제33회 한국정보처리학회 춘계학술발표대회 논문집 제17권 제1호 (2010. 4)

안드로이드 1.x 버전을 위한 블루투스 API 의 개발

박병조 · 양희재
경성대학교 컴퓨터공학과
pbj1024@nate.com, hjyang@ks.ac.kr

Development of Bluetooth API for Android 1.x Version

ByeongJo Park and Heejae Yang
Dept of Computer Engineering, Kyungsung University

요 약

최근 모바일 플랫폼으로 부상하고 있는 Google사의 안드로이드는 2008년 9월에 안드로이드의 첫 번째 공식 SDK 버전 1.0을 출시하여 현재(2010년 3월) SDK 버전 2.1까지 출시하였다. 안드로이드 플랫폼에서 블루투스 디바이스를 제어하기 위해 개발자들은 Android SDK의 Bluetooth API가 필요하게 되는데 이는 1.0~1.6버전까지는 지원 되지 않고 2.0 버전부터 공개가 되었다. 하지만 문제점은 안드로이드 2.0이 발표되기 이전에 나왔던 대부분의 안드로이드 단말은 2.0 업데이트 펌웨어를 현재까지도 지원하지 않고 있다는 것에 있다. Bluetooth API는 Android SDK 2.0 버전 이상에서만 사용 가능하기 때문에 개발자들이 개발한 블루투스 어플리케이션의 최소 펌웨어 버전 요구사항이 2.0 이상 일 수밖에 없는 것이다. 이 연구에서는 안드로이드 1.x 버전에서 사용될 수 있는 블루투스 API의 개발에 대해 알아보고자 한다.

1. 서론

구글에서는 안드로이드의 초기버전인 1.0버전을 2008년 9월에 출시하였다. 그 후에 계속적인 업데이트를 지속하여 1.5(CupCake), 1.6(Donut)의 버전을 거친 후 2010년 3월 현재 2.1(Eclair)버전까지 제공하고 있다. 하지만 안드로이드 단말 중에 가장 널리 사용되고 있는 HTC사의 G1 모델마저 현재까지도 안드로이드 펌웨어를 1.6버전까지 밖에 지원하지 않고 있으며 구글에서 2.0 버전의 소스트리를 2009년 10월 말에 공개했음에도 불구하고 G1의 사용자들은 2.0의 가장 핵심적인 기능인 멀티터치와 개발자에 의해서 만들어진 블루투스 어플리케이션을 사용 할 수 없다 [1].

개발자들이 블루투스를 제어하는 어플리케이션을 작성할 수 있게 하는 블루투스 API의 경우는 안드로이드 베타 버전에 해당되는 SDK 0.9 버전에서는 잠깐 지원하였으나 구글측에서 보안 외 여러 가지 이유 등으로 인해 정식버전인 1.0버전과 그 이후에는 지원하지 않았다. 구글에서는 1.6 버전까지 블루투스 API를 공개하지 않았고 결국 2.0 버전이 출시하면서 블루투스 API를 공개하여 개발자들은 블루투스 어플리케이션을 작성 할 수 있게 되었다. 하지만 G1 외 안드로이드가 적용된 여러 모바일 기기가 아직 2.0 버전의 펌웨어를 지원하지 않기 때문에 현재까지도 1.x버전이 적용된 안드로이드 단말에서 접속한 안드로이드 마켓에서는 해당 단말의 펌웨어 버전에 적용 가능한 어플리케이션만 제공하고 있다는 한계점이 있다. 그리고 1.x버전에서 찾을 수 있는 어플리케이션들 중에서는 블루투스를 이용한 어플리케이션의 종류도 매우 적은편이고 그 내용도 아주 기초적인 블루투스 On/Off 기능을 외부 위젯으로 작성하여 편의를 제공하는 정도에서 그치는 것이 대부분이다. 본 연구에서는 이러한 안드로이드의 문제점 중 하나인 1.x버전 펌웨어(Cupcake, Donut)에서의 블루투스 어플리케이션 작성 방법에 대해서 기술하고자 한다.

논문의 구성은 다음과 같다. 2절에서는 안드로이드 블루투스 API 문제에 대해 해결 방법을 안드로이드 비공식 펌웨어 업데이트와 안드로이드 블루투스 API를 개발자가 직접 작성하는 방식 두 가지로 나누어서 간략하게 설명한다. 3절에서는 블루투스 API 작성 방식의 실제적인 접근 방법에 대해 소개하기로 하며 4절에서는 실제적인 연구 결과에 대하여 설명한다.

