MOBILE VOICE TO SIGN LANGUAGE SYSTEM

by

Mohd Izuan Ahamad
8269

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Technology (Hons)
Information and Communication Technology

JANUARY 2008

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Mobile Voice to Sign Language

by

Mohd Izuan Ahamad

A project dissertation submitted to the
Information Technology Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION AND COMMUNICATION TECHNOLOGY)

Approved by,

____

(Low Tan Jung)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
January 2008

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

[illegible]

MOHD IZUAN AHAMAD

-ABSTRACT-

'Mobile Voice to Sign Language System'

Mohd Izuan Ahamad (8269)
Universiti Teknologi PETRONAS
Bandar Seri Iskandar, 31750 Perak Darul Ridzuan, Malaysia
wanc24@gmail.com

This report presents recent technologies using Mobile devices as a medium to interact between normal people and hearing disables people. This system is using Java mobile technologies to do voice processing on a mobile platform. This system will allow users to capture the voice match with appropriate sign. Voice recognition to control devices such as a robot already implemented in Java. Here some ideas to implement by using J2ME which is for small mobile devices to make it valuable for current situation or technology. There are at least 29 million people around the world who suffer from speech and hearing disabilities. It is somehow difficult for us to interact with them because of the unknown language used by them to communicate with each other. Sign language is a form of communication which is widely used by deaf and mute peoples. Thus, the only way of communication is learning their language which is sign language. As in the case in verbal language, sign language is differs from one region to another. However, when people using different signed languages meet, communication is significantly easier than when people of different language meet. Sign language, in this respect, gives access to an international deaf community to communicate. This report contains solution whereby one does not need to learn sign language to be able to communicate with the disabled. This system will convert the English language to sign language.

ACKNOWLEDGEMENT

First of all, I would like to thank to my supervisor, Mr. Low Tan Jung for his excellent guidance, courage and advised upon the completion this project.

Special thanks also go to my friends who help me to find material for developing this application and my colleagues who give a lot of support for this project

Last but not least my beloved family for their understanding and support during the completion of this project

Thank You
Mohd Izuan Ahamad
8269, Information Communication Technology
April 10, 2008

Table of Contents

List of Figures

List of Tables

List of Abbreviations

MV2S	Mobile Voice to Sign Language
J2ME	Java 2 Micro Edition
J2SE	Java 2 Standard Edition
FYP	Final Year Project
MIDP	Mobile Information Devices Profiles
JVM	Java Virtual Machine
CDLC	Connected Devices Limited Configuration
ASL	American Sign Language
PDA	Personal Digital Assistant
API	Application protocol interfaces
JSML	Java Speech Markup Language
XML	Extensible Markup Language
JSGF	Java Speech Grammar Format

CHAPTER 1
INTRODUCTION

1.1 Background of Study

This project will investigates the ways of communication and interaction between a normal people with a person with hearing (deaf) and speech (mute) disabilities using the speech recognition technology. From the existing statistics shows that, there are a lot of numbers people having these problems. Rapid development of mobile technology makes our world more interesting and a lot of peoples demanding on a new products. However, there are no such mobile products for peoples to learn sign language. Speech recognition technology provides the ability for computer to listen from user input (voice) and determine what have been said. That technology already implemented in Personal Computer and this project will transfer that kind of technology to the mobile devices using limited memory and processor.

1.2 Problems statement

Some of the peoples around the world may be born with lack of hearing or speech abilities. It is sometimes difficult to communicate between normal people and not every people understand sign language. It might be difficult for everyone to learn the sign language in a short period of time. Other than that, the cost for learning might be high and sometimes not every people can afford to learn sign languages. Current technology, there have already an application for peoples to learn sign language, but the application can only be installed in computer. Nowadays, the explosive growth of mobile technology, a lot of peoples around the world will prefer using mobile devices is most practical and easy to bring anywhere (mobility).

1.2.1 Problems with current system

There are some systems such as a translator designed for disabled only for learning purposes. It might not help people to communicate between all of them or normal people. This *Mobile Voice to sign system* is enhancement of the previous project which was developed by FYP student (July 2006). The old system designed using matlab and installed on personal computer attached with web camera. This new system will be using mobile phones as a platform and using J2ME technology. However, this system might have a limitation such as a memory and the processing speed might be slower than using personal computer.

1.3 Objective and scope of study

The objectives of this system are:

- To allow peoples learn basic sign language.
- To enable those who suffer from sudden lost of hearing/ speech abilities to learn sign-language effectively and within a short period of time.
- To reduce the discrimination with the disabled, by providing more interaction between the normal and the disabled
- To eliminate the needs of going to high cost sign-language
- To make the devices portable and people can bring anywhere they want to use as a medium to communicate.

The main scope is for this project is to create system that is able to capture the voice or speech and translate them into appropriate sign on real time. The mobile devices will be able to store the voice recorded by user and can match the voice with the sign language.

1.4 Advantages of MIDP

Mobile Information Device Profile (MIDP) comes at critical time, a time when new technologies growing like mobile phones and exploding the market.

Simultaneously, MIDP devices are achieving the kind of processing power, memory availability, and Internet connectivity that makes them an attractive platform for mobile networked applications. MIDP is already deployed on millions of handsets all over the world. There are a lot of advantages using MIDP. There are:

- **Portability**

  The advantages of using Java over using other tools for small device application development are portability. We can write device applications with C or C++, but the result would be specific to a single platform. An application written suing MIDP APIs will be directly portably to any MIDP device.

- **Security**

  A second compelling reason for using java for small device development is security. Java is well known for its ability to safely run downloaded code like applets. This is a perfect fit and easy while nifty application dynamically downloading to the mobile phone. Another thing, the Java Virtual Machine(JVM) used in the CLDC only implements a partial *bytecode* verifier, which means that part of the important task of *bytecode* verification must be performed off the MIDP device. Other than that, CLDC does not allow for application- defined *classloaders*. This means that most dynamic application delivery is dependent on device-specific mechanisms.

CHAPTER 2
LITERATURE REVIEW

Sign Language and Mobile technology is a part of this research. This section will be divided into many parts to create better understanding of the study.

2.1 Sign Language

A sign language (also signed language) is a language which uses manual communication, body language and lip patterns instead of sound to convey meaning-simultaneously combining hand shapes, orientation and movement of the hands, arms or body, and facial expressions to express fluidly a speaker's thoughts. Sign languages commonly develop in deaf communities, which can include interpreters and friends and families of deaf people as well as people who are deaf or hard of hearing themselves.

The recorded history of sign language in Western society extends from the 16th century. In 1755, Abbé de l'Épée founded the first public school for deaf children in Paris; Laurent Clerc was arguably its most famous graduate. He went to the United States with Thomas Hopkins Gallaudet to found the American School for the Deaf in Hartford, Connecticut.[9] Gallaudet's son, Edward Miner Gallaudet founded the first college for the deaf in 1857, which in 1864 became Gallaudet University in Washington, DC, the only liberal arts university for the deaf in the world.

2.2 American Sign Language (ASL)

ASL is a natural language as proved to the satisfaction of the linguistic community by William Stokoe, and contains phonology, morphology, semantics, syntax and pragmatics just like spoken languages. It is a *manual language* or *visual language*, meaning that the information is expressed not with combinations of sounds but with combinations of handshapes, palm orientations, movements of the hands, arms and body, location in relation to the body, and facial expressions. While spoken languages are produced by the vocal cords only, and can thus be easily written in linear patterns, ASL uses the hands, head and body, with constantly changing movements and orientations. Like other natural sign languages, it is "three dimensional" in this sense. [1][2] ASL is used natively and predominantly by the Deaf and hard-of-hearing of the United States and Canada.

2.3 Mobile Wireless Technology

Wireless computing is currently the cream filling in the technology applications in e-learning (Sbihli, 2002). Mobility is the most important feature of a wireless communication system. Currently wireless communication is one of the fastest growing segments in industry (Rogers and Edwards, 2003). Wireless networks are the pathways along which e-learning travels. Without a network it would not be possible to read a file from a server, share documents with remote customers, or send and receive e-mails (Jamlipour, 2003). Wireless communication offers the flexibility for users to access the geographical coverage of wireless systems. Nevertheless, it is limited: wireless networks use a fixed infrastructure for central administration (Mark and Zhuang, 2003).

