ASHESI UNIVERSITY COLLEGE

GHANAIAN LANGUAGE INTEGRATION ON ANDROID SMARTPHONES

NATHAN ADLAM FLETCHER

2014
Applied Project

ASHESI UNIVERSITY COLLEGE

GHANAIAN LANGUAGE INTEGRATION ON SMARTPHONES

By

NATHAN ADLAM FLETCHER

Dissertation submitted to the Department of Computer Science,

Ashesi University College.

In partial fulfilment of Science degree in Computer Science

JULY 2014

Declaration

I hereby declare that this dissertation is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature:……

Candidate's Name:……

Date:……..

I hereby declare that the preparation and presentation of the dissertation were supervised in accordance with the guidelines on supervision of dissertation laid down by Ashesi University College.

Supervisor's Signature:……..

Supervisor's Name:…….

Date:……..

Acknowledgements

My sincerest thanks goes to my parents Mrs. Olivia Fletcher and Mr. Norman Arthur Fletcher and my family for their support and constant help in seeing me through this project and my education in college. They have not let a moment of much needed help go by through my time spent in college. Sincerest thanks also goes to my supervisors Dr Nathan Amanquah and Prof. Aelaf Dafla. The advice I have received from both of you in and out of class cannot be compared to any other institution.

I am grateful to my friends and beta testers Thelma, Esinam, Jessica, Maame, Martha, Jeffery, Anna-Lisa and especially Peter whose technical advice contributed into building of the User Interface of this application.

Abstract

The main aim of this project is to integrate local languages spoken and written in Ghana into the input method services of smartphones. The high penetration of mobile devices in and across Africa continues to be an interesting phenomenon and this does not exclude smartphones. Since the introduction of smartphones, their production and customization has been skewed towards users and consumers in the West. Therefore the localization of the features of smartphones for the African environment has been very slowed if not stopped. The major result of this is that text based mobile messaging is limited to characters of only the dominant languages of the world such as English, French, Spanish and Portuguese. This leaves consumers especially in Ghana to resort to replacing the special characters of their local language with either numbers of symbols or totally ignore their use. This might conflict with the correct use of the local language in context.

The goal of this project is to right this by implementing an input method service also known as a soft keyboard (SoftKeyboard) for the android smartphone platform which would represent the special characters in local languages spoken in Ghana. The end result should be a user-friendly and intuitive and interactive keyboard that satisfies Akan, which is a major spoken language across Ghana and other select languages in the country.

Table of Contents

CHAPTER 1

INTRODUCTION

The population of Ghana consists of about 75 ethnic groups [1] some with similar and others with totally distinct languages from each other. Thus the Government of Ghana in its basic education initiative included the study of Ghanaian languages as a compulsory subject throughout primary and JSS [2]. In the end, the West African Exams Council conducts an exam of all subjects which include the local language studied by candidates before they move on to the Secondary Level of their education, after they have passed their Basic Education Certificate Examination. It is thus safe to conclude that every Ghanaian who has gone through basic education in Ghana is literate in the reading and writing of one local language.

With the rise in the use of mobile technology and mobile internet in Ghana, it is necessary for users to communicate in their local language both in speech and in text. Also I have observed that the increase in usage of social media like Facebook, Twitter and WhatsApp has allowed the youth, who mostly use them, to express themselves freely sometimes in their native languages. However the support of technologies such as word processing software and hardware like laptops for local languages is limited in Ghana to a few organizations such as the Kasahorow Foundation [3]. This therefore presents a need for more efforts to be put into the propagation of localization in technology especially in Ghana. This project seeks to achieve that with mobile technology as its focus.

This project is going to do this by analyzing the common languages spoken in Ghana and effectively come out with a keyboard input method for

smartphones which would represent most languages spoken in Ghana. Each identified major language spoken in Ghana would be represented as an individual layout and then a hybrid layout would be designed to integrate the selected languages into one layout. The chosen platform for the implementation of this project is the Android Operating System.

1.1 Objective

This project seeks to integrate the literacy of Ghanaian languages into modern technology. The problem that has been identified is the lack of adaptation of Ghanaian literature with technology, especially mobile technology. The evidence of this can be seen in the social media where local language words are misspelled and the special characters of the local languages such as 'ɛ' and 'ɔ' are replaced by the number three ('*3*') and the closing bracket ('*)*'). A second evidence of the lack of technical support for local languages in technology is the absence of a proper spell check and language pack for Ghanaian languages on any of the major mobile Operating Systems namely Android, Windows Phone, iOS and Blackberry. One can observe this when they look through the language settings of these mobile operating systems and not find Akan, Ga or Ewe in the list of languages provided as at the time of this research. In addition to this, there is a lack of a local language dictionary and spell check support on these platforms. Consequently it is very common to find present language packs both on computers and on mobile suggesting English words in place of local language words. It is only recently that a few organizations and groups of people have taken initiative to fix this such as the Google Developer Group in Ghana which have started compiling an open online database of translated words from English to select local languages [4]. This project

intends to do this by implementing an input method service for smartphones (android) which is adaptable to the local languages spoken and written across the country.

1.2 Previous Work

The problem this project addresses is a common one worldwide. The number of languages and their undocumented representations within technology is numerous. As such there have been attempts by these under-represented dialects to make their mark. The "Urdu Word Prediction System for Mobile Phones" [5]. The purpose of the project was to determine a way to type faster in Urdu which is one of the national languages of Pakistan. Similar to the goals of this project, that word prediction system proposed to develop algorithms to make Urdu much more comfortable to use on mobile devices. One thing identified in that project was that the research was open to the total alteration of the present keypad layout in order to achieve their goal. This project would attempt to design keyboard layouts for local languages on the android platform.

Other organizations have also attempted to integrate local languages spoken in Ghana into their systems. For example, in the fall of 2011 I participated in a Google Africa program which sought to translate the Google search homepage to local languages such as Hausa, Ga, Ewe and Akan. Also the efforts of Kasahorow has gone through great lengths to aggregate the use of their patch software which enables one to type the special language characters on the input devices of desktops and laptops [3]. This project seeks to contribute to the pile of effort to right this wrong

and make a positive impact in the integration of Ghanaian languages in technology.

CHAPTER 2

Design

2.1 Requirements

In order to create an acceptable input method service, it is important to fully identify what the requirements of the service are and to separate the functional or essential ones from the non-functional ones. Therefore listed below are the identified requirements for this project.

2.1.1 Functional requirements

1. The keyboard layout for Ghanaian languages should have a user-friendly layout and be easy to use.
    a. First things to be identified are all the characters that make up the Ghanaian language. Then an essential arrangement of these characters for the keyboard layout has to be designed.
2. The keyboard should combine English and Ghanaian languages
    a. In Ghana, most people speak an average of two languages. This is because English is not the first language of many Ghanaians. Therefore there is the need to implement another keyboard layout which combines both English and the local language to satisfy the Ghanaian language literate users of the application. It was observed that a casual conversation in Ghana consists of mixing both English and a local language. This layout would compensate for the presence of the mixed dialect in text messaging and other text functions. It would also be ergonomically feasible and convenient to use one

layout to type out this mixed language feature instead of switching layouts when needed.

3. Include default English keyboard layout in input method service
    a. The reason for this decision is to satisfy users who cannot type in only the Akan language and would need the English language QWERTY keyboard layout. Such users primarily would use the English keyboard and type in their local language occasionally. However this layout would be a secondary layout to serve as a smooth transition onto the mixed language layout mentioned earlier, which would be presented to the user as the default layout. All layouts would however receive the local language dictionary and word suggestion and correction feature.
4. The service should be able to be extended to accommodate more languages
    a. Though Akan is a major language spoken across Ghana, there are about 75 ethnic groups in Ghana [6]. Some of these ethnic groups speak languages which are variations of Akan. Others too speak totally different languages. This input method service should be able to extend its capacity to more local languages spoken in Ghana.
5. Implement a dictionary and word suggestion service for the various keyboard layouts.
    a. This implementation seeks to educate and increase the literacy of users within the language on the correct spellings of words where necessary.