2. 안드로이드 블루투스 API 문제점 해결 방법

2.1 펌웨어 업데이트

구글에서는 안드로이드를 Open source 정책으로 개발하면서 리눅스 git [2] 유틸리티를 통해 배포하였다. 현재에도 안드로이드 개발자 사이트에 접속하면 git을 통한 안드로이드 플랫폼의 소스트리를 구할 수 있다.

첫 번째로 소개하는 방법은 git을 통한 2.0 펌웨어 제작이다. 현재 Android open source project 홈페이지에서 G1에 대한 환경설정 방법을 제공하고 있으나 해당 페이지에서 소스트리 빌드하는 방법을 제시하는 마지막 버전은 Donut(1.5)버전에 해당된다. 본 연구에서 진행한 방법은 Donut버전에서의 환경설정 방법을 따랐으며, 여러 시행착오 끝에 컴파일에 성공하고 G1에서 작동하는 펌웨어 이미지를 얻을 수 있었다 [3]. 하지만 해당하는 펌웨어를 실제 우리가 연구하고 있는 단말(G1)에 적용 하였을 때 여러 가지 문제가 생겼다. 급격한 속도 하락과, 블루투스 장치 작동이 하지 않았고 여러 오류들로 인해 정상 사용하기 힘들었다.

현재에도 해외 여러 Android Forum에서는 수많은 개발자가 G1에서 정상적으로 작동하는 2.0 펌웨어 이미지를 만들려고 노력하고 있으며 현재로는 사용이 가능한 정도의 이미지도 몇 몇 나오기 시작했다. 하지만 이 방법을 통해 G1에 안드로이드 2.0 펌웨어를 적용하였다 하더라도 다른 문제점이 발생한다. 결국 2.0 펌웨어를 업데이트하여 G1에서 공식 Bluetooth API를 사용 할 수 있으나, 이는 안드로이드 2.0 버전 이상에서만 작동하므로 개발자가 블루투스 기능을 사용하는 어플리케이션을 작성 한 후에 안드로이드 마켓(애플의 앱스토어에 해당함)에 업로드 하더라도 실제 사용자도 2.0 버전 이상이 적용된 단말에서만 다운로드 받을 수 있고, 사용 할 수 있기 때문에 이는 제한적인 요소에 해당한다. 이에 대해서 2번째 방법에서는 조금 다른 방식으로 접근하여 1.5 버전 이상의 단말에서 블루투스 기능을 사용하는 방법을 소개한다.

2.2 블루투스 API 작성 가능 여부 확인

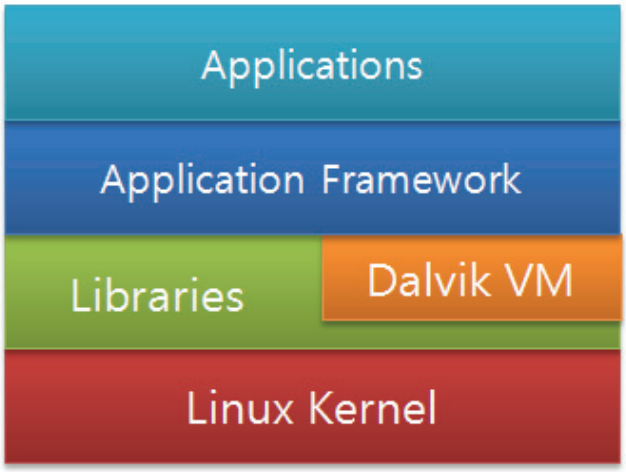

그림 1. 안드로이드 아키텍처

안드로이드 소프트웨어 스택은 위의 그림 1과 같은 형태의 구조를 가지고 있다. 가장 하층 레이어의 'Linux kernel'을 기반으로 하여 그 상단에 C/C++로 작성된 'Libraries'로 구성이 된다. 이 'Libraries'과 'Linux kernel '계층은 안드로이드의 'Dalvik Virtual Machine'을 통해 'Application Framework'와 'Application' 계층에 노출 된다. 안드로이드 소프트웨어 스택의 구조와 각 계층에 대한 자세한 설명은 참고 문헌 [4] 을 참고하기 바란다.

안드로이드 git의 소스 트리를 컴파일 하여 펌웨어 작성 시에 안드로이드의 소프트웨어 스택에서의 모든 계층을 포함하여 펌웨어 이미지가 만들어지게 된다. 본 논문에서 관여 된 주요 부분은 'Libraries' 레이어와 'Dalvik VM' 레이어이다. 일반적으로 PC에 설치되는 리눅스 운영체제 환경에서 블루투스 디바이스 제어는 대부분의 경우 BlueZ [5,6,7]라는 라이브러리로 블루투스 제어코드를 작성 할 수 있다. 본 연구에서는 이러한 환경이 안드로이드 플랫폼을 탑재한 임의의 단말에서도 적용 될 수 있는지를 알아보기 위해 실험용으로 진행한 단말에서 안드로이드 SDK에서 제공하는 ADB(Android Debug Bridge) [8] 의 Shell을 사용하여 BlueZ 라이브러리가 존재하는지 확인해본 결과, BlueZ 라이브러리가 설치 된 것을 알 수 있었다.