Wireless communication systems, such as cellular, PDAs (Personal Digital Assistant),satellite phones as well as wireless local area networks, have widespread use and have become an essential tool in many people's everyday life, both professional and personal (Sbihli, 2002). PDAs and cellular systems are omni directional in design, permitting service for many different users within a specified area. Microwaves involve unidirectional or point-to-point technology; unidirectional wireless technologies transmit only in one direction at one time (Rogers and Edwards, 2003). In multiple-access communication systems, which refer to mobile devices, the goals are universal coverage, quality of service similar to that of traditional service, low equipment cost for providers and end users, and lower number of cell sites required to achieve maximum penetration while maintaining quality of service (Dubendorf, 2003). Wireless computing technology benefits e-learning by increased customer strength, decreased cost and improved customer service. Wireless data networks must be able to overcome challenges like coverage, reliability, standards, speed and cost (Vacca, 2002).

2.4 Mobile Application

Mobile Applications are applications that use the Java™ Micro Edition (Java™ ME) platform, a set of technologies and specifications developed for small devices like pagers, mobile phones, and set-top boxes. Java ME uses smaller-footprint subsets of Java SE components, such as smaller virtual machines and leaner APIs, and defines a number of APIs that are specifically targeted at consumer and embedded devices.

The NetBeans Mobility Packs support the two base configurations of the Java ME platform, CLDC and and CDC. The Connected, Limited Device Configuration (CLDC) is for small wireless devices with intermittent network connections, like mobile phones, and personal digital assistants (PDAs). The Mobile Information Device Profile (MIDP), which is based on CLDC, was the first finished profile and thus the first finished Java ME application environment. MIDP-compliant devices are widely available worldwide.

The Connected Device Configuration (CDC) is for larger devices (in terms of memory and processing power) with robust network connections, such as set-top boxes, Internet appliances, and embedded servers.

2.5 Java 2 Micro Edition (J2ME)

Java 2 Micro Edition(J2ME) targets a variety of devices with limited memory and display categories PDAs, tablets, cell phones and household appliances. Once Java 2 Micro Edition(J2ME) is embedded or downloaded onto a device it can run all of the applications within the J2ME library. So, an email application written on J2ME could run on Palm, a Motorola Phone or an Amana microwave, assuming that the device is J2ME enabled. During the year 1999 Java introduced Virtual Machines that fit inside a wide range of computing devices, libraries and APIs that are specialized for each kind of computing devices, libraries and APIs that are specialized for each kind of computing device and tools for development and devices configuration.

J2ME became very popular because of two important issues:

- Modularity and customizability was essential requirements
- J2ME provides a range of virtual machines, each optimized to the different processor types and memory footprints commonly found in the consumer and embedded marketplace.

To support this flexibility, two concepts were essential configurations and profiles. A configuration defines a minimum platform for a horizontal class of devices, with similar memory and processing capacities. A profile defines a Java platform for a specific vertical market segment. Each profile is layered on top of configuration. So, a configuration defines a broad horizontal device family, and a profile defines a specialized device category in that family. An application written to a specific configuration/profile guarantees portability across all devices included in that configuration/profile association.

2.6 Speech Recognition in Mobile Devices

Research conducted by Gartner Dataquest in 2001 [9], predicted that by 2005 there would be 780million mobile phone and PDAs being shipped worldwide. Given these statistics, it is reasonable to assume that both mobile phones and Personal Digital Assistants play an important part in many people's daily lives. The ever growing power of these mobile devices, coupled with the provision of wireless and Bluetooth networking capabilities, opens the door for a host of new applications to be developed and for old application paradigms to be applied on new frontiers. With the evolution of the Java platform and language, Java has become suitable for developing speech applications which can command and control mobile devices. The Java Speech API [4], specifically the Cloudgarden implementation, provides speech recognition and synthesis tools for Java applications. The Java Speech Markup Language (JSML) [5], which is a subset of XML ,and the Java Speech Grammar Format (JSGF) [4] offer simple but powerful development structures for speech synthesis development and recognition grammars respectively.

This research project is concerned with building a framework with applications for command and control of a remote device by voice activation with speech recognition from a local control station over a wireless network in three different scenario configurations. In each scenario Java plays a critical role. The first of these scenarios or configurations is based on a PC workstation under the Windows operating system connected via a wireless network to a PC-based server. The server issues commands to a remote device. The second scenario will involve developing a distributed speech recognition engine, between an iPaq pocket PC and a PC based server which will issue the commands to the remote device. The third scenario will involve a mobile device, specifically an iPaq Pocket PC that will connect over a wireless network to a PC-based server. The server will, again, issue commands to the remote device. For purposes of this research project the remote device will be a Java controlled Lego MindStorms robot that will move and undertake certain actions under instructions relayed to it over a wireless interface. The robot could be replaced with practically any computing or electronic device which has a Java Virtual Machine installed. As part of this research it is our intention to develop a speech recognition engine using the Java programming language. The Java Media Framework in particular can facilitate the signal processing prerequisites of a speech engine.

2.6.1 Speech Recognition Process

To perform speech recognition, a signal must first be acquired, a digital sample of the signal is then taken and the signal is analyzed to extract the features of the speech. Traditionally speech recognition and synthesis systems have been developed using low level languages such as C. The Java Speech API achieves its goal of platform independence by leaving the implementation of the speech engine to the underlying operating system. However the current version of the Java platform has the necessary tools needed to develop a speech recognition engine based on the process as below:

