이기종 모바일 플랫폼 간 어플리케이션의 상호 호환을 위한 변환 솔루션 구현

Implementation of Conversion Solution for Interoperability of Applications Developed on Different Mobile Platforms

강경보*, 강동현*, 류종민*, 홍창표*, 이중훈**, 윤정한**, 좌정우*
제주대학교*, (주)WiSEGRAM**

Kang Kyung-Bo*, Kang Dong-Hyun*, Ryu Jong-Min*, Hong Chang-Pyo*, Lee Joong-Hoon**, Yoon Jung-Han**, Jwa Jeong-Woo*
Cheju National University *, WiSEGRAM Inc. **

요약

이동통신망의 진화와 더불어 다기능 휴대폰을 이용한 신규 사업모델들이 개발되고 있다. 이동통신사업자는 다기능 휴대폰을 이용하여 기존의 음성서비스에서 WAP, ME 등의 모바일 브라우저와 WIPI, J2ME, BREW 등의 이동멀티미디어 플랫폼을 이용한 무선인터넷 서비스를 제공하고 있다. 카메라, MP3, MPEG, 3D 게임엔진, DMB 등의 멀티미디어 솔루션, 블루투스, IrDA, W-LAN, 등의 PAN 기능, GPS 연동을 통한 위치정보 등의 다기능 휴대폰이 개발됨에 따라 이를 기반으로 한 다양한 모바일 어플리케이션이 이동통신사업자가 사용하는 플랫폼으로 개발되고 있다. 콘텐츠 사업자는 하나의 모바일 어플리케이션을 이동통신사업자가 제공하는 모바일 플랫폼에 맞추어 새로이 개발해야 하는 문제점을 갖고 있다. 본 논문에서는 이기종 모바일 플랫폼 간 어플리케이션 상호 호환을 위해 대표적인 무선인터넷 플랫폼인 WIPI와 BREW에서 변환 솔루션을 제안한다. WIPI와 BREW의 특성을 분석하고 이를 기반으로 하여 WIPI에서 BREW, BREW에서 WIPI로 모바일 어플리케이션을 변환하는 솔루션을 제안한다.

Abstract

Cellular network evolution and development of multi-function cellular phones introduce new mobile business mode. Cellular operators provide mobile internet services using the mobile browser such as ME and WAP and the mobile multimedia platform such as WIPI, J2ME, and BREW. New mobile applications are developed using mobile platforms provided by cellular operators as cellular phones having multimedia solutions such as camera, MP3, MPEG, 3D game engine, DMB, PAN such as bluetooth, IrDA, W-LAN, and location information using GPS are developed. Content providers have problems of redevelopment of a mobile application for different mobile platforms. In this paper, we propose a conversion solution for interoperability of applications developed on different mobile platforms of WIPI and BREW. We analyze APIs of WIPI and BREW and develop conversion solutions

I. 서 론

이동통신망이 고도화되고 다기능 휴대폰이 보급됨에 따라 이동통신망 서비스는 기존의 음성 서비스에서 모바일 브라우저와 이동멀티미디어 플랫폼을 사용하는 신규 사업모델들이 개발되고 있다. 최근에 이동통신 사업자는 방송, 금융, 텔레매틱스 등 산업간 융합을 위한 신규 사업 모델을 적극적으로 개발하고 있다. 다기능 휴대폰의 보급과 USN, PAN 등 신규 통신망이 보급되면서 시공간의 제약을 받지 않는 유비쿼터스 환경에서 서비스가 개발되고 있다[1][2]. 콘텐츠 공급자는 이동통신사업자가 제공하는 모바일 플랫폼에 맞추어 신규 사업모델에 따른 다양한 어플리케이션이 개발해야 하는 문제가 발생하게 된다. 국내 콘텐츠 사업자는 SKT, KTF, LGT의 모바일 플랫폼에 맞추어 동일한 어플리케이션을 별도로 개발해야 하므로 개발업무에 부담을 갖게 된다. 국내에서는 이와 같은 콘텐츠 사업자의 어려움을 덜어 주기 위해 표준 모바일 플랫폼으로 WIPI(Wireless Internet Platform for Interoperability)를 개발하여 사용하고 있다. WIPI 플랫폼은 국내 모바일 단말기의 표준 플랫폼을 제시하여 국내의 이동통신사와 휴대폰 제조사가 통합된 WIPI 플랫폼을 채택하여 콘텐츠의 개발업무의 통합 환경을 구축할 수 있도록 하였다. 하지만 국내에서는 WIPI 플랫폼을 사용하여 통합 환경을 구축할 수 있었지만 업체가 세계시장으로 발돋움하기에는 부족함이 많았다. 해외의 모바일 단말기 시장에서 WIPI 플랫폼이 보편화 되지 않았으

