KVM 을 위한 Qplus Profile 구현*

°이재호 김흥남 *이현철
한국전자통신연구원 인터넷정보가전연구부
*㈜오프너스
email: {bigleap, hnkim}@etri.re.kr , cloudhc@openers.co.kr

The Implementation of Qplus Profile For KVM

Jae-Ho Lee°, Heung-Nam Kim, *Hyun-chul Lee
Internet Appliance Technology Dept. ,ETRI
*OPENERS Co.,Ltd.

요 약

KVM(Kilobyte Virtual Machine)은 Sun Microsystems 의 자바가 갖고 있는 장점과 기능을 최대한 수용하면서 셀룰러 폰, 양방향 페이저, PDA 및 셋탑 박스와 같이 메모리 공간이 적고, 낮은 CPU 사양을 갖는 네트워크 디바이스에 적용할 수 있도록 만든 경량의 JVM(Java Virtual Machine)이다. 본 논문은 인터넷 정보가전 기기를 위한 실시간 운영체제인 Qplus(Q+)에서 자바 응용 프로그램이 구동 될 수 있도록 J2ME 에서 요구되는 KVM 용 Profile 계층을 구현하고, 이를 커널에 통합하여 D-TV 보드(SA110)와 Cerf 보드(SA1110)에서 동작실험을 하였다

1. 서론

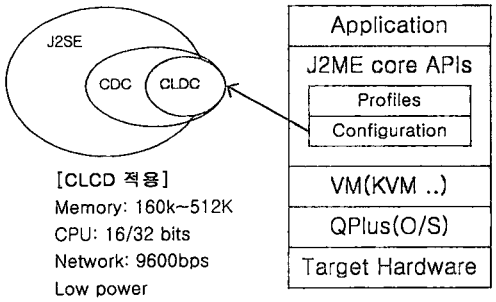
IMT2000 에 대한 관심과 더불어, 무선 인터넷 시장이 지속적으로 크게 성장하고 있다. 이러한 모바일 디바이스에 탑재된 실시간 운영체제 위에 KVM 과 같은 자바 플랫폼을 구축하면, KVM 기반으로 개발된 웹 브라우저, 게임과 같은 자바 응용프로그램이 운영체제와는 독립적으로 서로 다른 무선기기에서도 코드 수정 없이 수행될 수 있다. 이는 무선 인터넷 콘텐츠 제공자가 플랫폼에 제약되지 않고 독립적으로 응용프로그램들을 개발하고, 사용자의 모바일 디바이스에 다운로드 서비스를 관리함으로써 이익을 창출할 수 있는 새로운 사업 모델을 제시한다.

본 논문의 2 장에서 KVM 기반에서 동작되는 J2ME 와 전자통신연구원에서 개발된 Qplus 커널과 개발 툴(Esto)을 소개하고, 현재 국내에서 쓰이고 있는 무선환경의 자바 플랫폼에 대한 기술 동향을 관련연구로써 간략히 기술하였다. 3 장에서는 QPlus 실시간 운영체제를 사용하는 임베디드 시스템에서 KVM 기반의 응용프로그램이 실행될 수 있도록 커널과 디바이스에 맞는 프로파일 계층을 구현하였으며, 4 장에서 Qplus 커널 위에 이식된 KVM 에 대한 동작 검증실험 하였다.

2. 관련연구

2.1 J2ME (Java2 Platform Micro Edition) 개요

Sun 의 자바는 적용분야에 따라 Enterprise, Standard, Micro Edition 의 3 가지 솔루션이 있다(Java 2 EE > J2SE > J2ME).
이 중에 J2ME 는 휴대용 소형기기나 가전용 기기에서 자바를 실행하기위한 플랫폼으로 개발된 것으로, [그림 1]의 J2ME Configuration 에 따라 CDC 와 CLDC 로 나누어 진다. 개발에 적용한 기술은 CLDC 로 이는 KVM 과 J2SE 코어 라이브러리로 이루어진다.



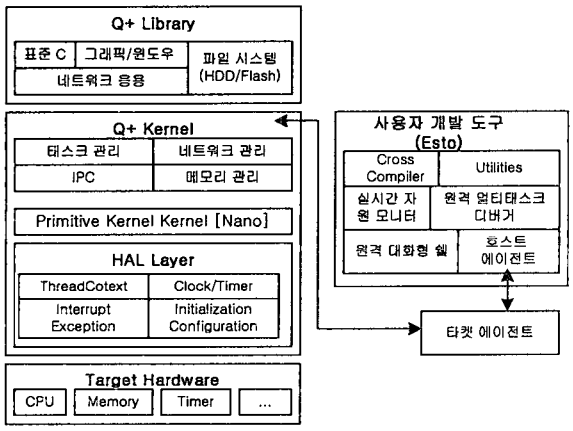
[그림 1] J2ME 의 개요

J2ME 의 Profile 은 Configuration(CLDC/CDC) 기반에서 타겟 디바이스에 사용자 API 를 지원하여 각 응용프로그램이 하드

하드웨어나 운영체제에 의존하지 않고 호환성 있게 동작될 수 있도록 설계해야 하며, 주로 GUI, 데이터 입출력, 네트워킹 부분이 관계된다. 본 논문에서 KVM 기반의 응용 프로그램을 작성하기 위한 사용자 수준의 API 를 만들기 위해 주로 이 부분의 Native Code 들을 수정 및 추가 작성 하였다.