2.1.2 Non-Functional requirements

1. Implement user-friendly design for the service
2. Provide settings UI to change user preferences for the input method service.
3. The application should be optimized to respond quickly.

The Android mobile platform is an extensive open source platform which is constantly changing and improving in both design and functional implementation. Therefore it is necessary to study how input works on this platform for each version and know what constraints might be faced. This led to a list of tasks stated below which were required to be done in order to successfully complete this project.

- The Input method service of the Android Operating System had to be studied
- Research on existing layouts for local languages chosen for the project
- Re-Design / Maintain layouts according to results of research and testing
- Experiment / Test layout and obtain feedback on its usability
- Implement Dictionary / Word Suggestion service to ensure correct input
- Word prediction as the Urdu prediction did was intended to be implemented however this was sidelined because of time constraints

2.2 Design Process

The iterative steps used in designing this was centered on defining, designing and building. Defining includes research and analysis, designing includes sketching, modeling, concepts and diagrams, while building involves prototyping and materializing of designs. In order to effectively design the user interface for local languages and the hybrid keyboard, other layouts of other languages had to be sampled. They were sampled to gather how they interacted with their users in terms of special characters and input processes. Some language keyboard layouts had characters less than the usual 26 letter layout of the QWERTY keyboard. Other languages like Mandarin had way more characters for their keyboard layout. However designers of these layouts found ways round complex problems they faced.

2.3 Research performed

2.3.1 Modern keyboard layout designs

In order to effectively design the different keyboard layouts stated, it was necessary to study how keyboard layouts came into being. The typewriter was designed by an inventor called Christopher Latham Sholes in the year 1867 [7]. Typewriters then were machines that produced characters from a piece of engraved steel striking ink onto paper to form the character on the head of that steel. Back then, Christopher Sholes made various models of the typewriter to improve their mechanical efficiency and speed and to stop them from jamming. As such, the QWETY layout was finally formed after a few iterations of the typewriter. In fact, the first layout was designed in an alphabetical form but due to the frequency of the letters ‘t’ and ‘h’, they had to be split to prevent mechanical jams. One can conclude that the

current QWERTY layout did not consider ergonomics but pure mechanical efficiency [8].

The Dvorak keyboard layout was a keyboard layout designed with ergonomics in mind. It was invented by August Dvorak in the early 1930's who produced evidence that his layout was much faster however it was a bit too late for the Dvorak layout as the QWERTY keyboard had gained much more adoption worldwide since it came out first [8]. Modern research on the "Analysis of Alternate Keyboards using Learning Curves" concludes that, it would unfortunately take longer to adopt to alternative input methods such as the Dvorak keyboard as compared to other keyboard input methods in their research [9].

In conclusion, the QWERTY keyboard layout is here to stay and its worldwide adoption is proof of it. Therefore the design of the soft keyboard layouts for local languages for the purpose of this project should be centered on the QWERTY keyboard and make minor design changes to alter its look and feel to make its use seamless on mobile devices already using QWERTY.

2.3.2 Soft keyboard layout design concepts

Research for the keyboard layout for the Akan language led to the University of Ghana language center and the department of linguistics. It was discovered that there existed special typewriters which were used to type out literature of various kinds in local languages. However, the department of linguistics did not have a model to show me. The reason for this was that, the department had discontinued the use of those typewriters and had moved on to the use of PC's with the Kasahorow local language

keyboard plugin and as such, the typewriters had been discarded. The layout from that typewriter would have been the intended layout to be used for the Akan soft keyboard layout for consistency. The Kasahorow plugin after installation on PC replaces the '{' and '}' symbols on the qwerty keyboard with the special Akan characters 'ɛ' and 'ɔ'. It also replaces the dollar symbol with the Ghana Cedi character ('₵'). Another asset obtained from the research was the official Akan Dictionary [10] which was a pilot project of about 1530 defined Akan words. This dictionary was published by the Department of Linguistics at the University of Ghana.

2.4 Akan soft keyboard layout.

The unavailability of the Akan typewriter layout provided the opportunity to formulate a custom layout to represent it. The initial approach was to eliminate all letters of the English alphabet that were not present in the Akan alphabet which included '*q*', '*j*', '*c*', '*v*' and '*z*'. The next step was to arrange the remaining letters in the layout of the QWERTY format and include the special characters of the Akan language i.e. 'ɛ' and 'ɔ'. The positioning of these special characters was supposed to be intuitive and seamless to the average Akan literate user. Therefore the character 'ɛ' was placed after '*e*' and the character 'ɔ' was placed after '*o*'. This placement was due to the fact that '*e*' and '*o*' preceded 'ɛ' and 'ɔ' in the order of the Akan alphabet. The biggest error that surfaces was that doing this made the top row above the home row of keys overly crowded shown in Figure 1: Initial Akan Keyboard Layout. There was thus no balance between the

number of keys on the top row in relation to the home row keys and the bottom row of keys.

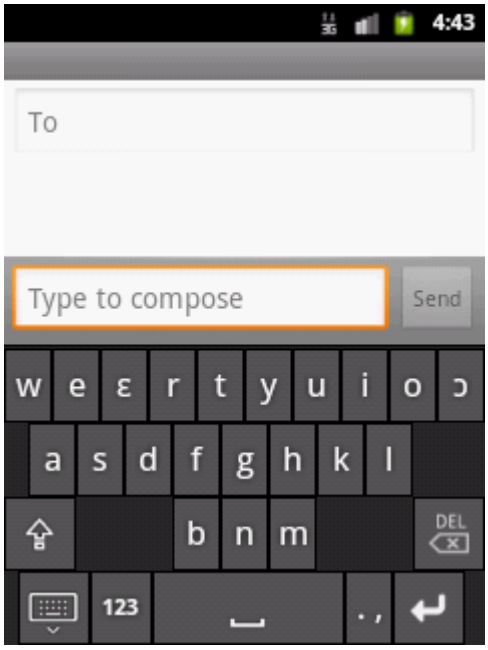

Figure 1: Initial Akan Keyboard Layout

The logic of this system changed to allow the extra characters to be placed around their preceding letter in the alphabet. This would then allow the dropping of either of the two characters from the top row to the home row as long as they were in proximity to their preceding character in the Akan alphabet, as shown in Figure 2: Akan Keyboard Final Layout.

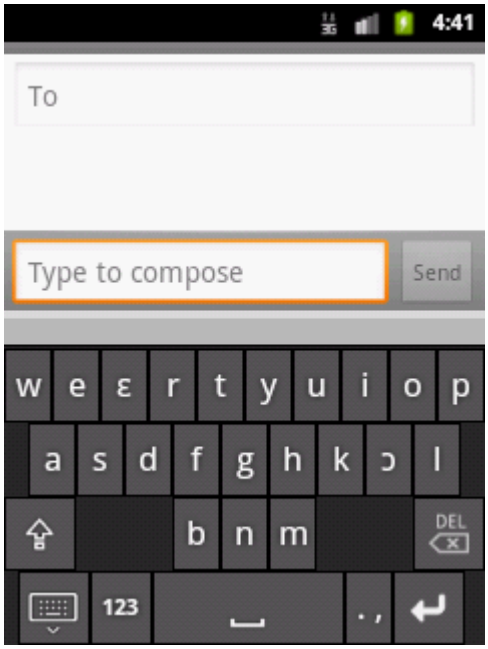

Figure 2: Akan Keyboard Final Layout

2.4.1 Akan Variation issues

As mentioned earlier, some of the 75 ethnic groups in Ghana possess similarities in their languages. [6] The major example here is in Akan which consists of sub ethnic groups or dialects namely 'Fante', 'Asante' and 'Akuapem' etc. This brought about a problem where some consonants such as 'z' would be present in one languages ethnic group but not the other. In order to determine a constant format, the Akan dictionary purchased was used as a standard to determine which consonants would be included and vice versa [10] . The Akan dictionary provided how a word should essentially be spelt in Akan, alongside its phonetic transcription in the three major dialects mentioned above. The phonetic transcription is how the word is pronounced in the various Akan dialects. This thus provides a standard way of spelling Akan words which does not include consonants such as 'z' and 'j' found in Fante and a standard alphabet for all Akan dialects [10].