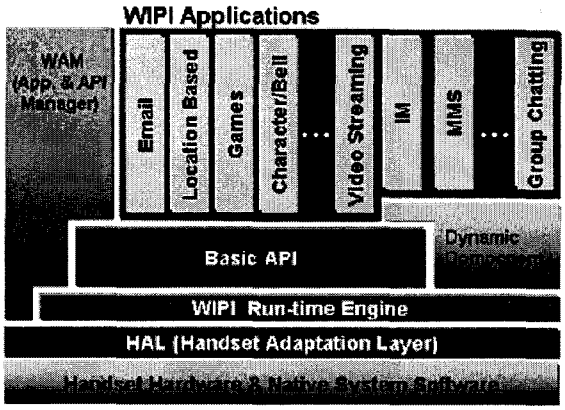
며 국제적으로 사용되는 플랫폼이 다르기 때문이다. 예를 들어, Qualcomm의 BREW(Binary Runtime Environment for Wireless), NOKIA의 Symbian, JAVA의 J2ME 등 모바일 플랫폼 등이 있다. 국내에서 WIPI로 개발된 모바일 어플리케이션을 다른 플랫폼에서 동작하게 위해서는 신규 플랫폼에 맞추어 새로이 개발해야 하고 역으로 다른 플랫폼에서 개발된 모바일 어플리케이션은 WIPI에서 새로이 개발해야 한다.

본 논문에서는 WIPI와 BREW에서 개발된 모바일 어플리케이션을 상호 호환하기 위한 변환 솔루션을 제안한다. 2장에서는 본 논문에서 사용되는 WIPI와 BREW의 특징과 모바일 어플리케이션 변환을 위한 방안에 대해 설명한다. 3장에서는 이기종 모바일 플랫폼 간 어플리케이션 상호 호환을 위한 변환 솔루션에 대해 기술하고 4장에서 결론을 내린다.

II. 관련 연구

1. WIPI 플랫폼

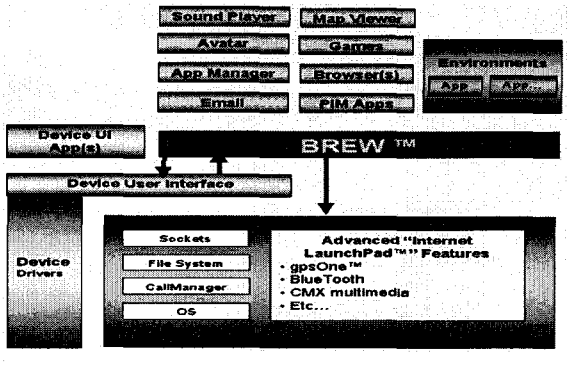
WIPI는 ETRI(한국전자통신연구원)와 정보통신부, 한국 무선인터넷 표준화 포럼 등이 공동 개발한 무선인터넷플랫폼 표준 규격이다. 그리고 정보통신부는 2005년 4월부터 출시되는 국내 휴대폰에 탑재를 의무화 하여 국내에서 무선인터넷플랫폼으로서의 비중이 크다[4]. WIPI 플랫폼의 개념적 구조는 그림 1에 나타낸 것과 같다. 모바일 플랫폼이란 모바일 표준 플랫폼 규격에 따라 작성된 응용프로그램을 실행 시킬 수 있는 단말기의 실행 환경(Runtime Execution Environment)을 모바일 플랫폼이라 하며 이 플랫폼은 응용프로그램 관리와 API 관리 기능(WAM : WIPI App. & API Manager)을 포함해야 한다. HAL(Handset Adaptation Layer)이란 플랫폼의 하드웨어 독립성을 유지하기 위한 추상화 계층으로 상위 Layer들이 HAL위에서 Native System과 무관하게 즉, 하드웨어 독립적으로 플랫폼이 구성될 수 있도록 해주며 하단의 단말기 기본 소프트웨어와 플랫폼을 연결해 주는 역할을 한다. 단말기 기본 소프트웨어(Native System Sofrware)는 플랫폼이 탑재되는 기반 소프트웨어를 말한다. 기본 API(Basic API)란 응용 프로그램 개발자가 사용하는 기본 API모음으로, Java와 C API로 구성되어 있다. 이 기본 API를 이용하여 WIPI의 응용프로그램은 구현이 되고 사용자들에게 모바일 콘텐츠로서의 역할을 수행할 수 있는 것이다.