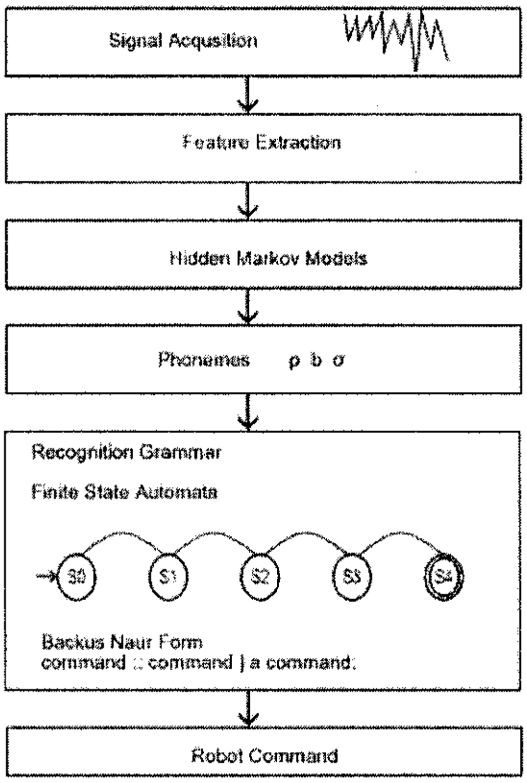


Figure 2.1: Speech recognition Process

2.6.2 Speech Recognition Engine

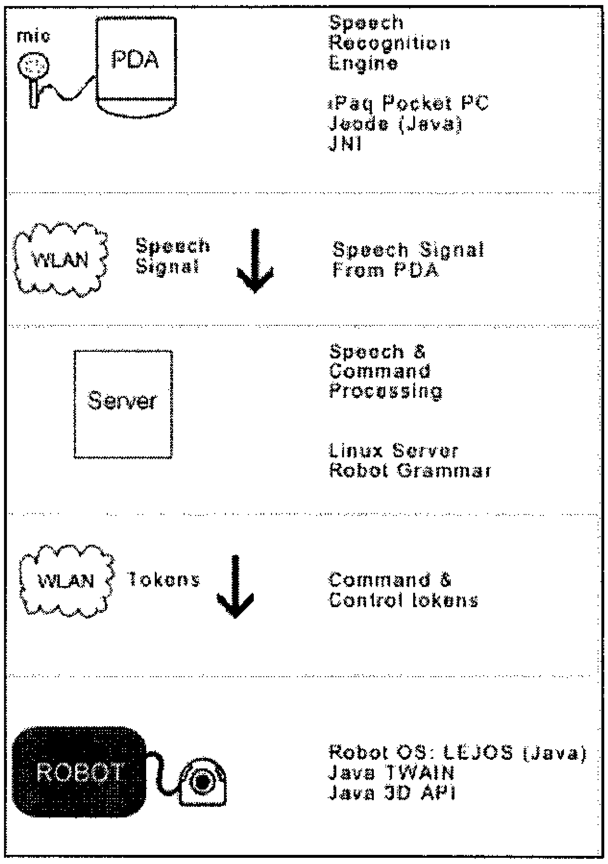


Figure 2.2: Speech Recognition Engine

The Java Speech API is used to provide a platform independent speech recognition and synthesis interface for Java applications. Sun supply a reference standard for the Java Speech API, but they do not provide an implementation. The API has additional requirements of a Microsoft SAPI 4/SAPI 5 or IBM via Voice speech engine on the Windows operating system. While these engines satisfy some of the needs in this project, an engine for the Linux operating system and for the mobile device are still required. Via Voice for Linux has recently been discontinued by IBM.

2.7 Voice Processing-Recognition

Many mobile phones now have voice activate phone books and dial commands. They are particularly recommended for in-car use. Natural language voice recognition for typing text is well established on PCs allowing users, after some initial training of the software, to speak continuously in their natural voice. As the processors within mobile devices become more powerful and yet even smaller, it will be possible to have the same functionality and degree of accuracy with mobile devices. However, the natural language voice recognition software does not have to be resident in the mobile device, it can be a service run from the mobile operator's server. This method of using 'voice portals' is likely to appear first.

Nuance, a Californian company, is a leader in the field of voice recognition. It's customers include American Airlines, Odeon Cinemas and Charles Schwab. With Unisys it has developed a voice activated and speaking yellow pages system in the Minneapolis area. Users can say commands like "*give me a list of pizza restaurants*". Video recorder commands like *back, re-wind* can be given. Having selected an entry, the user can then command the system to dial the number.

2.8 Java Speech Markup Language (JSML)

Speech synthesizer provides a computer with the ability to speak. Users and applications provide text to a speech synthesizer, which is then converted to audio. Speech synthesizers are developed to produce natural sounding speech output.

However, natural human speech is a complex process, and the ability of speech synthesizers to mimic human speech is limited in many ways. For example, speech synthesizers do not "understand" what they say, so they say, so they do not always use the right style or phrasing and do not provide the same nuance as people.

The Java Speech Markup Language(JSML) allows applications to annotate text with additional information that can improve the quality and naturalness of synthesized speech. JSML documents can include structural information about paragraphs and sentences. JSML allows control of the production of synthesized speech, including the pronunciation of words and phrases, the emphasis of words (stressing and accenting), the placements of boundaries and pauses, and the control of pitch and speaking rate. Finally, JSML allows markers to be embedded in text and allows synthesizer-specific control.

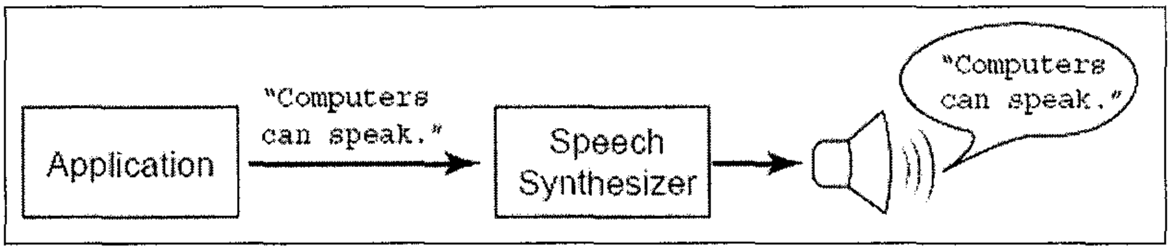


Figure 2.3: Text is converted to the audio output

2.9 Voice Control Application

2.9.1 Voice Signal

Application for controlling particular functions of mobile phone by voice have been developed by company VoiceSignal[19]. VoiceSignal offers a set of various mobile applications[20] utilizing speech recognition or synthesis. Voice enabled applications offer alternatives to basic mobile phone functions such as number dialing by digit or name, sending dictated text messages, reading text messages or search for location, restaurant, ringtone, or other topics. Both speech recognition and synthesis are done mostly in the mobile phone. As it stated on company website:" to be the most effective for the user, and avoid unnecessary delays, interruptions in service, extra connect-time charges and excessive battery drain we believe that most speech solutions should reside on the device". These applications are available for various software and hardware platforms.

2.9.2 Voice Controlled Yahoo Maps for Desktop

This application was to demonstrate voice controlled searching in Yahoo maps from desktop browser. This application consists of servlet and HTML page with JavaScript and Java Applet. Java applet takes care of recording and playing audio and communication with servlet, JavaScript for moving position in map to show requested position(Includes API for Yahoo Maps from Yahoo). Servlet part manages connections and sessions between clients and Voice Extensible Markup Language(VXML) browser. VXML browser is similar to HTML browser but instead of visual interaction it listens(speech recognition) and speaks(text-to-speech(TTS)). Before servlet is ready to accept connections it is necessary to connect VXML browsers to servlet from administration console and then to create any number of services with a VXML document for each browser. This allows basic scaling and configuration when running. It was developed with VXML browser Charlie.

2.9.3 Speech Recognition Using Neural Networks

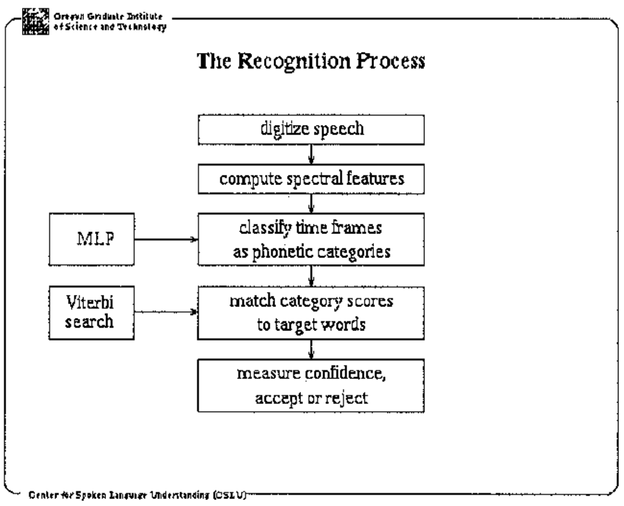

Oregon Graduate Institute of Science and Technology
The Recognition Process
digitize speech
compute spectral features
classify time frames as phonetic categories
MLP
Viterbi search
match category scores to target words
measure confidence, accept or reject
Center for Spoken Language Understanding (CSLU)

Figure 2.4: Diagram of recognition process

According to John-Paul Hosom, Ron Cole, and Mark Fanty(1999),There are four basic steps to performing recognition. The steps are:

I. Digitize the speech that we want to recognize; for telephone speech the sampling rate is 8000 samples per second.

II. Compute features that represent the spectral-domain content of the speech (regions of strong energy at particular frequencies). These features are computed every 10 msec, with one 10-msec section called a *frame.*

III. A neural network (also called an ANN, multi-layer perceptron, or MLP) is used to classify a set of these features into phonetic-based categories at each frame.

IV. A Viterbi search is used to match the neural-network output scores to the target words (the words that are assumed to be in the input speech), in order to determine the word that was most likely uttered.