2.2 실시간 운영체제: Qplus-T(Thread-based RTOS)

Qplus 실시간 운영체제는 전자통신연구원에서 개발한 RTOS 로서 쓰레드 기반(Qplus-T) 과 프로세스 기반(Qplus-P) 의 두 가지 모델이 있다. 본 논문에서는 Qplus-T 와 이와 연동하여 커널 및 응용프로그램을 개발 할 수 있는 개발도구 Esto 툴을 사용하였으며, [그림 2]와 같이 일반 상용 RTOS 와 비슷한 구조와 기능을 갖는다.



[그림 2] Qplus 실시간 운영체제의 구성과 기능

현재 임베디드 업계에서 사용되는 대부분의 상용 RTOS 에 대하여 외국에 값비싼 사용료를 지불하며 이용하고 있다. 이러한 현실에 반하여, 국산 RTOS 활성화를 위해 커널 소스와 함께 Esto 툴을 교육용으로 무료로 배포하는 사이트를 운영하고 있다[참고문헌].

2.3 국내 무선 플랫폼 시장 동향

자바를 사용하는 J2ME 기술은 WAP 프로토콜의 정적인 텍스트 데이터 전송과는 달리 동적인 그래픽을 이용 가능하기 때문에 대부분의 무선 인터넷 시장에서 널리 사용되고 있으며, 국내 통신 업체의 사용현황은 [표 1]과 같다.

특히 KVM 의 소스는 10000 라인 정도로 크기가 작고, C 언어로 구현되어 있어 이식성이 우수하다. 많은 업체에서 CLDC 를 기반으로 MIDP(Mobile Information Device Profile)를 제정하여 여러 유형의 모바일 디바이스의 프로파일(디스플레이, 입력장치,GUI) 의 API 들을 스펙에 맞추어 제작하고 있으므로 J2ME 기술이 더욱 확장될 것으로 전망된다.

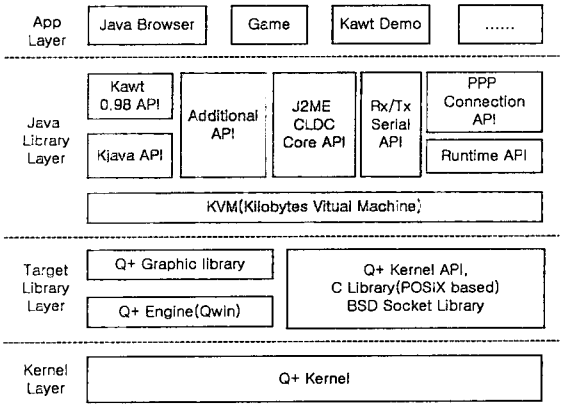
[표 1] 국내 무선시장에서의 자바 기술

자바 플랫폼	개발사	사업자
KVM	Sun Micorsystems	LG 텔레콤(2000)
GVM	신지소프트	SK 텔레콤(2000)
MAP	모빌탑	KTF(2001)
XVM(SKVM)	XCE	SK 텔레콤(2001)
BREW	퀄컴	KTF(시범서비스)

3. KVM 동작을 위한 Qplus Profile 구현

3.1 KVM 라이브러리 구조

일반적인 KVM 개발 시스템의 계층 구조는 [그림 3]과 같이 4 개의 계층으로 이루어진다. 가장 아래가 커널 계층이고, 바로 위에 Qplus 라이브러리로 구성되는 Target Library Layer 와 이를 바탕으로 KVM 과 코어 API 및 추가 API 라이브러리로 구성된 Java Library Layer 가 있다. 최상위 계층의 개발자들은 Java Library Layer 에서 제공하는 API 을 이용하여 다양한 응용 소프트웨어를 개발할 수 있으며, 그 밑에 계층 구성에 대해서는 고려하지 않고 설계할 수 있다.



[그림 3] KVM 라이브러리 계층 구조

다바이스 유형에 따라 작성해야 하는 Profile 의 주요 수정 대상인 GUI 클래스 라이브러리는 J2SE 와 호환성을 유지하는 kAWT 를 이용하였다. KVM 이식은 크게 2 가지 작업으로 이루어지는데, 첫째는 C 소스로 구성된 KVM 의 포팅이고, 둘째는 KVM 에 사용하는 코어 클래스 라이브러리의 커스트마이징이다. 이 작업에는 코어 클래스 라이브러리를 바탕으로 부가적인 라이브러리(kAWT API etc.)를 개발하는 과정이 포함된다. CLDC 소스 파일 디렉토리는 표와 같은 구성을 갖는다.