▶▶ 그림 1. WIPI 플랫폼 구조도

2. BREW 플랫폼

미국 Qualcomm 사가 CDMA(코드분할다중접속) 방식의 이동통신기기용으로 개발한 차세대 플랫폼이다. Brew는 기본적으로 C++ 언어를 사용하여 콘텐츠를 개발하며 해외에서는 Brew 플랫폼 상위에 Java VM을 탑재한 모바일 단말기를 출시하여 Java 언어를 사용한 콘텐츠의 개발도 가능하다[5]. 그림 2는 BREW 플랫폼의 개념적인 구조를 나타낸 것이다. Device Driver는 휴대폰의 장치들을 사용하기 위한 소프트웨어로 Device User Interface가 그 상위에 존재하여 BREW 플랫폼이 하드웨어와 분리되어 독립성을 유지하도록 해준다. CORE ASIC(Application-Specific Integrated Circuit) Chip Software는 별도의 Driver없이 BREW 플랫폼이 제어가 가능한 Device이다.



▶▶ 그림 2. BREW 플랫폼 구조도

이러한 구조로 BREW의 응용 프로그램은 장치와는 분리된 환경에서 BREW 플랫폼에서 제공하는 API를 사용하여 독립적인 개발이 가능하게 된다.

3. 소스의 변환

WIPI는 세부적으로 C와 Java로 구분되는 언어를 지원함으로서 WIPI에서 Brew로의 변환은 WIPI Jlet을 Brew로 변환

과 WIPI Clet을 Brew로 변환 두 가지 모두 지원이 되어야 하며 Brew는 기본적으로 C코드를 사용하기 때문에 WIPI Clet으로 변환 하나가 존재 한다. 즉 3가지의 변환을 통해서 두 플랫폼 간의 호환 솔루션을 제공하게 된다. 본 논문에서 제안하는 변환 솔루션에서는 표 1에서 기술된 소스 변환기술중 컴파일 기술에 의한 Transcode 방법을 사용하여 소스를 변환하고 별도의 Wrapper Library를 통해 대상 플랫폼에 변형된 소스의 구동환경을 조성하게 된다.

[표 1] 소스 변환 기술 비교

변환 방안	변환결과	장 점	단 점
컴파일 기술에 의한 Transcode	Source Code	• 변환 내용 검토가 용이 • 비교적 투명한 검증	• 문제 발생 시 원인규명이 명확하지 않음
소스의 직접 변환	Source Code	• 변환 후 디버깅 가능 • 원본 source의 흐름을 유지한 상태에서 변환	• 완벽한 1:1 mapping 불가 • 프로그래머 코딩스타일에 의존적
원본 소스의 무 변형 실행 (Library 사용)	Binary File	• 변환과 관련한 디버깅의 작업의 최소화됨	• Library에서 오류 발생시 해결이 어려움