여기서 블루투스를 제어하기 위해서는 'Libraries' 레이어에 직접 접근하는 코드를 작성해야하는데 문제점은 C, C++ 기반으로 되어있는 라이브러리를 활용해야 하지만 Android 어플리케이션 코드는 Java기반으로 작성 된다는 것이다. 하지만 구글에서는 1.5버전 이상에서 작동하는 안드로이드를 위한 NDK(Native Development Kit) [9,10]를 따로 마련해 두었다. NDK란 C, C++로 작성되는 native code를 사용해 안드로이드에서 동작하는 임베디드 컴포넌트를 제어 할 수 있도록 지원하는 도구이다.

앞선 상황을 미루어 보면 안드로이드 SDK에서 블루투스 API가 지원되지 않더라도 블루투스 장치를 제어 할 수 있는 어플리케이션이나 API를 만들 수 있는 여지가 있다는 것을 알 수 있다.

3. 블루투스 API 작성

2절에서 소개한 내용을 토대로 안드로이드 1.5, 1.6 버전에서 작동 될 수 있는 블루투스 API를 작성하고자 한다. 앞서 밝힌바와 같이 안드로이드 플랫폼에는 BlueZ 라이브러리를 기본적으로 포함하고 있기에 해당 라이브러리를 사용하여서 코드를 작성 할 수 있다. 이 라이브러리에서는 이미 블루투스 스택 [5,6,7] 에서 가장 상위층인 Application 계층의 바로 하위 레이어인 RFCOMM/SDP 레이어에서 프로그램을 작성 할 수 있도록 구현되어 있기 때문에, 본 연구에서는 직접적으로 블루투스 스택을 구현할 필요 없이 라이브러리 내 포함 된 함수들을 사용하여 간단하게 안드로이드 블루투스 장치 제어를 위한 JNI [10] 라이브러리를 작성 할 수 있다. 안드로이드 블루투스 장치 제어를 위한 프로그램 작성의 개략적인 순서는 다음과 같다.

1. BlueZ를 이용한 블루투스 제어 코드를 NDK에 인식 될 수 있는 코드 형태로 작성.

2. (1)의 과정에서 작성한 코드를 Android NDK를 통하여 라이브러리 파일 생성.
3. (2)의 과정에서 생성 된 라이브러리 파일을 안드로이드 어플리케이션 프로젝트에 포함.
4. 안드로이드 어플리케이션 프로젝트에 포함 된 블루투스 권한 부여하는 코드 작성.
5. 안드로이드 블루투스 제어 어플리케이션 코드 작성 시 필요한 부분을 (3)의 과정에서 생성된 라이브러리 호출 형태로 작성.

3.1 로컬 블루투스 장치 ID 확인 모듈 작성

안드로이드에 적용하기 이전에 PC에 블루투스 동글을 설치하여 미리 테스트를 해보았다. PC에서 동작하는 코드는 BlueZ 라이브러리의 간단한 함수조작을 통하여 운영체제 내에서 정의된 블루투스 장치의 Device ID를 리턴 하는 모듈을 제작하였다.

이 모듈을 이용하여 안드로이드에서 불러 올 수 있는 라이브러리를 작성하기 위하여 다음과 같은 작업을 하였다. 안드로이드 어플리케이션 프로젝트에서 native code를 불러들이는 메서드와 라이브러리를 로드하는 Java 클래스를 작성 한 후, 이를 javah의 C Header File Generator [10] 를 통하여 C 헤더 파일을 생성하였다. 이전에 작성한 Device ID를 가져오는 모듈을 JNI의 양식에 맞게 수정 한 후 안드로이드 어플리케이션에서 사용하기 위하여 프로젝트에 포함하였다. Android-NDK-r3 [9] 버전에서는 기본적으로 블루투스 라이브러리를 포함하고 있지 않기 때문에 이를 PC에서 설치 된 BlueZ라이브러리를 NDK의 라이브러리에 포함하여 컴파일 하였다. 하지만 실제 컴파일 중에서는 블루투스 라이브러리를 찾을 수 없다고 나와서 원인을 분석한 결과, 안드로이드 커널에서는 ARM EABI(Embedded Application Binary Interface)를 사용한다는 것에 있었다 [11]. 단순히 안드로이드가 BlueZ 3.36 버전에 기반을 하였다는 것에서 PC에서 사용되는 BlueZ 라이브러리를 그대로 사용하는 것이 아니라 NDK를 작성하는데 사용되는 라이브러리도 안드로이드 단말의 arm의 환경에 맞춰 컴파일 된 BlueZ 라이브러리가 필요하다는 것을 알 수 있었고, 이에 맞춰 다시 블루투스 Device ID를 반환 하는 JNI native code를 컴파일 하였다. 컴파일 된 목적 파일이 생성 된 것을 확인 한 후, 간단하게 작성한 안드로이드 어플리케이션의 Activity [8] 내에서 Native 메서드를 호출 하여 실제로 동작하는 것을 확인 할 수 있었다.

