Voice based E-mail and Skype-call Application for the Blind

M.D. N. Dilini
Department of Computer Science and Engineering, University of Moratuwa, SriLanka
nathiesha.12@cse.mrt.ac.lk

Abstract- In the modern world, communication has become easy and efficient due to advancement of technology and development of the Internet. However, this does not apply for the visually impaired people. With the new technological developments, blind population finds it is challenging to keep up with the modern communication methods such as email and Skype. The main barrier stands in between these communication methods and the blind users is, their inability to use the keyboard and to see the computer or phone screen. This paper aims at describing the system being developed for the visually impaired people to use email/Skype using a desktop and to take phone calls using a smart phone, without any previous training. The system will not require the user to make use of keyboard; instead, it will work only using mouse operations and speech conversion to text. Text to speech conversions would be used to give the output and to read the screen contents to the user. This system can also be used by any normal person, for example the ones who are not able to read (illiterate). The system is completely based on interactive voice responses, which will make it user friendly and easy and efficient to use.

Keywords— User Interface Design; Speech to Text; Text to Speech; Communication for Blind

I. INTRODUCTION

Today we live in an information world where the computer and mobile phones have become an inseparable part in everyone's lives. The computers, mobile phones and internet together with the advancement of technology have significantly affected the ways of communication. The old communication techniques have been replaced with email, Skype, Viber etc. Mobile phones also play a key role in modern communication. Most of these methods are very simple to access and use. The users only need to follow the guidelines on the screen and click or type the required inputs. However when it comes to the visually impaired or illiterate population, this process becomes really hectic and complex due to their inability to make decisions, as decision making and communicating the choices involve reading the contents on the screen and clicking a particular area for selection or typing using the keyboard.

All these communication applications are originally designed targeting the normal users and with the aim of improving usability for them. However when statistics is considered, it becomes evident that over 285 million of the world population is visually impaired while 39 million out of them are blind [1]. Nearly 12% of the world population is illiterate [2]. These numbers suggest that an effort should be made to facilitate the visually impaired people to use modern communication methods. All the visually impaired and illiterate people are deprived of the access to modern communication technology due to their visual inability. To overcome this challenge and to build an application for the blind people to use modern communication methods were the main objectives of the project.

The system enables user to use his voice as the main tool of communication. The user does not have to read the screen contents. Instead, the application would be reading out the contents on the screen for the user and each interface is designed in a way that makes the users to access function of system efficiently. Based on the audio output, user can give an input or select an option using his voice, or a simple mouse click or a key press, whereas in the Android system, a tap on the screen would do that. The system identifies the user voice commands and based on the identified commands, the system services can be accessed by the user. This simple procedure is capable of overcoming the barrier for the blind users to access these communication methods, up-to some satisfactory level.

The project was aimed at developing two applications, one for the Android phone and the other as a standalone desktop application. The system would allow a user to use it without previous training or any external support. Though the applications were developed primarily targeting the blind group of users, this can be used by anyone else, who finds it easier to listen and speak rather than reading and typing. This also allows people who cannot read or write, to access emails and Skype.

II. LITERATURE REVIEW

A wide variety of related work is available in literature on voice based systems or blind users [3, 4]. Among them the authors in [3] have proposed a system which will help the visually impaired people to access email services efficiently. That proposed system [3] helps in overcoming some drawbacks that were earlier faced by the blind people in accessing emails. They have eliminated the concept of using keyboard shortcuts along with screen readers which will help reducing the cognitive load of remembering keyboard shortcuts. Also with the use of the proposed system any naive user who does not know the location of keys on the keyboard need not worry as keyboard usage is eliminated. The user only needs to follow the instructions given by the IVR and use mouse clicks accordingly to get the respective services offered. Other than this the user is needed to feed in information through voice inputs when specified.

Unlike current systems which emphasize more on user friendliness of normal users, the research article focuses more on user friendliness of all types of people including normal people, visually impaired people as well as illiterate people. As

they suggests IVR and recognizing mouse click events while making the whole screen area clickable was incorporated into my system as it seem to be an effective way to take user input. Furthermore I additionally incorporated the key press event as an error correction mechanism, where by pressing any key user can try giving voice inputs again or visit a previous interface.

