to make sure the subjects and the researcher can understand each other. Questionnaires and prototype guides (see Appendix D, E) are prepared. During the user tests, the researcher does user observation at the same time. By this observation we can learn about our target audience better, such as what functions confuse them [35].

3.3.6 Performance testing

Since experimentation and evaluation are important parts in this project, we need to find a proper method to analyze performance. There are many measurement methods for video quality. The comparison of an original video with a processed video is usually used to measure the loss in encoding and decoding phases [78] of local video files. On the other hand, more and more video transmission and streaming research is related to the Internet. The performance tests of such projects focus on the workload of the network [15, 71] and user behaviour analysis [14]. From such projects we notice that the measurement for Internet-related video has different points from the measurement of local video quality.

The QoS of Internet-related video communication is usually determined by bandwidth and frame rate [77], while the memory and Central Processing Unit (CPU) usage of the server should also be considered. In this research, we propose to monitor the network data and the CPU usage to evaluate the performance of the prototypes from the server side. Since the clients' side of the prototypes are running on different platforms, it is hard to measure the performance of the prototypes independent of devices and OSs. Therefore, we chose to do the performance testing on the server side.

We decided to run all the prototypes continually and individually for forty eight hours. The CPU and memory data, and the network usage data on the server are monitored and logged automatically using Windows performance monitor [84]. Then a tool called Performance Analysis of Log (PAL) is used to analyze these data. Since the four prototypes of this research use a standard frame rate (15 fps), the frame rate is not a parameter included in the performance test. Furthermore, we test this scenario when several groups of users request the same service, how many resources CPU and memory would be consumed on the server. The results of the performance testing are described in Section 5.3.

3.3.7 Ethical considerations

Since Deaf people are the research target, we have to consider whether our research is likely to cause them physical or emotional harm, such as violating their rights to privacy by posing any sensitive questions, and recording their personal communication data for evaluation. We have the responsibility of explaining our request

and the rights of a subjects to our Deaf volunteers clearly. During our user tests, a SASL interpreter is used to handle Deaf users' feelings and their understanding of the research. Before a Deaf subject starts to use our prototypes and answer the questionnaire, the sign language interpreter helps the researcher to explain what our research project is about and what we want to ask him/her to do. Every subject joined in our test willingly. Since many subjects do not know English, we collected their signed agreement instead of asking them to sign a consent form.

3.4 Summary

There are many communication projects for Deaf users, and some communication products for hearing people can also be used by Deaf people. However, the gaps in the related work expose problems for local Deaf users. For example, MobileASL is a system which provides a portable sign language communication service on mobile via GPRS in America. Deaf people in South Africa are unable to use MobileASL for two reasons. Firstly, South African Deaf people are unable to obtain the software for there is no commercial product based on MobileASL. Secondly, the user feedback gathered in its research is not strong enough to show us if this system is the best one for South African Deaf communication. There were only fewer than twenty American Deaf subjects who tested MobileASL, and the project was not tested in South Africa.

Another example regards using communication products for hearing people. Many Deaf people are poorly educated. They do not know how to use software, and cannot read manuals and help documents. Another reason is that most communication products for hearing people prioritize voice in video transmission. Besides, since few Deaf people have a PC, Deaf people are rarely on-line for synchronous chatting. In addition, mobile video quality is poor, so it requires low level codec work like MobileASL to accommodate sign language. We would like to adapt 'hearing' video tools for Deaf users because they do not meet Deaf users' needs very well.

This chapter discusses the methodology for the thesis. The main research question is: how can we choose a promising video communication alternative to provide acceptable sign language communication service, and make sure it is easy to use for Deaf people's daily conversations? This research question can be split into three

question: what development technologies should be considered as alternatives, how to build up prototypes using the alternatives, and how to evaluate prototypes to find a suitable alternative for sign language communication?

Three research methods are used to answer these questions. Qualitative methods are used to answer the first question. A survey was conducted to collect Deaf user background for clarifying user requirements. Software engineering methods are used to answer the second question. A waterfall model is applied in development. Qualitative and quantitative methods are used to answer the third question. User tests were done with questionnaires. Subjects were observed during user tests. Performance testing was also conducted. The results of user tests and performance testing are analyzed by statistics.

Data collection has three components: user behaviour and background data collection, user feedback and performance testing. Deaf user background data was gathered with a survey. Internet usage was recorded and aggregated. We conducted two rounds of user tests. A pilot trial examined the prototypes with a small number of participants, while a full trial tested the prototypes with a larger number of Deaf people. An interpreter was used for all communication with the Deaf participants. All the prototypes were run continually and individually for forty eight hours. The CPU and memory data, and the network usage data on the server were monitored and logged. The next chapter describes the design details of the four prototypes.

Chapter 4

Prototype Design

The members of our target Deaf community have two main characteristics. The first is that they use SASL as their primary language to communicate with each other. The second is that they have limited computer literacy. Only a few communication products are used by them, such as Facebook. The first issue indicates that they have some specific requirements for a user interface. The second characteristic determines that they are unable to understand the technical details of a video communication system. These challenges should be factored into our prototypes. To address the challenges, we consider specifying the user requirements in two parts.

The first part is user requirements for features. All the four prototypes described a video communication service via the Internet. The basic steps of our prototypes are simple: login, search for a friend for a sign language conversation, do video communication, and logout. For this work flow, requirements for features include the following:

- Each user has a unique identity. The identity and related user information are stored on the server. User information can be changed by the owner.
- The user is able to choose the person he wants to chat with.
- The user is able to terminate the conversation he is having.
- A proper error message is shown when an incorrect user operation is made.
- A proper information message is shown when the server has a problem.

- Menus and buttons have proper labels. The contents of labels are associated with the actions.
- Audio data can be removed from video files, for Deaf people do not need a voice service during sign language communication.

The second part of the user requirements is for UI design. As mentioned in section 3.3.4, we gathered the user requirements through a survey. According to the results, the following requirements are listed:

- The UI of the system should be simple and clear.
- The interactive actions between the server side and the client side should be undemanding.
- No complex settings are shown on the client side. The work flow of the video communication looks uncomplicated and user-friendly. The complicated processes are handled on the server side.

This chapter answers the second research question: ‘how to build up prototypes using the alternatives’. This chapter describes how the four prototypes are designed and implemented in terms of system design and UI design. Section 4.1 introduces the design details of prototype-Flash. Section 4.2 discusses the specification of prototype-HTML5. Section 4.3 addresses the implementation of prototype-J2ME. Section 4.4 links the design details of prototype-Android. These prototypes are used to conduct the user and performance tests described in Chapter 5.

4.1 Prototype-Flash

Prototype-Flash provides a synchronous video streaming service inside a web browser on a PC. Prototype-Flash is based on Adobe Flash technology. The components of prototype-Flash are shown in Figure 4.1. The prototype consists of both client and server. The client Flash file is embedded in a web page.

Section 4.1.1 presents the system design of prototype-Flash. Section 4.1.2 discusses the user interface of prototype-Flash.

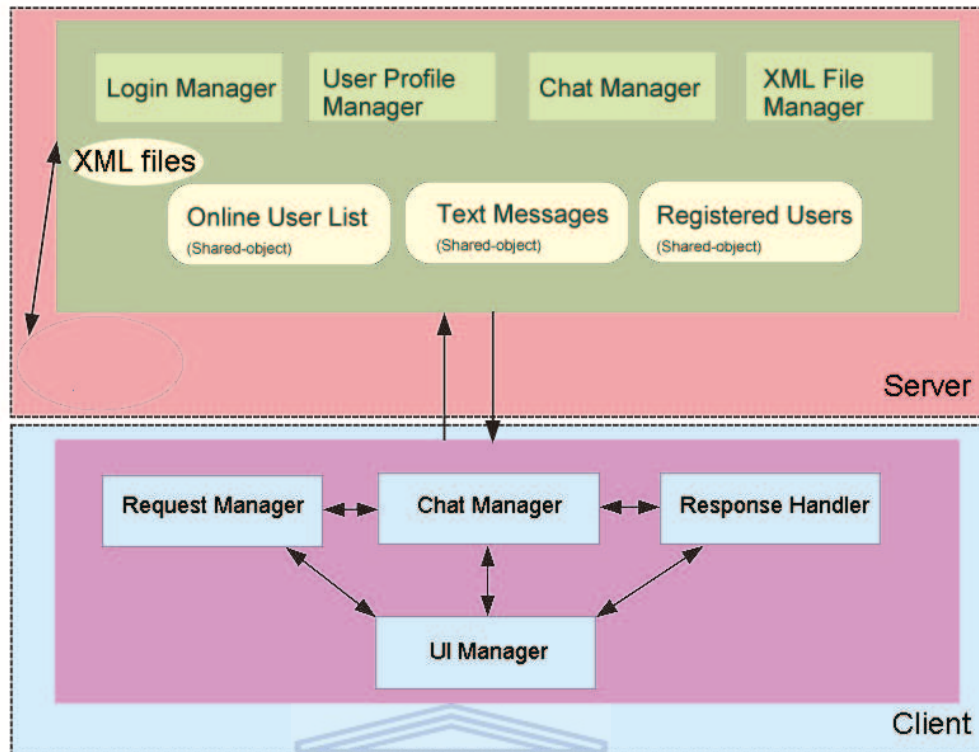


FIGURE 4.1: Prototype-Flash architecture

4.1.1 System design

Prototype-Flash has a client/server architecture. The client part is built with Adobe Flex. The components of the client part are shown in the bottom half of Figure 4.1:

- The UI manager model defines the user interface components and interacts with a user. There are two parts inside the UI manager. Part one is used to implement the functions related to the user interface. For example, when a user enters his/her user name, the UI should be switched from login frame to logged-in frame after authentication. The function to switch frames belongs to part one, written in ActionScript 3 [7]. Part two uses MXML [72] to describe the controls in Extensible Markup Language (XML) style. For example, in the login frame there is a button at the bottom, labelled 'login'. The definition of the UI elements belongs to part two.
- The chat manager is a module to manage the work flow of sign language communication. The work flow for starting two-way synchronous video communication in prototype-Flash is shown in Figure 4.2. When step four is

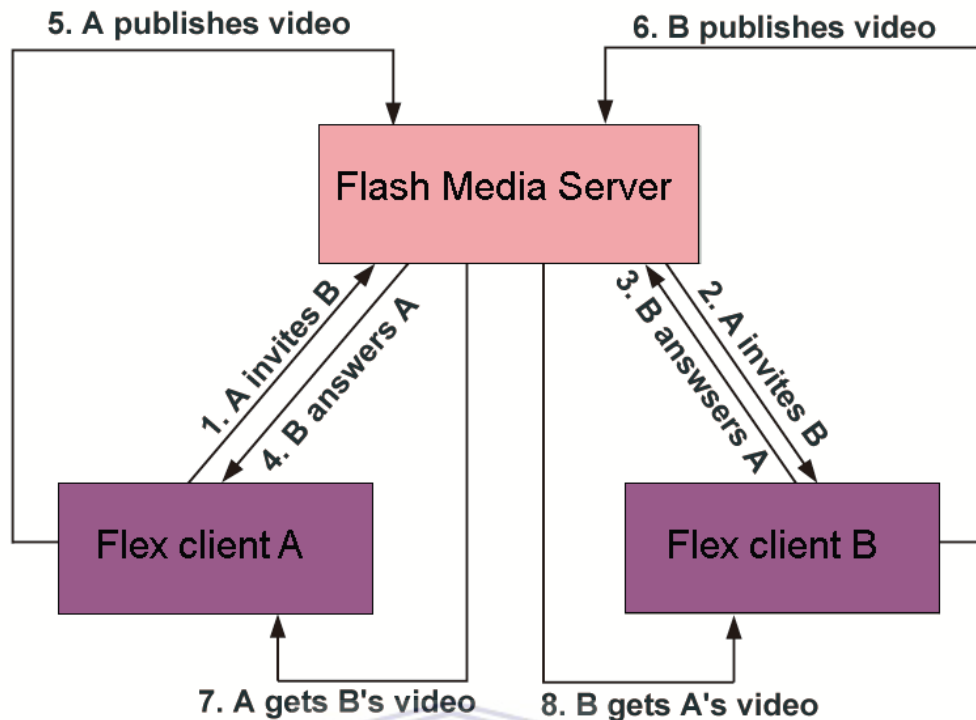


FIGURE 4.2: Synchronous video work flow

finished, an Adobe Flash setting dialogue is shown on an individual client. The video publishing starts after the user agrees to allow Flash access to the web-cam. When either A or B wants to end the video chat, he/she sends a message to the server, and the two-way communication is shut down on both sides.

- Figure 4.2 shows the work flow of a two-way synchronous video communication in prototype-Flash. A sends a command of invitation to the Flash media server to invite B. The server forwards this invitation command to B. B returns an command to the server to answer A's invitation. The server forwards this answer command to A. A starts to publish his live video to the server when the conversation begins. In the same time B also starts to publish live video. A receives B's video from the server since B's publish begin. On the other side, B also receives A's video since A's publish begin.
- The request manager sends requests to the server. The Flex client connects to the Flash server using the NetConnection [65] class, with RTMP. Steps 1, 3, 5, 6 in Figure 4.2 go through the request manager.

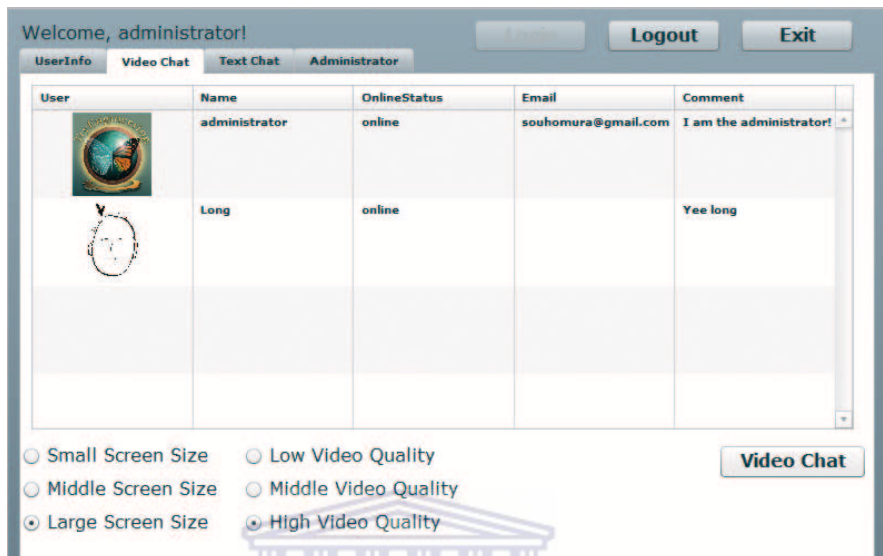
- The response Handler gets server responses and calls the specific functions in the UI manager, to reflect the responses in the user interface. Steps 2, 4, 7 and 8 in Figure 4.2 go through the response handler.

An open-sourced Flash media server frame called Red5 [3] is used in the server of prototype-Flash. Our server is built with Java running inside Red5. The components of the server are shown in the top half of Figure 4.1:

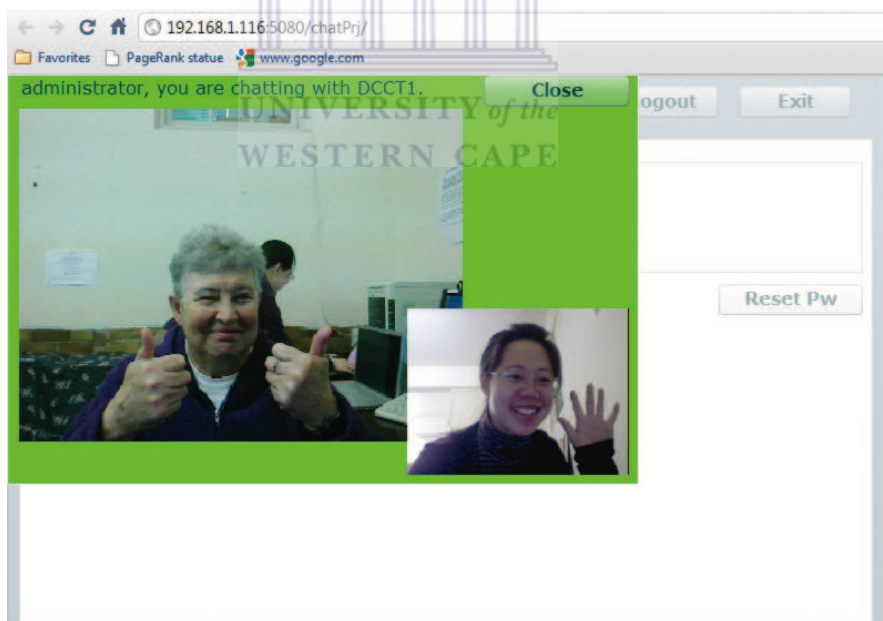
- The user profile is stored in XML files. These XML files save personal information, including name, password, email address and comment.
- Shared objects store the common and sharable information. These objects could be presented on every client. The on-line user list and text messages are stored in shared objects. The list of registered users is also stored in shared objects.
- The login manager is used to control login and logout actions.
- The user profile manager handles the modification of personal information. The administrator has an additional right to add and remove users, and can modify the personal information of another user. Other users can only modify their own information.
- The chat manager on the server side catches the chat start and chat end messages from client A and relays the messages to client B. Chat manager is also the transfer station for video streaming. The published video of client A is saved by chat manager temporarily, and client B asks for this video from chat manager.
- The XML file manager is used to write and read XML files. The files are read into data structures when the server starts. The information of a user is updated when the user logs in. The information is converted and stored when the user logs out.

4.1.2 User interface

Two screen-shots of prototype-Flash are shown in Figure 4.3. Figure 4.3(a) is the on-line user list view. In the ‘Video Chat’ tab, the on-line users are listed in a



(a) View of on-line user list for chat



(b) View of synchronous video chat

FIGURE 4.3: The user interface of prototype-Flash

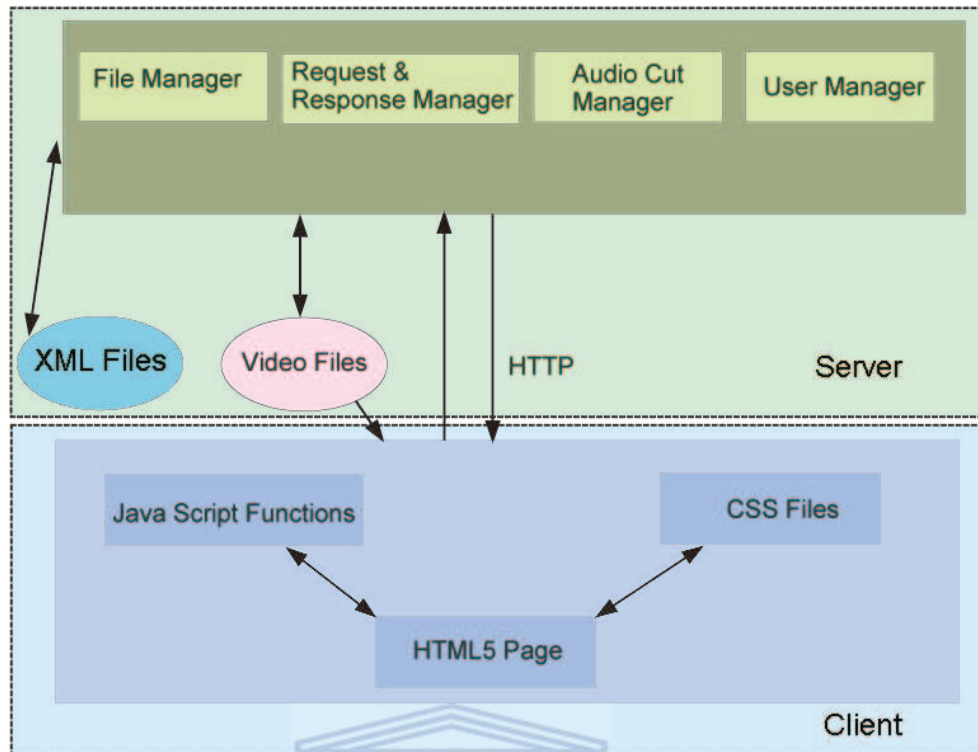


FIGURE 4.4: Prototype-HTML5 architecture

data grid controller. User profile information is also shown here, such as photo, user name, status and email. Six radio buttons allow video quality settings. They are separated into two groups. The first button group defines the screen size to stream the live video. The second button group determines the video quality used in video streaming. Figure 4.3(b) is the actual communication view. The top left view is of the incoming video. The size of this screen is determined by the first radio button group in Figure 4.3(a). The bottom right view is the sending video. The quality of this video is defined by the second radio button group in Figure 4.3(a).