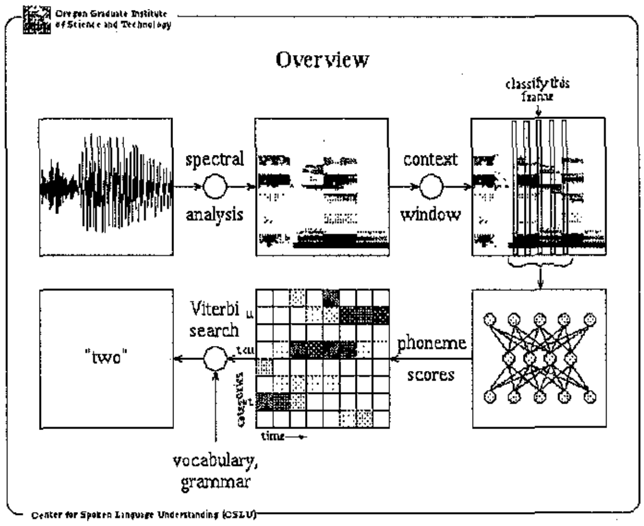


Figure 2.5: Graphical Overview of Recognition Process

The digitized waveform is converted into a spectral-domain representation; one of two sets of features may be used, depending on the recognizer. For the current general-purpose recognizer, use twelve mel-frequency cepstral coefficients (MFCC coefficients), twelve MFCC delta features that indicate the degree of spectral change, one energy feature, and one delta-energy feature (for a total of 26 features per frame). Cepstral-mean-subtraction (CMS) of the MFCC coefficients is done to remove some of the effects of noise. For the latest digit recognizer (to be released with version 2.0 of the Toolkit), use twelve MFCC features with CMS, one energy feature from MFCC analysis, twelve perceptual-linear prediction (PLP) features with noise reduction by RASTA processing, and one energy feature from PLP analysis (for a total of 26 features per frame).

To provide some information about the acoustic content, take a *context window* of features, as shown by the vertical lines in the upper-right diagram in Figure 2.5. This means simply taking the frame of interest as well as frames that are -60, -30, 30, and 60 msec away from the frame of interest. This is done to take into consideration the dynamic nature of speech: the identity of a phoneme will often depend not only on the spectral features at one point in time, but it will also depend on how the features change over time.

CHAPTER 3
METHODOLOGY/PROJECT WORK

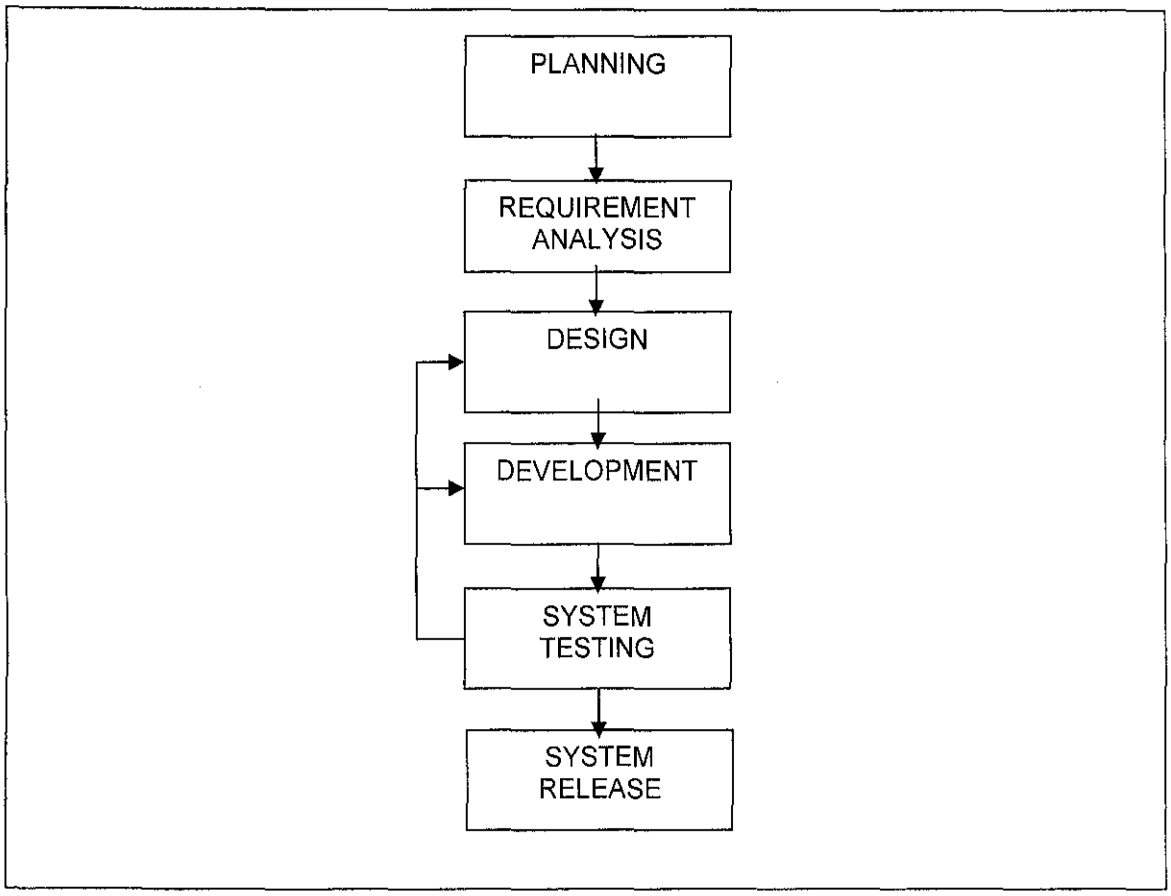


Figure 3: Project Development lifecycle

3.1 Planning

During this phase, every resource is gathered to start planning of the system. This involved feasibility analysis to ensure the project can be developed regarding on the requirements and resources. The source of information is from internet, journal, books and other previous student's project which is related to this system. The writer also needs to find any materials related to this project which is sound processing and voice recognition.

- 1GB ram(minimum)
- Microsoft windows XP(Service pack 2)
- J2ME™ Wireless Toolkit
- NetBeans™ Mobility Pack
- Microsoft® Visual Studio™ 2005
- Mobile devices support with MIDP 2.0