Another interesting work has been done in the same research area [4] that describes about a voice mail architecture that helps blind people to access e-mail and other multimedia functions of operating system (songs, text).It also focuses on mobile applications where SMS can be read by system itself. In that research paper, they describe the voice mail architecture used by blind people to access E-mail and multimedia functions of operating system easily and efficiently. This architecture they propose will also reduce cognitive load taken by blind to remember and type characters using keyboard. It also helps handicapped and illiterate people.

When analyzing the architecture of the latter proposed system [4], major components of the system were found, and they are G-mail (System read messages on recipient mailbox), RSS (Real Simple syndication for news), Song (listen songs), Book reader (system read book) and Drive browser (To search drives and folders). They also proposed a similar system for the mobile devices. Although the system I developed was less complex than the proposed system by the research article, the concepts discussed were adoptable to my system.

As the latter system proposes, the user can apply the same keyboard command by performing different mouse operation and voice operation in their system each voice operation mapping to certain keyboard operation and also voice operation map with certain keyboard operation enabling user to use keyboard or mouse or voice commands to get a work done. [5] Although this concept was not used in this system, that can be included in future developments to make the system more usable.

III. SYSTEM MODELS

A. System Requirements

The system requires two standalone applications, one for the desktop and other for the android phone. Each application needs to provide access for emails. User should be able to view the in-box and sent mails and also to send emails to his contacts. The android application should have the similar requirements, with the additional feature of taking phone calls. All these features should be incorporated with voice recognition and text to speech conversions. Fig. 1, shows the layered view of the system.

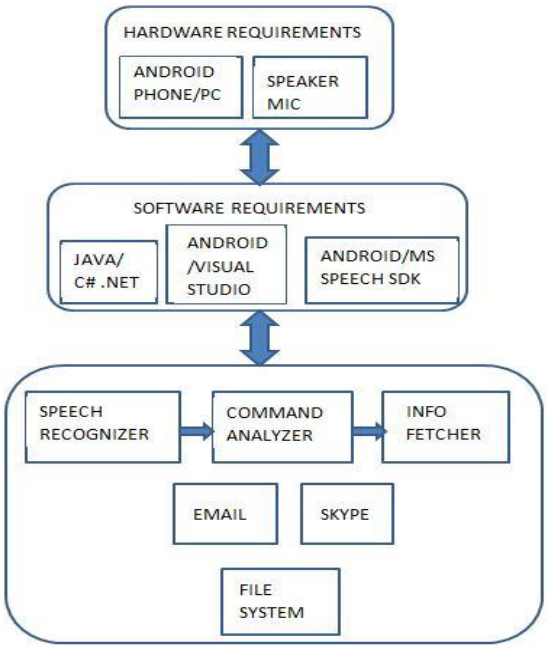

Fig.1. Layered View

The most significant non-functional requirements of the application, is system usability. As the target group is blind people, ease of accessing the services should not be compromised. The voice recognition functions and error correction mechanisms have been incorporated to the system to increase the ease of access. Also the system should be reliable, and available as if any error occurs, the blind users may not be able to fix it on their own.

Fig. 2.Use Case Diagram

Fig. 2, shows the use case diagram of the email access sub system in the windows application. Once a user log into the system, it announces the registered user names, where user can select one using voice commands. Then the user credentials will be loaded and the user is provided with the functionalities such as read the inbox, read sent items, edit contacts or compose new email.

If the user has not registered into the system before, he can select 'add new user' option using the speech commands and then enter the new user details, which would be stored in the system and can be accessed anytime. If in-box/sent items are selected, user in-box/sent mails would be loaded and the available emails in that particular folder would be read out for the user to listen. If user wishes to compose a new email, the recipient name can be selected out of the registered contact names that would be read out by the system. The user message would be recorded, stored and sent to the recipient as an audio attachment.