4.2 Prototype-HTML5

Prototype-HTML5 gives an asynchronous video interaction service inside a web browser on a PC. Prototype-HTML5 is based on HTML5, Java Script and Java. There is a Java server and a web page in prototype-HTML5. The components of prototype-HTML5 are shown in Figure 4.4.

Section 4.2.1 addresses the system design of prototype-HTML5. Section 4.2.2 describes the user interface of prototype-HTML5.

4.2.1 System design

Prototype-HTML5 also has client/server architecture. The client part of prototype-HTML5 is a web page based on HTML5, Cascading Style Sheets (CSS) and Java Script. The client components are shown in the bottom half of Figure 4.4:

- HTML5 page is the frame the user watches when prototype-HTML5 starts.
- CSS is used to describe the layout style of a HTML5 page. All the visual effects are indicated in CSS files, including the background colours, font sizes and positions of elements.
- Java Script functions respond to user events, and communicate between the HTML5 page and the server. Prototype-HTML5 provides an asynchronous video communication service between two clients by using Java Script functions. The work flow of the asynchronous communication is shown in Figure 4.5. The asynchronous video communication is similar to email communication. The client and the server are connected via HTTP. The HTTP connection is created by using a Document Object Model (DOM) API called XMLHttpRequest [75]. This API is used in Java Script functions. XMLHttpRequest can create a connection with a web server, sending the HTTP request directly to the server and handling the response from the server.
- The user can delete all video messages from the client side. In this case the corresponding video files are deleted from the server.
- There is no feature for video capture in prototype-HTML5. A user of prototype-HTML5 can use the default video capture software installed in his/her PC to record the video messages.
- Local video files can be attached to messages using the HTML form submit feature. The file information and the video stream are combined and sent together. The server then handles this data and saves the video stream.

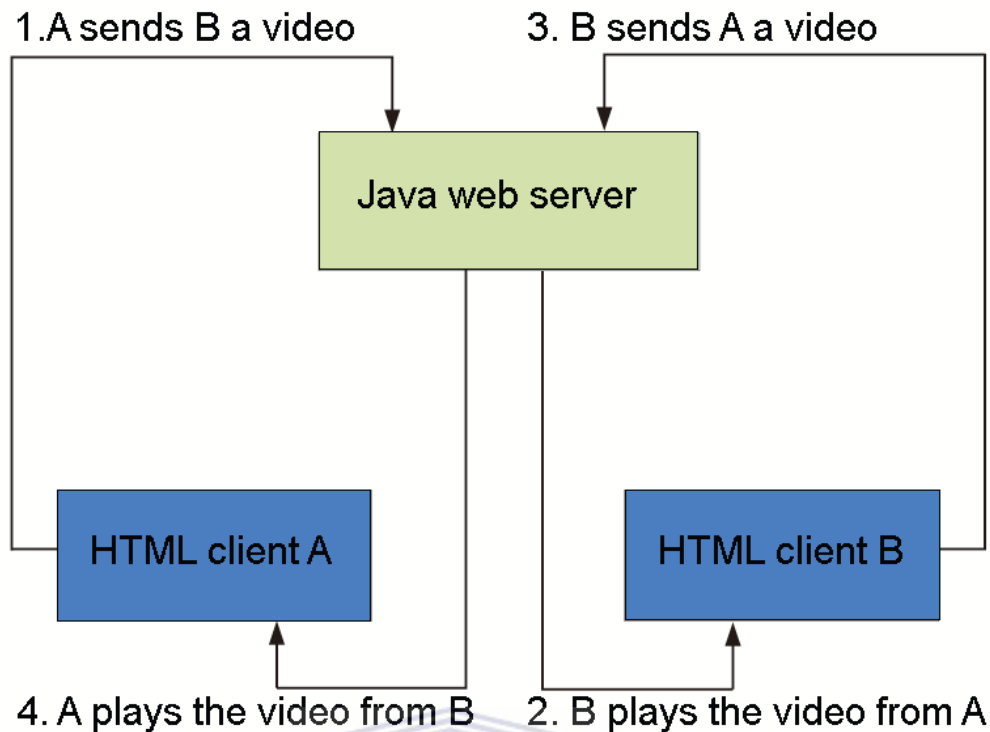


FIGURE 4.5: Asynchronous video in prototype-HTML5

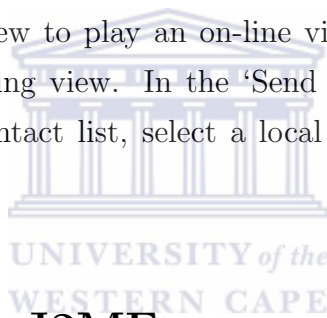
The Java server of prototype-HTML5 runs inside an open source web server framework called GlassFish. The components of the server side are shown in the top half of Figure 4.4:

- User profiles are stored in XML files. Personal information and contact lists are saved in a user profile.
- Video files are stored on the physical disk of the server. These video files can be deleted by the receiver from the client side.
- The file manager is used to control the files stored on the server. Three types of files are managed by the file manager: videos, user profiles, and a property file for server settings.
- The request and response manager handles the HTTP requests and responses, and is the main component of the server. All the other modules are used by the request and response manager directly or indirectly. Steps 1 and 3 in Figure 4.5 send the HTTP requests to the request and response manager, while steps 2 and 4 get HTTP responses from the request and response manager.

- The audio cut manager is used to remove the audio data from a video file. Open-source FFmpeg (www.ffmpeg.org) is used to implement the audio cut feature.
- The user manager maintains an on-line user list to record the status of the on-line users, and manages the profile data of all users.

4.2.2 User interface

Two screen-shots of prototype-HTML5 are shown in Figure 4.6. The layout style of prototype-HTML5 is similar to the layout style of a standard on-line mailbox. A tab control is used to switch the views from one to another. This tab control is similar to prototype-Flash. Figure 4.6(a) is the video playing view. In the ‘Video inbox’ tab, all the video messages sent to this user are listed. A player is presented on the bottom of the view to play an on-line video via its address link. Figure 4.6(b) is the video sending view. In the ‘Send video’ tab, a user can choose a number from his/her contact list, select a local video file, and upload it to the server.



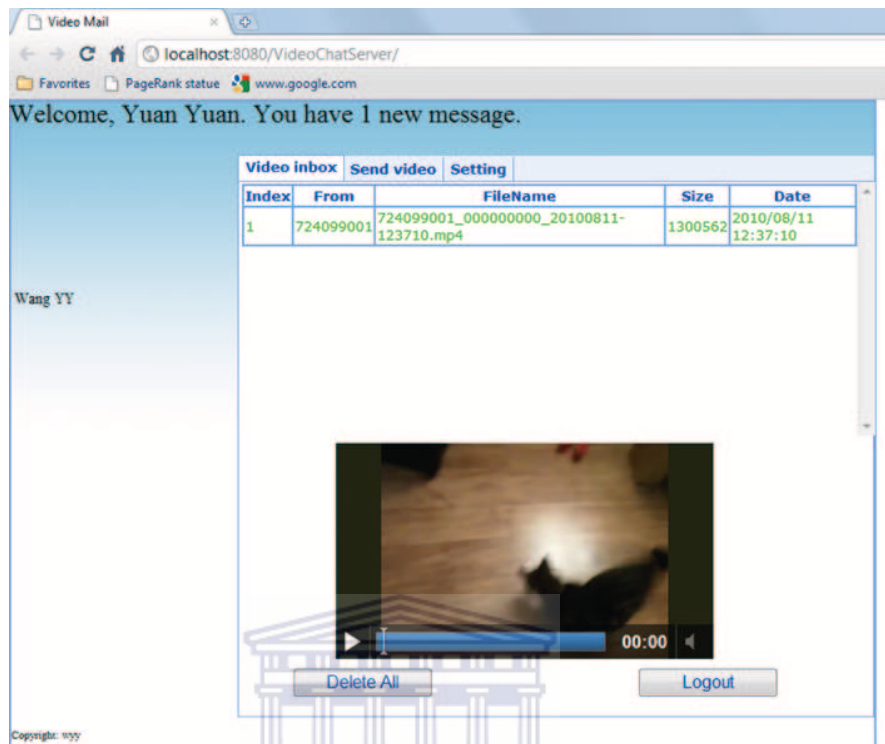
4.3 Prototype-J2ME

Prototype-J2ME provides a portable asynchronous video communication for Symbian OS. This prototype is based on Java and J2ME technology. A user of prototype-J2ME can send a video from a Symbian mobile phone via the Internet, and play video messages on a Symbian mobile phone.

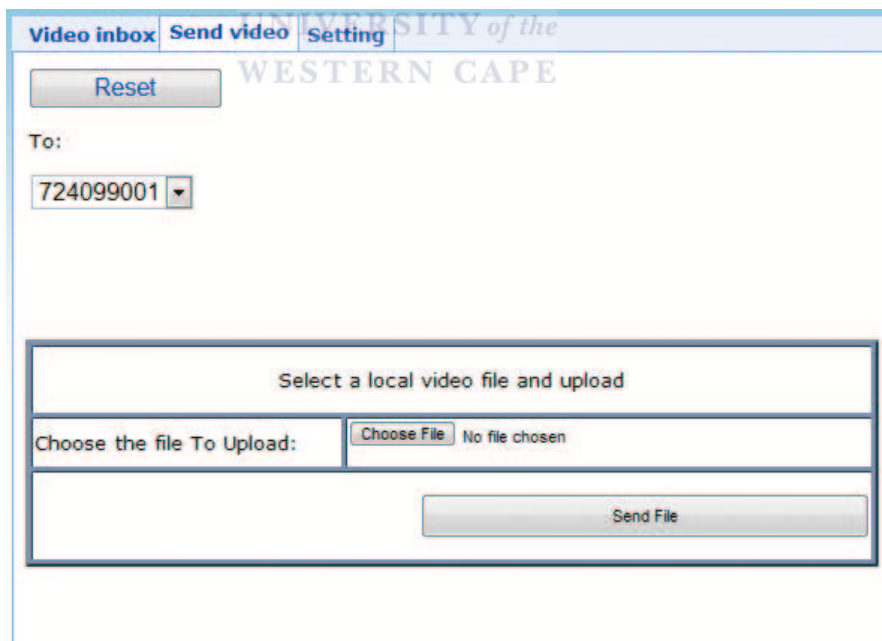
Section 4.3.1 details the system design of prototype-J2ME. Section 4.3.2 describes the user interface of prototype-J2ME.

4.3.1 System design

Prototype-J2ME has a client/server architecture. The client of prototype-J2ME runs on the Symbian OS, while the server of prototype-J2ME is shared with prototype-HTML5. Prototype-J2ME and prototype-HTML5 are quite similar. The clients of these two prototypes play and send video in an asynchronous way.



(a) Playing video view



(b) Sending video view

FIGURE 4.6: The user interface of prototype-HTML5

The connection between client and server is through HTTP. Thus, we can share the server of prototype-HTML5 with prototype-J2ME. Since these two have the same basic features, it is possible to use the same server handling user requirements. It makes source code management easier. Another advantage is that users of these two prototypes could use both PC and mobile to do asynchronous video communication, for user accounts are also shared.

From now on we call the client of prototype-J2ME ‘prototype-J2ME’, and the shared server ‘HTTP-server’. The work flow chart for the system design of prototype-J2ME is shown in Figure 4.7. Prototype-J2ME is built with J2ME. The features on the client side are described here:

- The user needs to register with a phone number before using this system.
- The user must specify the phone number he/she wants to communicate with. There are two ways to do this. The first way is to select the number from an address book. The second way is to input the phone number directly.
- The user downloads a video message from the server to the phone. The video message can be played after downloading.
- The user is able to capture a video message and send the message to the person he/she is communicating with.
- The client connects to the server via HTTP.

Since prototype-HTML5 and prototype-J2ME share the HTTP-server, the difference between these two prototypes is on the client side. Prototype-HTML5 does not provide a video capture service, while prototype-J2ME can capture new videos and send. Prototype-HTML5 users can delete all the video messages from the client side, while prototype-J2ME does not have this function. Besides, the data format of data sending and receiving is different.

4.3.2 User interface

Two screen-shots of prototype-J2ME are shown in Figure 4.8. Figure 4.8(a) is the main menu view. Figure 4.8(b) is the view of video capturing. Figure 4.8(c) is the view of video playing. Prototype-J2ME runs on the Symbian mobile OS

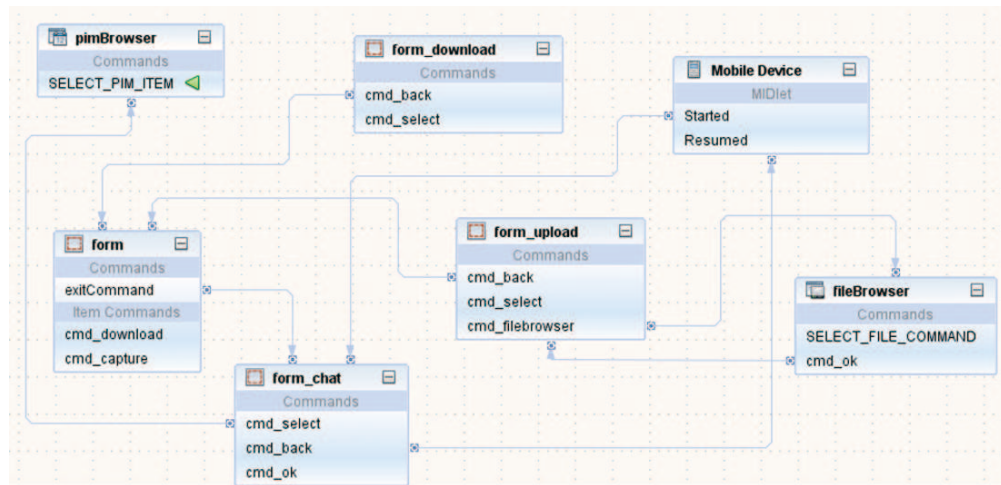


FIGURE 4.7: Work flow chart for prototype-J2ME: This is the design chart of prototype-J2ME. ‘Form’ is the start of the chart. ‘Form_chat’ is the main view of this prototype. ‘Form_download’ is for video download and video play. ‘Form_upload’ is for video capture and video update.

to provide asynchronous video communication. When a user (client A) opens the prototype-J2ME application on the Symbian OS, he/she should choose a user (client B) whom he wants to chat with. Then client A can open the main window of prototype-J2ME. There are two main features in the system: capture video messages and send them out, and download video mails. There is a message shown on the top of the main window to remind client A how many new video mails from client B he/she has got.

4.4 Prototype-Android

Prototype-Android provides a portable asynchronous video communication in Android OS. Prototype-Android is based on Android technology. A user of prototype-Android can check the video messages he/she has received and play the videos, and send video message to other users. Section 4.4.1 describes the system design of prototype-Android. Section 4.4.2 describes the user interface of prototype-Android.



(a) Main menu (b) Video capturing



(c) Video playing

FIGURE 4.8: The user interface of prototype-J2ME

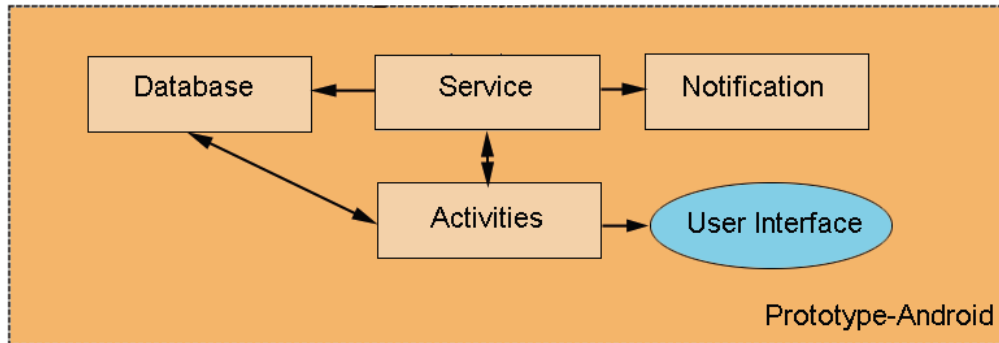


FIGURE 4.9: The architecture of prototype-Android

4.4.1 System design


Prototype-Android also has a client/server architecture. The client side of prototype-Android runs on an Android OS, while the server side runs on a Java web server. Prototype-J2ME and prototype-Android have the same features running on different mobile OSs. The clients of these two prototypes play video and send video in an asynchronous way. The connection between client and server is through HTTP. Thus, we can reuse the same server for prototype-Android, prototype-J2ME and prototype-HTML5. Since these prototypes have the same features, it is possible to use the same server handling user requirements. It makes source code management easier. The prototype-Android has the same advantage that users of these prototypes can use PCs, Symbian phones and Android phones to do asynchronous video communication, for user accounts are also shared. From now on we call the client of prototype-Android. Figure 4.9 shows the model of prototype-Android. The components in prototype-Android are as follows:

- Service is the component that communicates with the server. The service component is a specific item running in the background continually, to connect to the Java server and check video messages after a certain time. The frequency can be changed by a user. The default frequency of connecting to the HTTP-server is once per minute.
- Activity is the frame showing on the screen. User interactions are included in the activity component.
- User interface includes some XML files which use a specific format. All the visual effects are described in these XML files. The UI definition files are used in activity component.

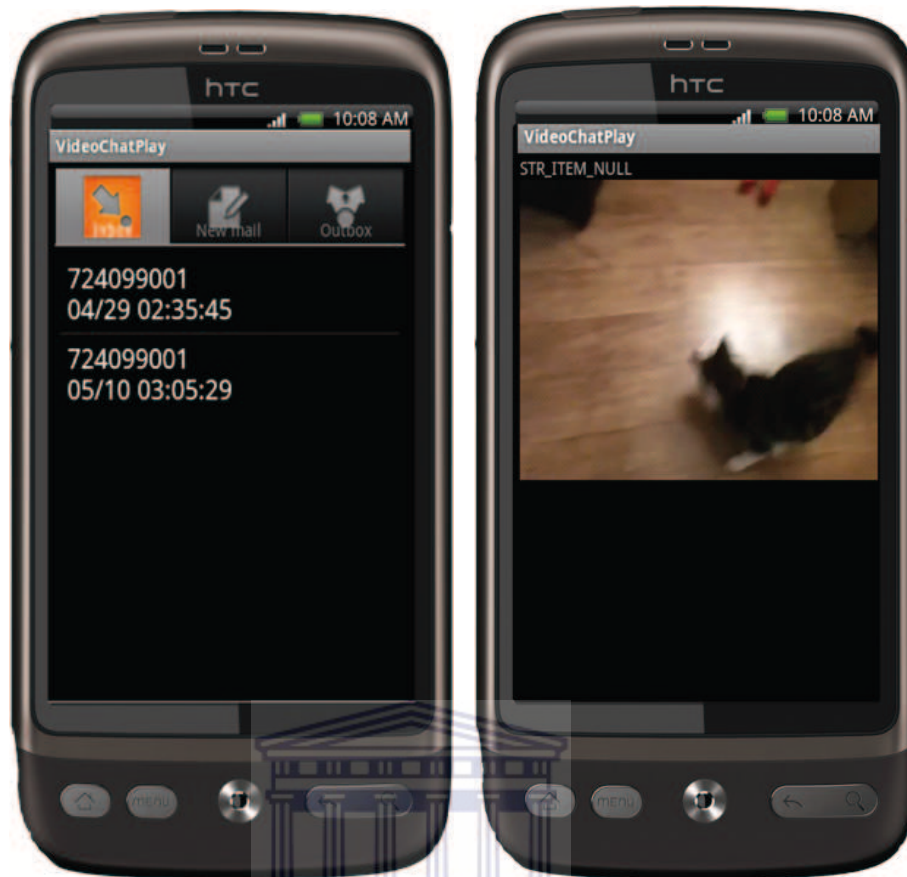
- A database is used to store the details of video messages. The history of video sending is also saved in the database.
- A notification appears when the service has found a new video message for the user.

