정보가전 및 내장형 장치를 위한 Java 기술

Java Technology for Consumer Electronics and Embedded Devices

하영국(Y.G. Ha) 전자지불연구팀 선임연구원
임신영(S.Y. Lim) 전자지불연구팀 책임연구원, 팀장
함호상(H.S. Ham) 전자상거래연구부 책임연구원, 부장

최근 휴대 전화 및 인터넷이 대중화 되면서 실생활에서 무선 네트워크 서비스를 이용하는 사용자가 급속히 증가하고 있으며 가전 업체를 중심으로 인터넷 TV, 디지털 백색가전 등과 같은 정보가전 제품들이 개발되고 있다. 이와 같은 내장형 장치들은 기존의 PC와는 다른 제한된 운영 환경을 제공하며, 이를 위한 경량의 Java 플랫폼으로서 EJAE, PJAE, J2ME 등이 등장하였다. 이러한 내장형 Java 플랫폼들은 기존의 Java가 가지고 있던 객체지향성, 분산성, 플랫폼 독립성, 보안성 등과 같은 주요 특징들을 대부분 수용하면서 제한적인 자원 및 연산 능력을 갖는 내장형 장치에 적합하도록 설계되어 있다. 본 논문에서는 내장형 Java 플랫폼의 기술 동향에 대하여 살펴보기로 한다.

I. 서 론

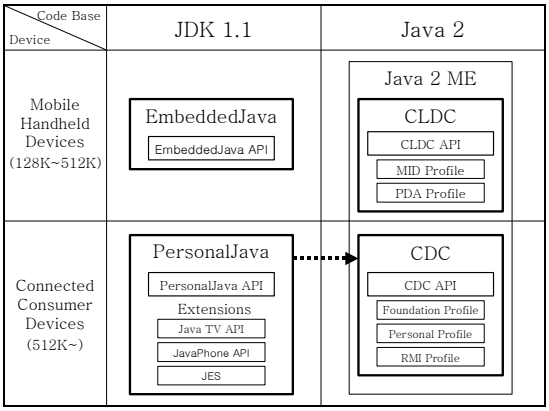
최근 들어 이동통신업자 및 ISP를 중심으로 무선 인터넷 서비스 경쟁이 심화되고 있는 가운데 인터넷 브라우저(WAP 또는 ME)를 탑재한 휴대 전화 및 웹 패드와 같은 무선 단말기의 보급이 급속도로 증가하고 있다. 또한 일반 가정에도 인터넷 사용이 보편화 됨에 따라 홈 네트워크 및 정보가전에 대한 관심이 날로 증대되고 있다. 이미 일부 가전 업체들을 중심으로 디지털 셋탑박스, 인터넷 TV, 인터넷 냉장고, 웹 폰 등이 개발되고 있으며 가까운 미래에는 거의 모든 가전제품 및 각종 전자제품(포스트 PC 및 Navigation/Automotive 장치) 등이 인터넷과 연결되어질 것으로 예상된다. 이러한 내장형 장치(Embedded Device)들은 기존의 PC와는 다르게 제한적인 시스템 리소스 및 연산 능력을 제공하므로 여기에 적합한 어플리케이션 운영 환경을 필요로 하게 된다.

이와 같은 내장형 시스템을 위한 Java 플랫폼으로서 EJAE(EmbeddedJava™ Application Environment), PJAE(PersonalJava™ Application Environment) 및 J2ME™(Java 2 Micro Edition) 등이 등장하게 되었으며 이들은 객체지향성, 분산성, 플랫폼 독립성, 보안성 등과 같은 기존 Java의 주요 특징들을 대부분 수용하면서 제한적인 내장형 환경에 적합하도록 설계되었다.

본 논문에서는 우선 내장형 시스템을 위한 Java 플랫폼 기술의 개요에 대하여 살펴보고, EJAE, PJAE 및 내장형 어플리케이션을 위한 Extension에 대하여 설명한다. 그리고 J2ME 플랫폼의 구조 및 J2ME 플랫폼을 구성하는 CDC(Connected Device Configuration)와 CLDC(Connected Limited Device Configuration) 및 어플리케이션을 위한 Profile에 대하여 살펴보고 결론을 맺기로 한다.

II. 내장형 Java 플랫폼 개요

내장형 시스템을 위한 Java 플랫폼으로서는 EJAE, PJAE 및 최근에 Java 2가 발표되면서 등장한 Java 2 Micro Edition 등이 있다. (그림 1)은 각각의 플랫폼의 구성 및 상호 관계를 나타낸다.



(그림 1) 내장형 Java 플랫폼의 구성

그림에서 보듯이 EmbeddedJava 및 PersonalJava는 JDK 1.1 명세를 기반으로 하고 있다. 따라서 EmbeddedJava와 PersonalJava API는 JDK 1.1 API를 Code Base로 하여 구성되며, Person- al-Java의 경우는 Core API 외에도 Java TV API, JavaPhone API, JES(Java Embedded Server)와 같은 Extension 패키지를 추가할 수 있다.

Java 2 ME는 Java 2 명세를 기반으로 하고 있는데 다시 대상 H/W 플랫폼에 따라 CLDC 및 CDC로 나뉘어 진다. CLDC는 주로 PDA나 이동 전화 및 무선 단말기 등과 같은 휴대용 장치에 적합하며, CDC는 셋탑박스, 인터넷 TV, 디지털 냉장고 등의 내장형 가전 장치를 주요 적용 대상으로 설계 되었다. CDC의 경우 PersonalJava가 Java 2 플랫폼에 맞추어 진화한 형태이기는 하나 기존 JDK 1.1 기반의 PersonalJava와 공존하며 함께 내장형 Java 솔루션으로서 사용되고 있다.