2.5 Hybrid keyboard layout

The hybrid keyboard to be designed would stand to represent a dual language format for typing. A normal cordial conversation in Ghana is not exclusively in one language. It would usually start in a preferred language (English or the local language) and then occasionally switch to a local language commonly understood by those in the conversation. Then there would most likely be a weave of switches between languages in order to express themselves better. That is the purpose of this keyboard layout; to cater for the constant moves between languages spoken in Ghana and then effectively represent that on the keyboard.

According to the summary report of the 2010 population census in Ghana, the dominant ethnic groups in Ghana are the Akan, Mole Dagbani, Ewe and Ga-Dangme. The alphabet of these ethnic groups contain similarities especially with the special characters which make up each language. These characters include but not limited to ɛ, ɔ, Ŋ, ʒ and ɣ. Some of these are present across the ethnic groups mentioned above such as the ɛ and ɔ. Therefore this hybrid layout proposed would present these two as primary keys on the layout. This means that they would be visible and would be placed according to the results of the final layout of the Akan keyboard.

From the alphabet figures below, one can observe that the special characters resemble some of the default letters of the English alphabet thus the remainder of the special characters would be converted into secondary characters of the letters in the English alphabet they resemble. Thus, they would be accessed by the "long-press" gesture of the keyboard layout. However, that was not implemented in this project but still remains as a consideration for future iterations of this project.

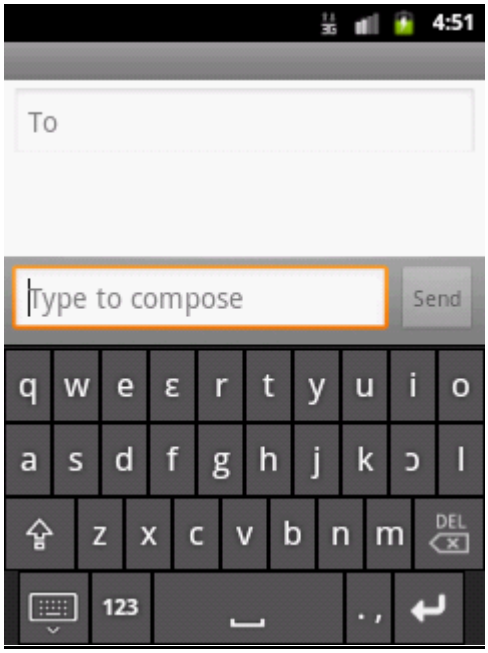

Figure 3: Hybrid Keyboard Layout

A a	B b	D d	Ɖ ɖ	Dz dz	E e	Ɛ ɛ	F f	Ƒ ƒ	G g	Gb gb	Ɣ ɣ
[a]	[b]	[d]	[ɖ]	[dz]	[e]	[ɛ]	[f]	[ɸ]	[g]	[gb]	[ɣ]
H h	I i	K k	Kp kp	L l	M m	N n	Ny ny	Ŋ ŋ	O o	Ɔ ɔ	P p
[h]	[i]	[k]	[kp]	[l]	[m]	[n]	[ɲ]	[ŋ]	[o]	[ɔ]	[p]
R r	S s	T t	Ts ts	U u	V v	Ʋ ʋ	W w	X x	Y y	Z z	
[ɾ]	[s]	[t]	[ʦ]	[u]	[v]	[β]	[w]	[x]	[y]	[z]	

Figure 4: Ewe Alphabet [11]

A a B b D d E e Ɛ ɛ F f G g H h I i J j K k L l M m
N n Ŋ ŋ O o Ɔ ɔ P p R r S s T t U u V v W w Y y Z z

Figure 5: Ga Alphabet [10]

A a B b D d E e Ɛ ɛ F f G g H h I i K k L l M m
N n O o Ɔ ɔ P p R r S s T t U u *V v* W w Y y

Figure 6: Akan Alphabet [10]

A a	B b	Ch ch	D d	Dz dz	E e	Ɛ ɛ	F f	G g
[aː]	[b]	[ʧ]	[d]	[dz]	[eː]	[ɛ]	[f]	[g]
Gb gb	Ɣ ɣ	H h	I i	J j	K k	Kp kp	L l	M m
[g͡b]	[ɣ]	[h]	[iː/ɪ]	[j]	[k]	[k͡p]	[l]	[m]
N n	Ny ny	Ŋ ŋ	O o	Ɔ ɔ	P p	R r	S s	Sh sh
[n]	[ɲ]	[ŋ]	[oː]	[oː]	[p]	[r]	[s]	[ʃ]
T t	U u	W w	Y y	Z z	Ʒ ʒ	’		
[t]	[uː/ʊ]	[w]	[y]	[z]	[ʒ]	[ʔ]		

Figure 7: Dagbani (Dagomba) Alphabet [10]

Figures 4 through to 7 show the alphabets of Akan, Ga, Ewe and Dagbani. Figure 4: Ewe Alphabet is the phonetic alphabet for Ewe while the rest represent the alphabet of their various dialects. Notice that in all of the dialects, they have the letters ɛ and ɔ present in all of them alongside some of the native Latin letters. In the Ewe phonetic alphabet, the Dz, Gb and Kp as well as all double letter combinations represent basic phonetic sounds used in the dialect but not present in the alphabet. However the double letter combinations found in Dagbani in Figure 7: Dagbani (Dagomba) Alphabet [10] are part of the Dagbani alphabet.

2.6 Function keys

Extra function keys that were included in the design consist of a ‘shift’ key (Figure 8: Shift function key to switch between upper case and lower case characters) to toggle between upper case and lower case lettering, a ‘symbols’ key (Figure 1) which toggles between the symbol-numeric keyboard and the alphabetic keyboard and finally a language switch key

(Figure 11) to hop through the different keyboard layouts present in the application. Also present were the 'backspace' and 'enter' keys (Figure 9) which are considered standard on all keyboard layouts.

Figure 8: Shift function key to switch between upper case and lower case characters

Figure 9: Enter key to move to a new line of text

Figure 10: Symbols key to switch to symbol and numeric layout

Figure 11: Language Switch key to switch between the layouts available on the keyboard

CHAPTER 3

Implementation

In order to implement this on the Android OS, the Android Developer Toolkit bundle (ADT) was downloaded from the Android developer website. The creation of a local language keyboard would require taking advantage of the input method services provided by the Android OS. The Android Developer website provided great tools, sample code and tutorial to get me started on the development of the input method service. However it was not very understandable all the time therefore I had to rely on tutorials from other websites such as GitHub [12] and Vogella [13]. The foundation of the input method service was gotten from the Android open source samples on input method services and soft keyboard samples. The term "input method service" is the label given to any Android application which seeks to take over the default keyboard service of the Android Operating system in any form be it numeric (the phone dialer keypad for example) or alphanumeric. Consequently it becomes another keyboard from a third party. The IDE used to develop this is Eclipse Kepler v22.4 but much later on, I had to upgrade to version 22.6 in order to cater for cross version issues in the Android Developer Toolkit (ADT).

