Embedded Linux-based smartphone platform for sharing WIPI contents

Jae-ho Lee, Sun-ja Kim, Sang-yun Lee
Wireless Internet Platform Team, Embedded S/W Technology Center, ETRI, Korea
{bigleap, sunjakim, sylee}@etri.re.kr

Abstract ¾ **Smartphone is becoming more and more popular in mobile market but applications running on smartphone are much fewer than on legacy 3G mobile phones. In order to overcome the poverty of smartphone applications, we have developed the sharable mobile platform that can be adopted to both smartphone and 3G mobile phone. This approach enables many applications for 3G mobile phones to run on smartphones without any modifications.**

In this paper, we introduce the Korean standard mobile platform called WIPI(Wireless Internet Platform for Interoperability) and analyze its architecture in detail. Then, we implement the middleware platform on linux-based smartphone for the compatibility with 3G mobile applications and incorporate it into the reference board similar to smartphone .

Keywords ¾ **WIPI, Linux, Smartphone, Mobile Contents**

1. Introduction

1.1 What is WIPI?

Korea had the first successful commercialized CDMA mobile communications system and CDMA(3GPP2) is one of the two main global standards for communications technologies along with WCDMA. Korean handset manufacturers are also making great progress in the global market. Samsung Electronics is currently ranked third and LG Electronics is ranked fifth for mobile handset production worldwide [1].

Korean telecommunication industries are attempting to make a standard mobile platform called WIPI, which is the common platform for running mobile applications independent of service provider or handset vendor. Many contents for WIPI were developed and commercialized in the end of 2003.

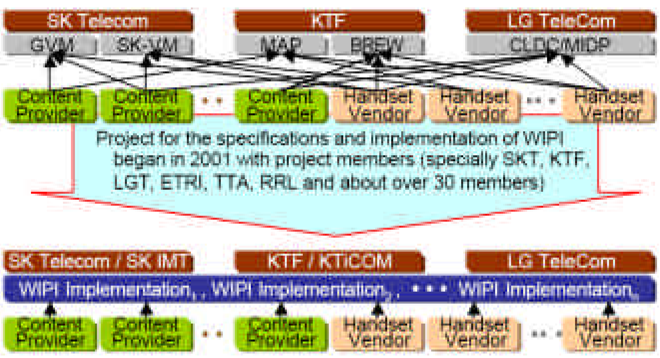


Figure 1 Motivation of emerging WIPI

Figure 1 describes the motivation for the development of WIPI. There are three big mobile carriers in Korea; SK Telecom, KTF and LG Telecom. Before the emergence of WIPI, KTF had adopted BREW as its mobile platform, while the other two carriers SK Telecom and LG Telecom used independent wireless technology based on J2ME. Because the three mobile operators have different technologies, content providers had to pay additional development costs due to modification which occurred several times to make the contents suitable for each mobile service provider. In addition, the availability of multiple platforms was one of the hindering factors that allowed proprietary wireless networks and suppressed freedom of content usage.

A common mobile platform benefits carriers, handset vendors, content developers and customers. The benefits towards each sector are highlighted below:

Carriers: fast delivery of new applications and services, more downloadable services over-the-air(OTA)

Handset vendors: reduction in engineering cost and time, easy working with 3rd party developers

Contents developers: open standard mobile platform for developing high quality contents, wider distribution channel and wider array of content services

Customers: choice of various contents independently for wireless carriers and handset vendors

The latest version of WIPI is 2.0, released by Korea Wireless Internet Standardization Forum (KWISF) that consists of about 30 Korean telecommunication industries.

Telecommunication Technology Association (TTA) in Korea adopted WIPI as a national mobile standard platform, which means all mobile handsets must support WIPI. In the near future, with WIPI technology-enabled handsets, consumers will be able to personalize their handset with applications such as games, infotainment, and location-based services. The advent and development of WIPI will pave the way for Korean telecom companies to lead the global wireless Internet industry.

1.2 Future of Linux-based smartphone

The 3G mobile communication system will be a significant step forward in the convergence of telecommunication and data communication industries. Enhancing the smartphone to merge multiple functions such as voice, data, internet and multi media services will meet these requirements.

Adoption of full-feature handsets – mobile terminals based on full-feature operating systems such as Palm, Linux or Windows Mobile - will represent the next stage of technology evolution in the mobile handset market, as shown in Figure 2. These devices will provide significantly greater design flexibility to OEMs, ease the process of launching new mobile interactive services, and put new advanced computing capabilities in the pockets of consumers [3].