CHAPTER 4

RESULT AND DISCUSSION

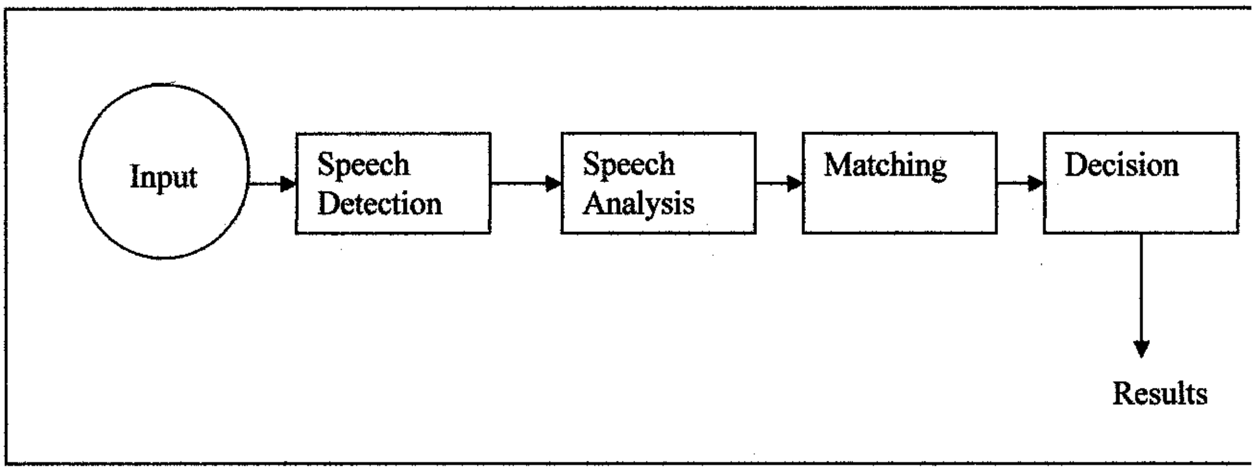


Figure 4.1: System Flow

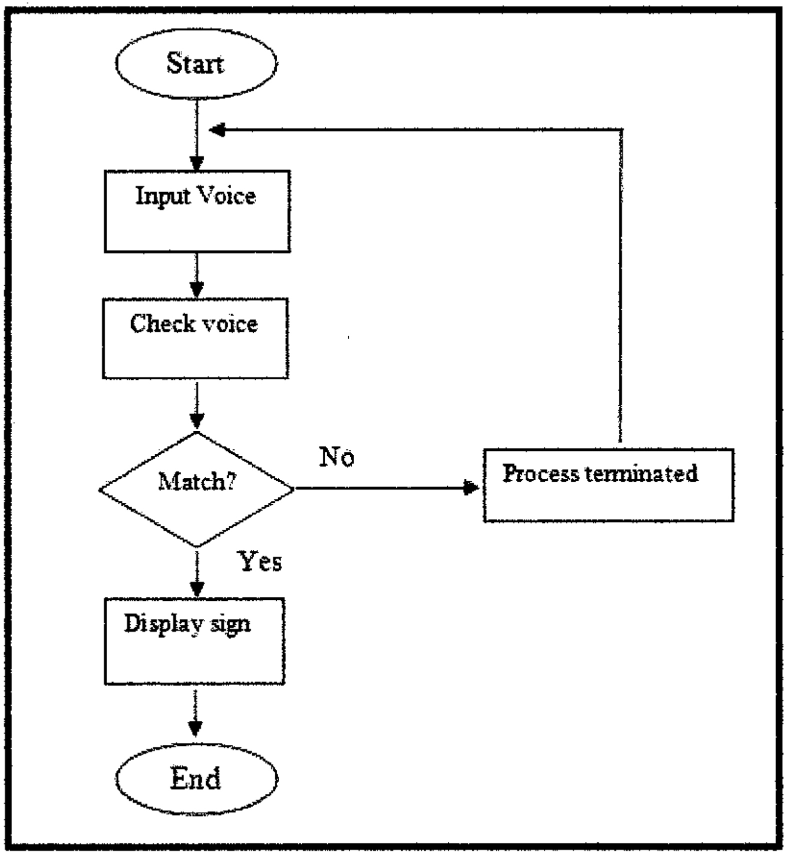


Figure 4.2: Flowchart

4.1 J2ME Architecture

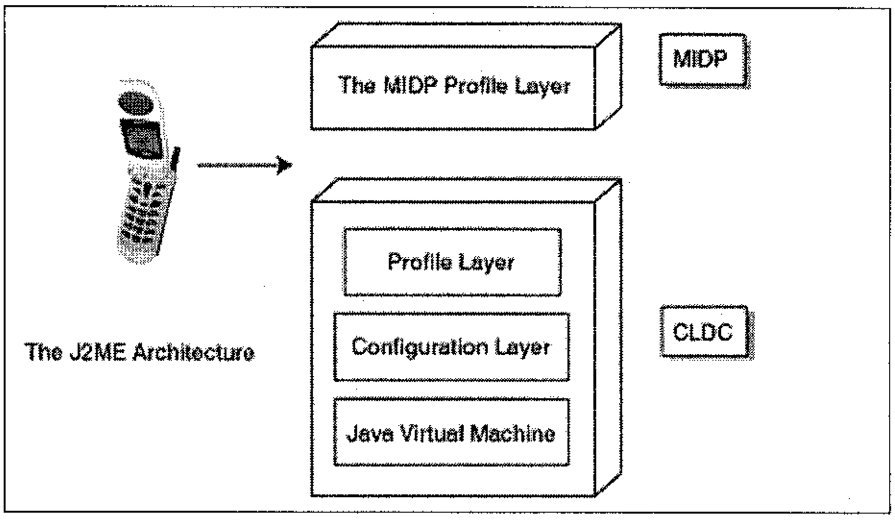


Figure 4.3: J2ME architecture

- **Java Virtual Machine layer:** This layer is an implementation of a Java Virtual Machine that is customized for a particular device's host operating system and supports a particular J2ME configuration.
- **Configuration layer:** The configuration layer defines the minimum set of Java Virtual Machine features and Java class libraries available on a particular category of devices. In a way, a configuration defines the commonality of the Java platform features and libraries that developers can assume to be available on all devices belonging to a particular category. This layer is less visible to users, but is very important to profile implementers.

- **Profile layer:** The profile layer defines the minimum set of application programming interfaces (APIs) available on a particular family of devices. Profiles are implemented upon a particular configuration. Applications are written for a particular profile and are thus portable to any device that supports that profile. A device can support multiple profiles. This is the layer that is most visible to users and application providers.

- **MIDP layer:** The Mobile Information Device Profile (MIDP) is a set of Java APIs that addresses issues such as user interface, persistence storage, and networking.

The J2ME architecture is based on *families* and *categories* of devices. A category defines a particular kind of device; cellular telephones, simple pagers, and organizers are separate categories. A family of devices is made up a of a group of categories that have similar requirements for memory and processing power. Together, cellular phones, simple pagers, and simple personal organizers make up a single family of small-footprint devices

Sun designed J2ME specifically for the consumer market of tiny commodities, such as smart cards and handhelds devices. J2ME provides portable application environment for the customer and embedded systems, including mobile devices. A user having a J2ME mobile device can download applications from open network and run them on his/her mobile device.

Java 2 Platform, Micro Edition (J2ME) is part of the Java 2 platform. While Java 2 Standard Edition (J2SE) targets desktop systems, and Java 2 Enterprise Edition (J2EE) targets the server backend applications, J2ME is a collection of APIs focusing on consumer and embedded devices, ranging from TV set-top boxes, telematics systems, residential gateways, to mobile phones and PDAs. Within each edition of the Java 2 platform, there are different Java Virtual Machine (JVM) implementations that are optimized for the type of systems they are targeted at. For example, the K Virtual Machine (KVM) is a JVM optimized for resource constrained devices, such as mobile phones and PDAs.

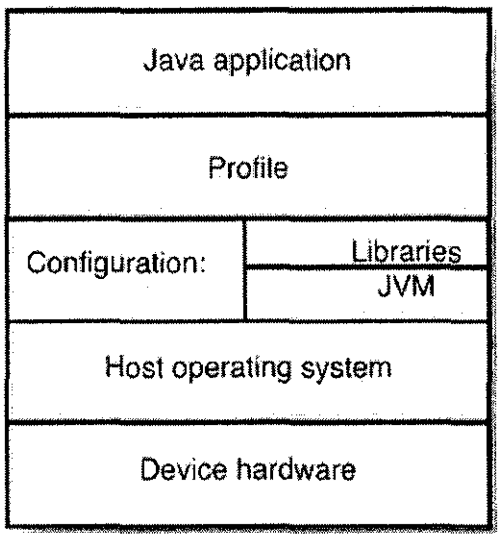


Figure 4.4: J2ME Platform

4.2 Java Speech Processing

Computerized speech can be divided into two categories: *speech recognition* and *speech synthesis*. Speech recognition is the art of turning analog sound waves captured by a microphone into words. These words are either commands to be acted on or data to be stored, displayed, manipulated, and so on. Speech synthesis is the art of taking the written word and transforming it into analog waveforms that can be heard over a speaker. When looked at in this light, the problem of teaching a computer how to listen and talk seems a little daunting.

The Java Speech API is a set of abstract classes and interfaces that represent a Java programmer's view of a speech engine, but it makes no assumptions about the underlying implementation of the engine. This engine can be either hardware or a software solution (or a hybrid of the two). It can be implemented locally or on a server. It can be written in Java or in any other language that can be called from Java. In fact, different compliant engines can even have different capabilities. One of them might have the capability to learn your speech patterns, whereas another engine might choose not to implement this. It is the engine's responsibility to handle any situations that it does not support in a graceful manner.

4.3 Speech Recognition

The other side of Java Speech is speech recognition. The state of the art in recognition is not nearly as advanced as speech synthesis. Recognizers receive information electronically via microphones. They then must try to determine what set of syllables to create from the set of phonemes (sounds) just received. These syllables must then be combined into words.

Java Speech supports dynamic grammars. This means that grammars can be modified at runtime. After a change is made to the grammar, it must be committed using the *commitChanges()* method of the recognizer. When these changes are committed, they are committed atomically, meaning all at once. Listing 12.9 shows a simple grammar.