This approach meant that the input method would be built from scratch. Another way that this task could have been implemented was to build on top of the AnySoftKeyboard application for Android. This application is an open source application that allows developers to build and develop their own input method service with their own layout and dictionary completion. It has had many iterations for different languages and is the base software that the mobile version of the Kasahorow application [14] [15]. It should

be noted that at the start of this project, the mobile version of the Kasahorow application had not been implemented. Even though these methods mentioned exist now, it should be noted that work on this project was already underway before the launch of the Kasahorow application.

3.1 Structure and class definitions

In order to implement the input method service for this project, it should be built as an Android application. The file and folder structure of Android applications include the android manifest file (*manifest.xml*), the source folder which contains the java code and programming logic (*src/*) and the resource folder which in this case contains the different keyboard layouts and the main app settings interface (*res/*).

As mentioned early on, some classes to develop the keyboard application were gotten from sample code provided by the Android SDK API level 10. The specific sample project extended for this is known as "SoftKeyboard" [16]. Classes used in the implementation of the app include *CandidateView*, *LatinKeyboard, LatinKeyboardView* and *SoftKeyboard* [16]. These were provided by the SoftKeyboard sample project by the Android API 10 included in the SDK. They work together to implement a Standard English keyboard. The LatinKeyboard file is responsible for handling the character codes provided by the layout xml and translate them from Unicode to characters. The LatinKeyboardView file is responsible for inflating the xml layout onto the screen. These work hand in hand to make sure that the keyboard is always displayed correctly on the screen. Finally, the CandidateView class is responsible for how the suggestions are displayed on the screen and last but not the least the manifext.xml file provides

properties of the application to the Android OS. It also outlines sections of the application and how they are connected with each other. Finally, the manifest.xml file tells the Android OS which permissions that application would need to run. Sample permissions include internet access, GPS and services such as the input method service which would be used in this application. In order to include a service, the service tags should be opened and defined in the manifest file. For this project the permission property value to include the application as an input method service is "android.permission.BIND_INPUT_METHOD". This had to be included in the manifest file (see figure8). These conditions in the manifest file are provided to the Android OS during installation of the application and a link is created in the input settings of the device to include the app as a service which can be selected and used. All this is detected and efficiently handled by the Android OS so that it is included in the "Language and Keyboard" section in the settings of the operating system. Figure 12: Manifest file of the application.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.example.android.softkeyboard"
        android:versionCode="1"
        android:versionName="1.0">
    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="19" />
    <supports-screens
        android:anyDensity="true"
        android:largeScreens="true"
        android:normalScreens="true"
        android:smallScreens="true" />
    <application android:label="@string/ime_name"
        android:allowBackup="true"
        android:icon="@drawable/sym_keyboard_done">
        <activity
            android:label="@string/app_name"
            android:name=".Main" >
            <intent-filter >
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <service android:name="SoftKeyboard"
                android:permission="android.permission.BIND_INPUT_METHOD"
                android:configChanges="orientation"
                android:alwaysRetainTaskState="true">
            <intent-filter>
                <action android:name="android.view.InputMethod" />
            </intent-filter>
            <meta-data android:name="android.view.im" android:resource="@xml/method" />
        </service>
    </application>
</manifest>
```

Figure 12: Manifest file of the application

Figure 12 above shows the manifest file of the app. The highlighted section shows the service opening and closing tags which tell the android OS that the application is an input method. It requires permission from the OS to "BIND_INPUT_METHOD". This would let the user know through an alert that the app has the ability to store information inputted by them, when they select the keyboard as the default input method. This alert is a precautionary measure by the Android OS to make sure that the user has agreed to any information that could be sent out through the application, and this is a regular feature of the Android OS.

The task at hand was to extend this to include Ghanaian languages and also include a dictionary and word suggestion system on the keyboard. In order

to do this, I first had to implement the Akan keyboard layout. As mentioned earlier, the keyboard layouts are located in the resource folder. The xml file which was created for the Akan layout made the necessary adjustments for Akan and included the Unicode values for the special characters in Akan which are 'ɛ' and 'ɔ'. Android represents these characters with their Unicode values in the xml file however it does not represent all Unicode values. This affected the upper case versions of 'ɛ' and 'ɔ' and they did not show in the uppercase layout of both the local language keyboard and the hybrid keyboard. Therefore the small cases of 'ɛ' and 'ɔ' would be used for the upper case versions for now. This is because their Unicode values are not included in Android OS versions up until Android 4.2 i.e. API level 17 and later versions. The implication of this is that, any older Android device which installs this application would not have the upper cases of 'ɛ' and 'ɔ' showing on their keyboard.

```xml
<?xml version="1.0" encoding="utf-8"?>

<Keyboard xmlns:android="http://schemas.android.com/apk/res/android"
    android:keyWidth="10%p"
    android:horizontalGap="0px"
    android:verticalGap="0px"
    android:keyHeight="@dimen/key_height"
    >
    <Row>
        <Key android:codes="119" android:keyLabel="w" android:keyEdgeFlags="left"/>
        <Key android:codes="101" android:keyLabel="e"/>
        <Key android:codes="603" android:keyLabel="ɛ"/>
        <Key android:codes="114" android:keyLabel="r"/>
        <Key android:codes="116" android:keyLabel="t"/>
        <Key android:codes="121" android:keyLabel="y"/>
        <Key android:codes="117" android:keyLabel="u"/>
        <Key android:codes="105" android:keyLabel="i"/>
        <Key android:codes="111" android:keyLabel="o"/>
        <Key android:codes="112" android:keyLabel="p" android:keyEdgeFlags="right"/>
    </Row>
    <Row>
        <Key android:codes="97" android:keyLabel="a" android:horizontalGap="5%p"
            android:keyEdgeFlags="left"/>
        <Key android:codes="115" android:keyLabel="s"/>
        <Key android:codes="100" android:keyLabel="d"/>
        <Key android:codes="102" android:keyLabel="f"/>
        <Key android:codes="103" android:keyLabel="g"/>
        <Key android:codes="104" android:keyLabel="h"/>
        <Key android:codes="107" android:keyLabel="k"/>
        <Key android:codes="596" android:keyLabel="ɔ"/>
        <Key android:codes="108" android:keyLabel="l" android:keyEdgeFlags="right"/>
    </Row>
    <Row>
        <Key android:codes="-1" android:keyIcon="@drawable/sym_keyboard_shift"
            android:keyWidth="15%p" android:isModifier="true"
            android:isSticky="true" android:keyEdgeFlags="left" />
```

Figure 13: Akan Keyboard xml layout

The format shown in the code snippet in Figure 13 is the structure used to create the Akan layout. It has been extended to include all the other Ghanaian languages mentioned to be implemented. Also implemented was a hybrid keyboard layout for both English and Ghanaian languages. This was done by first implementing a default QWERTY layout and then including the extra characters found in the various Ghanaian languages. The result of this was that it made the keyboard layout look cluttered therefore a different strategy was adopted. The alphabet of the various languages in

Ghana (see figures 4 through to 7) have a lot of the special characters appearing in each language. The characters 'ɛ' and 'ɔ' are present in Akan, Ewe, Ga and Dagbani therefore, the strategy adopted was to include just those two characters in the hybrid keyboard. The hypotheses developed was that these characters would give a fair representation of languages across Ghana since they are common in these major ethnic groups. The keyboard layout xml file is provided in the figure below.