B. System Design

The windows application, once installed on the PC will open up automatically. The user can start using it by a simple mouse click. The MIC of the PC will capture user's voice inputs. With the use of Microsoft speech SDKs, speech would be recognized and accordingly the next step would be taken and relevant data would be fetched from the files and user would be notified by the reader. The files will contain the authentication details and user contact lists for email and Skype as well. The hardware and software modules proposed and the flow of information is depicted in Fig. 3.

The android application architecture is also quite similar to the architecture of windows application. Roughly, the hardware requirements for the Android version of the application are touch screen device, preferably of size 4.0‖ x 4.0ll, android OS version 2.3.6 or higher, CPU speed ≥ 400 MHz, at least 30 MB of free phone memory, with support for SD card installation. And the system requires at least 80 MB of secondary memory.

The android application should be supported by android API level 2.3 or higher. Data persistence should be addressed using a relational database in the mobile device and PC. Authentication is supported using user name and password. The architecture is flexible and extensible, ensuring re-usability for the next phase of the development and response time constraints due to less processing power.

The UML 3-tier architecture diagram shown in Fig. 3, depicts the main layers of the architecture of the application, the packages within each one of them and interactions between these packages.

As shown in Fig. 3, the UI layer could interact directly with the service layer if there is no application logic in the operation. But if data needs to be processed first it should be done in the logic section rather than the Data Access Object or UI layer. In the domain layer, the model section contains the plain old Java objects and the logic section would use Data Access Objects and other services from the services layer and return data to the UI layer.

The service layer would contain features of the device like file access for the domain and UI layers. Fig. 4, depicts the class diagram (Logical view) of the system Fig. 5, depicts the activity diagram (process view) of accessing in-box. When the user initiates a request to see the in-box, the user name would be passed with the request.

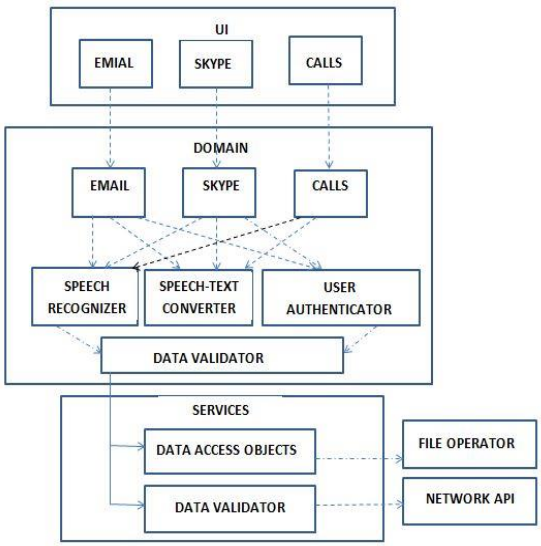

Fig.3. Three-tier architecture diagram

The application would check whether the user name is valid, and if verified correct, the application would fetch the credentials from the file server. The email address and password would be passed back to the application. Using these credentials, the application would connect to Gmail and user would be logged into the system. If the user wishes to read the in-box, the application would access the in-box, and text to speech converter would read the in-box

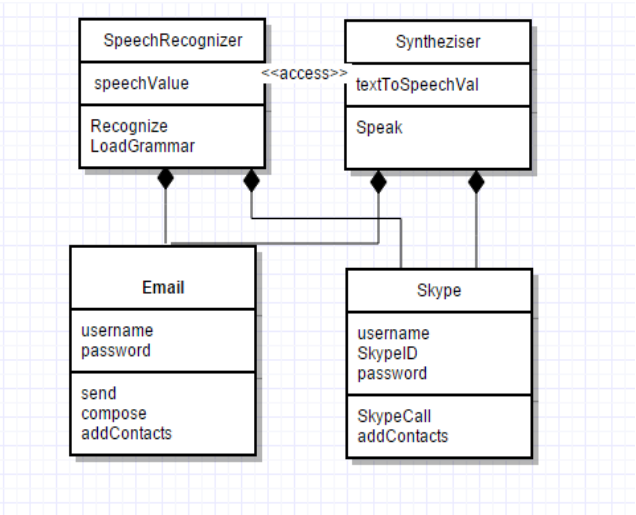

Fig.4. Class Diagram

Although the original plan was to design and develop a database for the storage of data, with the project progress, the design slightly deviated. The user information of the system had to be stored and accessed when needed. However only the basic log-in information like user name, email address, Skype ID and contact list of emails and Skype IDs the user frequently uses had to be stored. Furthermore as this was a desktop application for blind, most probably there would be a single user or two