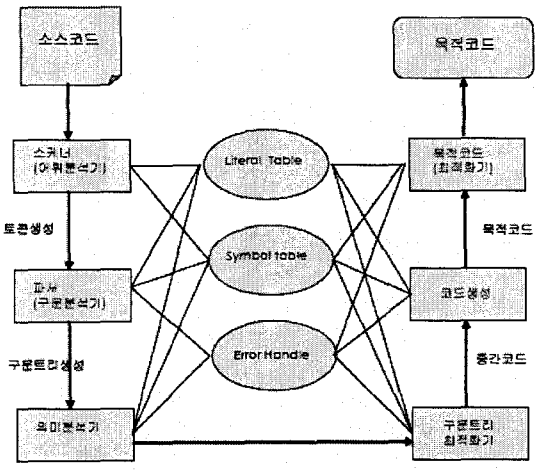
WIPI Java에서 Brew로의 변환을 예로 하여 변환의 각 단계를 살펴보면 다음과 같다.

- WIPI JAVA로 제작된 소스 코드를 읽음
- C++ 코드로 변환
  - Brew의 소스 코드 형태로 제작, Arm C++의 문법과 규약에 맞도록 한다.
- 변환 된 소스는 미리 설정된 구동환경과 같이 컴파일

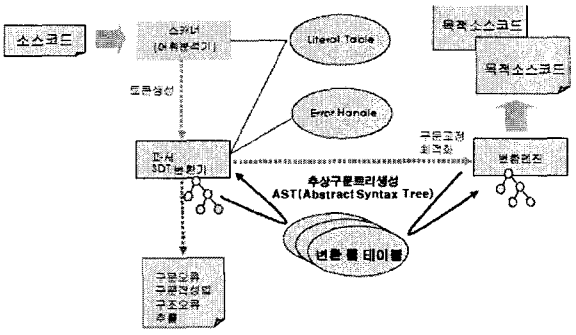
3.1. 컴파일 기술을 이용한 Transcode

일반적인 컴파일의 과정은 그림 3과 같이 표현될 수 있다. 그러나 본 솔루션에서 변환된 소스의 목표는 최종 목적 코드가 아닌 원시소스를 다른 플랫폼 즉 Brew의 원시소스로 만드는 것으로 전체의 단계가 필요치 않다. 컴파일러 이론이 적립되지 않은 컴퓨팅 초창기에 소스 프로그램에 대한 중간코드를 생성하지 않고 직접 목적 기계 코드를 생성하는 단일패스(One-Pass) 컴파일러 설계 방법을 사용하였는데, 본 연구 시스템의 주요한 기능인 코드 변환체계는 이 단일패스(One-Pass) 컴파일러의 설계 방법을 사용한다.

그림 4는 컴파일 과정을 나타낸 것이다. 변환기의 처리 개요를 살펴보면 어휘 분석기에 의해 토큰 화된 원시 프로그램은 문법-지시적 변환방법(SDT : Syntex Direction Translation)을 이용한 파서에 의해서 Bottom-Up 방법으로 추상구문트리(AST)를 만들고 AST상에 변환될 소스가 구동될 환경에 적합한 소스코드로 일대일 mapping된 정보를 이용하여 소스변환에 적용하는 개념이다.



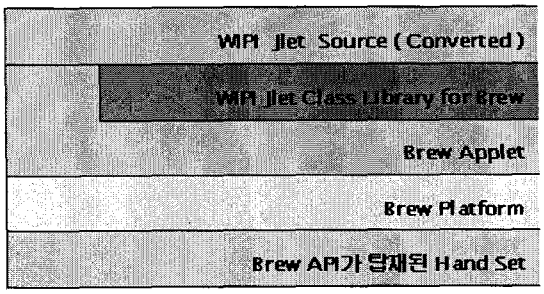
▶▶ 그림 3. 컴파일 과정



▶▶ 그림 4. 소스 변환의 과정

3.2. 구동환경의 조성

변환된 소스는 대상 플랫폼에서 작동시키기 위해 구동 환경을 조성해 주어야 한다. 소스를 변환하여 다른 원시 소스를 만드는 데는 한계가 있기 때문에 기본적으로 API Library가 필요하다. 이 Library는 변환되기 이전의 콘텐츠가 구동되던 플랫폼의 API를 담고 있어야 하며 대상 플랫폼의 언어를 사용하여 작성되어야 한다. 그리고 플랫폼 마다 콘텐츠를 구동시켜주고 이벤트를 처리하는 방식이 다르기 때문에 이를 맞춰줄 프로그램이 있어야 한다.