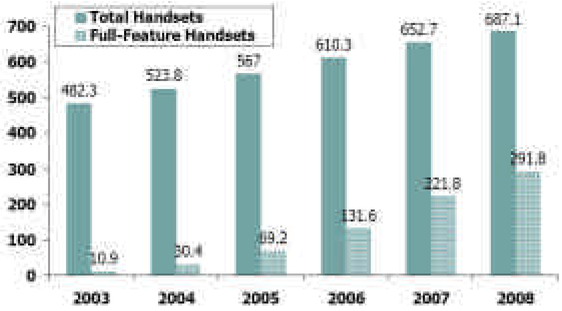


Figure 2. Full-feature handset market

Linux may become the preferred operating system in full-featured mobile terminals as well as a variety of embedded systems. The world of telecommunications, electronics and service industries are closely watching whether linux will dominate the smartphone market and the development of the linux-based smartphone for commercialization. For examples, Motorola announced its first embedded Linux based smartphone, A760, earlier last year. NTT DoCoMo, has adopted Linux for its 3G phone. Samsung is shipping a smartphone powered by embedded Linux from Mizi Research. This mobile teminal, SCH-i519, is being distributed in China currently.

Table1 shows that Linux will dominate the smartphone market, beating out rival operating systems. Zelos[3] says that Linux scored the highest on the two criteria that matter most to OEMs and carriers: open-ness and low cost.

Table1. Long-Term success scores for mobile platforms

Platform	Business Viability (2)	Complet-ness (2)	Cost (3)	End-User Appleal (1)	Openness (3)	Weighted Score
Linux	4	2	5	1	5	43
Palm	2	4	2	4	3	31
Symbian	3	5	3	1	4	38
Windows	5	4	2	4	2	34

Linux-powered smartphone will hold an important position in the near future, so rich applications as well as new mobile platforms will be required. It may be an ideal case that existing content already developed for legacy 2G or 3G mobile phones are executable on smartphone without any changes. The remainder of this paper investigates the existing mobile platform, the WIPI, and presents our approach to adopt the WIPI platform on smartphone.

2. Analysis and Design

2.1. WIPI architecture for mobile terminal

There are worldwide platforms such as J2ME of SUN, BREW of QUALCOMM, Symbian etc. Comparing with other platforms WIPI has several key advantages as the handset platform of choice for interoperability.

WIPI supports multiple programming languages such as C/C++ and Java, and downloads and runs all applications as a type of binary from the contents distribution server. Although the application is written in java language, ahead-of-time compiler (AOTC) converts byte code compiled by java compiler (javac) into machine code that can be directly executed on handset.

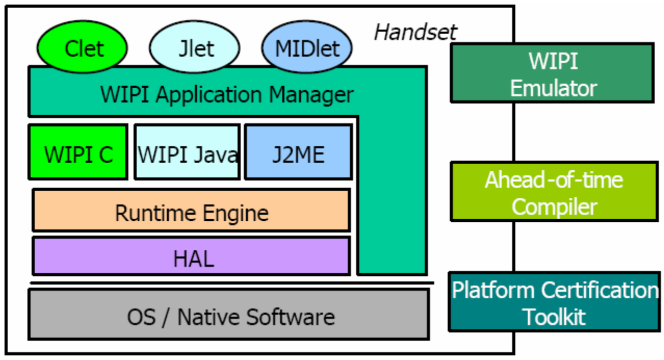


Figure 3. General WIPI architecture

Figure3 shows the general WIPI architecture. The lowest layer consists of hardware, OS, and native software. For CDMA enabled-handset, Qualcomm's REX/DMSS is used. For smartphone, native system software is based on WinCE or Linux, Symbian, PalmOS etc.

The handset adaptation layer (HAL) provides high portability for the above layers such as runtime engine, WIPI-C, WIPI-JAVA and WIPI applications. HAL hides the complex operation of the underlying hardware devices and simplifies the access and control of them.

Runtime engine provides the execution environment for WIPI applications like java virtual machine (JVM) in Java world. The main difference is that JVM loads and executes java class files while WIPI engine takes and executes platform-specific machine code. Runtime engine consists of linker/loader, memory manager, garbage collector, thread manager, synchronization manager and runtime library for supporting the converted code. Event handler and exception handler are needed for preserving the semantic of the java language.