[표 2] CLDC 소스 구성

Directory	Descriprtion
api	코어클래스 라이브러리
bin	실행이미지
build	플랫폼별 Makefile
docs	KVM 관련문서
jam	Java Application Manager native code
kawt	kAWT 클래스 라이브러리 및 kawt 관련문서
kvm	KVM 런타임 native code
samples	예제 프로그램
tools	Java Code Compactor 및 preverfify관련소스

이 중 Qplus 커널 및 디바이스 의존성 때문에 수정하거나 새롭게 추가 구현한 작업 디렉토리는 다음과 같다.

- kvm/VmCommon : 시스템 property
- kvm/VmExtra: 네트워크, 시리얼 통신, startup 코드
- kvm/VmQplus: 커널 종속적인 코드, GUI native code
- tools/jcc/src/runtime: Native Call Function Lookup Table

3.2 KVM 개발 환경

Qplus-T 커널에 KVM 을 추가하기위한 개발환경은 호스트 부분과 타켓 부분으로 나누어지므로 각각에서 동작할 수 있는 아래의 C 컴파일러가 필요하다.

	C compiler	Java Compiler
Host	MS Visual C++ 6.0	Sun Microsystems JDK1.3.1 for windows
Target	Cygnus GCC 2.9 (StrongARM cross comiler)	None

호스트에서는 자바 소스를 컴파일하여 바이트 코드인 클래스 파일을 만드는 과정(소스 컴파일 단계)과 이를 통해 생성된 클래스 파일을 검사(사전 검사 단계)하는 두 가지 과정으로 구분된다. 소스 컴파일 단계에서는 JDK 에서 제공하는 자바 컴파일러(javac)를 사용하고, 검사 단계에서는 preverify 툴을 이용한다. 타겟 실행환경은 호스트 개발단계를 통해 생성된 클래스 파일을 타겟의 메모리에 다운로드하여 실행시키게 된다.

4. 동작 검증

4.1 KVM 응용프로그램 수행을 위한 실험 환경

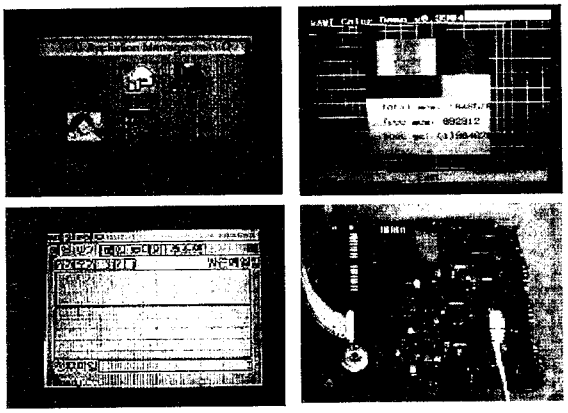
실험 검증을 위해 Qplus 커널과 KVM 이 이식된 하드웨어 사양은 아래의 [표 2]와 같다. 실행을 위해서는 KVM 컴파일 과정에서 생성되는 클래스 라이브러리(clazz.jar)를 타켓 파일 시스템의 루트에 위치 시키기 위해 타겟의 플래시 드라이버를 작성하고 Qplus 커널에서 관리하는 파일 시스템을 사용하였다. 또한 호스트에서 만들어진 클래스 라이브러리(clazz.jar)를 타겟의 플래시 메모리에 복사할 수 있도록 타겟 파일 시스템에서 동작되는 jsh 과 다운로드를 위한 tftp 를 구현하여 사용하였다.

[표 2] 실험용 하드웨어 사양

CPU		Intel StrongARM SA-1110
OS		Qplus-T
Main Memory	SDRAM	32M
	FLASH	16M
Display	LCD	Kyocera 5.7" VGA LCD (320 X 240 X 256 Color)
Touch Panel / Audio		UCB1200 Codec
확장 슬롯		Compact Flash II
외부 인터페이스		USB Slave
시리얼 인터페이스		Three 9-pin RS-232C Connectors
이더넷		Cirrus Logic CS8900A

4.2 실행결과 및 향후 연구과제

[그림 7]은 인텔의 SA1110 CPU 를 탑재한 보드에서 기존 개발된 kAWT 데모들을 실행시킨 화면이다. kAWT 는 J2ME CLDC 에 포함된 kJava 가 표준 자바의 UI 컴포넌트와 많은 차이를 보이고 있으므로 추가적인 소스 코드의 수정 없이 기존 AWT 를 사용하기위해 진행중인 프로젝트다.



[그림 7] KVM 기반의 응용프로그램 실행화면

향후 과제로서 자바 플랫폼의 무선 인터넷 표준화 동향에 대해 조사하고, 이들과 Qplus 커널이 실제 하드웨어에 이식되어 실용화할 수 있는 방안들을 연구해야 할 것이다.

5. 참고문헌

[1] ETRI, "조립형 실시간 OS 개발", December, 2000

[2] Sun Microsystems, "Java™ 2 Pltform Micro Edition(J2ME™) Technology for Creating Mobile Devices"

[3] Sun Microsystmes, "KVM Porting Guide", KVM 1.0.2. FCS February 26, 2001

[4] www.embenix.com/plus, Qplus 및 Esto 툴 배포사이트

[5] Intel, StrongARM SA-1100 Microprocessor Technical Reference Mannual, December 1998

[6] www.kawt.de, kAWT 프로젝트 사이트