3.2 주변 장치 스캔 모듈 작성

주변의 블루투스 장치를 스캔하여 스캔 된 장치의 MAC 주소를 반환하는 모듈을 작성하였다. BlueZ의 스캔 모듈을 호출하였으며, 이 모듈에서 스캔한 결과를 저장하는 inquiry_info [6,7] 구조체들의 배열을 안드로이드의 Java환경에서 인식 할 수 있는 jobjectArray [10] 형태로 변환하여 반환 하는 모듈을 작성하였다. 이 모듈에서는 C++ 코드를 사용하였기에 NDK에서 사용되는 make파일의 내용에 LOCAL_DEFAULT_CPP_EXTENTION [9] 옵션을 부여하였다.

3.3 클라이언트 모듈 작성

안드로이드에서 PC 블루투스 동글로 데이터를 송신하는 모듈을 작성하였다. [6,7] 모듈의 구성 통신 방식은 RFCOMM [5] 을 통해 통신하였고, PC의 블루투스 동글에서 서버로 설정된 소켓 포트가 1로 되어 있었기에 안드로이드 클라이언트에서는 블루투스 동글 서버의 1번 포트로 접속하는 소켓을 생성하였다. 생성된 소켓 스트림을 통하여 "Hello, ConetLIB"이라는 15Byte의 메시지를 전송하는 모듈을 작성하였다. 메시지 전송은 안드로이드 어플리케이션 내에서 버튼을 생성하여 버튼이벤트를 통해 처리하였으며, 윈도우 하이퍼터미널의 블루투스 COM 포트에 연결하여 실제 데이터가 전송 된 것을 확인 할 수 있었다. (그림 2)

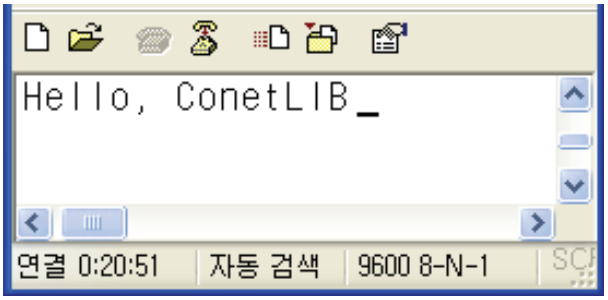

그림 2. 안드로이드 단말과 PC 블루투스 동글 간의 통신

4. 연구 결과 및 분석

본 논문에서는 기존에 구글에서 지원하지 않던 안드로이드 1.x버전에서 블루투스를 사용하기위한 방법으로 NDK를 사용하여 안드로이드 어플리케이션 프로젝트에 사용 될 수 있는 블루투스 API 작성 방법을 제안하였다. 연구 결과로는 안드로이드의 소프트웨어 스택의 전반적인 구조에 대한 이해를 통하여 안드로이드 단말의 블루투스 장치를 제어하는 API를 개발 할 수 있었다.

연구 개발환경은 PC에서는 Linux Fedora 10과 Windows XP 운영체제를 사용하였고, PC 블루투스 장치로는 블루투스 동글(DBT-122), BlueZ 라이브러리는 3.36을 사용하였다. Android 단말은 ADP1(Android development Phone 1)을 사용하였고 사용된 단말의 펌웨어 버전은 1.6(Donut)버전이다. 안드로이드 어플리케이션 개발에 사용한 SDK는 Android-SDK-1.5, NDK로는 Android-NDK-r3를 사용하였다. 안드로이드 1.5 SDK를 사용하여 어플리케이션을 개발하였기에 1.5 펌웨어를 사용하는 단말에서도 동일하게 적용 될 수 있다.