Since prototype-Android shares the `glshttp-server` with prototype-HTML5 and prototype-J2ME, the difference between prototype-Android and the other two prototypes are on the client side. The user interaction of prototype-Android is similar to email communication, just like prototype-HTML5. All the video messages are listed in prototype-Android, while a prototype-J2ME user can only check the messages one by one. Prototype-HTML5 users can delete all the video messages from the client. Prototype-Android does not have this feature. Prototype-Android and prototype-J2ME use the same format to translate data between the client side and the server side. This format is different from the data format of prototype-HTML5.

4.4.2 User interface



Screen-shots of prototype-Android are shown in Figure 4.10. The user chooses a tab to switch views. Tab control is similar between prototype-Android and the other prototypes. Prototype-Android uses a standard Android application style to design the UI. It should be noted that only activities are presented in Figure 4.10. Figure 4.10(a) is the video message list view. All received video messages are listed in this view. Some information about the messages is also presented. Figure 4.10(b) is the video playing view. When a user touches a piece of a video message shown in Figure 4.10(b), the details of the message is shown. A user can touch the ‘play’ button to play a video. Figure 4.10(c) is the video sending view. The user inputs the necessary information and chooses a video file to send. He/she can select an existing video file, or capture a new one. Figure 4.10(d) shows the video capturing view. When a user selects the ‘capture a new one’ radio button in Figure 4.10(c), this view is shown to capture a video.



(a) View of video message list

(b) View of playing video



(c) View of sending video

(d) View of mail history

FIGURE 4.10: The user interface of prototype-Android

4.5 Summary

This chapter detailed the implementation of the prototypes to answer the second research question in Section 3.1. There are four prototypes built for experimentation: prototype-Flash, prototype-HTML5, prototype-J2ME and prototype-Android. Prototype-Flash is based on Flash technology. It provides a synchronous video streaming service inside a web browser on a PC. Prototype-Flash has a client/server architecture. The client side of prototype-Flash is built with Flex, while the server is built with Java running inside Red5. Prototype-HTML5 gives an asynchronous video interaction service inside a web browser on a PC. It is based on HTML5, Java Script and Java. Prototype-HTML5 also has a client/server architecture. The client part of prototype-HTML5 is a web page based on HTML5, CSS and Java Script. The Java server of prototype-HTML5 runs inside an open-source web server framework called GlassFish. Prototype-J2ME provides a portable asynchronous video communication for the Symbian OS, while prototype-Android provides a portable asynchronous video communication in an Android OS. Prototype-HTML5, prototype-J2ME and prototype-Android all transmit video files in an asynchronous way to a shared server. The connection between client and server is through HTTP. The features of these three prototypes are similar: the user can play incoming video and send video to other users. Thus, we can share the server of prototype-HTML5 with prototype-J2ME and prototype-Android. Since these three have the same basic features, it is possible to use the same server handling user requirements. It makes source code management easier. Another advantage is that users of these three prototypes can use PCs, Symbian phones and Android phones to do asynchronous video communication for user accounts are also shared. We call the shared server ‘HTTP-server’. Both the system design and the user interface of each prototype were described individually. The features of each prototype were listed, and user interfaces were shown with screenshots. All four prototypes were tested with users and in the lab. These tests are described in the next chapter.

Chapter 5

Results

This chapter discusses the results of experimentation with the prototypes and analyzes the results. The results are used to answer the third research question: ‘how to evaluate prototypes to find a suitable alternative for sign language communication’. Section 5.1 provides user behaviour and technology background data. Section 5.2 presents the results of user tests with the four prototypes, including both pilot and full trials. Section 5.3 presents the results of the performance tests of the Flash and HTTP servers. Section 5.4 synthesizes the results and analyzes them.

5.1 User behaviour and technology background

Two types of data were collected to understand the Deaf users’ background and their use of communication technology. The first is Deaf user background data. A survey was conducted to gather user profiles and their technology background data. We collected general information (gender, home language, education level), PC related and mobile related background data. A questionnaire was prepared, a presentation was shown to the subjects with an interpreter. The second is Internet usage data. This is the record of what Deaf users do on the Internet at the Bastion in the PC lab. Section 5.1.1 details Deaf users’ background data. Section 5.1.2 details Internet usage data.

5.1.1 Technology background

A survey was done with DCCT members to collect user and technology background data (see Appendix A). Thirty four DCCT members were surveyed. 50% of Deaf users go on the Internet weekly. Most (79%) of these DCCT members have these remote communication experience: 26% Deaf users use text messaging on a PC (like email) daily, almost 68% of DCCT members use text messaging on mobile phones (SMS) to communicate with others, one Deaf user has video call experience, but no Deaf user uses video chat on PC, text relay or video relay in their daily communication.

From the above data we can see that many Deaf people have a base of remote communication. It will not be hard for the experienced users to learn a new communication system for Deaf people. However, most (96%) of the experienced users do not choose synchronous video communication, but asynchronous text messaging. Do Deaf people prefer text messaging? Or, do they like to use asynchronous systems? We will discuss this question later with more data.

When the Deaf people are using PCs and mobile phones, 75% of them complain that the devices are too hard for them to use. Another 6% give up the devices as their Deaf friends do not use them. 32% of them want to learn how to use a computer. These Deaf people do not have the necessary PC skills, and they just need to be educated. 26% of them want to know what a computer can do. They do not know that a PC cannot only open documents and do typing exercises, but can also let Deaf people communicate with their friends remotely. If the benefits of a PC and the Internet are shown to them, they will be interested in computer devices. 35% of Deaf people suggested finding it easy to use communication systems and share this with their friends. They cannot find suitable communication tools running on a PC. The situation of mobile phones is different from that of PCs. Almost 68% of the surveyed Deaf users are looking for communication systems for Deaf people running on mobile phones. They would like to use mobile phones if there is a sign language communication tool for them. 29% of Deaf users need to learn to use a mobile phone.

44% of the subjects have had computer experience. 40% of these users use PCs with Windows OS. 5% of PC users use a Mac OS. The other 55% of computer users do not know what OS they use. This result tells us that many Deaf people do not have a concept of an OS. Thus, OS-related application is hard for them to

understand. They will be confused if an application can be installed on one PC, while the installation always fails in another PC. We listed some popular PC-based communication systems, including MSN, Google talk, Facebook, Tokbox, Camfrog and Skype. 28% of the Deaf users have used Facebook, while 25% of the Deaf people have used none of the above. Comparing the usage of Tokbox (3%), Camfrog (9%) and Skype (9%), it seems that asynchronous communication is preferred by Deaf people, for they do not need to focus on devices for a long time. The fact that 96% of remote communication experienced users choose asynchronous text messaging is also proof of this opinion.

88% of the surveyed people have a cell phone. 72% of the mobile phone owners use a Nokia cell phone, which means they use a Symbian OS. Over 90% of mobile phone users use SMS to communicate, and 40% of these DCCT members are Mxit (www.mxitlifestyle.com) users. 13% of Deaf users use mobile phone to go on the Internet. Most (87%) of these mobile phone users are interested in a video communication system for Deaf people on their mobile phones. Over 58% of them want to try a mobile video communication application if it is free. Deaf people are concerned about costs. Free applications are what they want. About the UI of the application, 58% of the Deaf people say they like pictures with simple words. They also want the perception of data charges to be less than the cost of SMS. Since asynchronous video transmission connects to the Internet intermittently while synchronous video streaming uses network continually during the conversation, the cost of the former is less than the latter. Cost is another reason for choosing asynchronous video.

5.1.2 Internet usage

BANG opened a PC lab at the Bastion in 2004. Six PCs with free Internet connection are available for DCCT members. DCCT staff help BANG to record the Internet usage data. BANG has aggregated the data collected by Deaf lab managers since 2007. This data lists what the DCCT members have done when they connect to the Internet. The number of users for each year is about one hundred. Figure 5.1 shows pie graphs from 2007 to 2010 for eight Internet items:

- ‘IM’ is an abbreviation for Instant Messaging (IM). IM counts when users use IM applications like MSN, Google talk and other instant chat software.

- ‘Video’ includes video conferencing usage of systems like Skype and Camfrog.
- ‘Facebook’ indicates usage of this popular social networking system.
- ‘Mail’ counts the usage of email.
- ‘Search’ counts the number of users who connect to the Internet for specific reasons, like job hunting or house rental.
- ‘Quick read’ shows the the number of users who go to web sites to get common information, such as local and global news. Youtube and Yahoo (www.yahoo.com) are counted here.
- ‘Education’ counts how many Deaf users use on-line education systems.
- ‘Other’ includes other actions on the Internet.

From Figure 5.1 we can see that email was the most popular Internet item in 2007. By 2008 the usage of other items has increased while the usage of email was decreased. In 2009, Facebook gained a similar number of users to email. In the meantime, many Deaf users became interested in reading news. The situation for 2010 is quite similar to 2009. However, the usage of video has decreased again. We believe this is due to the lab managers’ restriction of video downloading to avoid reaching the monthly download cap.

Figure 5.2 focuses on the usages of IM, video, Facebook, email and quick read from 2007 to 2010. Deaf people appear to like asynchronous communication services with pictures and video. As only a few PCs are available for many DCCT members, asynchronous communication appears more suitable for them than synchronous communication. A Deaf user can leave a message for friends, and friends might reply to this message after a week or a month. Pictures and video are visual, and therefore satisfy Deaf people much more than text messages. Facebook provides an asynchronous picture and video service. Quick read supports on-line video playing. These two items have increased continually from 2007. Email is primarily text. The main feature of both video applications and IM systems is synchronous communication. The usage of email, video and IM has decreased over the years.

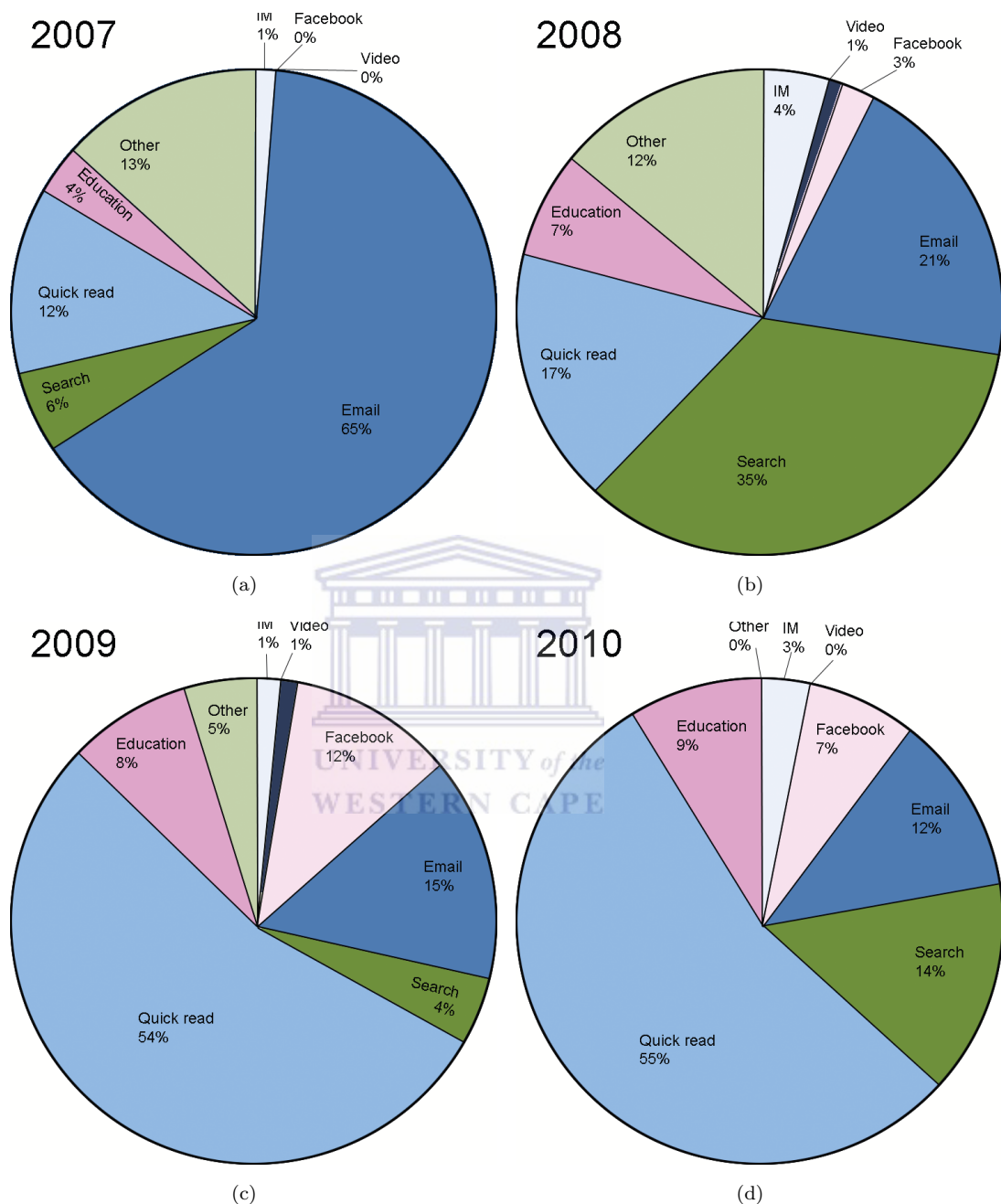


FIGURE 5.1: Internet usage at the Bastion by year: DCCT staff have recorded what Deaf users do when they go to the PC lab, and BANG has continually aggregated the data together. We have counted the usage for eight Internet items. For example, in 2007 there were 92 Deaf users who went on the Internet. The total usage of the service of these users is 1019 times, and email users used email 661 times. Thus, the email usage for 2007 is 65%.

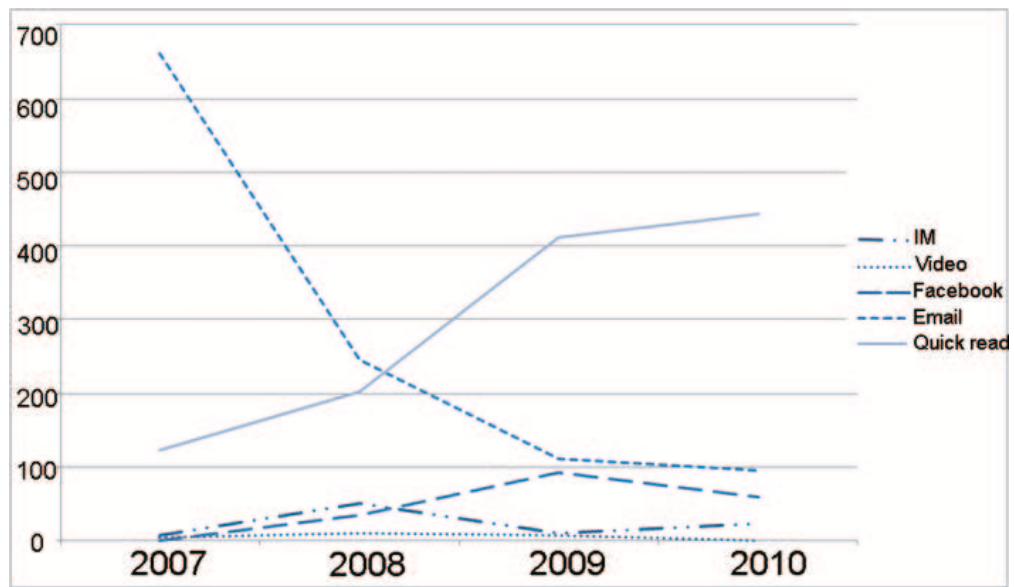


FIGURE 5.2: Comparison of Internet usage of key items

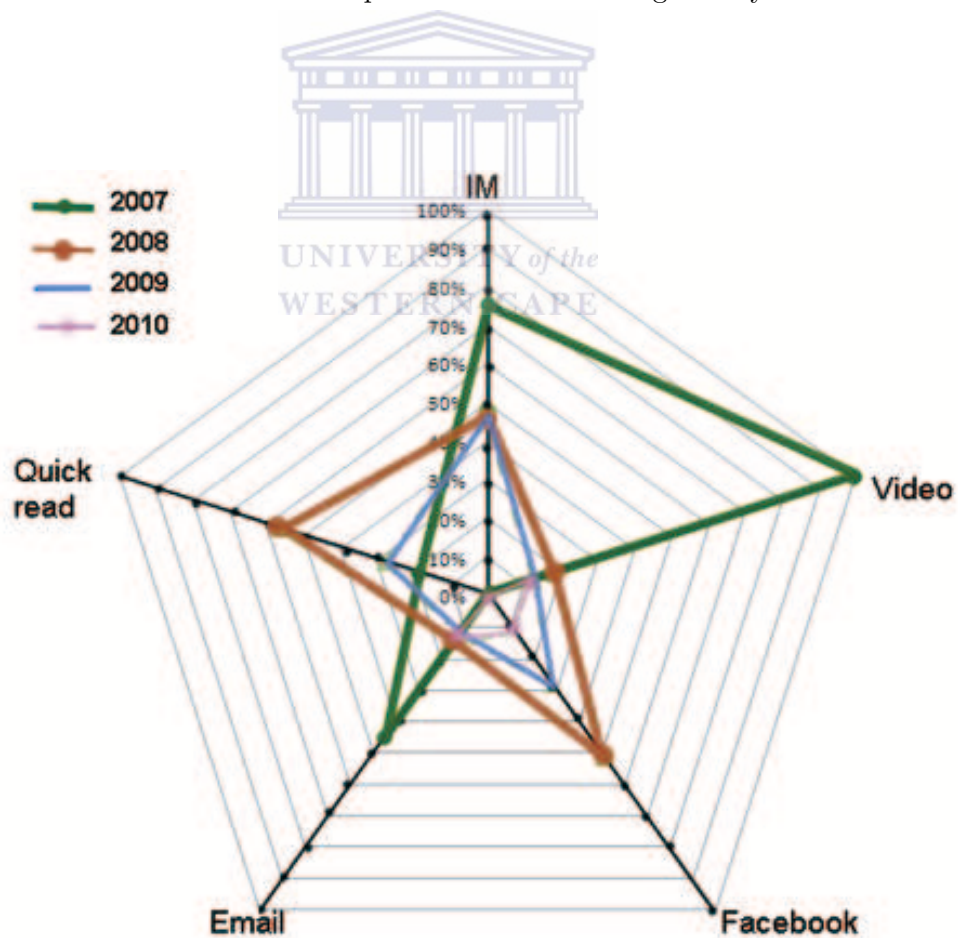


FIGURE 5.3: Internet usage of the top five users

TABLE 5.1: Internet usage counts

Year	Login times	User number	Some Internet items				
			IM	Video	Facebook	Email	Quick read
2007	929	92	8	4	0	661	123
2008	1025	129	51	11	35	245	203
2009	677	157	11	8	92	112	411
2010	753	195	24	0	60	96	444

Table 5.1 shows the actual counts of Internet items presented in Figure 5.2. This table shows that although the user number has increased year by year, the total number of logins increased and then decreased. The reason is shown in Figure 5.3. Figure 5.3 shows how many Internet resources have been used by the top five users in DCCT. In 2007, the total login times of the top five users was four hundred and ten. This number is almost half of the login times of the whole year. At the same time, the top five users spent much resources. For example, in 2007 the top five users used 46% email resources. This is a reason for the high email usage in 2007. The email usage of the top five users in 2008 is 15%, in 2009 is 13% and in 2010 is 12%. It means more and more Deaf people have started to use the Internet resource, although most of them are not as experienced as the top five users.