```
<Keyboard xmlns:android="http://schemas.android.com/apk/res/android"
    android:keyWidth="10%p"
    android:horizontalGap="0px"
    android:verticalGap="3px"
    android:keyHeight="@dimen/key_height"
    ><!-- unit is %p -->

    <Row>
        <Key android:codes="113" android:keyLabel="q" />
        <Key android:codes="119" android:keyLabel="w"/>
        <Key android:codes="101" android:keyLabel="e"/>
        <Key android:codes="603" android:keyLabel="ɛ"/>
        <Key android:codes="114" android:keyLabel="r"/>
        <Key android:codes="116" android:keyLabel="t"/>
        <Key android:codes="121" android:keyLabel="y"/>
        <Key android:codes="117" android:keyLabel="u"/>
        <Key android:codes="105" android:keyLabel="i"/>
        <Key android:codes="111" android:keyLabel="o"/>
        <Key android:codes="112" android:keyLabel="p" android:keyEdgeFlags="right"/>
    </Row>

    <Row>
        <Key android:codes="97" android:keyLabel="a"
             android:keyEdgeFlags="left"/>
        <Key android:codes="115" android:keyLabel="s"/>
        <Key android:codes="100" android:keyLabel="d"/>
        <Key android:codes="102" android:keyLabel="f"/>
        <Key android:codes="103" android:keyLabel="g"/>
        <Key android:codes="104" android:keyLabel="h"/>
        <Key android:codes="106" android:keyLabel="j"/>
        <Key android:codes="107" android:keyLabel="k"/>
        <Key android:codes="596" android:keyLabel="ɔ"/>
        <Key android:codes="108" android:keyLabel="l" android:keyEdgeFlags="right"/>
    </Row>
```

Figure 14: Hybrid keyboard layout for Ghanaian languages

3.2 Classes Developed

In order to represent local languages in Ghana efficiently it was necessary to build classes that would support word suggestion and storage. The classes developed to support the dictionary database and word suggestion

system include Word and DatabaseHandlerWord. They work by storing a list of words in the Ghanaian languages into a SQLite database and retrieve a list of words that look like the current word being typed in the input in view at the moment. The list of words from the Ghanaian languages used to populate the SQLite database were gotten from a spreadsheet compiled by the Google Developer Group project to translate a list of English words to Akan, Ga and Ewe [4]. Akan words were sorted out and cross checked with words provided in the Akan dictionary previously purchased.

3.2.1 Word Class

The "Word" class is an object class that represents words that would be stored into the database and retrieved as word suggestions as the user types on the keyboard. The class has a constructor that define the object. The second constructor is "**public** Word(String word, **double** freq)" is particularly useful. It creates a word and assigns a frequency value to the word. This frequency value would determine which word would be suggested to the user first as the keyboard is being typed on. The input parameters are the word itself represented as a string and the frequency of the word which is a double value that sets how frequent words are used in order to suggest them first when retrieving word suggestions. This class implements getter and setter methods that can set a single word to a Word object, retrieve that word from the object, set a new frequency value for that particular word, and retrieve a the frequency value for that particular word.

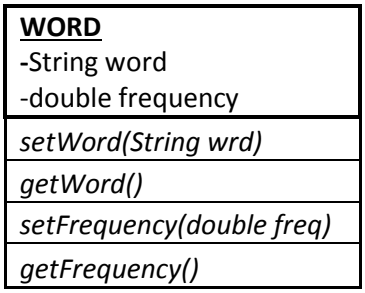

Figure 15: Class diagram describing Word Class.

3.2.2 DatabaseHandlerWord Class

The DatabaseHandlerWord class is the class responsible for putting words into the SQLITE database. This class creates the various tables and implements methods that create the database (*onCreate()*), retrieve a single word or entry (*getWord(String wrd)*), add words to the database (*addWord(Word wrd, String tableName)*) and retrieves a list of all words similar to the input parameter. These are methods used during the active use of the keyboard by the user. There are other methods that were developed to test the class to see if it was working accordingly. These classes include getAllwords(String tableName, String actualWord) which lists out all words in the database and getWordCount(String tableName) which returns the number of words in a particular table in the databse. Due to time constraints the method updateWord(Word wrd) was not implemented. This class uses the Word class mentioned above to add words and their frequencies to the database and queries the tables to retrieve words as necessary. The database schema is composed of tables which hold words from different languages. Due to the availability of the Akan words, only the Akan table was created to represent Ghanaian languages. The tables in the schema are English, Akan and Hybrid. Each table has two columns Word and Frequency just like the Word class described above.

3.2.3 Main Class

This class was created to perform the insertion of words into the database when the application is installed. It declared the DatabaseHandlerWord and Word objects and inputs Akan words into the database from a text file which contains a list of Akan words. English words were not added to the database as it takes a long time to record them into the database. This is because the database of words had to be built from a text file which contained a list of about five thousand English words. This had to be done every time the application was installed to be tested and took between thirty to forty minutes to complete. This process usually fails because the test phone used would run out of memory. The limited time for this project would however not allow a much more efficient solution to be pursued. The main class is connected to a main activity in the manifest file which launches it once the application is installed.

3.2.4 CollectSuggestions Class

This is a private class which was created in the Softkeyboard class provided by the Android sample mentioned early on. It was made to be able to read from the database of stored words and return a list of suggested to the user as they typed. Its main job is to constantly call methods from the database that query the table to return words that are similar to the current word being typed in the database. As shown in Figure 16, this private class extends AsyncTask which is a class in Android that performs tasks on a separate thread from the main thread which the program runs on, and returns a result. It then passes the result which is a List to the setSuggestions() method in the SoftKeyboard class.

```
//AsyncTask extension of collectWords so that the whole process is threaded. This is an inner class
private class CollectSuggestions extends  AsyncTask <String, Void ,ArrayList<String>>{
    ArrayList<String> list = new ArrayList<String>();

    @Override
    protected ArrayList<String> doInBackground(String... arg0) {
        DatabaseHandlerWord mdb = new DatabaseHandlerWord(getApplication());
        //Toast.makeText(getApplicationContext(), "Currently Composing "+ mComposing.toString(), Toast.LENGTH_SHORT).show();
        String sudoResult =  mComposing.toString();
        if (words != null){
            words = mdb.getAllwords("akan", sudoResult);
        }
        Object[] wd = new Object[words.size()];
        wd = words.toArray();


        for (int i=0; i <= wd.length-1; i++){
            list.add(((Word) wd[i]).getWord());
        }
        //db.close();
        return list;
    }

    protected void onPostExecute(ArrayList<String> wordList){
        setSuggestions(list,true,true);
        //updateCandidates();
    }

}
```

Figure 16: Code snippet for CollectSuggestions class

3.3 Challenges

Challenges faced during the development of the application include.

1. Finding the correct Android platform to work with. The special characters in Ghanaian languages are spread across the extended Latin character set. This character set was not included fully in all the iterations of the various Android platforms. Thus it was a challenge going through the different API versions of Android until the base platform was found which Android 4.2.2 is. Higher versions of Android after this supported the application.
2. Obtaining an adequate phone for testing. The test phone used for development had the version number 2.3 which refers to API level 10. Though the adequate version to run the app on is Android 4.2.2 and above i.e. API 17, API 10 supported displaying only the small letter cases of 'ɛ' and 'ɔ'. This made testing of upper cases quite

difficult as they only returned blank spaces. The upper cases of these character could be replaced by their smaller cases. However, in order to do that it would have to be stated in the xml layout file of the keyboard by inserting the Unicode value within the 'android:codes' attribute of the key tag in the xml layout file. See the highlited section of Figure 17.