▶▶ 그림 5. 변환된 콘텐츠의 작동환경

WIPI Jlet의 콘텐츠를 Brew의 콘텐츠로의 변환을

예로 들면 그림 5와 같은 동작환경에서 변환된 콘텐츠를 작동시키게 된다. 위와 같은 환경을 구성하면 변환된 소스는 기존의 플랫폼의 API를 그대로 사용할 수 있고, 이벤트의 처리 또한 기존의 플랫폼에서 작동되는 방식을 그대로 사용할 수 있다. 즉 Wrapper Library와 최소한의 Brew Applet가 존재하기 때문에 변환기에서 원본 소스에 가하는 변환이 최소한으로 적어진다.

III. 이기종 모바일 플랫폼 간 어플리케이션 변환 솔루션

1. 시스템의 구성

솔루션의 시스템은 크게 3개의 모듈로 구분된다. 대상 콘텐츠의 소스를 직접 변환할 변환기와 대상 플랫폼에 맞춰 제작된 Wrapper Library, 변환된 콘텐츠를 구동하고 모든 이벤트를 전달해줄 최소한의 응용 프로그램으로 시스템을 구분한다. 변환기는 원본 소스를 읽어 최소한의 변환을 수행하는데 변환의 대상은 문법과 대상 플랫폼의 규약에 어긋나는 것이다. 대상 플랫폼에 맞춰 제작된 Wrapper Library는 원본 콘텐츠가 구동되던 플랫폼의 API를 모두 담고 있어야 하고 그 기능과 동일하게 동작해야 하지만 플랫폼에 특성에 따라 구현이 불가하거나 구현의 대상에서 제외 될 수 있다.

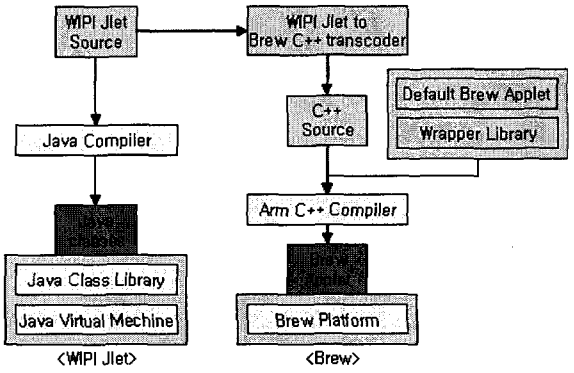
마지막으로 변환된 소스가 대상 플랫폼의 구동환경과 맞지 않기 때문에 콘텐츠를 구동하고 각각의 상황에 맞는 이벤트를 발생시켜 줄 최소한의 응용 프로그램이 필요한데 이 프로그램의 핵심은 이벤트의 전달이다. 모바일 단말기의 콘텐츠들은 모바일이라는 특성상 Applet 구조로 구성되며 이벤트의 발생에 따라 콘텐츠가 작동하게 된다. 따라서 플랫폼에 맞추어 이벤트를 적절히 전달을 해줘야 변환된 콘텐츠가 정상작동을 할 수 있다. 전달될 이벤트도 Library와 마찬가지로 원본 콘텐츠가 구동되던 플랫폼의 모든 이벤트를 전달 해줘야 맞지만 플랫폼의 특성에 맞추어 발생하지 않는 이벤트와 상황에 따라 전달 할 필요가 없는 이벤트 등이 존재한다.