본 연구에서는 테스트를 위해 간단한 블루투스 통신 어플리케이션을 작성하였고 다음 그림 3의 아키텍처에서

Application레이어 하단에 위치하는 API for Bluetooth(Java Class) 레이어에 해당하는 Android Bluetooth API Class를 개발하였다. 그리고 이 클래스가 사용하는 라이브러리인 Bluetooth JNI Library(C++ Shared Library)레이어의 공유 라이브러리를 작성하였다. JNI 라이브러리에서는 Linux Library에 해당하는 BlueZ 블루투스 스택을 제어하는 형식으로 구성하였다.

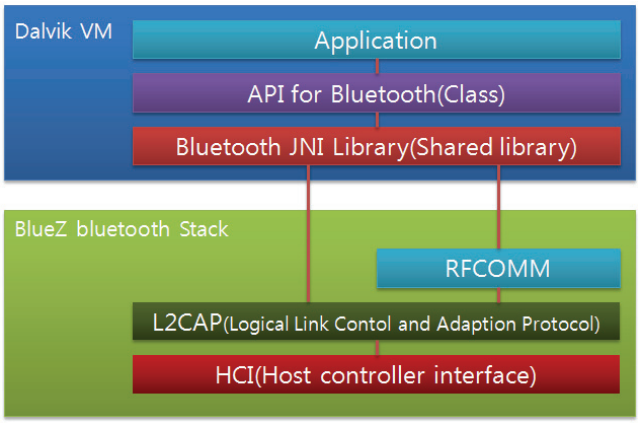

그림 3. 블루투스 프로그램 아키텍처

본 연구에서 진행한 방식과 같이 개발자가 임의로 제작한 공유라이브러리(.so)는 안드로이드 단말의 파일시스템 내에서 /data/local과 같은 공유라이브러리 디렉터리에 위치해야한다. 하지만 이 라이브러리(Bluetooth JNI Library)가 일반 사용자 단말의 파일 시스템에는 포함 되어 있지 않기 때문에, 이 라이브러리를 사용하는 어플리케이션을 배포하기 위해서는 안드로이드 어플리케이션 파일 (.apk) [8] 에 JNI 라이브러리를 포함시켜야 한다. 안드로이드 어플리케이션 프로젝트에서 JNI라이브러리를 import 한 후, apk파일을 출판 하게 되면 JNI라이브러리가 포함된 apk파일을 생성 할 수 있게 된다. 라이브러리가 어느 곳에 위치하는지 확인하기 위하여 본 연구에서 개발한 어플리케이션 apk파일을 단말에 설치 한 후 위치를 확인 한 결과 이 라이브러리는 /data/data/'PACKAGE_NAME'/lib 디렉터리 내에 lib'SHARED_LIBRARY_NAME'.so 형태로 위치한 것을 확인 할 수 있었다. 결과적으로, 본 연구에서 제작한 라이브러리를 사용하여 안드로이드 블루투스 어플리케이션을 개발하기 위해서는 이 라이브러리를 apk에 포함하고 안드로이드 프로젝트에서 불러오는 방식으로 배포 가능한 프로그램을 작성 할 수 있다.

참고 문헌

[1] HTC, "G1 overview"
http://www.htc.com/www/product/g1/overview.html
[2] Jon Loeliger, "Version Control with Git: Powerful Tools and Techniques for Collaborative Software Development", O'ReillyMedia, 2009
[3] Android Open source Project, "Building for Dream Or Sapphire"
http://source.android.com/documentation/building-for-dream
[4] Adam M. Dutko, "an introduction to the new google mobile Linux framework, android", ACM Linux Journal, 2008, ISSN:1075-3583
[5] Jennifer Bray, Charles F Sturman, 이문수 역 "한국어판 블루투스 (Bluetooth:Connect Without Cables)", 홍릉과학출판사, 2001
[6] Marcel Holtmann, Andreas Vedral, "Bluetooth programming for Linux", Wireless Technologies Congress, 2003
http://www.irexpert.ir/UploadedFiles/Expert/BillBoardFiles/wtc2003_slides.pdf
[7] Albert Huang, MIT University CSAIL LAB, "An Introduction to Bluetooth Programming"
http://people.csail.mit.edu/albert/bluez-intro/
[8] Reto Meier, 조성만 역 "Professional Android Application Development", 제이펍, 2009
[9] Android Developers, "NDK"
http://developer.android.com/sdk/ndk/index.html
[10] Rob Gordon, "Essential Jni: Java Native Interface (Essential Java)", Prentice Hall PTR, 1998
[11] KLDP Wiki, "Android Porting On Real Target",
http://wiki.kldp.org/wiki.php/AndroidPortingOnRealTarget