Performance Comparison of Myanmar Language IME between Native C/C++ and pure Java on Android Platform

Nandar Pwint Oo
University of Computer Studies, Yangon
nandarpwintoo@gmail.com

Abstract

Nowadays, Myanmar language input method editor (IME) is widely used in embedded system of today Android portable devices such as mobile phones, tablets and e-books readers etc. as an important part of human-computer interaction (HCI). However, to have an efficient inputting performance like Chinese prediction input method and Latin prediction input method, there are still many issues. This paper tried to show the performance comparison of Myanmar prediction IME development between native C/C++ language and pure Java. The experiments indicated that the native C/C++ used IME can raise the faster inputting performance than pure Java especially in searching the candidate syllable or word suggestion for users.

Keyword: JNI, Android, IME, HCI.

1. Introduction

Google has launched Android mobile phone platform as an open source platform since November 2007. Android mobile platform relies on Linux version 2.6 for core system services such as security, memory, process management, network stack, driver model and so on. Linux becomes the main embedded operating system because of its lower cost, powerful and easy-to-transplanting. Android SDK (Software Development Kit) provides necessary tools and API for the application development with Java language. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack. The native libraries include a set of "c libraries", a custom implementation and optimized for embedded use, are small size fast code paths. Because of the open source nature of Android, many countries have tried to develop input method editor with their own languages on Android platform. In recent year in Myanmar, there are 40,000 users are using mobile phones. The current mobile devices can use not only text messaging but also web-mail such as Gmail, etc. It indicates that one has an opportunity to input a long text via mobile phone. The increase of text input on mobile devices drives the demand for improving a text input method.

The currently used approach in the development of the mobile phone embedded input method for Myanmar language is based on only one language such as Java [6]. If the hardware resources, power consumption and the response time are taken into account, the way that simply using Java to develop the prediction input method applications is proved not as good as imagined. After opening an input method, most of the computing ability of CPU is allocated to execute the "search engine" module. Thus, in prediction input method, the implementing of the "search engine" module through the C language which can faster program execution, the whole performance of the input

method is bound to be improved. In this paper, two implementation concept of the prediction input method based on Android platform is analyzed and measured the performance according to the merit of memory consumption, time consumption and code complexity. The rest of this paper is organized as follows. The related research of prediction input method in embedded system is discussed in the next section. Section 3 describes the implementation background of input method in embedded system. The architecture and development concept of both input method are presented in Section 4 and comparison experiment are evaluated in Section 5. In Section 6, it is summarized with conclusion.

2. Related Work

Android is also one of the embedded systems that revolve around the Java abstraction layer embedded into Android. The modified Linux kernel 2.6 is almost hidden deep inside the Android operating environment. In Linux, the *make* process for C/C++ applications can directly be optimized via special compiler can further boosting application performance. Thus, most of the computing intensive application is developed with native C/C++ in Android platform. Y. Wu, J. Luo and L. Luo [8] proposed the concept of porting mobile web application engine to the Android platform. Mobile web application engine xFace of native code (C++) runs on the Android (Java) platform through JNI called can greatly reduce the Android platform migration workload. Yao Xia-xia, Wu Yan-hui and He Jin [7] discussed a realization of Chinese input based on Android with C language which can improve the performance of the input method is raised. L. Sangchul and J. J. Wook [4] showed the difference in terms of performance between an Android application using native code library from C source and an Android application using the same algorithm written in Java language only. Their conducted experiments are on JNI communication delay, integer calculation, floating-point calculation, memory access algorithm, and heap memory allocation algorithm. However, performance measures between the transplanting of prediction search engine module that is written in C/C++ to Android embedded system and pure prediction Java input method in Android embedded system for Myanmar language is not analyzed till now.

3. Background

In embedded software development process, the process of converting the source code for embedded software into an executable binary image involves three distinct steps. First, all of the source files must be compiled or assembled into and object file. Second, all of the object files that result from the first step must be linked together to produce a single object file known as *Relocatable* program. Finally, physical address must be assigned to the relative offsets within the *Relocatable* program and it is ready to be run on the embedded system.