[표 2] WIPI Jlet 콘텐츠를 Brew 콘텐츠로 변환 시 구성요소

구 성 요 소	구 현 내 용
변 환 기	• Java언어를 C++언어로 변환 기능
Wrapper Library	• WIPI Jlet Class Library를 C++언어를 사용하여 구현
Brew Applet	• Brew 플랫폼에서 구동될 Applet 구현 • 모든 이벤트는 변환된 소스로 전달 기능

2. 시스템의 동작

시스템의 전체적인 동작은 그림 6에 나타낸 것과 같다. 동작단계 중 첫 단계인 소스의 변환 단계는 시스템의 구성요소 중 변환기를 통해 원본 WIPI Jlet 소스를 C++의 코드에 맞게 변환을 한다. Java 언어와 C++ 언어는 모두 Class를 사용할 수 있기 때문에 변환이 가능하다. 추가적으로 Wrapper Library를 구현함에 있어 플랫폼의 특성에 맞춰 API를 기존과 상이하게 사용해야 할 경우도 변환기에서 변환을 가해준다. 하지만 모든 것을 자동으로 변환을 한다는 것은 불가하므로 변환된 소스의 검토는 필요하다.



▶▶ 그림 6. 기존과 변환 후의 실행과정 비교

변환된 소스는 이미 구현된 Wrapper Library와 최소한의 Brew Applet과 함께 컴파일을 한다. 이때 주의해야 할 점은 WIPI와 Brew가 콘텐츠의 용량제한 기준이 다르다는 것이다. WIPI와 Brew를 비교했을 때 Brew는 250kb를 지원하는데 이는 WIPI 보다 용량의 제한이 훨씬 작다. 따라서 Wrapper Library를 모두 포함하게 된다면 용량 제한을 초과하게 되므로 선별적으로 포함을 시켜야한다.

추가적으로 Wrapper Library, Brew Applet을 구현함에 있어 생성기는 제약사항이 있을 수 있는데 예를 들면 WIPI Jlet의 API 중에 Brew 플랫폼을 사용하여 구현이 불가한 API는 사용을 할 수 없고 콘텐츠의 개발자가 API를 사용한 의도대로 구현을 해야 한다.

IV. 결 론

본 논문에서는 WIPI플랫폼 혹은 Brew 플랫폼에 콘텐츠를 소스 변환을 통해 다른 플랫폼에서도 작동할 수 있는 솔루션을 구현하였다. WIPI 플랫폼 중에 WIPI Jlet의 콘텐츠를 변환하여 Brew 플랫폼에서 구동하는 솔루션의 시스템에 맞춰 구현을 하였지만 동일 과정을 통하여 Brew와 WIPI Clet, Jlet 플랫폼에서 콘텐츠의 호환 솔루션을 설계, 개발할 수 있다. 이

밖에 국내의 MIDP, GVM 등과 해외의 Symbian 등의 플랫폼에 특성을 적용을 해보지는 못했지만 본 논문의 솔루션을 사용하여 완성된 콘텐츠를 다른 플랫폼에 적용하기 위해 이중 개발을 하던 방식을 탈피하여 콘텐츠를 소스변환을 통해 다른 플랫폼에서도 작동될 수 있게 함으로써 콘텐츠 개발의 비용 절감과 능률 향상을 더욱 고취할 수 있으리라 예상된다.

▌참 고 문 헌▌

[1] 김성환, 양석호, "휴대폰을 위한 모바일 자바 프로그래밍", pp.22-39, (주)피어슨에듀케이션코리아, 서울, 2005.
[2] 배석희, 한상훈, 전영준,, "위피 WIPI 모바일 프로그래밍 기술을 통일한 위피 입문서", pp.14-30, 대림출판사, 서울, 2004.
[3] Ray Rischpater, "Software Development for the QUALLCOMM BREW Platform", apress, 2005.
[4] 안창남, "WIPI SDK 기술교육 자료", WIPI-EDU-SDK_기술교육_kwisa-1.1.2, AROMA SOFT, 2003.
[5] 박상태, "Brew 기본 개념", Geotel, 2001.
[6] KTF, "WIPI 개발자 가이드", 2004.
[7] www.qualcomm.com/brew/ko/developer