maximum, therefore the amount of data to be stored would be minimal and the security expectations were not very high. As the use of a separate database degrades the performance of the system, it was decided during the feasibility study, that a simple file system would be sufficient for the storage of data.

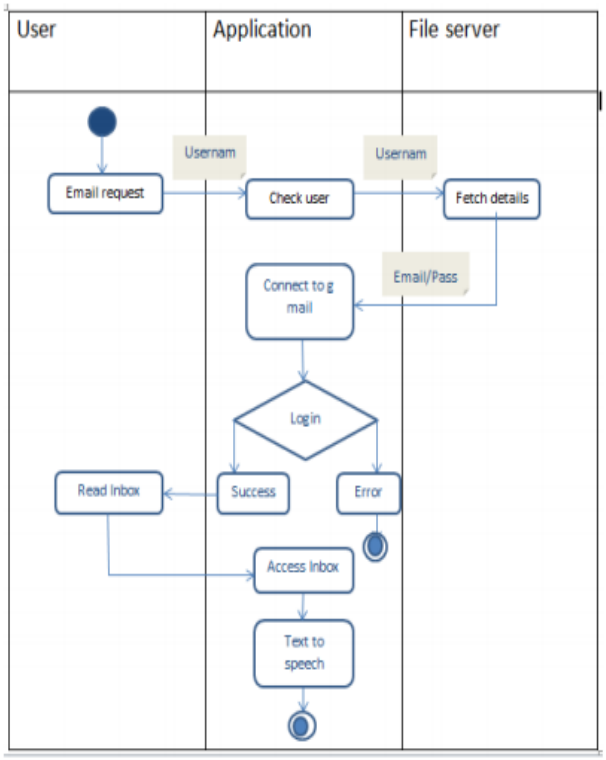

Fig.5. Activity Flow Diagram

IV. System Implementation

A. Implementation Procedure

The system development followed the rational unified process. The development phase was completed in two iterations. The first iteration focused on completing the windows application whereas in the second iteration development of the similar android application was done. When developing the desktop application, .Net architecture was used. The system was developed using the language C# and available MSDN libraries were made use for most of the functions. The android application was developed using the android framework. Java was used for the coding of logic and XML was used for the development of interfaces.

Visual Studio was used as the IDE for the development of desktop application. As it supported the .net architecture and the use of C# and as the application made use of MSDN APIs for some functions, Visual Studio seems to fit the purpose most. Android Studio was used as the IDE for the development of the android application. The built in functions supported the development of android application and it was easy to use Android APIs. Other than the IDEs, OpenProj was used for the project planning purpose. The project schedule and resource allocation was done using that.

Several open source speech engines like CMU Sphinx and Kaldi were evaluated, but Microsoft speech recognition engine seem to fit the purpose for the user speech input recognition, most because it could be easily integrated with the C# code base. Furthermore MS speech engine supported many languages like English, Chinese, German, French etc. and it also supports voice training. Other than that Microsoft text to speech conversions were also used. MSDN APIs were used for this purpose. In the Android application, the available Android APIs for text to speech conversion were used to provide user audio guidance. For the speech recognition purpose, Google speech recognizer engine was incorporated into to the application because of its high accuracy than the available other speech recognizer. Online tutorials and forums were frequently used during the android and desktop application development.

The system does not require any initial data for its functioning. The user is supposed to enter the log in credentials when a new user account is created. This information would be stored in separate text files. Whenever the user needs to log in, these data would be fetched and used. Additionally the contact names and their details like name, telephone number, email address and Skype IDs will also be stored in text files and fetched whenever the user requires. For the demonstration purpose these files were filled with real values.