5.2 User tests

As introduced in section 3.3.5, we conducted two user tests, a pilot trial and a full trial. Both of these two trials were done at the Bastion, with DCCT staff and members. Section 5.2.1 lists the devices for the trials, and the installation details. Section 5.2.2 presents the results of the pilot trial. Section 5.2.3 presents the results of the full trial. Section 5.2.4 analyzes the overall results of the user tests.

5.2.1 Preparation

In the experimentation, computers are prepared for prototype-Flash and prototype-HTML5, while mobile phones are prepared for prototype-J2ME and prototype-Android. The following are all the devices used in the experimentation:

- A Sony VGN-Z12GN laptop which has an Intel Core 2 duo processor 2.4GHz CPU and 2GB memory running Windows Vista business OS. There is a web-cam installed on this laptop.
- A MacBook Pro laptop which has an Intel Core 2 duo processor 2.2GHz CPU and 2GB memory running MAC OS X. There is a web-cam installed on this laptop.
- A Nokia E66 mobile phone running the Symbian OS.
- A HTC Desire mobile phone running the Android OS.
- A local network.
- A wireless router.

The Sony laptop runs the servers. It also acts as a client of prototype-Flash and prototype-HTML5. The Mac laptop works as a client of prototype-Flash and prototype-HTML5. The Nokia phone runs the client application of prototype-J2ME. The HTC phone runs the client application of prototype-Android. The local network and the wireless router are used to build a wireless network for all the devices.

The installation of prototype-Flash has the following steps:

- Install Java Development Kit (JDK) 6 on the Sony laptop. We downloaded the install file for Windows from the Java website (java.sun.com), and installed the JDK in path `c:\java\jdk`. Java Runtime Environment (JRE) was also installed. The path is `c:\java\jre`.
- Add Java paths into the environment variables. Folder `c:\java\jdk\bin` and `c:\java\jre\bin` were added into the environment variable 'path'. Folder `c:\java\jdk\bin`, `c:\java\jdk\lib` and file `c:\java\jdk\lib\tools.jar` were added into the variable 'classpath'.
- Install Apache Ant 1.7 on the Sony laptop. We downloaded the package from the Apache website (ant.apache.org), and unzipped the package in path `c:\ant`.
- Add Ant paths into the environment variables. Folder `c:\ant\bin` was added into the variable 'path'.

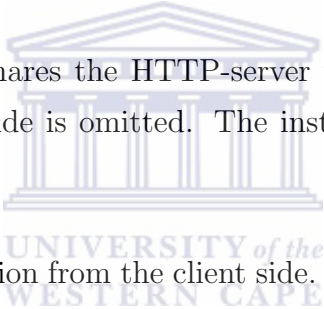
- Install Red5 0.9 on the Sony laptop. We downloaded the install file for Windows from Red5 website (osflash.org/red5), and installed the software in path `c:\red5`.
- Add Red5 paths into environment variables. Folder `c:\red5\lib` was added into the variable 'classpath'.
- Open communication ports. Port 5080 and port 1935 were opened for RTMP communication.
- Try to run the Red5 server. We restarted the laptop and entered the following address in a web browser: 'http://localhost:5080', and the welcome page of Red5 was shown.
- Install prototype-Flash into the Red5 server. We copied the folder 'chatPrj' into path `c:\red5\webapps`. The IP address of the Sony laptop is '192.168.0.186'. We opened file `C:\red5\webapps\chatPrj\WEB-INF\red5-web.properties`, and added the IP address into the value of variable 'webapp.virtualHosts'.
- Modify the connection from the client side. We opened the client side source code file 'chatPrjClient.mxml' with tool FlashDevelop (www.flashdevelop.org). We set '192.168.0.186' in line 67, and compiled the project. A Flash file 'chatPrjClient.swf' was built. We copied the Flash file into path `C:\red5\webapps\chatPrj`.
- Start prototype-Flash server. We restarted the Red5 service. It is necessary when the settings of folder 'webapps' were changed.
- Run prototype-Flash from the Mac laptop. We entered the following address in a web browser: 'http://192.168.0.186:5080/chatPrj', and the login dialog of prototype-Flash was shown.

JDK is also necessary for the other three prototypes. Since we installed JDK in prototype-Flash installation, it is omitted in the following introduction. The installation of prototype-HTML5 has the following steps:

- Install Glassfish on the Sony laptop. We downloaded the install file of Netbeans from the Java website, and installed it into the computer. (Glassfish is included in Netbeans.)

- Install Wamp server 2.0. We downloaded the install file of the Wamp server from the website (www.wampserver.com) and installed it in path `c:\wamp`.
- Copy work folder into the Wamp server. We copied the work folder 'file' into `c:\wamp\www`.
- Modify IP setting of the HTTP-server. Open file `C:\wamp\www\file\FilePath.ini`, change the value of variable 'FilePathRoot' into '192.168.0.186'.
- Start HTTP-server. We started the Wamp server and set it on-line. We opened the prototype-HTML5 project with Netbeans, and ran the project in Glassfish.
- Run prototype-HTML5 on the Mac laptop. We entered the following address in a web browser: 'http://192.168.0.186:8080/VideoChatServer/', and the login dialogue of prototype-HTML5 was shown.

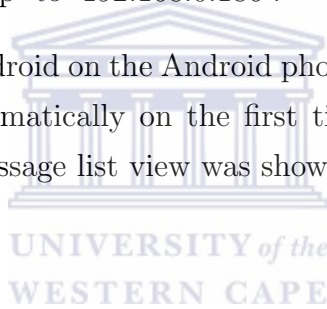
Since prototype-J2ME shares the HTTP-server with prototype-HTML5, the installation of the server side is omitted. The installation of prototype-J2ME has the following steps:

- 
- Modify the connection from the client side. We opened the prototype-J2ME project with Netbeans, and changed the content of line 26 of file 'MessageBox.java'. Then we rebuilt the project. A new version of file 'VideoCapturePlay.jar' was presented in the folder 'dist' under the root folder of the prototype-J2ME project.
 - Install prototype-J2ME on the Symbian phone. We connected the Symbian phone with the Sony laptop using bluetooth. We sent the file 'VideoCapturePlay.jar' from the laptop to the phone and installed it.
 - Register the Symbian phone. Two applications were installed on the phone. We used 'RegisterMIDlet' to register the phone for the first time.
 - Run prototype-J2ME on the Symbian phone. We clicked 'HelloMIDlet', and the prototype-J2ME was started.

Since prototype-Android share the HTTP-server with prototype-HTML5 and prototype-J2ME, the installation of the server side is omitted. The installation of prototype-Android has the following steps:

- Install Android SDK 2.1 on the Sony laptop. We downloaded the package for Windows from the Android website (developer.android.com), and unzipped it in path `c:\android-sdk`.
- Add Android SDK paths into the environment variables. Folder `c:\android-sdk\tools` was added into the variable 'path'.
- Install prototype-Android on the Android phone. We connected the phone with the Sony laptop using a standard Universal Serial Bus (USB) cable. We used the Android Debug Bridge (ADB) tool to install file 'VideoChat-Play.apk' onto the phone. File 'VideoChatPlay.apk' was stored in folder 'bin' of the root folder of prototype-Android project.
- Modify the connection from the client side. We opened prototype-Android on the Android phone and went to the performance menu. We changed the content of 'Server ip' to '192.168.0.186'.
- Run prototype-Android on the Android phone. The prototype registered the mobile phone automatically on the first time running prototype-Android. Then the video message list view was shown on the phone.

5.2.2 Pilot trial



The pilot trial was conducted with two groups of Deaf subjects, four Deaf users in each group. One group evaluated the two browser-based prototypes, and the other group gave feedback about the two mobile prototypes. The subjects in the pilot trial have computer and mobile phone experience. They use a computer and mobile phone daily, and often go on the Internet. These subjects also have rich experience in communication with hearing people.

We prepared user guides (see Appendix D) for the prototypes, and set up the prototypes in the BANG lab at the Bastion. Screen-shots and a brief explanation are listed in the user guides. We printed out the user guides, and gave them to our subjects. They followed the user guides step by step, trying the prototypes. There was no SASL interpreter for the pilot trial. We communicated with our subjects directly using written English, and our limited sign language.

Table 5.2 shows the results of the pilot trial. The user feedback is processed into percentile scores. There are three evaluation items in this trial. They are:

the total impression of the prototype, the video quality and the UI. Prototype-Android gained the highest evaluation in all three items, while prototype-Flash got the lowest evaluation in all items. Table 5.2 indicates that mobile technology satisfied Deaf people more than browser-based technology, and Android is the most promising of the four prototypes.

We observed these subjects during the trial. The Deaf subjects were satisfied with the prototypes. However, the user guides were not clear enough for them. Most of the subjects were confused when login views appeared, for the user names and passwords we prepared for the trials were complex. The guides for the mobile prototypes were incomplete. The introduction of an opening menu was omitted. Many subjects were not familiar with the mobile devices we prepared. They had trouble with their operation. The communication during the pilot trial was inadequate to understand the prototypes. They did not ask any questions after the trial, because of concerns for the language barrier.

Many details about doing the demo were adjusted based on the results of the user observations. In the pilot trial, subjects tried whole features, from login to logout. Since some features are not necessary for sign language communication, they were cut out for the subsequent full trial. The user guides were rewritten, including many animations to show the operation steps. Some pictures used in the user guides were changed to give the subjects a realistic feeling of communication. The demo video files were changed for the same reason. An interpreter was hired for communication during the full trial. We explained more details during the full trial to let the Deaf users understand what the prototypes can do for them. The operation of the devices was introduced before the full trials started.

TABLE 5.2: Prototype evaluation results in the pilot trial

	Percentile scores		
	Impression	Video quality	UI
Prototype-Flash	75	75	75
Prototype-HTML5	75	90	95
Prototype-J2ME	80	90	90
Prototype-Android	95	95	100

5.2.3 Full trial

In the full trial, thirty eight Deaf subjects used all four prototypes and gave their feedback. There was a SASL interpreter present. The samples we got included teenagers, young and old people. Over 55% of the Deaf users are unemployed, so almost 45% have jobs. Some of the subjects had never used computers or mobile phones, while some knew computers and mobile phones well.

All the Deaf users were very interested to use the prototypes. Most of them were especially interested in certain prototypes (prototype-Flash and prototype-Android) and wondered when they could use the system in their real life. Most of the subjects had never participated in such a trial before. They used the prototypes just as using a commercial product like Skype. They considered the realistic factors which could influence their decision. For example, some subjects chose prototype-Android as the best one in the four prototypes, while they chose prototype-J2ME as the one that they would like to use in their real life, because they knew the cost of the Nokia phone was cheaper than the cost of an Android phone. The cost of the phone is the reason that they choose prototype-J2ME. This reality is the reason we need to do these tests in the real-life environment. The result of this trial could show us Deaf people's real feelings.

The questionnaire (see AppendixC.1) has two kinds of questions. The first kind is to choose one prototype from the four on the basis of the description of the problem. There are three questions in this part. The first is to choose a favourite from the four prototypes. The second is to choose a most practical prototype. The third is to choose an easy to use prototype. The first kind of question was added into the questionnaire after the pilot trial. In the pilot trial we only asked the subjects to evaluate the prototypes individually. The first kind of question lets Deaf users compare the four prototypes with their impressions. The raw data of this part is shown in Table 5.3.

TABLE 5.3: Best choice evaluation in full trial

	Counts		
	Favourite	Practicality	Easy to use
Prototype-Flash	9	7	3
Prototype-HTML	1	1	2
Prototype-J2ME	5	5	7

TABLE 5.3: Best choice evaluation in full trial

	Counts		
	Favourite	Practicality	Easy to use
Prototype-Android	23	25	18
none	0	0	7

The **favourite** prototype. This question is to select a favourite prototype from the four. Prototype-Android was selected by 61% of the participants. Most Deaf participants were very satisfied with the quality of prototype-Android. They also thought prototype-Android was very easy to use. Many subjects asked the researcher where they could get such a phone and use such a service in their real life. It is clear that prototype-Android brought Deaf people a good user experience. The author was very happy to see the smiles on the faces after the subjects finished the test. The score for prototype-Flash was lower than prototype-Android but higher than the other prototypes. 24% of the subjects chose prototype-Flash as their favourite. The biggest difference between prototype-Flash and the other prototypes is that the former provides synchronous video service. This feature satisfied Deaf people. Although some subjects were confused about the UI of prototype-Flash and thought the video size for chatting should be bigger, they were happy with prototype-Flash. 13% of Deaf people chose prototype-J2ME and another 3% subjects selected prototype-HTML5. The quality of prototype-J2ME was not as good as prototype-Android. Users also needed to press many buttons to select services. These two weak points made prototype-J2ME mediocre. Prototype-HTML5 looks like on-line email. It is easy to use, and the quality of it is quite acceptable. However, prototype-HTML5 did not attract the attention of our subjects. Some Deaf people complained about the video size. They said it should be bigger.

The most **practical** prototype. This question is to choose a prototype that would be most useful in real life. The difference between this question and the previous one is that in this question more real-life factors should be considered. For example, the price of a phone is cheaper than a PC. A subject might choose prototype-J2ME for the low cost, even if his/her favourite is prototype-Flash. Prototype-Android got the support of most of the subjects, with 66% votes. Although some Deaf users liked prototype-Flash, they were concerned about the video quality of prototype-Flash. Some subjects thought prototype-Flash was an

exciting program. However, it was more like a toy than a communication tool. 18% of the subjects chose prototype-Flash. 13% of the Deaf people chose prototype-J2ME and another 3% subjects selected prototype-HTML5. The results of this question are very similar to the results of the previous one.

The most **easy to use** prototype. This question is to select the one most easy to use. Since many subjects did not have much computer or mobile phone experience, we made visual user guides for each prototype. The interpreter explained how the prototypes worked (in SASL), then our audience tried those prototypes step by step. The criterion is simple and clear: if there is a prototype that a user could use without the guide, this is the one the researcher is looking for. 50% of the subjects believed prototype-Android was the most easy to use prototype. They said prototype-Android was clear and simple. They preferred to use finger touch rather than move a mouse or press a button. On the other hand, 18% of the Deaf users would like the traditional way to use mobile applications, thus they selected prototype-J2ME. 18% of the subjects chose ‘none’, which means they thought they needed the guide before using any prototype. Only 8% chose prototype-Flash and 5% selected prototype-HTML5. As mentioned in section 5.1.1, the author found that in our sample, the number of mobile phone users is much bigger than the number of computer users. The result of this question confirmed this fact. Most Deaf users said they needed to learn more about computers.

TABLE 5.4: Prototype evaluation results in the full trial

	Percentile scores		
	Impression	Video quality	UI
Prototype-Flash	76	44	56
Prototype-HTML5	67	77	57
Prototype-J2ME	64	72	73
Prototype-Android	88	95	84

The second kind of question is to evaluate aspects of each prototype with percentile scores. There are three evaluation items in this trial. They are: the total impression of the prototype, the video quality and the UI. We provided five choices for each question in this part. The five choices correspond to five levels of satisfaction. The questions in this part are the same as the questions in the pilot trial.

The raw data is shown in Table C.1. We processed the results into percentage scores and show them in Table 5.4. All four were evaluated on the same aspects.

The evaluation of **prototype-Flash**. Many people were excited when they found they could do real-time sign language communication using this prototype. However, most of them complained about the video quality. Deaf people are used to signing with a fast speed, and they had to slow down because of the network delay. The subjects also thought that the video size of prototype-Flash should be bigger. They believed the video size was the biggest defect of its UI design.

The evaluation of **prototype-HTML5**. Basically, Deaf users were satisfied with prototype-HTML5. However, this prototype had no highlights to impress them. The video quality of prototype-HTML5 was considered to be acceptable. About the UI, Deaf people thought it was fine, but the video size should be bigger.

The evaluation of **prototype-J2ME**. Many people were interested in this prototype because it could be used anywhere. The only thing that bothered our audience was that they had to press buttons many times to complete an action. The video quality of prototype-J2ME was acceptable. Some Deaf users even thought the quality was fantastic. Many subjects loved the UI design of prototype-J2ME. They said it was simple and clear. They could completely understand the meaning of each menu.

The evaluation of **prototype-Android**. It is obvious that prototype-Android is preferred in this test. It gained the highest evaluation scores for each aspect. Most Deaf users were very interested in it. The subjects said they loved this prototype very much, and some of them said they wanted to keep the demo phone. Both the quality and the UI satisfied our audience. They thought the video was very clear and they could understand how to use this prototype very easily. It should be pointed out that the touch panel simplifies the operational progress greatly. For example, a user of prototype-J2ME needs to click buttons twice to send the captured video to another user, while a user of prototype-Android only touches a button once to do the sending. The switch between video sending and video playing views is another example. In prototype-J2ME a user clicks buttons three times to switch from video sending view to video playing view. In prototype-Android a user does the same action in one step. It might be an important reason why more phone users selected prototype-Android instead of prototype-J2ME.

During the full trial we observed the subjects while they were using the prototypes. They asked many questions after the test. Most of the questions were related to the mobile prototypes. Someone asked if prototype-J2ME could run on his phone. Someone wondered if there was a size limitation for sending video files. Many Deaf users wanted to buy an Android phone and asked us how they could get the device. A subject said the SMS service was too expensive to use and he wanted to use our prototype as soon as possible. Most of the subjects were satisfied with the mobile prototypes, and they wanted to learn much more details about them. There were also some questions about the browser-based prototypes. A Deaf user asked us how he could use prototype-Flash on his own PC. Another Deaf user said prototype-Flash was very convenient. Many subjects wanted to learn more computer skills in order to use the browser-based prototypes. During the full trial we noticed that some the subjects did not know how to control the mouse. It was difficult for these Deaf users to use the browser-based prototypes, even if there were the user guides for them.

5.2.4 Analysis of user tests

Comparing Table 5.4 with Table 5.2 we find that the average scores in the full trial are lower than the scores in the pilot. A possible reason is that the subjects in the full trial had less experience using computer and mobile phone applications. They only focused on the user experience they could get from our prototypes. The invisible criterion in their minds about a ‘good communication application’ is simple and strict. They are looking for a prototype that could let them do sign language communication comfortably just like they do face to face. However, from the result we could see that prototype-Android is the one that fits their needs best. Deaf people do not appear to care about synchronous video communication very much. They do not like to sit in front of a PC for a whole day just chatting. They are more interested in a mobile prototype for such a prototype could be used anywhere. And the video quality of the two mobile prototypes satisfied Deaf people more than the browser-based prototypes.