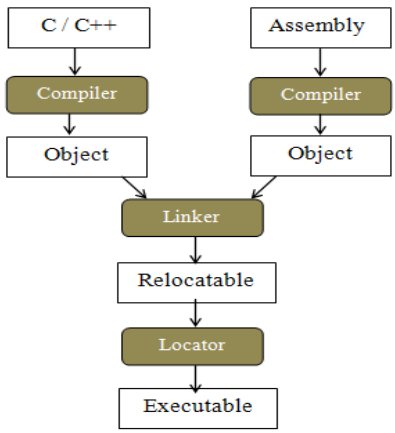


Figure 1. Embedded software development process

The standard Android development environment from Google with necessary plug-in facilities can execute code on either the host-based emulator or a real device, which is normally connected via USB. The environment only support in Android development is ARM-based target devices. Currently, Java is making its way into the embedded systems and mobile devices in android platform. The program written in Java is compiled into machine independent binary class byte codes. A Dalvik Virtual Machine executes these classes. The Java platform additionally specifies the Java Native Interface (JNI). JNI allows Java code that runs within a DVM to interoperate with applications or libraries that are written in other languages C and C++, etc and compiled to the host CPU. Thus, JNI plays an important role in embedded system as it provides a mechanism to interact with libraries specific to the platform

3.1. Java Native Interface (JNI)

The Java Native Interface is the native programming interface for Java that allows Java code that runs within a Java Virtual Machine (VM) to operate with applications and libraries written in other languages, such as C, C++, and assembly. Programmers use the JNI to write native methods when an application cannot be written entirely in the Java programming language. In the JNI framework, native functions are implemented in separate .c or .cpp files. When the JVM invokes the function, it passes a JNIEnv pointer, a jobject pointer, and any Java arguments declared by the Java method. Native data types can be mapped to/from Java data types. For compound types such as objects, arrays and strings the native code must explicitly convert the data by calling methods in the JNIEnv. It significantly reduces the effort of development cost and the Android platform migration workload. Native code C++ runs on the Android Java platform through JNI called. JNI's called the Java Native Interface, which is provided by Java to allow JAVA and the C/C++ interface to call each other. JNI provides feasibility for porting to the Android platform. Native code can be generated the dynamic library called by JNI in JAVA. According to the specificity of Android applications, application framework and start-up code can be written in JAVA language, and then called C/C++ program by JNI in that start-up code. Static initialize dynamic link library (library.so) contains the implementation of the computing engine JNI native method. A native method declaration must contain the *native* modifier. The native modifier indicates that this method is implemented in another language. The implementation for it will be stored in a spate C file and will be compiled into a dynamic link library (library.so). Dynamic link library is loaded before the computing JNI native method is called.

3.2. Android Native Development Kit(NDK)

The Android NDK is the toolset that allows Android application developers build performance critical portions of their apps in native code by embed component that make use of native code. Android application run in the Dalvik virtual machine, the NDK allows developer implement parts of application using native-code language such as C/C++. The NDK allows reusing a large of existing C/C++ code and in some cases increase runtime speed. Contents of the NDK include a set of cross-tool chain (compiler, linker, etc.) that can generate native ARM binaries on Linux, OS X and Window platform. A build system files that lets work efficiently with sources without having to handle the tool-chain, platform, CPU, ABI details by creating a build files – the build system compiles the C and C++ sources JNI

which describe in build files and then places the shared libraries directly in the application project.

4. Input Method Editor Implementation

The input method is the connection between the users and the edit text box of the application in user's mobile phone. To develop input method with native C/C++, Java language is used to develop the virtual soft keyboard and native C/C++ language through JNI is used to develop computing intensive part of Myanmar syllable/word search engine part as depicts in the following figure.