```
<Keyboard xmlns:android="http://schemas.android.com/apk/res/android"
    android:keyWidth="10%p"
    android:horizontalGap="0px"
    android:verticalGap="0px"
    android:keyHeight="@dimen/key_height"
    >

    <Row>
        <Key android:codes="119" android:keyLabel="W" android:keyEdgeFlags="left"/>
        <Key android:codes="69" android:keyLabel="E"/>
        <Key android:keyOutputText="&#603;" android:codes="603" android:keyLabel="Ɛ"/>
        <Key android:codes="82" android:keyLabel="R"/>
        <Key android:codes="84" android:keyLabel="T"/>
        <Key android:codes="89" android:keyLabel="Y"/>
        <Key android:codes="85" android:keyLabel="U"/>
        <Key android:codes="73" android:keyLabel="I"/>
        <Key android:codes="79" android:keyLabel="O"/>
        <Key android:codes="80" android:keyLabel="P" android:keyEdgeFlags="right"/>
    </Row>
```

Figure 17: The highlighted section shows the upper case Ɛ being displayed however the 'android:codes' attribute has the value 603 which is the Unicode value of small case ɛ

CHAPTER 4

Testing

4.1 Installation

In order to install the application one has to copy the generated APK file on to the device. Navigating to the file on the phone and tapping it would open a prompt which would ask if the application is to be installed on the phone. By tapping on yes in the alert, the application would be installed on the phone

4.2 Activation

After installation, it is required that the application is activated and set as the default input method editor for the phone. In the settings menu, one would have to go to the Language and Input settings on the Android device. Once there, under the section named 'Keyboards and input methods' the name of the application would be part of the list of input methods installed on the device. For the purpose of this project, the application was nicknamed "Chale Keyboard". This is because, the word 'chale' is well known in Ghana to mean friend and since one of the goals of the app was to be user friendly, it was thus named 'Chale Keyboard'.

After tapping on the checkbox to enable the app, the Default button shown in Figure 18 has to be tapped and then app ('Chale Keyboard') would have to be selected as the default keyboard input for the device.

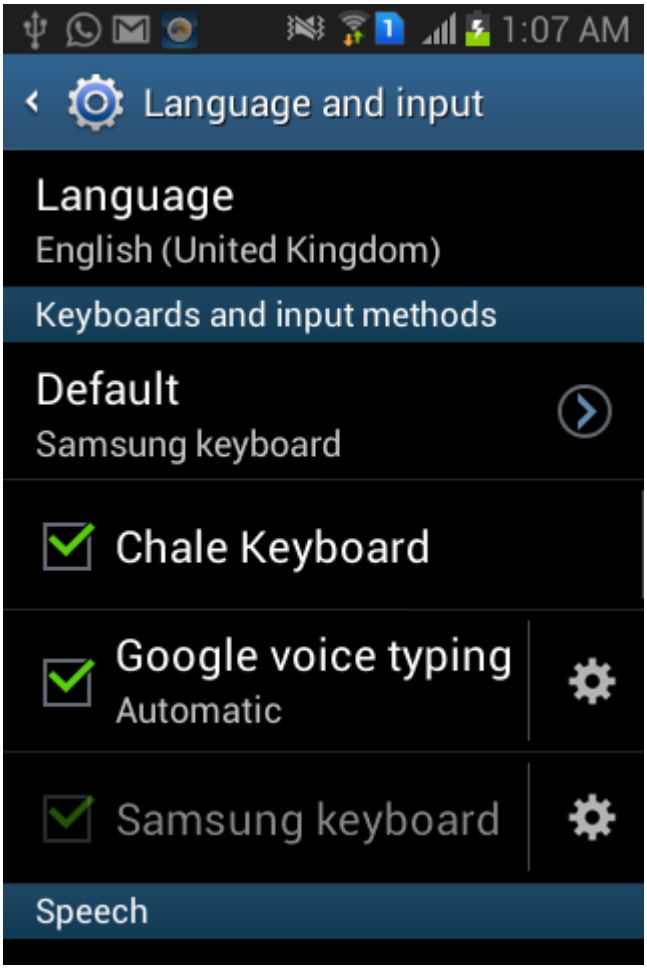

Figure 18: Activation process of the Local language keyboard

4.3 Alpha Testing

The application was used to create posts on social medial platforms such as Facebook, Google Plus, Twitter and chat applications install on the test phone like WhatsApp. All these were able to correctly represent the characters on their platforms.

4.4 Beta Testing

A lot of focus was not given to Beta testing of the application however after completing the Akan keyboard layout, I presented it to a few students on campus to have a go at it by making a draft of a text message. The results were quite remarkable. The users were very excited to see the local

language characters on the keyboard and without asking, started providing reasons why they would use it and why it was such a good idea. Snapshots of text message drafts are provided in the figures below.

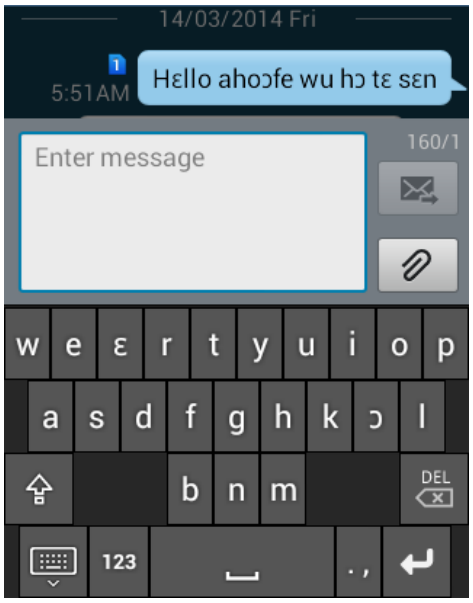

Figure 19: One text message sent by a Beta tester to a recipient on the test phone

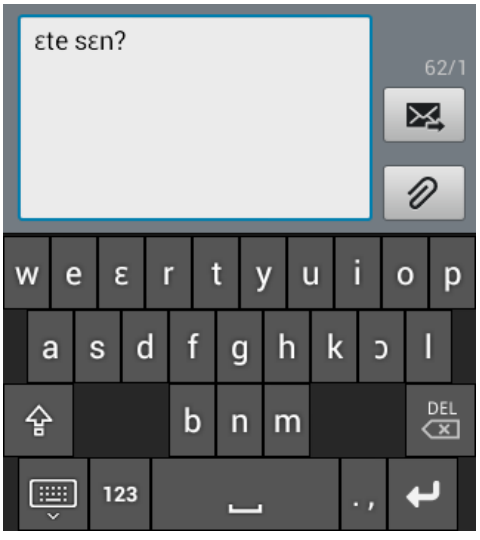

Figure 20: Draft of a message being sent by a Beta tester to a recipient

4.5 Experiment Performed

This section tries to determine the efficiency and effective use of the Ghanaian Language keyboard in a real world environment. This is done by performing a Beta test experiment on users to see how useable the application is by letting participants of the experiment type a three sentence passage on the keyboard application as though they were sending a text message to a friend. This experiment is similar to experiments performed to analyze alternate keyboards [9].

The test conducted is in four phases. The first phase is where the participant is given a three sentence English message to type on the keyboard as seen in Figure 24. They were to avoid the use of short hand and required to type the full form of words in the sentence. As they type, they would be timed to see how long it takes for them to type on the familiar QWERTY layout. After that, the timed result is recorded and they are given about a minute to rest while they are provided with the second phrase.

The second phase of the experiment is where the participant is given another three sentence passage to type. This time, the format of the passage is a mixture of English and Akan seen in Figure 23. Akan was selected as the preferred Ghanaian language to test with because it is dominant over other Ghanaian languages statistically especially in the southern part of Ghana and therefore it would be more probable to find participants who are literate in it [6] [2]. As mentioned early on, Ghanaians usually communicate with each other informally by mixing English with a local language. It was observed that when such communication is done on social media, they use the number three (‘3’) and the closing bracket (‘)’)

to represent the Ghanaian characters ‘ɛ’ and ‘ɔ’. Thus it makes the section of the experiment very essential. After that, the timed result is recorded and they are given about a minute to rest while they are provided with the third phase.

The third phase is where the same process is conducted once again however the three sentence passage is in Akan only in Figure 25. Finally the first process where the participant only types in English on the QWERTY keyboard is performed again as a control for the experiment. This control would serve as a determinant to see if the participant has adapted to the design and layout of the keyboard which might thus have affected typing speed. The results of the experiment were compiled into the table and graph in Figure 21 and Figure 22.

4.5.1 Experimental Process