4.4 Speech Synthesis

Speech synthesis, also known as text-to-speech (TTS) conversion, is the process of converting text into human recognizable speech based on language and other vocal requirements. Speech synthesis can be used to enhance the user experience in many situations but care must be taken to ensure the user is comfortable with its use.Speech synthesis has proven to be a great benefit in many ways. It is often used to assist the visually impaired as well as provide safety and efficiency in situations where the user needs to keep his eyes focused elsewhere. In the most successful applications of speech synthesis it is often central to the product requirements. If it is added on as an afterthought or a novelty it is rarely appreciated; people have high expectations when it comes to speech.

Natural sounding speech synthesis has been the goal of many development teams for a long time, yet it remains a significant challenge. People learn to speak at a very young age and continue to use their speaking and listening skills over the course of their lives, so it is very easy for people to recognize even the most minor flaws in speech synthesis.

Example of java classes:

Class: javax.speech.Central

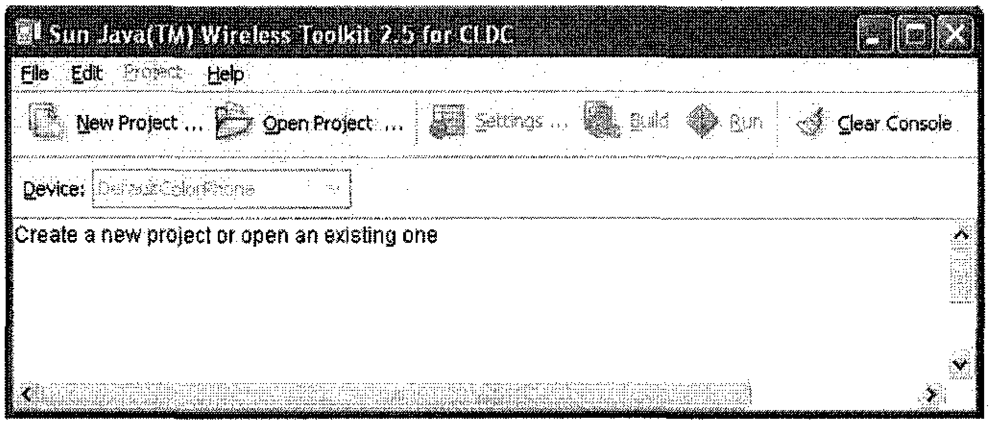
Class: javax.speech.synthesis.SynthesiserModeDesc

Class: javax.speech.synthesis.Synthesiser

These classes are used for converting text into speech using the selected voice. Synthesizers must be allocated before they can be used and this may take some time if high quality voices are supported which make use of large data files.

4.5 Sun Java Wireless Toolkit

The tools to develop this system is using Sun Java wireless toolkit 2.5.2. This tool will help developer more easily get started to the new system. This technology is still relatively new. Strong development tools, especially IDEs, will help J2ME's adoption among wireless application developers.J2ME Wireless toolkit contains a reference implementation of J2ME/MIDP(Mobile Information Device Profile) and has multiple device emulators that run on Unix/Linux and windows platforms. This tool also has performance profile tools and real memory usage monitors. All those tools can be administrated from a central control panel called kToolBar.kToolBar can also build a package ready-to-deploy JAR/Jad Programs from development directories. However, mastering all these tools and testing applications on all emulators can prove tedious.


Sun Java(TM) Wireless Toolkit 2.5 for CLDC
File Edit Project Help
New Project ... Open Project ... Settings ... Build Run Clear Console
Device: DefaultColorPhone

Create a new project or open an existing one

Figure 4.5: Sun java wireless toolkit 2.5

4.6 Mobile Media API(MMAPI)

The Mobile Media API (MMAPI) provides a powerful, flexible and simple interface to multimedia capabilities. The function is for playing and recording audio and video data.

4.6.1 Mobile Media API Architecture

The Mobile Media API is based on four fundamental concepts:

I. A player knows how to interpret media data. One type of player, for example, might know how to produce sound based on MP3 audio data. Another type of player might be capable of showing a QuickTime movie. Players are represented by implementations of the javax.microedition.media.Player interface.

II. Use one or more controls to modify the behavior of a player. developer can get the controls from a Player instance and use them while the player is rendering data from media. For example, can use a VolumeControl to modify the volume of a sampled audio Player. Controls are represented by implementations of the javax.microedition.media.Control interface; specific control subinterfaces are in the javax.microedition.media.control package.

III. A data source knows how to get media data from its original location to a player. Media data can be stored in a variety of locations, from remote servers to resource files or RMS databases. Media data may be transported from its original location to the player using HTTP, a streaming protocol like RTP, or some other mechanism. javax.microedition.media.protocol.DataSource is the abstract parent class for all data sources in the Mobile Media API.

IV. Finally, a manager ties everything together and serves as the entry point to the API. The javax.microedition.media.Manager class contains static methods for obtaining Players or DataSources.

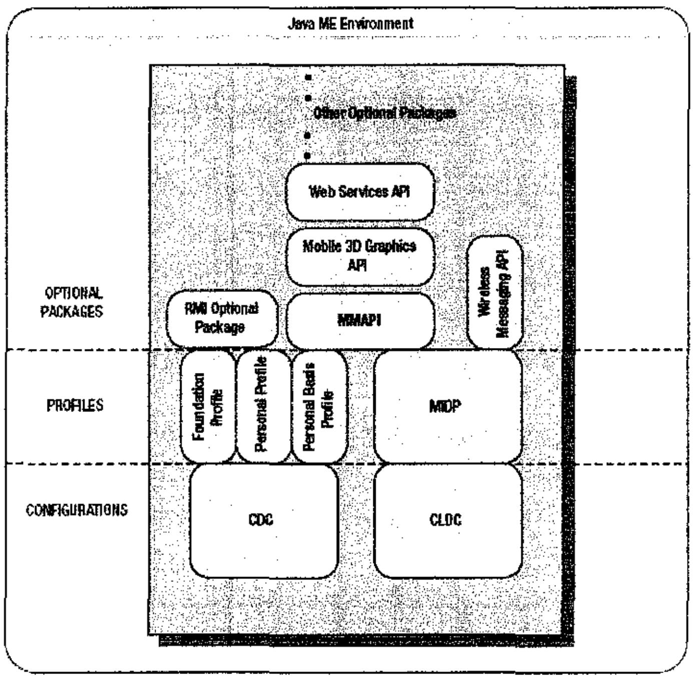


Figure 4.6: J2ME environment showing the MMAPI as an optional package

4.7 Implementation

4.7.1 Prototype

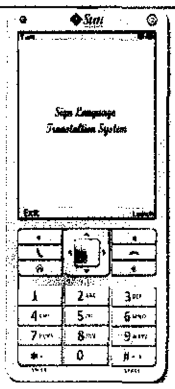
	**Description:** This is the main menu for the user while running this midlet.

Table 4.1: main Menu

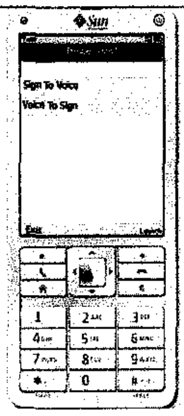
	**Description:** User can choose whether using sign to voice or voice to sign conversion

Table 4.2: main menu: User selection

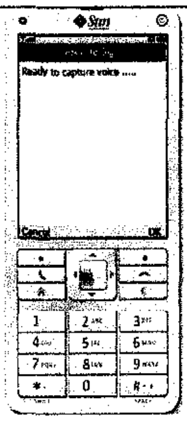
 Ready to capture voice	**Description:** The application is now ready to capture voice input by the user. System can only capture 3 seconds voice input.