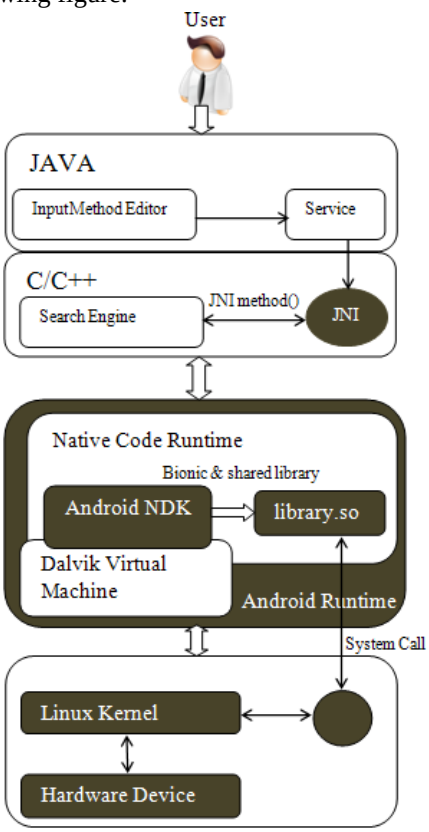


Figure 2. Myanmar Language Input Method Framework with native C/C++ on Android platform.

4.1. Input Method Editor with Native C/C++

To transplant this native C/C++ source code into embedded system Android phone, it is need to compile the native method with C/C++ file into dynamic link library with Android NDK. Android NDK use Linux gcc library as a compiler. Thus, Cygwin is used to make ndk-build on window platform. After cross-compilation, this library is loaded into mobile devices system library so that the input method editor (soft keyboard) that is written in Java can call it successfully.

The data that used by native C/C++ search engine file is offline data. For optimization of this data into embedded system, the offline probability of this syllable/word with xml format according to the following figure is needed to compress into binary dictionary.

```
<?xml version="1.0" encoding="UTF-8" ?>
<wordlist>
        <w f="3344">နှစ်</w>
        <w f="2023">တွေ</w>
        <w f="1890">ပေး</w>
        <w f="1496">အောင်</w>
        <w f="1242">များ</w>
        <w f="1061">ကို</w>
        <w f="1058">သွား</w>
        <w f="890">ယမ်</w>
        <w f="763">အား</w>
</wordlist>
```

Figure 3. Sample of Myanmar syllable probability list in xml format

The task of prediction input method editor is to search the underlying data "on the fly" as the user types on soft keyboard and to show the relevant records on candidate view which have keywords matching user typed partial keywords approximately. The underlying search engine of native C is used fuzzy search algorithm that use trie index to make prefix search. Thus, the syllable/word probability xml file is converted to binary dictionary that have trie index as depicts in the following figure.

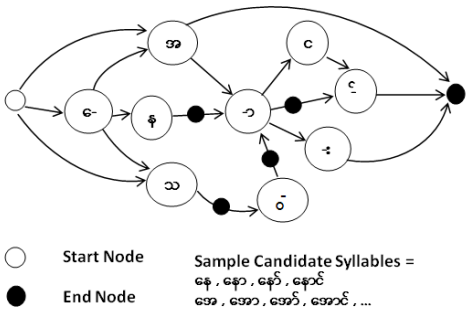


Figure 4. Sample of Myanmar syllable binary dictionary with trie index

4.2. Input Method Editor with Pure Java

With pure Java, all of the Input Method Editor and search engine part is built in Android application level. It does not need to use external native C share library and can develop as normal Android application development as shown in figure 5.

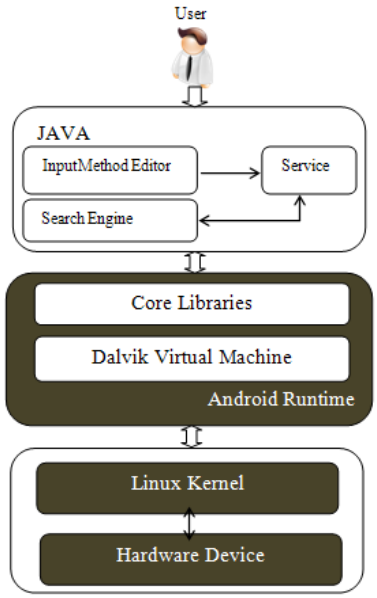


Figure 5. Myanmar Language Input Method Framework with pure Java on Android platform.