Figure 5.4 presents a comparison between browser-based and mobile alternatives based on Table 5.3. In the full trial, most of our subjects were more interested in mobile prototypes, especially prototype-Android. Combined with the results

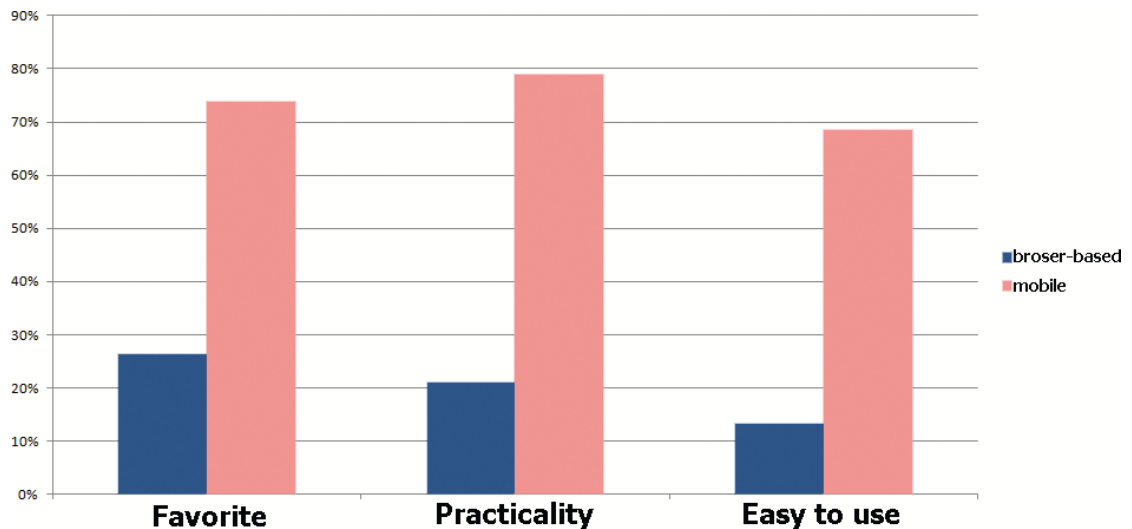


FIGURE 5.4: Comparison of two alternatives: A comparison between the browser-based alternative and the mobile alternative is shown in this figure. The result is based on Table 5.3.

in section 5.1.1, we think mobile technology is more suitable for sign language communication development in South Africa than browser-based technology.

5.3 Performance tests

Each prototype for the performance tests is the same version for the user tests. To record the performance data from the server, we ran the server of each prototype individually for forty eight hours. The machine we used for running the server was a virtual machine with Windows OS on the local network. The virtual machine was clean. Only the necessary applications (see section 5.2.1) were installed on the machine. We formatted the virtual machine and ran the Flash server initially. When the performance test of Flash server was done, we formatted the virtual machine again and tested the HTTP-server. Windows performance monitor was used to record logs on the virtual machine. The logs were saved in a text file on the virtual machine. We connected to the server from other client devices (see section 5.2.1) to do the video communication irregularly during the period. Then PAL tool was used to analyze the log files recorded by the Windows performance monitor. There are three performance ‘objects’ we monitored during this test. They are: CPU, memory and bandwidth. CPU usage and memory usage indicate the workload of the server from two different angles. Bandwidth used during client

and server conversations was logged to help us evaluate how frequent the video communication occurred.

Section 5.3.1 describes the results of Flash server. Section 5.3.2 details the results of HTTP-server. As mentioned in section 4.3.1 and section 4.4.1, prototype-J2ME and prototype-Android share the server with prototype-HTML5. Therefore, the results in section 5.3.2 are not only for prototype-HTML5, but also for prototype-J2ME and prototype-Android. Section 5.3.3 analyzes the performance data.

5.3.1 Flash server

Figure 5.5 shows performance data for the Flash server. We used the Sony laptop and the Mac laptop to connect to the server, to simulate the conversation between two users. The CPU usage is the CPU resources used by the Flash server. Windows performance monitor records the CPU usage of the user process. Since only the server service was started on the virtual machine, this usage data was the CPU usage of the Flash server. It is easy to see that the server spends all processor resources from the start, even when there is no video chat. The reason for this is the shared objects the server maintained. The Flash server is updating the status of the shared objects continually. The memory usage is almost a straight line. The server does not ask for much memory during conversations. About network workload, we can see the data transmission from the network spikes when there are user requirements. In the meantime, the CPU usage decreases. The server uses some network resources when communication starts and ends. The identity of the published video streaming of client A is transferred to client B when communication starts. The peak of bandwidth usage appears when the conversation ends, with a speed of 516 bytes per second. The server shuts down the published video streaming of the clients. The command of closing video communication is also sent out at that time. These actions use more bandwidth than starting communication. From Figure 5.5 we can see that the Flash server uses much more bandwidth when a communication starts and ends, than for opening and closing a video broadcast. And the bandwidth used when closing a conversation is more than for opening a conversation.

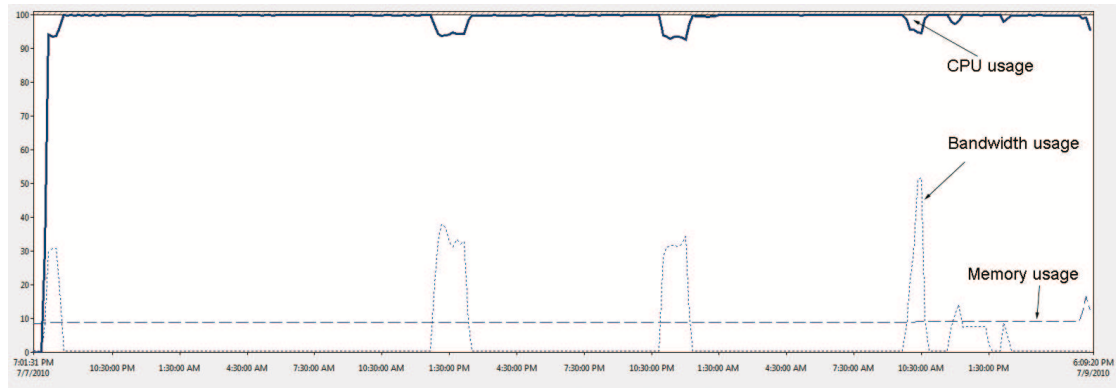


FIGURE 5.5: Processor and memory usage of prototype-Flash

5.3.2 HTTP server

Figure 5.6 shows the system resources used by prototype-HTML5, prototype-J2ME and prototype-Android. We ran the HTTP-server on the virtual machine, and connected to the server from the Sony laptop, the Mac laptop, the Nokia phone and the HTC phone. The HTTP-server spends CPU resources during video transmissions, and releases the resources after the communication ends immediately. The communication between the server and the clients is asynchronous. The server does not need to maintain data like the Flash server. The memory usage is almost a straight line. We thought a part of the memory resources is used by running Glassfish server and Wamp server. The bandwidth usage looks like a straight line, as the bandwidth usage for data transmission is too small to observe. The basic usage of bandwidth expended by the HTTP-server is about 63 bytes per second.

5.3.3 Analysis of performance data

Comparing Figure 5.5 with Figure 5.6, the HTTP-server seems more stable than the Flash server. The HTTP-server spends less CPU and bandwidth resources than the Flash server. Although the HTTP-server uses more memory resources than the Flash server, the overall performance is acceptable. The Flash server spends more bandwidth on opening and closing a synchronous video conversation than during the conversation period. This means that once a communication starts, it will not use much network resources of the server. However, the data size of synchronous video streaming is huge. Therefore, the total bandwidth spent by

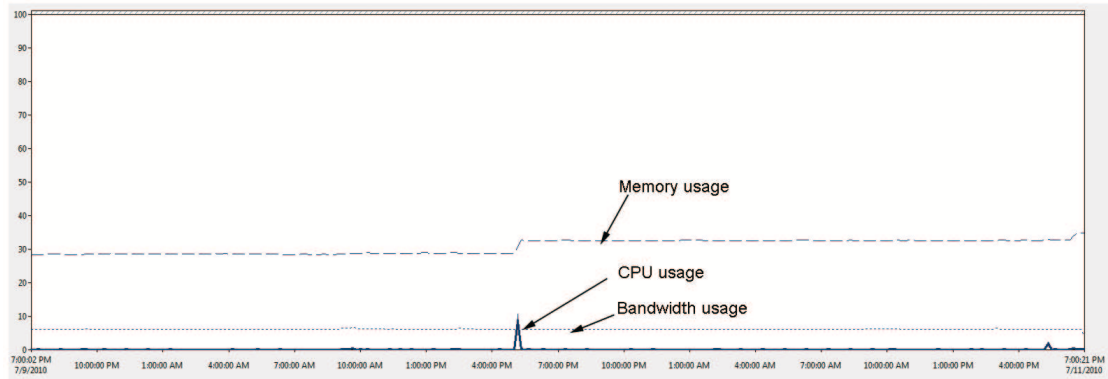


FIGURE 5.6: Processor and memory usage of prototype-HTML5: Since prototype-HTML5 shares the HTTP-server with prototype-J2ME and prototype-Android, this figure is also show the data usage of prototype-J2ME and prototype-Android.

the Flash server is more than the HTTP-server. The HTTP-server transmits asynchronous video, and the bandwidth usage is related to the size of the transmitted video file. In our performance tests, the bandwidth usage of the HTTP-server was stable.

5.4 Synthesized results

We can now analyze the four prototypes with Deaf users' opinions and performance data. Figure 5.7 presents both the qualitative and quantitative data processed as percentile scores for each of the four prototypes. The data from Table 5.4 is the qualitative data shown in the bottom half of each hexagon. The results of the performance tests are the quantitative data shown in the top half of each hexagon. The items of qualitative data are impression, video quality and the UI. The items of the quantitative data are processor, memory and bandwidth. There are three items for both qualitative and quantitative evaluation respectively. The data for each evaluation point is converted to a score. We calculated the actual values of the performance data to relative scores. For example, dx is the value of CPU usage of the Flash server, and dy is the value of CPU usage of the HTTP-server. The relationship of dx and dy is: $dx > dy$. The quantitative scores of the two servers are sx and sy . We calculated sy as a full score ($sy = 100$), and the score of sx is: $sx = \frac{100}{dx/dy}$. The centre of the hexagram refers to zero, and the vertexes

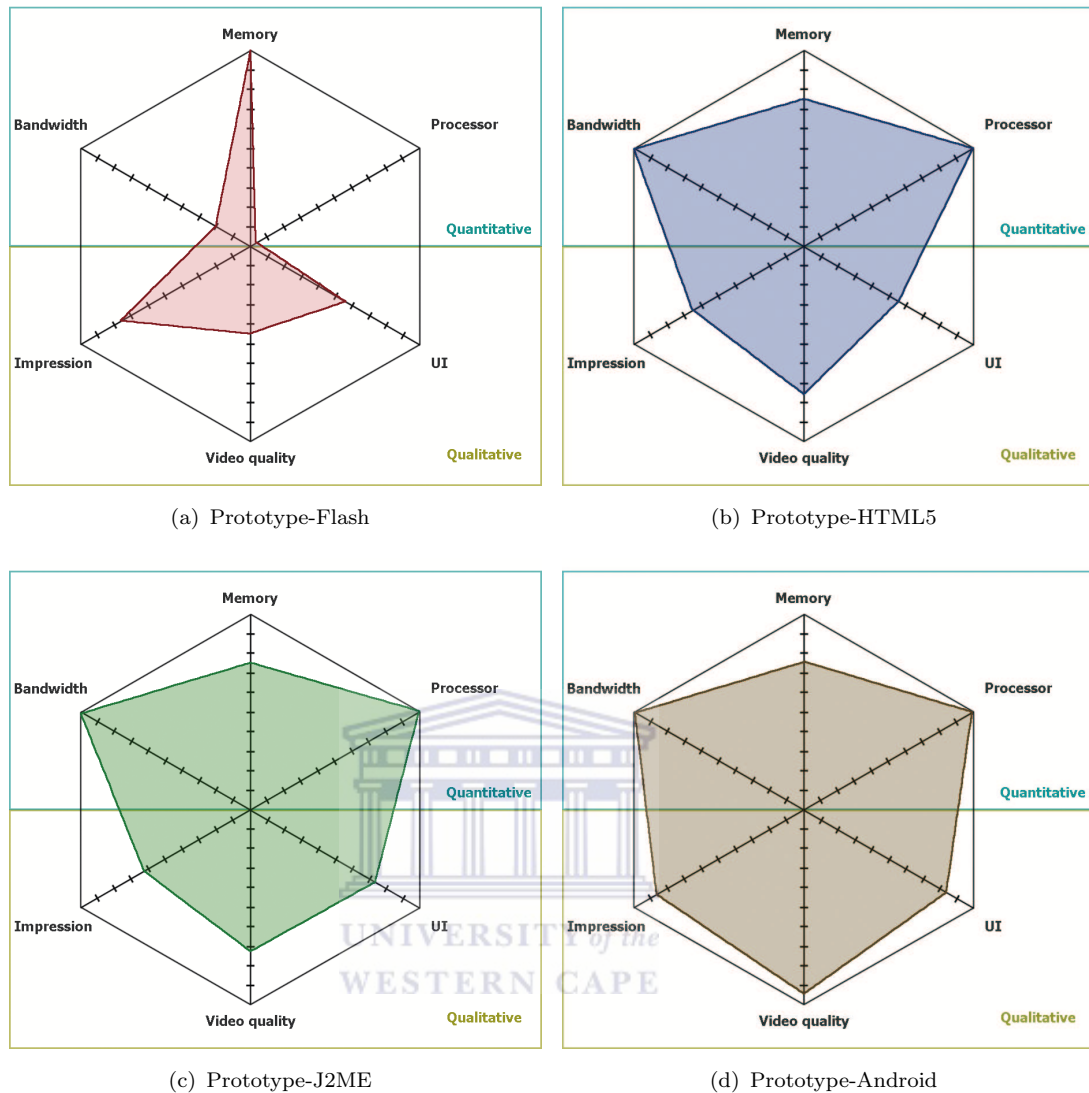


FIGURE 5.7: Integration of qualitative and quantitative data

means one hundred. The project with the largest area has the best combined scores, and that is prototype-Android.

Figure 5.7(a) shows the synthesized result for prototype-Flash. Both the advantages and disadvantages of prototype-Flash are extreme. The best evaluation item of prototype-Flash is memory usage. The worst evaluation items of prototype-Flash are CPU usage, bandwidth usage, video quality feedback and UI feedback. We believe improvement of the video quality and UI are very necessary for prototype-Flash. At this phase, this prototype is not suitable for sign language communication.

Figure 5.7(b) shows the synthesized result for prototype-HTML5. Prototype-HTML5 has a relatively average evaluation. The best evaluation items of prototype-HTML5 are CPU and bandwidth usages. The evaluations of other items are average. The usage of network resources increases when a conversion starts. This disadvantage limits prototype-HTML5 to a small-scale application. The quality of prototype-HTML5 has satisfied Deaf people. The UI of this prototype needs to be reconsidered. This prototype did not impress Deaf people. It is not suitable for sign language communication.

Figure 5.7(c) shows the synthesized result for prototype-J2ME. Prototype-J2ME also has an average evaluation. Since the three asynchronous prototypes share the HTTP-server, they have the same advantages and disadvantages with regard to performance items. The best evaluation item of prototype-J2ME are CPU and bandwidth usages. The evaluations of the other items are average. The video quality of prototype-J2ME satisfied Deaf people. The UI of prototype-J2ME is fine. The results of prototype-J2ME are better than prototype-Flash, and similar to prototype-HTML5. It also is not really suitable for sign language communication.

Figure 5.7(d) shows the synthesized result for prototype-Android. Prototype-Android is considered to be the best of the four prototypes. It has the same advantages and disadvantages as prototype-HTML5 and prototype-J2ME about performance items. The best evaluation item of prototype-Android are CPU usage, bandwidth usage, impression feedback, video quality and UI. The evaluation of memory usage is average. The majority of the audience was very satisfied with the quality and the UI. We believe prototype-Android is the most promising one for sign language communication.

5.5 Summary

The synthesized results of user tests and performance tests show that mobile technology is more popular than browser-based technology with Deaf people. It is consistent with the Deaf user background data. As mentioned in section 5.1.1, 88% of the volunteers have mobile phones, and almost 68% of the Deaf users usually use text messaging applications (SMS, Mxit) to communicate. Many Deaf users have mobile experience, and this experience helps them to access our mobile

prototypes. Most (86%) of these mobile phone users are looking for video communication systems for Deaf people. Their preferred video communication system should be easy to use, with a simple UI style. Prototype-J2ME and prototype-Android fit these needs well. Thus, they got high evaluations in the full trial. During the full trial, many Deaf users were interested in the Android phone. They wanted to buy such a device. This result is consistent with the answer to question 10 in Table A.1. Although many Deaf users do not like mobile phones, 70% of them change their minds when they found a video communication system for Deaf people. The highlighted prototype-Android made the subjects become interested in the platform device. There was one synchronous prototype and three asynchronous prototypes prepared for the research. The impression and UI evaluations of the synchronous and asynchronous prototypes were similar, while the evaluation of the asynchronous prototypes video quality is much higher than the synchronous prototype. It seems that the quality of synchronous video is not suitable for sign language communication, while asynchronous video satisfies Deaf people. It also explains the results of Internet usage about the low usage of video applications and the increase in use of Facebook and Quick read.

Based on all the above results, prototype-Android is considered to be the best of the four prototypes. The usage of CPU and bandwidth is low, while the usage of memory is acceptable. In the meanwhile, the majority of the audience was very satisfied with the quality and UI. We believe prototype-Android is the proper one for sign language communication.

Chapter 6

Conclusion

This thesis describes the design and evaluation of four alternatives for sign language video communication. These prototypes were measured and compared to identify a promising video communication alternative for Deaf people, in order to help them use network technology comfortably. Chapter 1 introduces background knowledge of this research. The definition of Deaf is introduced, and the South African Deaf culture is mentioned. Chapter 2 addresses related work. Chapter 3 discusses research questions. Some research methods are used to solve the research questions. The experimental design is also presented here. Chapter 4 describes the implementation of the whole system. Chapter 5 explains the analysis of the results of the experimentation.

Section 6.1 draws conclusions about communication alternatives for Deaf people. Section 6.2 discusses how to conduct user trials with Deaf subjects. Section 6.3 points out the limitation of this research. Section 6.3 suggests future work.

6.1 Communication alternatives for Deaf people

In this section the contents of the thesis are briefly summarized. Each subsection summarizes a chapter, and Section 6.1.6 draws a conclusion.

6.1.1 Introduction

This thesis describes the process to identify a promising video communication technology that provides acceptable sign language video communication for Deaf people. Some Deaf related projects, such as MobileASL and BANG H.264 asynchronous video system, were introduced. Two kinds of browser-based technologies, Flash and HTML5, and two kinds of mobile technologies, J2ME and Android were chosen. An evaluation of the four alternatives for sign language video communication is conducted. These prototypes were measured and compared to identify the most promising video communication alternative for Deaf people.

6.1.2 Related work

Text relay, video relay and sign language video communication were introduced as the main directions of Deaf related projects. Text relay allows Deaf users to use text devices to communicate with hearing people via an interpreter. Video relay provides a video service for Deaf people, helping them to communicate with hearing users in sign language via an interpreter. Sign language video systems focus on Deaf-to-Deaf communication. Some examples were introduced. Many video systems for hearing people were also introduced in this chapter. Although these applications are not for Deaf people specifically, they can also be used by Deaf people, albeit with some limitations. The related technologies were presented as well. The evaluation of video systems was also discussed, especially the need for qualitative and quantitative evaluation with Deaf subjects due to cultural considerations.