Table 4.3: Capture voice input

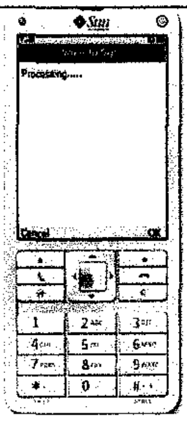
 Processing.....	**Description:** After the voice input, the system will process the voice (Using voice recognition and matching). In this situation, the voice input will be matched to the appropriate sign

Table 4.4: Processing voice input

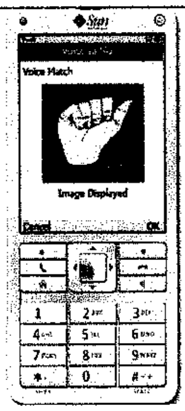
	**Description:** After processing, the system now can display the sign on the screen. If users want to find another sign, they can input another voice.

Table 4.5: Display the output

4.7.2 Example Scenario

a) User Start the application

b) The system will ask the user to input the voice

c) After captured the voice, the system will replay the voice and at the same time will process the voice

d) During the processing, the voice will be matched with the correct sign

e) The screen will display the sign to the user.

4.7.3 System Architecture

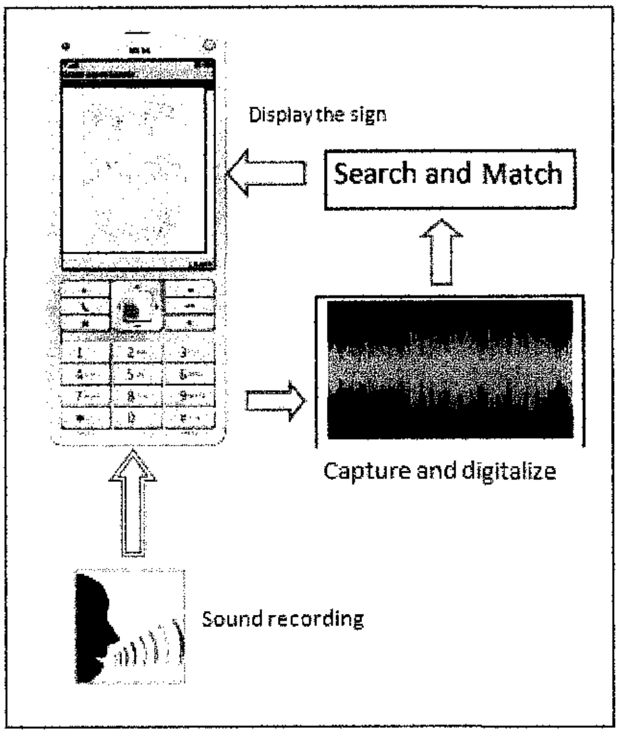


Figure 4.6: MV2S system architecture

4.7.4 Voice/Speech Recording

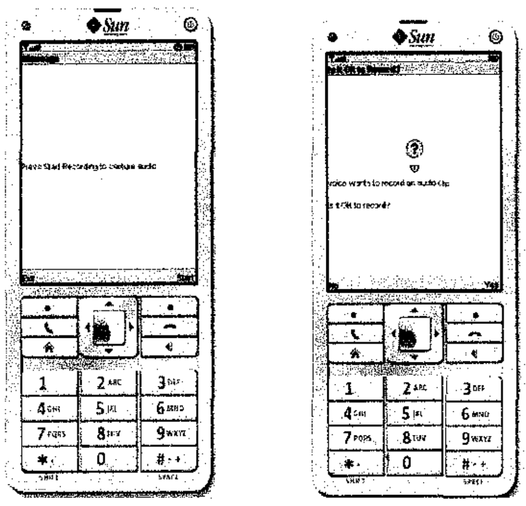

Sun
Press Start Recording to capture audio
Sun
voice wants to record an audio clip
Is it OK to record?
Yes

Figure 4.7: Record voice input

Audio recording process support audio.capture system property. If this property is true, then audio encodings system property must not return null, but must provide the formats in which audio can be recorded. Most of the mobile devices support the audio/amr and audio/amr-wb(wide band) formats.

To capture media, use RecordControl control to capture audio. This control doesn't distinguish between the media types and is a simple mechanism to start, stop and control the data recording. A player instance to capture audio is created by passing the special locator *capture:// audio* to the *Manager.createPlayer(Stringlocator)*method which will then, once realized give up a RecordControl object to use. Recorded data must be either stored somewhere or played back to the user(then discarded). Record control provides two methods for setting the location of recorded data. The *serRecordStream(OutputStream stream)* directs data to an OutputStream, and *setRecordLocation(Stringlocator)* directs it to the location in the form of locators user to create Player instance using *Manager.createPlayer(Stringlocator)method.*

4.7.5 Audio Capturing

An implementation supports capture of audio via MIDlet if it return true for the support.audio.capture system property. If this property is true, then the audio encodings system property must not return null, but must provide the format in which audio can be recorded. Many mobile devices supports the audio/amr and audio/amr-wb(wide band) formats. To capture media, use the RecordControl control, whether for audio or video. This control doesn't distinguish between the media types and is a simple mechanism to start, stop, and control the data recording. It provides several methods to achieve these tasks. Simple capture and playback of audio can be accomplished easily by using the two streams, ByteArrayOutputStream and ByteArrayInputStream. Because data doesn't need to be stored anywhere, one player instance can be used to record the raw audio data into the byte array, and another instance created to retrieve data from this buffer using the ByteArrayInputStream. For this project, the data can be recorded up to 10 seconds. The process can completely done if the voice can match with the certain sign language.

4.7.6 Voice Processing

Digitally recorded audio data is often referred to as sampled audio. Sampled audio represents successive samples of audio data. Audio data in physical terms is represented as a signal (of a sound wave). By taking discrete samples of the amplitude of a sound wave and representing it suing bits and bytes, digital audio data is generated. In a sense, sampled audio is only an approximation of the actual sound wave because the samples of the original and not recreating the whole thing. The sample of producing sampled audio is called Pulse Code Modulation (PCM). Although sampled audio is mostly encoded using PCM, it can be saved in a variety of format for transmission, retrieval and playback. The common storage for mobile phones is Adaptive Multi-Rate (AMR). In J2ME, voice processing normally using ByteArray.

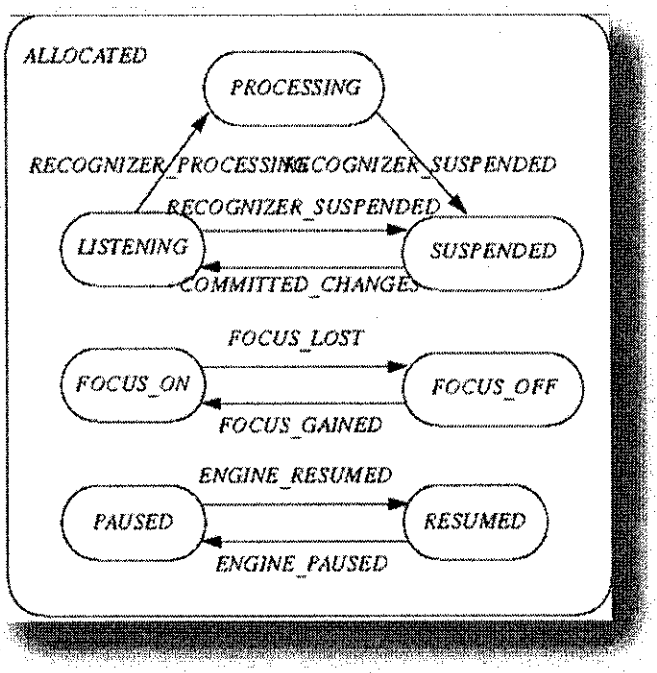


Figure 4.8: Recognizer stated in Java

The current audio functionality of the Java Speech API is incompletely specified. Once a standard mechanism established for streaming input and output audio on the Java platform .The API will be revised to incorporate that functionality. In this release of the API, the only established audio functionality is provided through the RecognizerAudioListener interface and the RecognizerAudioEvent class. Audio events issued by a recognizer are intended to support simple feedback mechanisms for a user. The three types of RecognizerAudioEvent are as follows: SPEECH_STARTED and SPEECH_STOPPED: These events are issued when possible speech input is detected in the audio input stream. These events are usually based on a crude mechanism for speech detection so a SPEECH_STARTED event is not always followed by output of a result. Furthermore, one SPEECH_STARTED may be followed by multiple results, and one result might cover multiple SPEECH_STARTED events.