The currently used Myanmar Language input method on Android platform (iTextMM) [6] only used pure Java for prediction input method. To avoid excessive traversing time and memory usage, pure Java IME used position aware rule based matching algorithm to save memory consumption and to get quick responsiveness.

In search engine module, instead of using offline binary dictionary with trie index, the candidate syllables are built at run time according to the user's typed partial keywords approximately. The main concept is after accepting the user typed partial keywords, the search engine module combines it with corresponding Finals, Visarga and Anusara, etc. Finally, the illegal combined syllables are reduced according to syllable dictionary and give candidate list to user. In calculation of candidate syllable, search engine module extremely has to use time consuming for-loop to execute within Dalvik. Moreover, this approach cannot take the leverage of offline probability of syllable as native C fuzzy search engine.

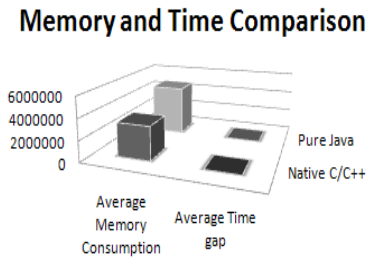
5. Performance Comparison

Using the native C to develop the search engine for prediction IME is better than the pure Java. Figure 6 showed the experimental results comparison of the two IME according to the merit of memory consumption and time gap in millisecond as shown in (1) and (2).

$$\text{Average Memory Use (bytes)} = \frac{\sum_{i=0}^{n}(TM-FM)}{n} \quad (1)$$

$$\text{Average Time gap (millis)} = \frac{\sum_{i=0}^{n}(AS-BS)}{n} \quad (2)$$

TM = Total memory
FM = Free memory
AS = time in millisecond after call to search engine
BS = time in millisecond before call to search engine



	Average Memory Consumption	Average Time gap
Native C/C++	3207088	49
Pure Java	4719456	1218

Figure 6. Experimental Results Comparison

In addition, the number of line of code to develop search engine module with Java (2970 lines) is greater than the native C. With external native C, it is only need to loads the dynamic library (System.loadlibrary("library");) containing the implementation of the search engine native method.

The experimental results showed that the native/C++ input method editor has an absolute advantage in searching candidate syllable because the pure Java approach runs directly in virtual machine, and the native C method can run directly on the underlying hardware.

6. Conclusion

This paper proposed an implementation concept of prediction input method based on Android platform not only with native C also pure Java. The experimental results showed that prediction input method application for Myanmar language takes up less resource and responses faster with native C because of Android embedded system.

References

[1] A.H. Dominque, "Mobile Devices-An Introduction to the Android Operating Environment Design, Architecture, and Performance Implications".

[2] C. Walls, and M. Graphics, "Android Development for Embedded System Beyond Mobile".

[3] H. U. Shuai-lai, L. U. Qiang and Y. Ji-wen, "An Implementation of Chinese Input Method on Linux Platform", Microcomputer Development, 2002.

[4] J. Smakov, "JNI Examples for Andorid", April 25,2009.

[5] L. Sangchul, and J. J. Wook, "Evaluation Performance of Android Platform Using Native C for Embedded System".

[6] N. P. Oo and N. L. Thein, "iTextMM: Intelligent Text Input System for Myanmar Language on Android Smartphone", 3rd FTRA International Conference on Information Technology Convergence and Services of ITCS,20-23,Oct 2011.

[7] S. Shinji, and A. Yutaka, T. Shigeaki, "Network-based Context-Aware Input Method Editor", 6th International Conference on Networking and Services, 2010.

[8] Y. Xia-xia, W. Yan-hui and H. Jin, "An Innovation of Chinese Input Based on Android Multimedia Mobile Device", Networking and Distributed Computing (ICNDC), 1st International Conference,20-21,Oct,2010

[9] Y. Wu, J. Luo, L. Luo "Porting mobile web application engine to the Andorid platform", 10th IEEE International Conference on Computer and Information Technology (CIT 2010).

[10] Z. Jiachun,L. Yiqin and L. Jin, "An Approach to Chinese Input based on Embedded Linux", Micro Computer Information, 2006.

.