6.1.3 Methodology

The main research question is: how can we choose a promising video communication alternative to provide acceptable sign language communication service, and make sure it is easy to use for Deaf people's daily conversation? This research question was split into three questions. In order to answer the research questions, three research methods were used: qualitative, software engineering and quantitative methods. Experimentation had three components: user behaviour and

background data collection, user tests and performance testing. Deaf user background data was gathered with a survey, and Internet usage was aggregated. We conducted two rounds of user tests. A pilot trial examined the prototypes with a small number of participants, and a full trial tested the prototypes with thirty eight Deaf people. Performance testing collected performance data in terms of bandwidth, CPU and memory usage in the laboratory.

6.1.4 Prototype design

Four prototypes were built for the research. Prototype-Flash is based on Flash technology. It provides a synchronous video streaming service inside a web browser on a PC. Prototype-HTML5 provides asynchronous video interaction inside a web browser on a PC. It is based on HTML5, Java Script and Java. Prototype-J2ME provides a portable asynchronous video communication for a Symbian OS, and prototype-Android provides a portable asynchronous video communication in Android OS. Prototype-HTML5, prototype-J2ME and prototype-Android all transmit video files in an asynchronous way. Since these three have the same basic features, it is possible to use the same server. It makes source code management easier. Another advantage is that users of these three prototypes could use PCs, Symbian phones and Android phones to do asynchronous video communication, for user accounts are also shared. We call the shared server ‘HTTP-server’. Prototype-J2ME refers to the client side of the prototype, and so does prototype-Android.

6.1.5 Results

The first part of the results come from a user behaviour and technology background. To understand the Deaf people’s background and their usage of communication technology, Deaf user background and Internet usage data was collected. Deaf user background data was gathered with a survey of thirty four people at the Bastion. We found that many Deaf people use mobile phones to do daily communication. Internet usage was collected in the BANG PC lab at the Bastion. DCCT staff recorded the data and we analyzed the data from 2007 to 2010 with eight Internet items: IM, video, email, search, Facebook, quick read, education and other. From the results of Internet usage we can see that Deaf people appear

to like asynchronous communication services with pictures and video. As only a few PCs are available for many DCCT members, asynchronous communication appears more suitable for them than synchronous communication.

The second part of the results comes from user testing. We conducted two user tests, a pilot trial and a full trial. Both of these were done at the Bastion with Deaf people. The pilot trial was conducted with eight Deaf subjects who were in two groups. One group evaluated the browser-based prototypes, and the other group evaluated the mobile prototypes. We prepared a questionnaire for the subjects, and observed them during the trial. In the full trial, thirty eight Deaf subjects used all four prototypes and gave their feedback. There was a SASL interpreter with us for facilitating communication. The user feedback of the full trial shows us that most of our subjects were mostly interested in mobile prototypes, especially prototype-Android.

The third part of the results comes from the performance tests. There are three performance ‘objects’ that we monitored during this test. They are: CPU, memory and bandwidth. The HTTP-server looks more stable than the Flash server. The former spends less CPU and bandwidth resources than the latter. Although the HTTP-server uses more memory resources than the Flash server, the overall performance is acceptable. The Flash server spends more bandwidth on opening and closing a synchronous video conversation than during the conversation period. The HTTP-server transmits asynchronous video, and the bandwidth usage is related to the size of the transmitted video file. From our performance tests the HTTP-server spent less bandwidth in video transmission than the Flash server, and the former looks more stable than the latter.

6.1.6 Conclusion

From the experimental results we can see that mobile alternatives are better than browser-based technology. The evaluation of Android is better than J2ME. About the video transmission results, asynchronous video appears to satisfy the Deaf users more, for the quality is higher than the quality of synchronous video. We believe prototype-Android is the most promising one for sign language communication because of the following advantages:

- Prototype-Android runs on mobile phones to provide a portable video service. Many Deaf people have their own mobile phones, and they would like to use mobile applications rather than PC software.
- Prototype-Android provides an asynchronous video service. Asynchronous video is more popular than synchronous video for Deaf people. Asynchronous video does not interrupt Deaf people's work and study. A Deaf user can leave a video message for his/her friends during a tea break, and watch the reply after work.
- The video quality of prototype-Android is quite good. Most of the Deaf subjects were satisfied with the quality, and thought the video quality was good enough for sign language communication.
- The UI of prototype-Android is simple and clear. Many Deaf users did not have much computer or mobile phone experience. However, they can use the prototype without a user guide.

6.2 Suggestions for conducting trials with Deaf people

We learned many things about conducting trials with Deaf people. The biggest problem in this part is the communication between the researchers and the Deaf participants. Although we learned some SASL from a class, it was not enough for us to explain our prototypes clearly, and understand the Deaf users' opinions. Thus, the interpreter is necessary for trials and surveys. The communication between the researcher and the interpreter is also important. A professional interpreter does not change what a researcher says. It means that the researcher will not get any advice from the interpreter, even if the interpreter realizes something should be changed because of his/her experience of communication with Deaf people. Therefore, all the words used in the introductions and documents should be clear and contain ambiguity.

Another thing we want to mention is that the process of survey and interview should not be too long. Since Deaf people need to focus on the signing of the interpreter during the period, they feel tired and lose interest if the process takes a long time. According to our experience, half an hour is the maximum time

for the survey and interview. For the same reason, the number of questions in a questionnaire should also be considered. We prefer no more than thirty questions with multiple choice answers.

Deaf people do not always go to the Bastion, and it took us three days to gather enough Deaf subjects for our final trial. We went to the Bastion on a Monday twice, and gathered twenty one subjects. However, the third time we went there was on the third Sunday, and we gathered seventeen Deaf users in half a day. If we had known that over one hundred Deaf members of DCCT would attend on the third Sunday, we could have saved much time and money on the trials.

6.3 Limitations of this research

In our full trial, the researcher introduced how to use the prototypes before the Deaf subjects tried to use them. The result might change if our subjects tried the prototypes without the intervention of the researcher. In the meanwhile, all the subjects tried the prototypes for a short time. Their feedback might change if they were to use the prototypes for days or weeks.

We collected the Deaf user requirements from DCCT members, and tested the prototypes with DCCT participants at the Bastion. Our results are applicable to the particular environment DCCT only.

We collected the performance data on the server side. The performance data of the client side was not gathered in this research. The four prototypes run on different platforms, and it is hard to measure the performance data independent of hardware and OSs. Another reason is that the features of each prototype are variable. We did not find a way to measure the client performance of each prototype with a common standard.

We did not compare our prototypes with other related work, such as MobileASL. The participants of MobileASL have the different social culture with our subjects. The economic and education background of them are variable. It would be interesting if we test our prototypes and mobileASL software with one subject group.

We used different devices (PCs, a Nokia E65 phone and a HTC desire phone) to test our prototypes with Deaf users. The features of the devices might affect the

results of evaluation. For example, a Deaf subject would like to use prototype-J2ME because he has a Symbian phone with button interface. Another Deaf subject might think prototype-Flash is his favourite for the screen size of PC is much bigger than the screen size of phones. The results would be more persuasive if we could done user trails without the above factors.

6.4 Future work

The prototype-Android can be improved. Prototype-Android should be similar to cellular video conferencing and/or Short Message Services (SMS), depending on the temporal modality, whether real-time or asynchronous. At this stage, the video messages of prototype-Android only include one video file. One could attach more than one video into a message, and text messaging can also be added to the prototype. The text messaging can be both synchronous and asynchronous. A Deaf user can leave a text message with videos to his/her off-line friends. The user can also do real-time text chatting with another on-line friend using our prototype.

One might also think about combing prototype-HTML5, prototype-J2ME and prototype-Android together. These three prototypes share the server and can exchange video files now, but their features are not integrated. There will be one video system for Deaf people, with three clients running on a PC, the Symbian OS and the Android OS. The features and the UI of each client are the same.

One might collect the performance data (bandwidth, CPU and memory usage) on the client side. A comparison of our prototypes and related work can also be done. Our target community is always concerned about the costs, which concerns the bandwidth consumption of prototype-J2ME and prototype-Android. More evaluations about bandwidth consumption can be done. For example, one can compare the bandwidth usage of prototype-J2ME, prototype-Android and mobileASL. One can also monitor the actual bandwidth data spent on voice-on and voice-off video transmission to calculate how bandwidth usage is reduced.

MobileASL did some work to improve the video quality. Skin segmentation and activity recognition [10] were used to get clear video with limited processor resources. We want to do similar research in the next phase to increase the video quality.

Some of the physical problems associated with mobile devices we were unable to fix, such as having the high quality video camera next to the display and having a wide angle camera to view the torso of a signing user instead of the floating head. It would be nice to have these features on a mobile phone.



Appendix A

Technology background questionnaire

A.1 Questionnaire

Part 1: Personal information



1. Gender:

- Male. Female.

2. Age:

- 15–25. 25–35. 35–40. 40–50. 50 or older.

3. My home language is:

- South African Sign Language. Xhosa. English. Afrikaans. Zulu. Other.

If **OTHER PLEASE** fill in your home language in the space provided:

4. If you know written English, how would you rate your level of English:

- Poor. Average. Good. Excellent.

5. I grew up in a:

City. Town. Informal Settlement. Rural Area.

6. Now, I live in a:

City. Town. Informal Settlement. Rural Area.

7. I use a:

Mobile phone. Computer. Both. Neither.

8. If “7 is none”, why do not use a computer or mobile phone?

I do not want to pay for them. It is hard to use for me. No friends of mine use them. I do not know what they can do for me. Other.

If OTHER PLEASE fill in your reason:

9. If you dont use computer, which of the following can change your mind?

If I learn to use computer. If I understand how computer can change my life. If I can use computer to communicate with other Deaf people easily. Other.

If OTHER PLEASE fill in what will change your mind:

10. If you dont use mobile phone, which of the following can change your mind:

If I learn to use mobile phone. If there is communicate software for Deaf. Other.

If OTHER PLEASE specify in the space provided:

11. How often do you use the Internet:

Daily. Weekly. Monthly. Rarely (less than once a month).

12. Which of the following do you use? (You may select more than one):

- Face to face with sign language. Written English using computer. (e.g. email, chat). Written English using mobile phone. (e.g. SMS). Video call. Video chat in Internet. Text relay service. Video relay service. Written English using pen and paper. Other.

If OTHER PLEASE fill in your primary communication method:

Part 2: Mobile background

13. What brand do you use:

- Nokia. Samsung. LG. ZTE.

14. I normally use the phone for:

- SMS. MXIT. Internet. Playing games. Other.

If OTHER PLEASE fill in what you use the phone for in the space provided:

15. If “14 is Internet, do you want to use video chat software in the phone?

- Yes, I want to use it any way! Yes, if it is free. Yes, if it is free and easy to use. No.

16. Which of the following is your favorite user interface style of mobile phone software?

- Pictures and few words. A few words and simple functions.
 Words and a lot of functions.

Part 3: PC background

17. The operation system (OS) of my computer is:

- Microsoft Windows. MAC OS. Ubuntu or Linux OS . I do not know

18. Which of the following have you used?

- MSN. GTalk. Facebook. Tokbox. Camfrog. Skype. None.

19. What point do you most dislike of the chat software?

- Always ask me to update. Need to install at first. A lot of unusable functions for me. No or less help documents. The video screen is too small to sign. Other.

20. What point do you most like of the chat software?

- Good user interface. Simple and clear menu options. All my friends are using it. High video quality. Play video frequently. Other.

A.2 Results

TABLE A.1: Technology background data Part 1: Personal information

No.	Question	Counts of answers							
		A	B	C	D	E	F	G	H
1	Gender	17	17	—	—	—	—	—	—
2	Age	1	20	4	7	2	—	—	—
3	Mother language	19	9	19	4	0	0	—	—
4	English level	4	12	18	1	—	—	—	—
5	Grown up place	11	9	1	13	—	—	—	—
6	Living place	14	6	9	2	—	—	—	—
7	Communication device experience	15	3	15	1	—	—	—	—
8	(7)Why not use PC or mobile	0	14	2	0	—	—	—	—
9	(8)In what case you will want to use PC	11	9	12	0	—	—	—	—
10	(8)In what case you will want to use mobile	10	23	0	—	—	—	—	—
11	Internet frequency	7	10	3	12	—	—	—	—
12	Communication methods	17	9	23	1	0	0	0	4

TABLE A.2: Technology background data Part 2: Mobile background

No.	Question	Counts of answers				
		A	B	C	D	E
1	Brand	18	4	2	1	—
2	Favorite service	27	12	4	6	1
3	Video communication system for Deaf people?	3	15	5	4	—
4	Favorite UI	15	7	6	—	—

TABLE A.3: Technology background data Part 3: PC background

No.	Question	Counts of answers						
		A	B	C	D	E	F	G
1	OS	8	1	0	11	—	—	—
2	Favorite application	6	2	9	1	3	3	8
3	Disadvantage of the previous application	5	2	2	3	4	10	—
4	Advantage of the previous application	2	5	5	2	2	11	—



Appendix B

Pilot trial questionnaire

TABLE B.1: Pilot trial result Part 1: Browser-based prototypes evaluation

Prototype name	Question	User1	User2	User3	User4
Prototype-Flash	Impression	Good	Fine	Good	Good
	Video quality	Fine	Good	Good	Good
	UI	Fantastic	Fine	Good	Fine
	Favorite	Yes	No	Yes	Yes
	Practicality	NG	OK	OK	OK
	Easy to use	OK	NG	NG	OK
Prototype-HTML	Impression	Good	Good	Fine	Good
	Video quality	Good	Fantastic	Good	Fantastic
	UI	Fantastic	Good	Fantastic	Fantastic
	Favorite	Yes	No	No	Yes
	Practicality	OK	OK	OK	OK
	Easy to use	OK	OK	OK	OK

TABLE B.2: Pilot trial result Part 2: Mobile prototypes evaluation

Prototype name	Question	User1	User2	User3	User4
Prototype-J2ME	Impression	Fine	Good	Good	Fantastic
	Video quality	Good	Fantastic	Good	Fantastic
	UI	Fine	Fantastic	Fantastic	Fantastic
	Favorite	Yes	Yes	No	Yes
	Practicality	OK	OK	OK	OK
	Easy to use	OK	NG	OK	OK
Prototype-Android	Impression	Fantastic	Fantastic	Good	Fantastic
	Video quality	Fantastic	Fantastic	Good	Fantastic
	UI	Fantastic	Fantastic	Fantastic	Fantastic
	Favorite	Yes	Yes	Yes	Yes
	Practicality	OK	OK	OK	OK
	Easy to use	OK	NG	OK	OK



Appendix C

Full trial questionnaire

C.1 Questionnaire

21. Which one do you most like?

- Prototype-Flash. Prototype-HTML5. Prototype-J2ME.
 Prototype-Android. None.

22. Which one would you like to use in real-life?

- Prototype-Flash. Prototype-HTML5. Prototype-J2ME.
 Prototype-Android. None.

23. Which one could you use without a guide?

- Prototype-Flash. Prototype-HTML5. Prototype-J2ME.
 Prototype-Android. None.

About prototype-Flash

24. How do you think about this application?

horrible ———— fantastic

25. How do you think about video quality?

horrible ———— fantastic

26. How do you think about the user interface?

horrible ———— fantastic

About prototype-HTML5

27. How do you think about this application?

horrible ———— fantastic

28. How do you think about video quality?

horrible ———— fantastic

29. How do you think about the user interface?

horrible ———— fantastic

About prototype-J2ME

30. How do you think about this application?

horrible ———— fantastic

31. How do you think about video quality?

horrible ———— fantastic

32. How do you think about the user interface?

horrible ———— fantastic

About prototype-Android

33. How do you think about this application?

horrible ———— fantastic

34. How do you think about video quality?

horrible ———— fantastic

35. How do you think about the user interface?

horrible ———— fantastic

C.2 Results

TABLE C.1: Full trial results: individual prototype evaluation

Prototype name	Question	Counts				
		Horrible	Normal	Fine	Good	Fantastic
Prototype-Flash	Impression	3	9	3	1	22
	Video quality	14	12	6	3	3
	UI	6	15	5	4	8
Prototype-HTML	Impression	4	7	10	5	12
	Video quality	2	5	7	6	18
	UI	6	13	8	2	9
Prototype-J2ME	Impression	10	4	6	5	13
	Video quality	4	6	5	10	13
	UI	7	6	0	6	19
Prototype-Android	Impression	1	3	4	2	28
	Video quality	0	0	3	4	31
	UI	3	3	4	1	27

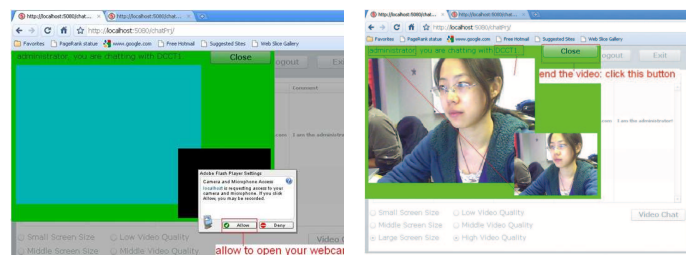


Appendix D

User Guide (Pilot Trial)

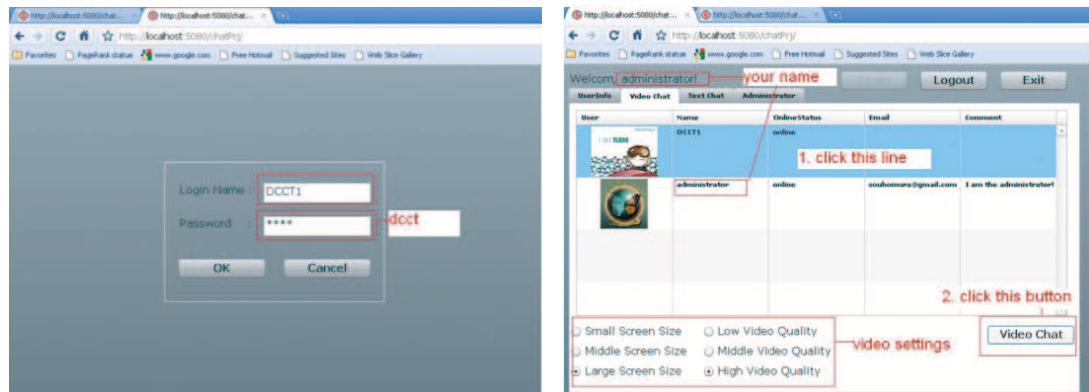


- (a) In the login dialog box, enter the following user name and password: administrator/0000.
- (b) The video settings determine the video quality and the screen size in Figure D.1(e).
- (c) The administrator will see that the whole window is frozen. Please wait for the reply from DCCT1

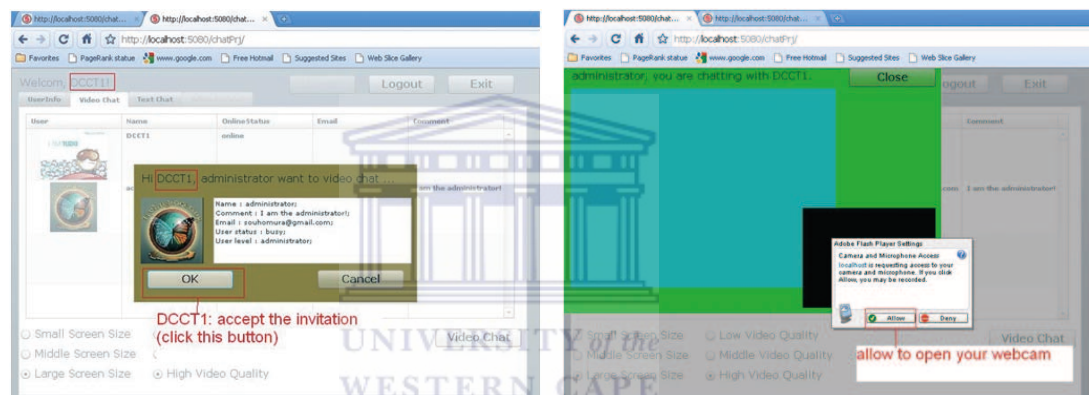


- (d) The video communication will not start unless you and DCCT1 both press the OK button.
- (e) The communication can be ended if you click the close button.