API layer provides both C libraries and Java libraries for applications developing Clet, Jlet, or MIDlet. Clet is a program written in WIPI-C and Jlet is written in WIPI-Java. WIPI 2.0 version also provides the compatibility with MIDlet. MIDlet is a small application written in CLDC/MIDP of J2ME platform. Each WIPI application has an event handler and the life cycles such as start, pause, destroy and resume.

WIPI application manager(WAM) installs, deletes, lists, or searches applications. WAM can be also written using WIPI library.

2.2. More WIPI technology

The previous section explained the WIPI software architecture installed on the mobile terminal. But there are additional tools that help developers test and emulate WIPI applications, or verify and certify the platform on specific hardware.

AOTC internally consists of 2 main parts, a Java-to-C translator and a cross-compiler. Java-to-C translator converts java class files into C source code, and then cross-compiler converts it into directly executable image. WIPI-JAVA implementation includes about 300 native functions, used for performance improvement or direct access to hardware resources. Because the native code is dependant on the underlying operating system, it has to be rewritten for each platform. These native functions written in C must be linked with java classes translated by Java-to-C translator. The details are beyond the scope of this overview. There are several research papers on AOTC technologies such as GCJ[4], Caffeine[5] and Toba[6].

WIPI emulator provides an integrated development environment (IDE) for WIPI content developers who may have difficulty in accessing the handset platform. Once WIPI application runs without any problem on WIPI emulator, it insures that the application is interoperable with WIPI platform on any device.

Now, WIPI implementation has been only ported to CDMA-enabled handset and WIN32-based PC for emulating WIPI. In order to check the interoperability of implementation from different company, each platform is tested, verified or certified by platform certification toolkit (PCT) and HAL certification toolkit (HCT)

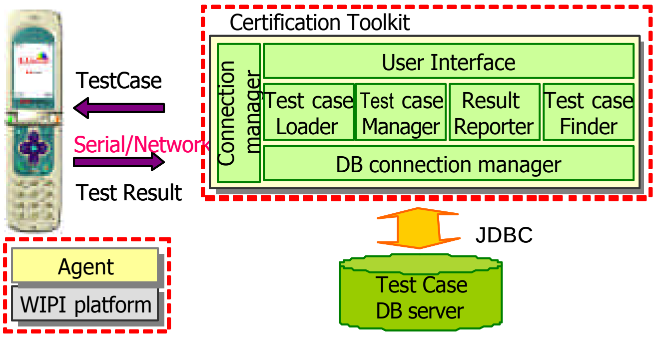


Figure 4. PCT overview

PCT is a suite of tests, tools that determines whether or not a WIPI library (WIPI-C/WIPI-JAVA) implementation complies with WIPI specification. As shown in Figure4, PCT consists of 3 different parts; PCT tool on host, database server for saving test suites and PCT agent on mobile terminal. These parts are developed and managed by specific companies that are authorized by KWISF memberships. Database sever contains sharable test case suites for verifying the platform. The tool is a GUI-based tool that communicates with PCT agent. PCT agent program is a Clet or Jlet, so it is downloadable through distributed channel in wireless environment and executable on WIPI. To check the interoperability of the platform, PCT server sends test query to PCT agent. PCT agent receives it and executes server's order by using built-in WIPI library on mobile terminal, then sends the results to server. Finally, server determines whether or not platform complies with standard specification, by comparing the result to the value stored in its database.

HCT is a suite of tests, tools that checks whether or not HAL implementation complies with WIPI specification. It is used by mobile terminal vendors to test whether the HAL is correctly ported to their product. It also consists of sever side and client side like a PCT. Because HAL is ported to vendor-specific software at first step, HCT agent has to be developed using native software, not WIPI library. This means HCT agent needs to be rewritten for every target platform. The process of verifying the platform is very similar to PCT.

Until now, there has been no study on HCT, while commercial-quality PCT was already developed by EXE-MOBILE in 2003. This paper implements HAL for linux-based smartphone and verifies by HCT that we designed and developed. This work will be the starting point of WIPI reference implementation for linux-based smartphone.

3. WIPI platform for linux-based smartphone

WIPI 1.2-based reference implementation exists only for REX-based handset platform and WIN32-based emulator at current work, in spite of the trends in increasing the number of smartphone-type mobile terminal users. Our approach designs and implements embedded linux-based smartphone platform complying with WIPI specification. It has important meaning that existing contents or tools for supporting WIPI service – already developed Clet/Jlet/MIDlet, AOTC, PCT, HCT, contents distribution server - can be reusable without any modification or with at least limited modification.