No special algorithm was used for the application. However in each form of desktop application, same procedure was called. When a form object is created a text to speech convector needs to be declared and initialized. The output device of the synthesizer needs to be set to the default audio device. Using the speak method; user should be guided about the interface using audio.

Then a speech recognizer engine object needs to be declared and initialized. Additionally semantic result value objects that contain key values and their codes also needed to be declared to insert the key words used within the given interface. (Ex-in-box, email, Skype, compose). Then the choice objects needed to be created and semantic result value objects using implicit conversion from Semantic result values to grammar builder objects. Once the speech recognizer object and associated semantic result values are created, the event handler for the event speech recognized needs to be declared. The method associated would trigger as soon as a speech is detected. For the speech recognition grammar need to be loaded as well.

A separate method need to be declared to handle the speech recognized event. Once a speech is recognized, the corresponding string value is stored into a variable and necessary actions should be taken based on the string value. Meanwhile user should be notified about the detected string by saying it aloud, where user can try again if needed. If the recognized string value matches with any of the specified key values, based on that user should be directed to the necessary and relevant interfaces. This is done using a series of if else statements

B. Main Interfaces

1). User Interfaces-Android system

First-time user of the mobile application should see the welcome page when he/she opens the application. The system would read out the contents of the screen and when the user is ready, he can simply click anywhere on the screen and he would be directed to the net interface.

The next interface would list the functions available in the system and would read them to the user. The user can say one of the given key words and access a particular service. The system would be listening to the user voice inputs and once a voice command is detected, the system would say it back. If user agrees with it, he can proceed by simply click in the mouse. If user wants to try giving the voice input again, he can make right click.

Fig. 6, depicts the user interface, when the user selects email service. A couple of options will be available like log in, add new users, back and close. The Skype system would have similar interfaces. The windows application would have similar interfaces.

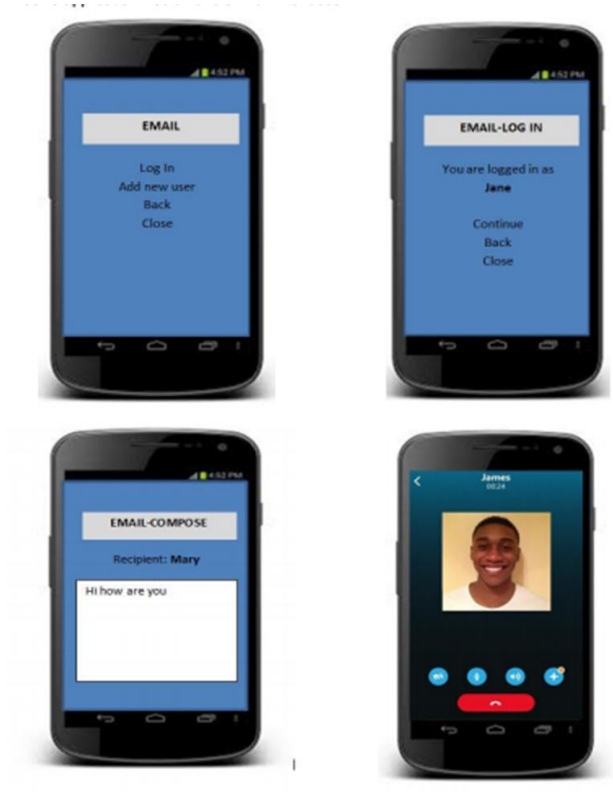

Fig.6. User Interfaces

2). Hardware Interfaces

Since neither the mobile application nor the PC have any designated hardware, it does not have any direct hardware interfaces. The system makes use of MIC and speakers of PC and mobile phone. The physical connection to Mic and speakers is managed by the underlying operating system on the mobile phone and the desktop application.