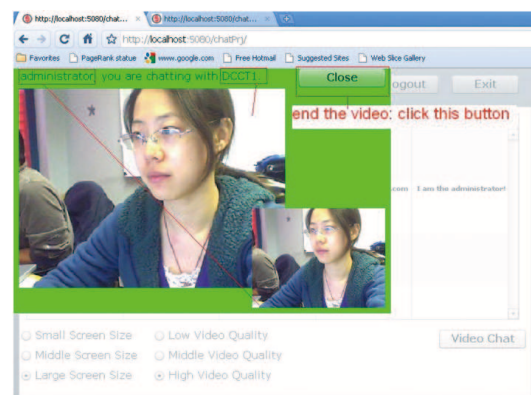
FIGURE D.1: User guide for prototype-Flash (administrator)



- (a) In the login dialog, enter the following user name and password: DCCT1/dcct.
- (b) When DCCT1 logs in the prototype-Flash, administrator switches to the second tab and wait for the administrator.

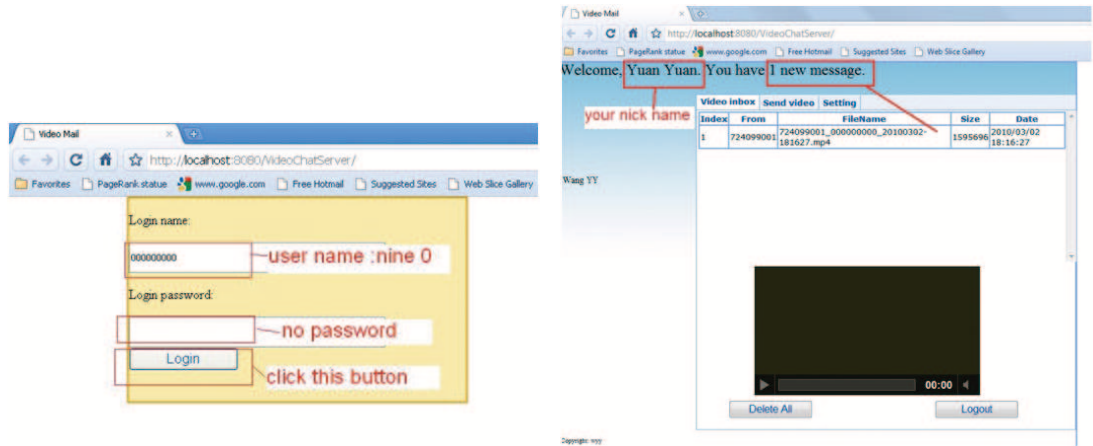


- (c) Then you will see the whole window is frozen and a dialog is popped up. Please press OK button. It is an invitation from administrator to start a video communication.
- (d) The video chat dialog and a small Flash setting dialog will be presented. Please click the OK button on the Flash setting dialog to access to your web-cam. (The video communication will not start unless you and administrator both press the OK button.)



- (e) The communication can be ended if you click close button.

FIGURE D.2: User guide for prototype-Flash (DCCT1)



(a) In the login dialog, enter the following user name:000000000 (nine 0) and press login button. (b) The inbox tab will be shown. You can check the new video messages for you.

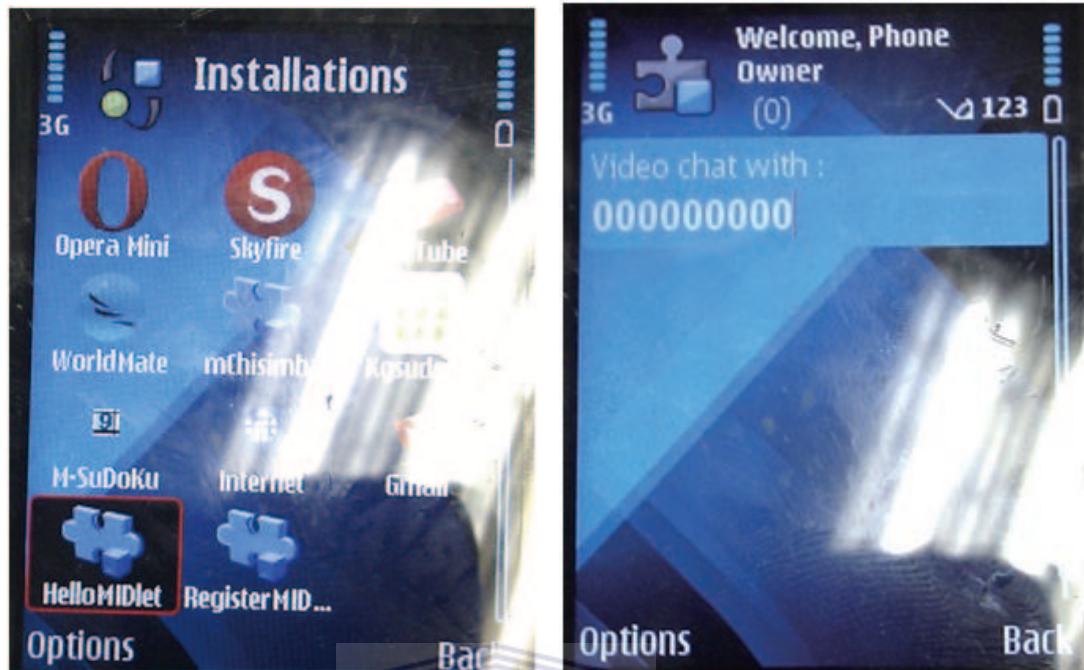


(c) If you have new messages, you can click the line listed out in the table. The thumbnail of the message will be presented in the online video player. Click the play button (the triangle button) and the message will be played. (d) To send a message to other user, please switch to the next tab and do the upload.

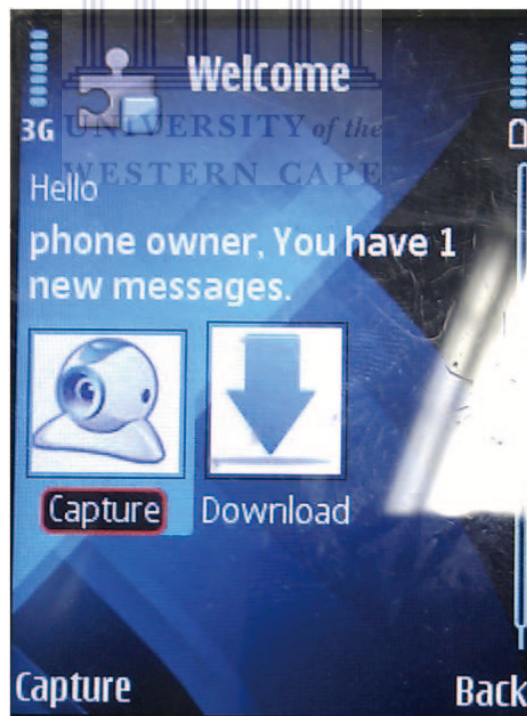


(e)

FIGURE D.3: User guide for prototype-HTML5

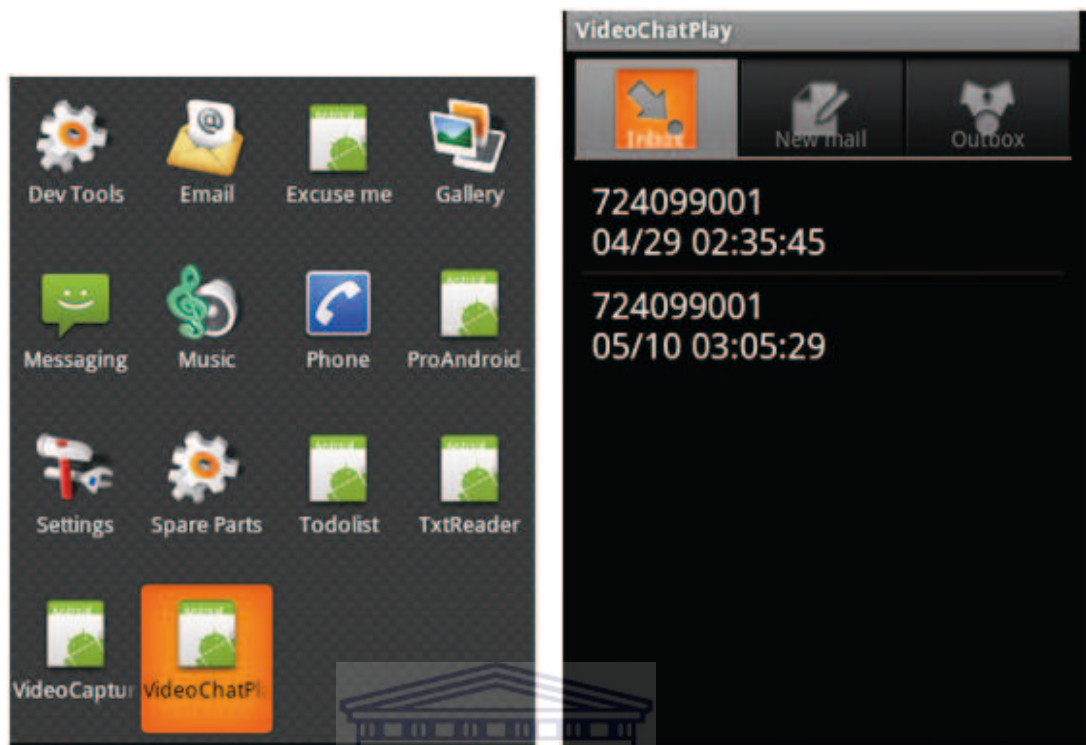


(a) Go to Menu — Installations — HelloMIDlet. (b) Click HelloMIDlet. Then input the phone number 000000000 to start the chat.



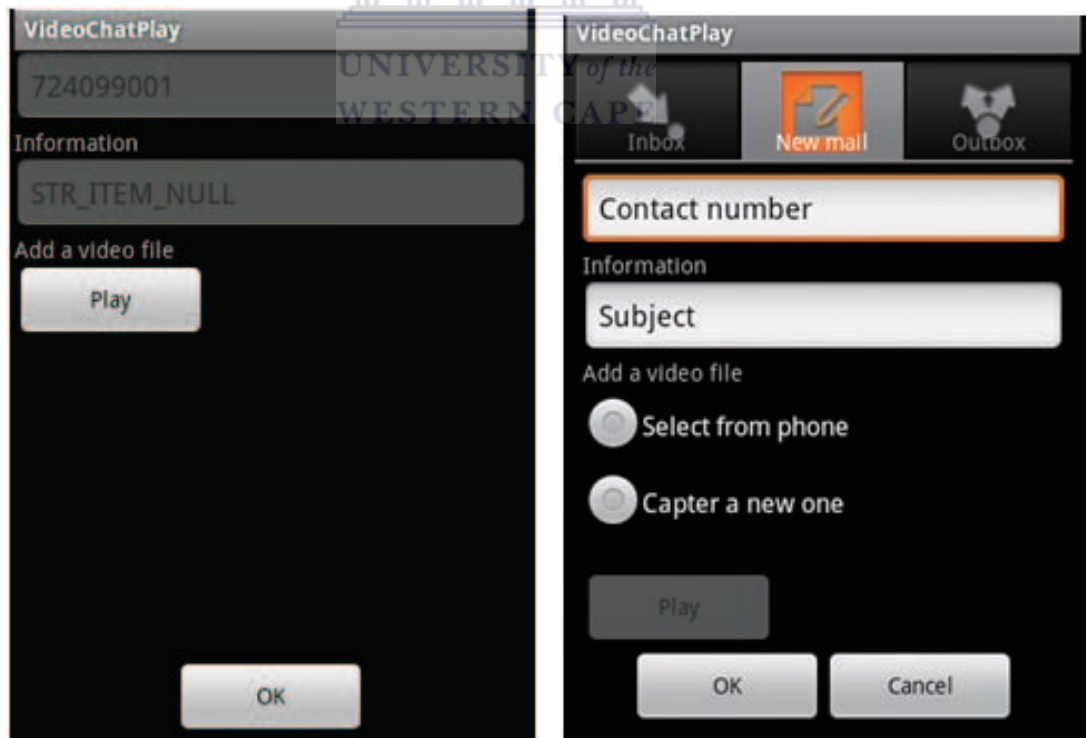
(c) If you have new messages, you can click the line listed in the table. The thumbnail of the message will be presented in the online video player. Click the play button (the triangular button) and the message will be played.

FIGURE D.4: User guide for prototype-J2ME



(a) Go to Application — VideoCapturePlay

(b) Click VideoCapturePlay. The following picture will be shown. The video messages are listed out.

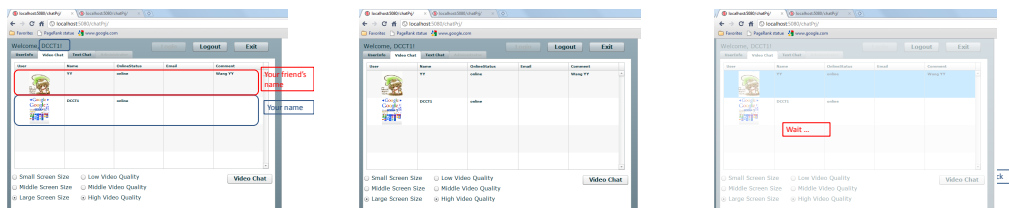


(c) Click the first message to take a look at details. If you want to play video, click Play button. (d) Go to New mail tab to sent video message to others.

FIGURE D.5: User guide for prototype-Android.

Appendix E

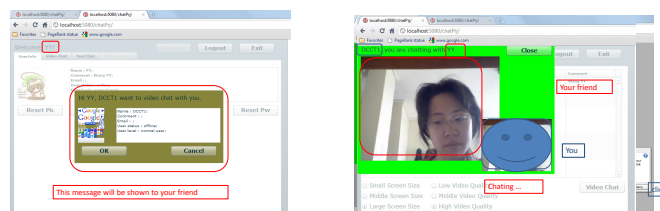
User Guide (Full Trial)



(a)

(b)

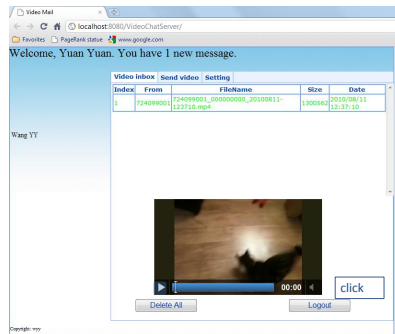
(c)



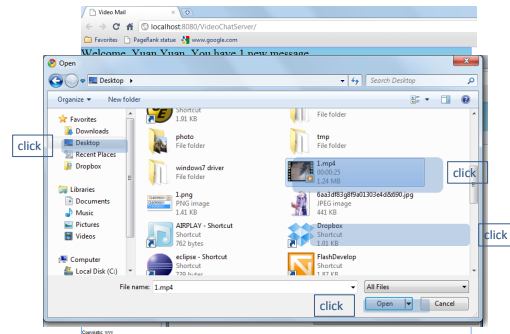
(d)

(e)

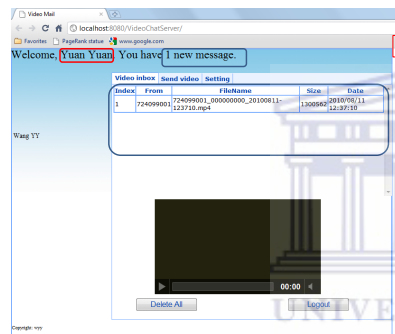
FIGURE E.1: User guide for prototype-Flash: Many animations are included in the document. More details are presented on the CD.



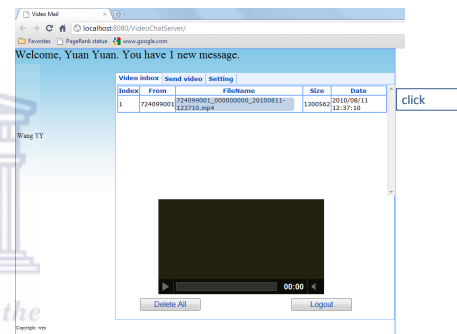
(a)



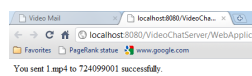
(b)



(c)



(d)



(e)

FIGURE E.2: User guide for prototype-HTML5: Many animations are included in the document. More details are presented on the CD.

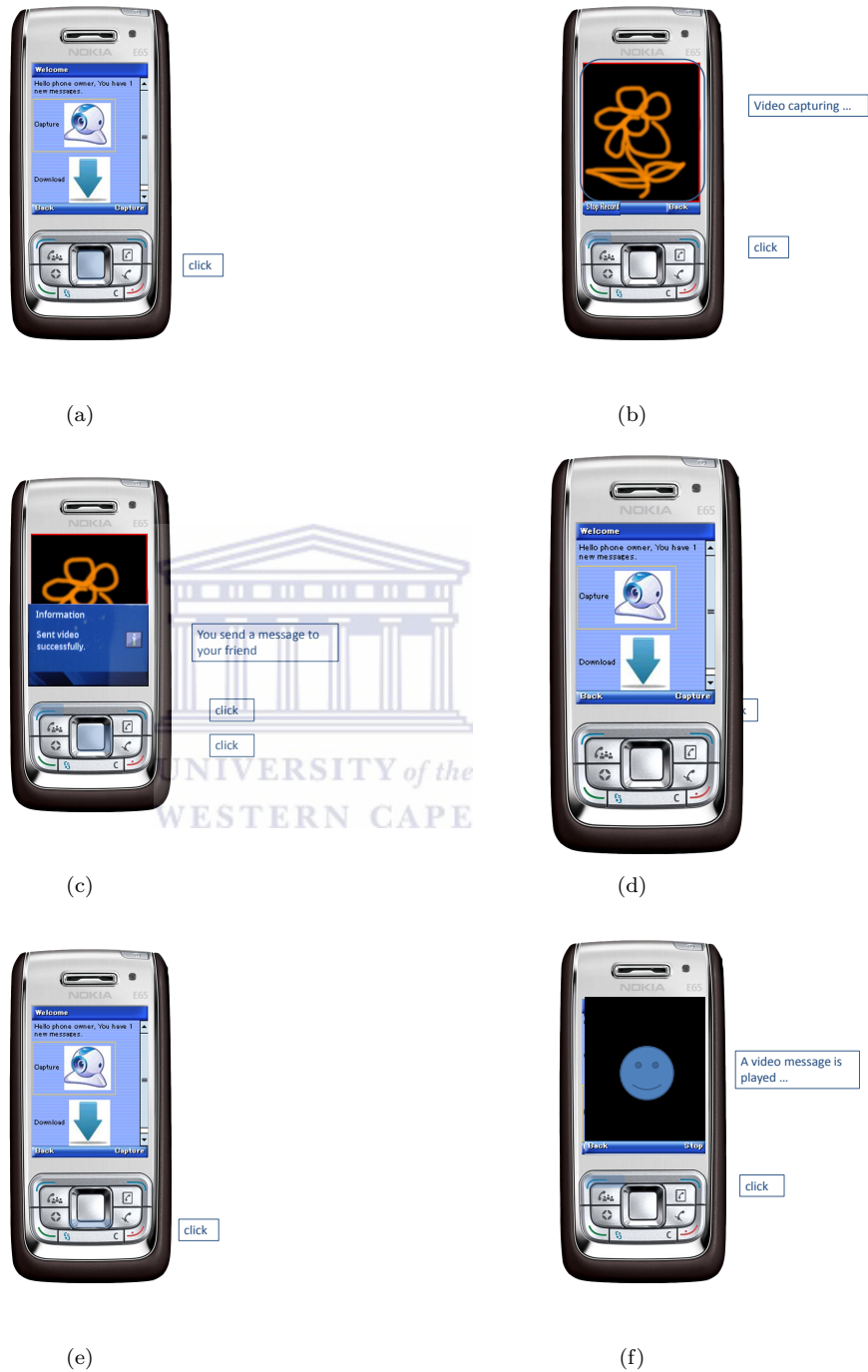


FIGURE E.3: User guide for prototype-J2ME: Many animations are included in the document. More details are presented on the CD.