The participants selected were picked randomly for the experiment. As mentioned early on, Akan was the selected Ghanaian language to be used for the experiment. Therefore randomly selected candidates were asked if they were literate in the Akan language before proceeding with the experiment. All candidates satisfied this condition. Another condition was that, candidates should have an android smartphone since this project focuses on the Android OS. This condition was to make sure that candidates were already comfortable with the phone and its default keyboard before the installation of the keyboard app. Once the app was installed on their device, it was set as the default input method service from the device settings. After installation, participants are given about two minutes to explore the keyboard and how its buttons work. Swipe was not implemented

on the Ghanaian keyboard and also word suggestions were not included in the version of the application used to test. Therefore this required participants to type on the keyboard by tapping individual characters. This would add to the test to determine how familiar the participants would be with the keyboard. At this point, they are encouraged to comment and ask questions whenever they are not sure of certain features. The experiment would allow the participants to be able to switch between the different keyboard layouts whenever they thought it was necessary when typing the different phrases. The experiment begins after this.

4.6 Results

During the implementation of the application a lot of observations were made about the Android OS in terms of its functionality and limits. One lesson learned was the different ways the operating system allowed the developer to perform the same tasks. For example, commands can be accessed either from the xml view or from the java code. This helped in making changes to the interface on the fly.

Another observation was the character encoding limits of early android versions. Up until android 4.2, not all Unicode values were represented in the OS. This limiting factor explains why not all characters were represented in the early versions of the OS. Finally, different brands customized and interpreted the functionality of the system in many different ways. This can be seen in the differences between the HTC brands of phones and the Samsung brand of phones. Some major models of HTC phones append extra whitespace characters to the ‘ɛ’ and ‘ɔ’ symbols of Ghanaian languages. This

makes it very difficult to use the application and totally flaws it when one tries to type Ghanaian language characters on them.

4.6.1 Experiment Results

Experiment Results: Completion time in seconds

	Time (seconds)			
Participant No	**English time**	**Mixed Language time**	**Akan only time**	**English time 2**
Participant 1	89	85	71	89
Participant 2	47	46	67	55
Participant 3	82	52	49	71
Participant 4	61	62	50	34

Figure 21: Results in seconds showing how long it took each participant to finish typing each text message

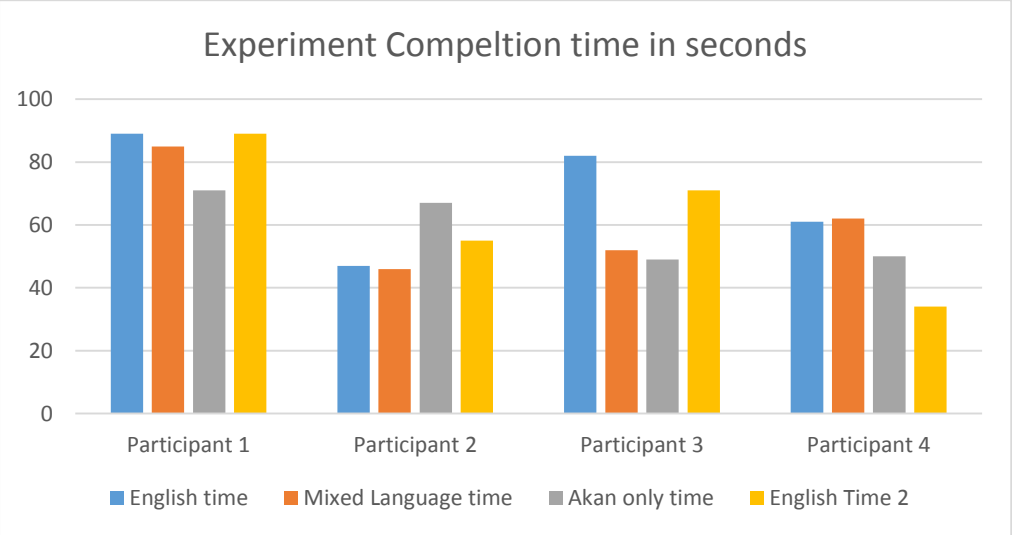

Figure 22: Bar graph representing timed results obtained in Figure 12

During the experiment, it was observed that three out of the four candidates used the hybrid keyboard when typing the English and mixed language phrases. They then switched to using the Akan only keyboard to type the Akan phrase. Two out of the four candidates requested to use word prediction and suggestion since they stated that they used it all the time to type on their phones.

The results show that text messages sent in English and mixed language (Figure 24 and Figure 23) did not have much completion time difference and an improved completion time in one candidate. The English text

Hi, how are you? Have you eaten this afternoon? I am very hungry and I want to eat beans with fried plantain.

Figure 24: English only passage used in experiment

message once typed again, showed either a maintained or reduced completion time. The Akan only text message showed a reduced completion time as compared to the other text messages with all participants except the Akan only result of Participant 2. According to the results of the data, one can conclude that the Ghanaian language keyboard application is useable for this small sample. However, this does not represent a general statistic. More testing has to be done to make more general statements.

Hi, ɛte sɛn? Have you eaten this afternoon? Me deɛ I am hungry oo na I want to eat kɔkɔɔ and beans.

Figure 23: Mixed language passage for experiment (English and Akan)

Ei, ɛte sɛn? W'adidi anɔpa yo? Me deɛ ɛkɔm de me oo na me pɛ sɛ me di yɔ kɛ gari.

Figure 25: Akan only passage for experiment

During the experiment, the use of the dictionary was not the main focus. The focus of the experiment was to determine how users interact with the keyboard interface and how they would tackle the tasks given to them. Enabling a dictionary or suggestions would not allow the participants to explore the full interface of the app.

4.6.2 Data Normalization

Though the phrases used in the experiment had the same meaning, they were not the same in terms of the number of characters they were made up of. Therefore in order to have more accurate results, the characters which made up the passages were normalized to a character count of a hundred. This means that the data would be adjusted to satisfy a character count of a hundred characters including white spaces. Figure 26 shows the new times obtained after characters in the passages had been normalized and Figure 27 shows its corresponding bar graph.

	Character count for text typed (including white spaces)			
	109	99	81	109
	Normalized Time for 100 characters (seconds)			
Participant No	**English time**	**Mixed Language time**	**Akan only time**	**English time 2**
Participant 1	81.7	85.9	87.7	81.7
Participant 2	43.1	46.5	82.7	50.5
Participant 3	75.2	52.5	60.5	65.1
Participant 4	56.0	62.6	61.7	31.2

Figure 26: Timed results after characters had been normalized

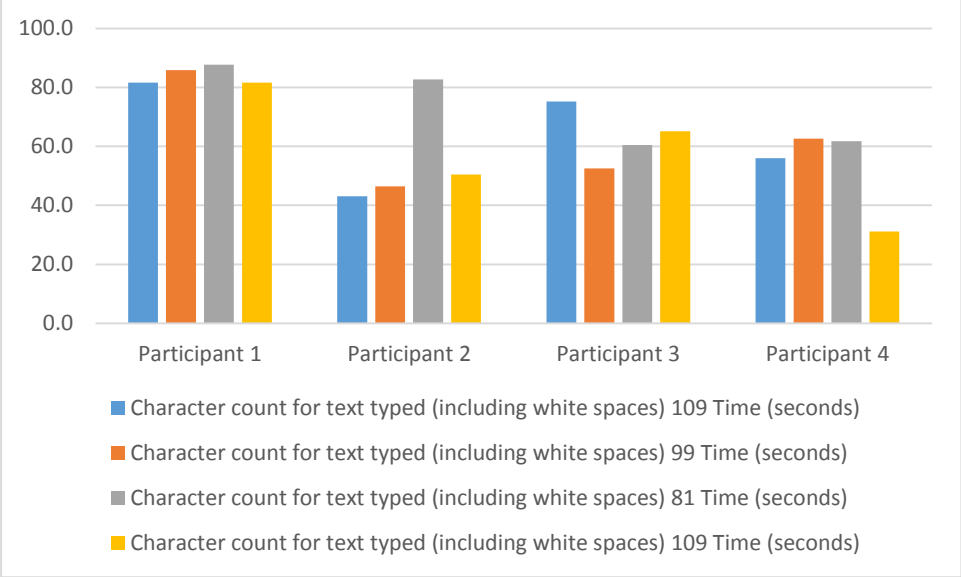

Figure 27: Corresponding graph for normalized data

The normalized results show that results show that it took marginally longer for the participants to type the Akan text however it was generally slightly better for participants to type the mixed language passage. This means that it would be easier for users to adapt to the hybrid keyboard.

4.6.3 Participant observations and suggestions

During the experiment, it was observed that some participants struggled to type the first English only passage. They kept on cycling through the layouts till they got to the English layout or realized they could just type it on the Hybrid layout. Most participants did not notice the difference between the Hybrid and English layouts. Pointing this out to them made them switch between these layouts when typing the Mixed and English only passages.