3). Software Interfaces

The system does not use any purchased components. All the components are developed with the uses of APIs when developing the desktop application. The language used for coding is C# in windows application. And Java is used for the Android application development. Therefore even though the functions of the 2 systems for PC and android are quite same, the components developed for one cannot be used for the other. However the error correction functions and speech recognition functions within a single application can be reused within the same application for different interfaces.

4). Communication Interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems for both the mobile application and the PC.

V. TESTING

The testing process was carried out with the mission of finding as many bugs as possible is the 2 sub-systems, finding important problems and any risks to the quality of the system, certifying the system is up to standards and fulfilling the process mandates. Accomplishing the above mentioned mission helped to increase the user satisfaction. It also helped in standardization. Furthermore it decreased the development time, and integration more easy and less time consuming.

This Testing approach supported the following objectives of identifying existing project information and the software that should be tested, listing the recommended test requirements, recommending and describing the testing strategies to be employed and listing the deliverable elements of the test activities. The following components were tested and they all fit the purpose.

- Speech recognition module android/desktop
- Gmail sending module
- In-box Access module
- Skype calling module
- Face recognition and user authentication module
- Phone calling module
- MSDN API for speech recognition
- MSDN API for text to speech conversion
- Google offline library for speech recognition
- G mail mobile app
- SkypeCOM API

These 3rd party libraries and applications have been successfully integrated into the system. Those have been thoroughly tested by the relevant parties before release. However testing was done before integrating those into the system to make sure they fit the purpose.

Unit testing was done for each unit in android application and desktop application. Unit tests were written using JUnit for the Android application and using NUnit for the desktop application. The user interfaces were also tested using test

classes. The application successfully passed all the written unit tests. Additionally, the android system was tested using the tool Testdroid, where the system was installed into different mobile devices and tablets and tested for the compatibility. The system was compatible with 70% of available devices.

VI. Conclusion and Future Work

The main aim of the project was to facilitate blind users to access email Skype and calls, which are modern ways of communication, incorporated with the technology. The challenge was to develop a mechanism to overcome the inability of the user to see the screen or to input using keyboard. Speech recognition could be successfully used for this purpose. Not compromising the usability of the system meanwhile was another key concern.

Different mechanisms like declaring the keyword objects in the system dictionary in advance could boost the accuracy in speech recognition. In the android application, although android APIs were used at first, later Google speech recognizing engine was incorporated into the system, because of its high accuracy and increased usability. However this requires internet connection and adds latency to the application which can be considered as a drawback. An efficient error correction mechanism (listening to the key press event) was also incorporated to the system in order to increase usability. However as the system could not be tested with a group of blind users for real, the level of usability could not be measured accurately.

As future work, the system needs to be tested under real conditions, with the target group of users. Based on the observations, further adjustments or improvements are intended to be carried out. During the final evaluations, it was pointed out that the android application has a higher response time due to the use of Google speech recognizer. Necessary actions needed to take to improve the response time.

Furthermore, a similar windows application targeting the windows mobile phones is to be developed. As the desktop application already uses .Net and C# languages, porting that application into a windows mobile application would require less time and cost. If the android application and windows application is detected to be error free with furthermore testing, they will be releases in Google play store and windows play store as free applications, for the blind users to make use of.

References

[1] The WHO website. Available: http://www.who.int/mediacentre/factsheets/fs282/en/

[2] Robin Leonard Arlene, “Statistics on Vision Impairment: A Resource Manual”, April, 2002

[3] Jagtap Nilesh, Pawan Alai, Chavhan Swapnil and Bendre M.R.. “Voice Based System in Desktop and Mobile Devices for Blind People”. In International Journal of Emerging Technology and Advanced Engineering (IJETAE), 2014 on Pages 404-407 (Volume 4, issue 2).

[4] Jae Sung Cha, Dong Kyun Lim and Yong-Nyuo Shin, “Design and Implementation of a Voice Based Navigation for Visually Impaired Persons”, June, 2013

[5] Disabilities, Opportunities, Internetworking, and Technology Website [Online]. Available: http://www.washington.edu/doit/how-can-people-who-are-blind-operate-computers

[6] IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.