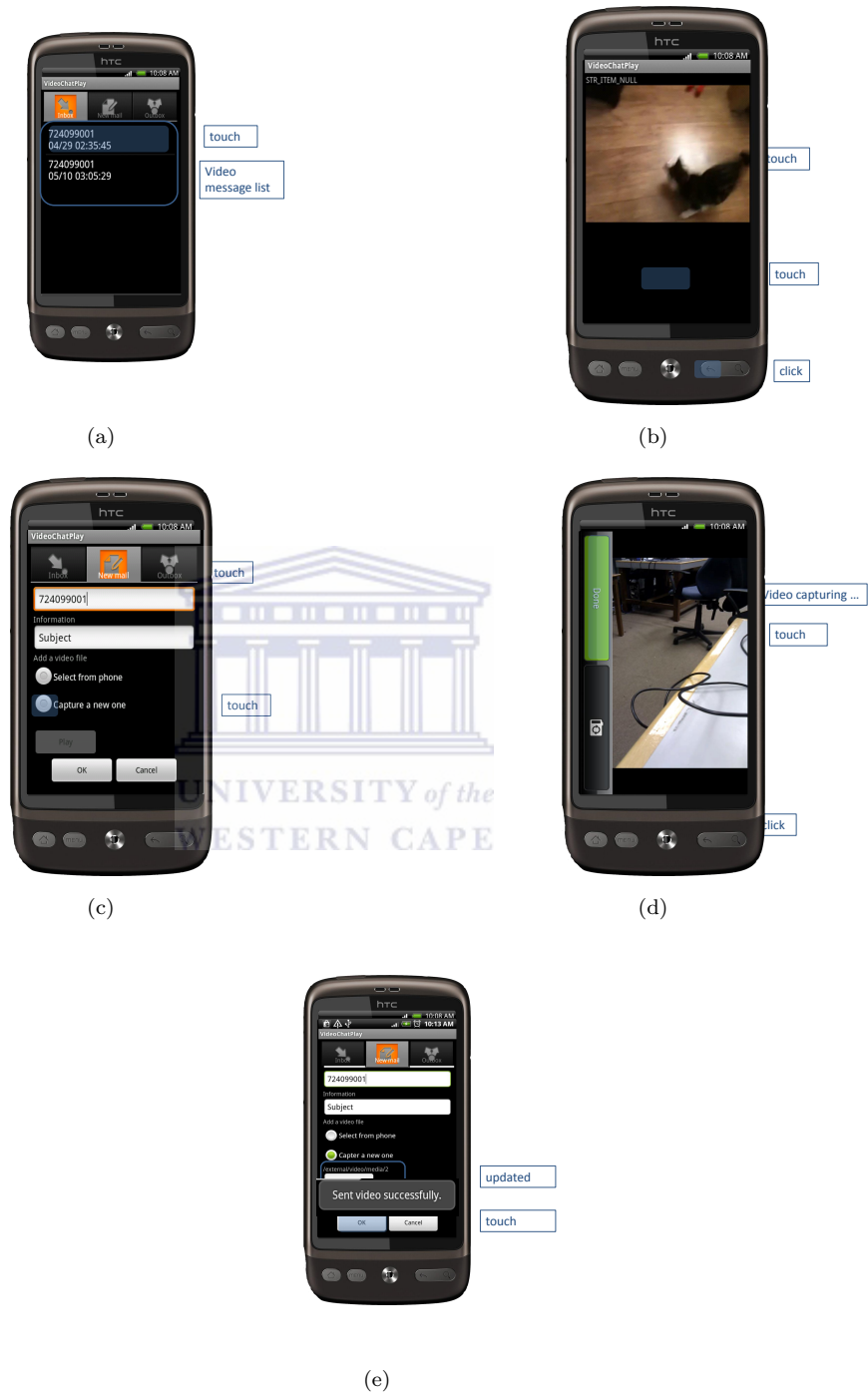


FIGURE E.4: User guide for prototype-Android: Many animations are included in the document. More details are presented on the CD.

Bibliography

- [1] D. Aarons and P. Akach. 6 South African Sign Language: one language or many? *Language in South Africa*, page 127, 2002.
- [2] D. Aarons and M. Glaser. A Deaf adult literacy collective. *Stellenbosch Papers in Linguistics*, 34:1–18, 2002.
- [3] C. Allen and J. Grden. Introducing Red5. *The Essential Guide to Open Source Flash Development*, pages 309–335, 2008.
- [4] J. Bankoski. WebM and VP8 land in Chromium. Technical report, Google, May 2010. blog.chromium.org/2010/05/webm-and-vp8-land-in-chromium.html.
- [5] B. Boehm and W. Hansen. The spiral model as a tool for evolutionary acquisition. *CrossTalk*, 14(5):4–11, 2001.
- [6] I. Bongartz, A.R. Conn, N. Gould, and P.L. Toint. CUTE: Constrained and unconstrained testing environment. *Proc. ACM TOMS*, 21(1):123–160, 1995.
- [7] K.S. Brandsson. Adobe Flex 3 and Actionscript 3.0. files.itslearning.com/data/776/2791/report.pdf, 2008.
- [8] F. Buttussi, L. Chittaro, E. Carchietti, and M. Coppo. Using mobile devices to support communication between emergency medical responders and deaf people. In *Proc. 12th international conference on Human computer interaction with mobile devices and services*, pages 7–16. ACM, 2010.
- [9] K. Carle and C. Zacharias. Introducing Youtube HTML5 supported videos. youtube-global.blogspot.com/2010/01/introducing-youtube-html5-supported.html, Jan 2010.
- [10] A. Cavender. MobileASL: Making cell phones accessible to the Deaf community. 2008. csta.villanova.edu:8080/handle/2378/362.

- [11] A. Cavender, R.E. Ladner, and E.A. Riskin. MobileASL:: intelligibility of sign language video as constrained by mobile phone technology. In *Proc. 8th international ACM SIGACCESS conference on Computers and accessibility*, page 78. ACM, 2006.
- [12] A.C. Cavender, N. Cherniavsky, J. Chon, R.E. Ladner, E.A. Riskin, R. Vanam, and J.O. Wobbrock. MobileASL: Overcoming the technical challenges of mobile video conversation in sign language. students.washington.edu/rahulv/lrec2010.pdf, 2010.
- [13] A.C. Cavender, R.E. Ladner, and E.A. Riskin. MobileASL: Making Cell Phones Accessible to the Deaf Community. mobileasl.cs.washington.edu/downloads/07-08-23-TDI.ppt, Aug 2007.
- [14] K.T. Chen, C.Y. Huang, P. Huang, and C.L. Lei. Quantifying Skype user satisfaction. In *Proc. Applications, technologies, architectures, and protocols for computer communications*, page 410. ACM, 2006.
- [15] M. Chesire, A. Wolman, G.M. Voelker, and H.M. Levy. Measurement and analysis of a streaming-media workload. In *Proc. 3rd conference on USENIX Symposium on Internet Technologies and Systems*, volume 3, page 1. USENIX Association, 2001.
- [16] D.N. Chin. Empirical evaluation of user models and user-adapted systems. *User modeling and user-adapted interaction*, 11(1):181–194, 2001.
- [17] M.F. Cohen. *An introduction to logic and scientific method*. Read Books, 2007.
- [18] P.L. Correia and F. Pereira. Objective evaluation of video segmentation quality. *IEEE Transactions on Image Processing*, 12(2):186–200, 2003.
- [19] R. Croke. Regulatory imperatives for IP videophones and the national video relay Service. Master’s thesis, University of Illinois at Urbana-Champaign, 2007. pactlab.spcomm.uiuc.edu/blog/ryancroke/thesisryancrokeFINAL.doc.
- [20] J. Dalrymple. Apple shows off Safari’s HTML 5 support. Technical report, Macworld, Mar 2009. www.pcworld.com/businesscenter/article/160934/apple_shows_off_safaris_html_5_support.html.

- [21] G.B. Dantzig. *Linear programming and extensions*. Princeton University Press, 1998.
- [22] D. Ding, J. Yang, Q. Li, W. Liu, and L. Wang. What can expressive semantics tell: Retrieval model for a flash-movie search engine. *Lecture notes in computer science*, 3568:123–133, 2005.
- [23] C. Doctorow. Students create video-chat program for deaf kids. <http://www.boingboing.net/2010/06/21/students-create-vid.html>, 2010.
- [24] B. Dougherty. Try our new HTML5 player. Technical report, Vimeo, Jan 2010. vimeo.com/blog:268.
- [25] K. Eilers-crandall and C. Aidala. Distance learning opportunities for Deaf learners. In *Proc. SSSConference Proceedings*, page 139, 2000.
- [26] F.H.P. Fitzek and M. Reisslein. MPEG-4 and H. 263 video traces for network performance evaluation. *IEEE Network*, 15(6):40–54, 2002.
- [27] E. Fossey, C. Harvey, F. McDermott, and L. Davidson. Understanding and evaluating qualitative research. *Australian and New Zealand Journal of Psychiatry*, 36(6):717–732, 2002.
- [28] H. Frowein, H. Kamphuis, and E. Rikken. Sign language interpretation via mobile videotelephony. In *Proc. 18th International Symposium on Human Factors in Telecommunication, Bergen, Norway*, pages 5–7, 2001.
- [29] J.J. Garrett. Ajax: A new approach to web applications. www.adaptivepath.com/ideas/essays/archives/000385.php, 2005.
- [30] J. Ghosh and R. Cameron. A quick tour of Silverlight 3 development. *Silverlight Recipes*, pages 1–29, 2009.
- [31] M. Glaser and W.D. Tucker. Web-based telephony bridges for the Deaf. In *Proc. SATNAC 2001, Wild Coast Sun, South Africa*, pages 472–677, 2001.
- [32] M. Glaser and W.D. Tucker. Telecommunications bridging between Deaf and hearing users in South Africa. *Proc. CVHI 2004, Granada, Spain*, 2004.
- [33] Google. *Chromium browser vs Google Chrome*, Aug 2010. www.google.com/support/forum/p/Chrome/thread?tid=380b7ffc5d845696&hl=en.

- [34] Google. Issue 21318: query FFmpeg libraries for codec support. Technical report, Google, Sep 2010. code.google.com/p/chromium/issues/detail?id=21318.
- [35] S.R. Gulliver and G. Ghinea. How level and type of deafness affect user perception of multimedia video clips. *Universal Access in the Information Society*, 2(4):374–386, 2003.
- [36] D. Hachamovitch. HTML5 video. blogs.msdn.com/b/ie/archive/2010/04/29/html5-video.aspx, Apr 2010.
- [37] P. Jägenstedt. (re-)Introducing <video>— Official blog for core developers at Opera. Technical report, Opera, Dec 2009. my.opera.com/core/blog/2009/12/31/re-introducing-video.
- [38] M. Jain and C. Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput. *ACM SIGCOMM Computer Communication Review*, 32(4):295–308, 2002.
- [39] A. Jordan. Effective implementation of interpreting services in an emergency department: Including deaf patients in a major source of public health care. In *Proc. 130th Annual Meeting of APHA*, 2002. apha.confex.com/apha/130am/techprogram/paper_39531.htm.
- [40] P.C. Jorgensen and C. Erickson. Object-oriented integration testing. *Communications of the ACM*, 37(9):30–38, 1994.
- [41] S. Jumisko-Pyykkö and J. Hänninen. Evaluation of subjective video quality of mobile devices. In *Proc. 13th annual ACM international conference on Multimedia*, pages 535–538. ACM, 2005.
- [42] M. Ketterl, R. Mertens, and O. Vornberger. Vector Graphics for Web Lectures: Experiences with Adobe Flash 9 and SVG. *Interactive Technology and Smart Education*, 4(4):182–191, 2008.
- [43] P. Ladd. *Understanding deaf culture: In search of deafhood*. Multilingual Matters Ltd, 2003.
- [44] B. Lesser, G. Guilizzoni, R. Reinhardt, J. Lott, and J. Watkins. *Programming flash communication server*. O’Reilly Media, Inc., 2005.

- [45] J. Leyden. BT trials mobile SMS to voice landline. www.theregister.co.uk/2004/01/08/bt_trials_mobile_sms/, 2004.
- [46] D. Loguinov and H. Radha. Measurement study of low-bitrate internet video streaming. In *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 281–293. ACM, 2001.
- [47] Z. Ma and W.D. Tucker. Asynchronous video telephony for the Deaf. In *Proc. SATNAC2007*, pages 134–139, 2007.
- [48] Z. Ma and W.D. Tucker. Adapting x264 to asynchronous video telephony for the Deaf. In *Proc. SATNAC2008*, pages 127–132, 2008.
- [49] Q.H. Mahmoud. *MMAPI 1.1, MIDP 2.0 media API, and advanced multimedia supplements*. Sun, Jan 2005. developers.sun.com/mobility/midp/articles/mmapioverview/.
- [50] I. McClelland. Product assessment and user trials. *Evaluation of Human Work: A Practical Ergonomics Methodology*, page 249, 1995.
- [51] H. McCracken. Microsoft previews the revamped Internet Explorer 9 platform. technologizer.com/2010/03/16/ie9-platform-preview/, Mar 2010.
- [52] R. Meier. *Professional Android application development*. Wiley-India, 2008.
- [53] C. Mills. Opera supports the WebM video format. Technical report, Opera, May 2010. dev.opera.com/forums/topic/576501.
- [54] B. Morris. *The Symbian OS architecture sourcebook: design and evolution of a mobile phone OS*. Wiley, 2007.
- [55] Mozilla. *Mozilla Firefox 3.5 release notes*, Jun 2009. www.mozilla.com/en-US/firefox/3.5/releasenotes/.
- [56] Mozilla. Bug 566243 - merge mozilla-webmedia repository to mozilla-central. Technical report, Mozilla, Aug 2010. bugzilla.mozilla.org/show_bug.cgi?id=566243.
- [57] B.S. Ngubane. For the launch of the pilot project: Telephone interpreting services for South Africa (TISSA). May 2002. www.dacst.gov.za/speeches/minister/mar2002/tissa_launch.htm.

- [58] J.J. Noble and T. Anderson. *Flex 3 Cookbook*. O'Reilly Media, 2008.
- [59] A. Norton. Enabling the disabled: word to mouth: talkingSMS is a valuable tool for the hearing and speech-impaired. *Winter 2002*, page 42, 2002.
- [60] C. Padden. *The Deaf community and the culture of Deaf people*, page 343. RoutledgeFalmer, 2000.
- [61] L.D. Paulson. Building rich web applications with Ajax. *IEEE Computer*, 38(10):14, 2005.
- [62] J. Penton, W.D. Tucker, and M. Glaser. Telgo323: An H. 323 bridge for Deaf telephony. In *Proc. SATNAC2002*, pages 309–313, 2002.
- [63] S. Pfeiffer and C. Parker. Accessibility for the HTML5 <video> element. In *Proce. W4A2009*, pages 98–100. ACM, 2009.
- [64] M.R. Power, D. Power, and L. Horstmanshof. Deaf people communicating via SMS, TTY, relay service, fax, and computers in Australia. *Journal of Deaf Studies and Deaf Education*, 12(1):80, 2007.
- [65] D. Richardson. *Foundation ActionScript 3.0 for Flash and Flex*. Friends of ED, 2009. www.friendsofed.com/samples/9781430219187.pdf.
- [66] W.B. Sanders and C. Cumararatunge. *ActionScript 3.0 Design Patterns*. O'Reilly, 2007.
- [67] D.G. Schmitt and M.L. Slowiaczek. An experimental study of synthesized speech intelligibility using text created by telecommunication device for the deaf (TDD) users. In *Proc. IEEE GLOBECOM*, pages 996–999. IEEE, 2002.
- [68] S. Schwarz and T. Roth-Berghofer. Towards goal elicitation by user observation. In *Proc. LLWA2003*, 2003.
- [69] C.B. Seaman. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on Software Engineering*, 25(4):557–572, 1999.
- [70] S. Sofaer. Qualitative research methods. *International Journal for Quality in Health Care*, 14(4):329, 2002.
- [71] K. Sripanidkulchai, B. Maggs, and H. Zhang. An analysis of live streaming workloads on the Internet. In *Proc. 4th ACM SIGCOMM conference on Internet measurement, Portland, United States of America*, pages 41–54. ACM, 2004.

- [72] Y. Stavrakas, M. Gergatsoulis, and P. Rondogiannis. Multidimensional XML. *Distributed Communities on the Web*, pages 100–109, 2000.
- [73] G.J. Sullivan, P. Topiwala, and A. Luthra. The H. 264/AVC advanced video coding standard: Overview and introduction to the fidelity range extensions. In *Proc. 27th Applications of Digital Image Processing*, volume 5558, page 53.
- [74] W.D. Tucker. *Softbridge: a socially aware framework for communication bridges over digital divides*. PhD thesis, University of the Western Cape, 2009. pubs.cs.uct.ac.za/archive/00/00/05/24/.
- [75] A. van Kesteren and D. Jackson. *The xmlhttprequest object*. W3C, 2007. www.w3.org/TR/XMLHttpRequest/.
- [76] R. Vanam, J. Chon, E.A. Riskin, R.E. Ladner, F.M. Ciaramello, and S.S. Hemami. Rate-distortion-complexity optimization of an H.264/AVC encoder for real-time videoconferencing on a mobile device. mobileas1.cs.washington.edu/downloads/vpqm2010_CiaramelloVanam.pdf, 2010.
- [77] Y. Wang, M. Claypool, and Z. Zuo. An empirical study of realvideo performance across the internet. In *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 295–309. ACM, 2001.
- [78] A.A. Webster, C.T. Jones, M.H. Pinson, S.D. Voran, and S. Wolf. An objective video quality assessment system based on human perception. In *Proc. SPIE1993*, volume 1913, pages 15–26, 1993.
- [79] E.J. Weyuker and F.I. Vokolos. Experience with performance testing of software systems: issues, an approach, and case study. *IEEE Transactions on Software Engineering*, 26(12):1147–1156, 2002.
- [80] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Ttechnology*, 13(7):560–576, 2003.
- [81] F. Williams and P.R. Monge. *Reasoning with statistics: How to read quantitative research*. Harcourt College Publishers, 2001.
- [82] W.W. Woelders, H.W. Frowein, J. Nielsen, P. Questa, and G. Sandini. New developments in low-bit rate videotelephony for people who are deaf. *Journal of Speech, Language and Hearing Research*, 40(6):1425, 1997.

- [83] L. Yi and W.D. Tucker. Automatic voice relay with open source Kiara. In *Proc. SATNAC2009, Swaziland*, pages 143–148, 2009.
- [84] X. Zhang, J.L. Freschl, and J.M. Schopf. A performance study of monitoring and information services for distributed systems. In *Proc. 12th HPDC2003*).