Both during and after the experiment, users were asked to give suggestions for the keyboard. A few of the responses are listed below.

- "I think it should be more intuitive and put all the ɛ and ɔ on the symbol layout. It should work like the emoji keyboard"
- "Does it have Swipe? I always use swipe. It makes me text without even looking on my phone"
- "Where is the dictionary? I use the dictionary. It's faster"
- "I think there shouldn't be multiple keyboard layouts. The button to switch language layouts should show just the special characters instead of cycling through the layouts whenever I want to say something"

These suggestions and responses show that there is the need for the full implementation of this project but though the sample size used for this experiment is not big enough, it give an idea of what to expect with larger tests and experiments.

CHAPTER 5

Conclusion and Recommendation

5.1 Conclusion

The development of this input method service for Ghanaian language integration unto smartphone devices especially in and across Ghana is far from perfect. However, the steady progress of work on its implementation and study shows that it is still a viable field which needs a lot of intellectual contribution in order to make its implementation perfect and necessary for effective adoption across the country. I believe that the project has been a success and the processes used to implement it can serve as a guide for expansion to local dialects of other countries. The following list states the various accomplishments of this project.

- An input method editor (IME) was implemented on the Android OS
- Developed a hybrid keyboard to input both English and Akan
- Implemented database and word classes to hold a dictionary for the keyboard.
- Experimented to test how feasible and useable this project would be if fully implemented

5.2 Recommendations

There are a few things I consider to be lacking in this application. When compared to other implementations of keyboards especially on the various app markets like SwiftKey [17] and GO Keyboard [18], they seem to have gone ahead to set a certain standard for third-party input services. Some of the features they both commonly poses include:

- Auto-correct: A feature across many softkeyboards which corrects the mistakes of users as they type.
- Custom themes: A customization feature which allows users to change the color theme of the keyboard to match the user's preferences.
- Swipe: A feature which produces the intended word the user wants. The user uses this feature by swiping their fingers sequentially along the letters that make up the word they intend to type.

The results of the tests performed, though a user test, provides an idea on the viability of the input method service and its adoption for Android smartphone users in the country. In the future, this test should be done on a larger sample of about a hundred people to get a clear reading on the efficiency of this keyboard project.

Works Cited

[1] Ghana Embassy Washington D.C USA, "Population," 2014. [Online]. Available: http://www.ghanaembassy.org/index.php?page=population.

[2] Ghana Education Service, "Basic Education Curriculum," 2012. [Online]. Available: http://www.ges.gov.gh/?q=content/basic-education-curriculum.

[3] Kasahorow Foundation, "Download," February 2014. [Online]. Available: http://www.kasahorow.org/download.

[4] Google Developer Group Ghana, "Ongoing Translation - English to Ga/Akan/Ewe - Google Translate," Google, 17 July 2012. [Online]. Available: https://groups.google.com/forum/#!searchin/ghana-gtug/reinvent$20the$20wheel/ghana-gtug/miekyPh8vdg/6xFwTa7qs20J. [Accessed 11 March 2014].

[5] B. F. K. M. a. S. M. S. Sana Shahzadi, "Urdu Word Prediction System for Mobile Phones," *World Applied Sciences Journal,* p. 8, 2013.

[6] Ghana Statistical Service, "2010 Population and Housing Census Summary Results of Final Report," Ghana Statistical Service, Accra, 2012.

[7] Encyclopaedia Britannica Online, "Christopher Latham Sholes," Encyclopædia Britannica Inc, [Online]. Available: http://www.britannica.com/EBchecked/topic/541481/Christopher-Latham-Sholes. [Accessed 6 April 2014].

[8] N. Baker, "Why do we all use Qwerty keyboards?," BBC, 11 August 2010. [Online]. Available: http://www.bbc.com/news/technology-10925456. [Accessed 7 March 2014].

[9] A. M. Anderson, G. A. Mirka, S. M. B. Joines and D. B. Kaber, "Human Factors: The Journal of the Human Factors and Ergonomics Study," *Analysis of Alternative Keyboards Using Learning Curves,* vol. 51.1, pp. 35-45, 2009.

[10] Depaqrtment of Linguistics (University of Ghana), Akan Dictionary Pilot Project, Accra: Combert Impressions Ltd, 2006.

[11] S. Ager, "Writing," Omniglot, 1998. [Online]. Available: http://www.omniglot.com/writing/. [Accessed 15 April 2014].

[12] GitHub Inc, "Android-KeyboardView-Example," GitHub Inc., [Online]. Available: https://github.com/rciovati/Android-KeyboardView-Example. [Accessed November 2013].

[13] L. Vogel, "Android Development - Tutorial," Vogella, January 2014. [Online]. Available: http://www.vogella.com/tutorials/Android/article.html. [Accessed January 2014].

[14] M. E. Danan, "AnySoftKeyboard," 16 March 2014. [Online]. Available: https://play.google.com/store/apps/details?id=com.menny.android.anysoftkeyboard. [Accessed 8 April 2014].

[15] Kasahorow, "Kasahorow," Kasahorow Foundation, 15 February 2014. [Online]. Available: https://play.google.com/store/apps/details?id=com.kasahorow.android.keyboard.app. [Accessed 8 April 2014].

[16] Google Inc., "Creating an Input Method," Google Inc., [Online]. Available: http://developer.android.com/guide/topics/text/creating-input-method.html. [Accessed 21 April 2014].

[17] TouchType Ltd., "Company," TouchType Ltd., 2008. [Online]. Available: http://www.swiftkey.net/en/our-company/. [Accessed 15 April 2014].

[18] GO Dev Team, "GO Keyboard," GO Dev Team, [Online]. Available: https://play.google.com/store/apps/details?id=com.jb.gokeyboard. [Accessed 15 April 2014].

[19] Stackoverflow, "Unicode support for android," Stack Exchange Inc., 20 June 2012. [Online]. Available: http://stackoverflow.com/questions/5300391/unicode-support-for-android. [Accessed 8 April 2014].