This section designs the prototype of smartphone software architecture to support WIPI, implements HAL and HCT, and verifies HAL using HCT.

3.1. Smartphone architecture

Figure 5 shows the embedded linux-based smartphone software stack containing WIPI. HAL is implemented with Qplus[10] and linux-based libraries, instead of REX/DMSS in CDMA-enabled handset. Qplus is a highly configurable embedded linux system developed by ETRI in Korea. Qplus supports various embedded system such as HomeServer of ETRI, iPAQ of Compaq, Zaurus of sharp and S3C2400 of Samsung.

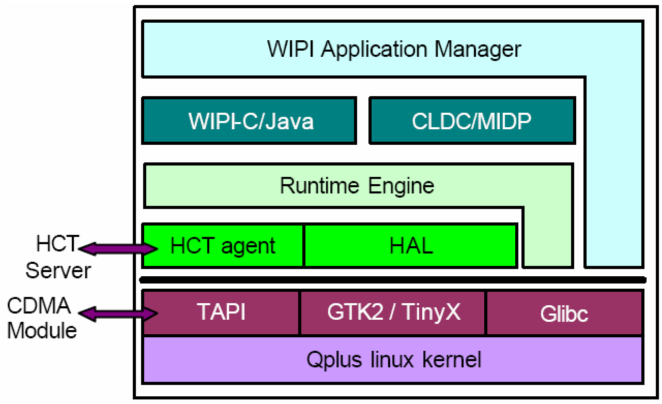


Figure 5. Linux-powered smartphone stack

TAPI provides telephony API to handle modem or external CDMA card. TinyX/GTK2 is used for creating GUI on LCD. Glibc is used as the standard C library in the linux system.

HAL is implemented using native software such as Glibc, TAPI, GTK2/TinyX. HCT agent is a kind of linux application that can communicate with HCT server to verify the HAL implementation on linux-based smartphone.

Runtime engine and WIPI libraries are also rewritten in linux-based code, while WAM can be reusable because it was developed with WIPI library for handset platform.

The next section focuses on HAL and HCT as the starting point of WIPI reference implementation for smartphone.

3.2. HAL and HCT

HAL APIs for the linux-based platform provide the following functions for each part.

System: gets or sets the mobile terminal information for platform, protects critical section, initializes or exits the platform, synchronization etc.

Call: controls telephony device.
Handset device: controls LED, backlight, vibrator, etc.
Network: opens, accesses or closes the network(socket)
Serial: opens, closes, reads, writes and controls serial
Media: controls multi-media devices, processes and plays multi-media data
Time: generates and manages system or user-defined timer
Utility: converts between unicode and local code
File: handles file or directory.
Input method: processes character input
Font: processes font
Frame buffer: print the data onto the screen or LCD
Virtual key: maps between function keys on application and keys on mobile terminal

HCT is an easy-to-use GUI-based tool for testing, verifying, certifying the HAL implementation for each platform. The tool consists of project manager and report manager as shown in Figure 6. The Project manager creates, deletes and modifies project from its menu. During project creation, in order to determine where parts of HAL are to be tested, test cases list should be configured from pre-defined test suites list in HCT database. The tool sends the user-defined test cases to the HCT agent. The agent executes the test cases on iPAQ and returns the result to the tool. While communicating with agent, report manager shows the processing status of HAL API. After completion of test, the summary of results is saved in the form of HTML file to enable ease of view later.

4. Experimentation

HAL is deeply dependent on the underlying operating system and physical device. Especially, it is very difficult to develop HAL for the latest original equipment manufacturer (OEM) devices because most of vendors do not open hardware specification or device driver code. So we select iPAQ of Compaq as the reference hardware for building and testing linux-supported WIPI platform, where it is easy to update kernel and add custom functionality as required by developers.

Comparing with iPAQ, commercial handsets have different device features, small size screen window and telephony-related interface. Vibrator and telephony-related operation in handset are replaced by icons in iPAQ. Display window size is also reduced to 160x240 for the existing WIPI contents developed for phone as shown in Figure 6.

The operation between HCT tool on Windows-based PC and HCT agent on Qplus-enabled iPAQ is denoted in Figure6 and Figure7.