AUDIO_LEVEL: This event is issued periodically to indicate the volume of audio input to the recognizer. The level is a float and varies on a scale from 0.0 to 1.0: silence to maximum volume. The audio level is often displayed visually as a "VU Meter" - the scale on a stereo system that goes up and down with the volume. All the RecognizerAudioEvents are produced as audio reaches the input to the recognizer. Because recognizers use internal buffers between audio input and the recognition process, the audio events can run ahead of the recognition process.

4.7.7 Displaying Video

VideoControl, which is the primary control for displaying video, is used to control how the video looks when displayed in a MIDlet, including its positioning and screen focus. The controls for playback, pause, rewind, and forward are provided by the underlying Player instance methods, and are therefore, generic. This video formats can display depend on MMAPI implementation. Most implementations support MPEG-4, but an increasing number support 3GPP as well. VideoControl also provie control over the positioning of the video, but the main method that initializes the display is the one inherited from GUIContol. This method, initDisplayMode(int mode, object arg), provides the basis on which the video is displayed on the device screen. This method must be call only once for each video display. Essentially, this method tells the implementation how to display the video. There are 2 types of displaying which are Form and canvas.

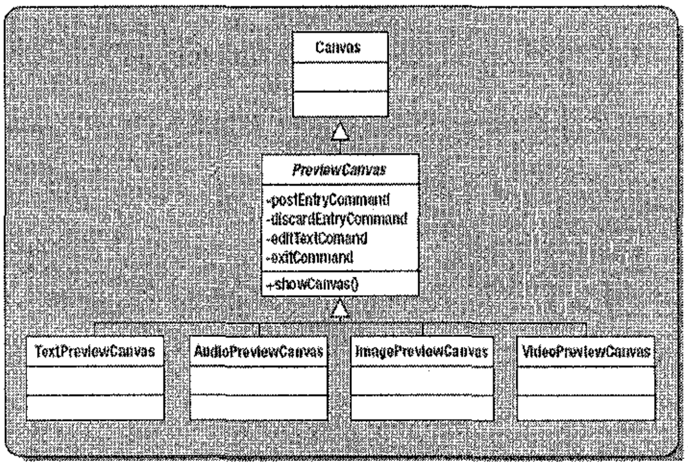


Figure 4.9: Display canvas for MV2S

4.8 Problems

During the implementation of the prototype, a few problems occurred. Most of the problems had to do with limitation in the technologies chosen for the project or the fact that the author was not particularly accustomed to working with them. The first problem with the initial design, and the one that influenced the entire project the most, was that there were no packages in Java capable of sound matching. Since recognizing sound is a crucial part of the system's functionality, a speech recognition engine was used instead. Naturally, the prototype could have been developed without speech recognition, using dummy classes to simulate sound matching. This alternative was opted against on the grounds that speech recognition is at least as advanced and demanding as sound matching. Hence, if the system worked with speech recognition, the feasibility of using sound matching in its place would to all intents and purposes established.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

Speech recognition technology helps in various ways to ease or solve the problems in communication between organizations. There are a lot of achievements have been made such as a product developed by IBM which is The Dragon Natural Speaking. Nevertheless, the recognition accuracy of this software is still lack due to the particular of human voice which is unique.

Natural language to sign language translation is the main focus in this research. The fundamental idea of this system is to translate voice (input) using mobile devices to sign language. This system will match the input (voice captured by user using microphone) with the stored images or video of the sign language. This process will use pattern recognition to match the voice and display appropriate results. Hence, this system will give the interactive way of communication between normal and disable peoples.

This project aims to create an application that allows translation of English language to sign language. With the minimal training for the system, it will be able to recognize the commands and execute actions. For the prototype, this system can be able to translate the simple commands or instructions and can trigger the output for users.

5.1 Suggested Future work/ Recommendations

Nowadays, mobile devices are widely used in the world. Peoples can easily get the devices with the low cost. The technologies also keep on changing every time with the current situation. Using sound matching and recognition is an obvious extension of this project. It would probably make sense to examine what implementation the most efficient in this report. The performance of the system compared to a more conventional distributed system needs to be studied to determine if there is any advantage at all to using mobile devices. It's also have been shown that the system is definitely possible to develop using existing techniques and tools, even with some minor limitation. Most of these restrictions are introduced because of the technology used in the prototype and it seems likely most of them could be worked around or eliminated with time or by using alternative approaches. Another future enhancement for this project is to translate a sentences rather than a word. So, peoples can really understand and can learn Sign Language effectively.

REFERENCES

1. Down, Sharon. Make a Difference. University of Arkansas (2005

2. Tennant, Richard. *The American Sign Language Hand shape Dictionary.* (1998).

3. Sun Microsystems 2001. Java 2 Micro Edition (http://java.sun.com/products/Mobile)

4. Sing Li and Jonathan Knudsen, Beginning J2ME: From Novice to Professional. Apress(2008)

5. Voice Processing : Recognition, Command and response (http://www.managingchange.com/mediums/mobile/voice.htm)

6. Nathan Tippy, Introduction to Java Speech API. Object Computing Inc (OCI)

7. IBM corporate website (http://www.ibm.com)

8. Tony Aires and Dr. Brian Nolan,Institute of technology Blanchardstown(2006), Voice Activated Command and Control with Java-enabled Speech over Wifi.

9. B.Sasidhar and B.V Durga Kumar.The Effect of Mobile Devices and Wireless Technology in E-learning. Sunway Academic Journal (2005).

10. Daniel Jurafsky and James H.Martin ,Prentice Hall.An Introduction to Natural Language Processing,Computional Linguistics, and speech Recognition (2000).

11. Paul Tremblett, Instant Wireless Java with J2ME. McGraw Hill(2002)

12. http://www.e-zest.net/j2me.html

13. Terrence Lawai Wan, Translator System (English to sign Language).(2007)

14. Canlas, Loida ,Laurent Clerc: Apostle to the Deaf People of the New World. Laurent Clerc National Deaf Education Center at Gallaudet University(2006)

15. Garner Group, Research on PDA and mobile phone sales by 2005, (http://info-edge/com(2004))

16. Stephen Potts, Alex Pestrikov, and Mike Kopack, Sams Publishing. Java 2 Unleashed(2004).

17. Jafar Ajdari, Java 2 Mobile Information Device Profile(MIDP).Helsinki University of Technology(2001).

18. Goyal, V. Pro Java ME MMAPI Mobile Media API for Java Micro Edition. Apress. (2006).

19. Voice Signal(http://www.voicesignal.com)

20. Voice Signal-Technology Description(http://www.voicesignal.com/solution)

21. John-Paul Hosom, Ron Cole, and Mark Fanty. Speech Recognition Using Neural Language. Oregon Graduate Institute of Science and Technology(1999).

22. Daniel Hagglund, Mobile Agents Using Sound. Monash University of Melbourne (2003).

Appendix A

Gantt chart

TASK	**DAYS**	**JULY**	**AUGUST**	**SEPTEMBER**	**OCTOBER**	**NOVEMBER**	**DECEMBER**	**JANUARY**	**FEBRUARY**	**MARCH**	**APRIL**
1. System Planning/Analysis	**48**										
Collecting information for project	20	▨	▨								
Study of gathered information	18	▨	▨								
Feasibility study	10		▨								
2.System Design	55										
Gathered information to produce system flow	30		▨	▨							
Generate system requirement	10			▨							
Generate system specifications	15			▨	▨						
3.System Development	110										
Development of system interface	50				▨	▨					
Development of system logic	20				▨	▨					
Interaction between the interface and the system logic	40					▨	▨	▨			
4.System testing	30										
unit testing	15								▨		
system test	15								▨		
5 Implementation	2										
final presentation of the project	1									▨	▨
Project Submission	1									▨	▨

Figure 1: Project Timeline