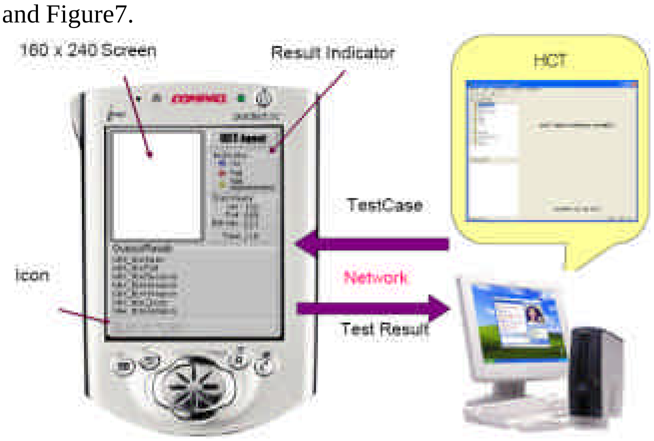
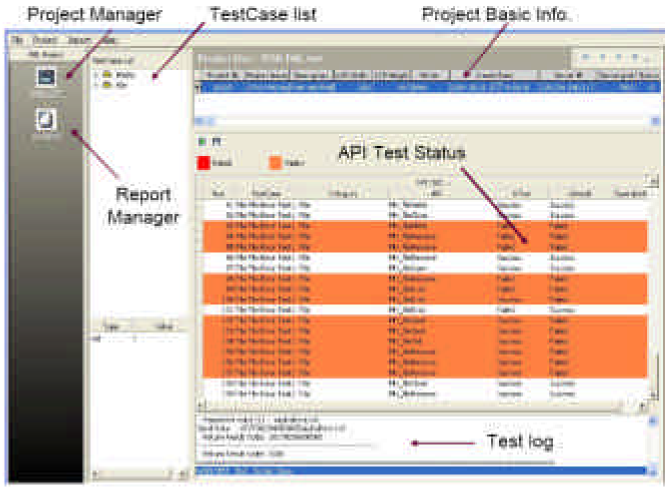


Figure 6. HCT agent on iPAQ



Fingure 7. HCT on host

5. Conclusion and future work

This paper introduces the mobile trends in Korea and explains the WIPI platform, which is aimed at providing contents interoperability among mobile operators. Now, WIPI technology has already been formalized in commercial mobile terminals as national standard in Korea. Many WIPI applications have also been developed using the WIPI emulator.

This paper demonstrates the current WIPI platform and expands the embedded linux area to enable legacy 3G mobile contents to be executed on linux-powered smartphone without any modification. It means WIPI platform has high possibility to be applied as new mobile platform for another mobile OS such as Symbian, Palm, Nucleus as well as linux or REX.

Currently we have developed some part of the WIPI reference implementation and HAL API for linux-based smartphone. But we plan to implement WIPI library and runtime engine by the end of 2004, and then add more test suites to PCT or HCT. In AOTC technology, to run Java-based WIPI application, research topics such as reducing the converted code size and runtime library size are bleeding edge research areas. We are pursuing advanced research in reducing the size of the code converted by AOTC in resource-limited mobile terminals.

REFERENCES

[1] Strategy Analytics, "Global Handset Market", market report, http://www.strategyanalytics.com
[2] http://www.linuxdevices.com.
[3] Seamus McAteer, "Defining the market for fullfeatured handset", Zelos Group, Inc.
[4] "Guide to GNU gcj", http://gcc.gnu.org/java/
[5] Cheng-Hsueh A.Hsieh, John C.Gyllenhaal, and Wen-mei W.Hwu, "Java Bytecode to Native code Translation: The Caffeine Prototype and Preliminary Results" Proceedings of the 29th International Symposium on Microarchitecuture, December 1996.
[6] T.A.Proebsting, G.Townsend, P.Bridges, J.H.Hartman, T.Newsham, and S.A. Watterson, "Toba: Java for Applications – A Way Ahead of Time(WAT) Compiler", Proceeding of the Third Conference on Object-Oriented Technologies and Systems(COOTS), USENIX Association Press, 4153; 1997
[7] KWISF, WIPI 1.2 specification, www.wipi.or.kr
[8] KWISF, WIPI 2.0 specification, www.wipi.or .kr
[9] WIPI developer site, www.developer.wipi.or.kr
[10]Embedded S/W Technology Center, ETRI, QPlus, http://qplus.etri