III. EJAE와 PJAE

앞서 설명한 바와 같이 EmbeddedJava 및 Pers-onalJava는 JDK 1.1을 기반으로 개발되었으나 Java 2 ME 플랫폼이 발표된 이후에도 내장형 장치를 위한 Java 개발 환경으로 독자적인 위치를 차지하고 있다. 이 장에서는 Embedded-Java 및 PersonalJava에 대하여 살펴보고 내장형 Application 개발을 위한 PersonalJava 기반의 Extension 패키지에 대해서 설명한다.

1. EmbeddedJava

EmbeddedJava 응용 환경(EJAE)은 전용의 기능(Dedicated Function)을 갖는 장치를 위한 내장형 어플리케이션 개발 환경을 제공한다[1]. 다시 말해서 EmbeddedJava는 Core API의 개념을 제공하지 않으며 이를 탑재한 장치는 한 가지 목적의 기능을 위한 API만을 내장하고 있게 된다. 따라서 다른 종류의 장치를 위한 EmbeddedJava의 구현은 기능에 따라 모두 달라질 수 있는데 이러한 방식의 장점은 PersonalJava에 비해서 적은 양의 Memory Foot-print(256K~512K)를 요구한다는 점이다.

유사한 장치들을 대상으로 하는 J2ME의 CLDC 및 MIDP 명세가 발표된 이후에도 EmbeddedJava는 독자적인 내장형 프레임워크로서 그 위치를 지키고 있는데 플랫폼 기반의 Java 환경을 필요로 하지 않는 내장형 시스템 개발을 위한 Black-box 솔루션(API를 노출하지 않는)으로서 그 역할을 수행하고 있다.

가. EJAE 명세

EmbeddedJava 응용 환경은 Java 언어 명세(The Java Language Specification[2])에서 정의하고 있는 모든 사양을 지원하며, Java VM 명세(The Java Virtual Machine Specification[3]) 중 Class 로딩 및 Verification 기능을 선택사항으로 한 것 외에 모든 사양을 지원한다. 단 이 두 가지 선택사양은 개별적으로 구현 될 수 없고 함께 구현되거나 모두 제외되어야 한다[1].

나. EJAE API

EJAE API 명세는 JDK 1.1 API로부터 도출되었다. <표 1>은 EmbeddedJava API의 패키지 목록을 보여주는데 이들 중 지원하지 않는(unsupported) 패키지를 제외한 모든 패키지가 구현될 수도 있으며, 선택(configurable) 패키지 중 불필요한 Class, Method, Field 혹은 해당 패키지 전체는 구현시 생략될 수 있다[1].

<표 1> EmbeddedJava API

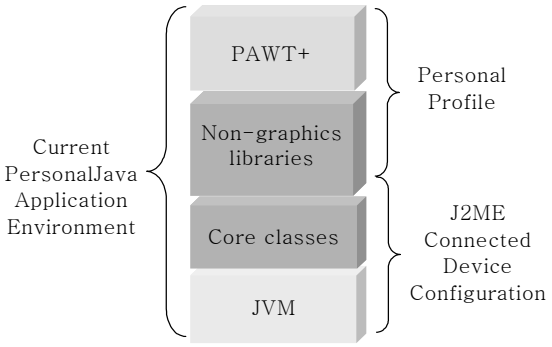
패키지 명	지원 수준
java.applet	unsupported
java.awt.*	configuarable
java.beans	configuarable
java.io	configuarable
java.lang.*	configuarable
java.math	configuarable
java.net	configuarable
java.rmi.*	configuarable
java.security.*	configuarable
java.sql	configuarable
java.text.*	configuarable
java.util.*	configuarable

2. PersonalJava

PersonalJava 응용 환경(PJAE)은 네트워크와 연결된 디지털 가전제품과 같은 내장형 장치를 위한 Java 플랫폼으로서 EmbeddedJava와는 다르게 범용의 기능(General-purpose Function)을 제공하기 위한 Core API가 모든 장치에 내장되어야 한다.

이와 같은 특징은 PersonalJava의 실행 환경이 EmbeddedJava에 비해 보다 많은 Memory Footprint(512K~)를 요구하는 반면 인터넷을 통해 다운로드한 Application(Java Applet)을 실행 가능한 내장형 시스템을 구현할 수 있도록 한다는 장점을 제공한다[4]. Core API 이외에 PJAE를 기반으로 하는 선택적인 확장 패키지로는 JES, Java TV API, JavaPhone API 등이 있으며 이러한 확장 패키지들은 Sun Microsystems 및 관련 전문가 그룹이 주도하는 JCP(Java Community Process) 프로그램을 통하여 공개적으로 개발된다.

현재 PersonalJava 플랫폼의 차기 버전은 J2ME가 등장하면서 CDC 명세에 흡수 통합되었으나 기존의 JDK 1.1을 Code Base로 하는 PersonalJava Application과의 호환성을 제공하기 위한 CDC 기반의 Personal Profile이 제공된다. (그림 2)는 PJAE 및 J2ME CDC 간의 매핑을 나타낸다.



(그림 2) PJAE와 J2ME

가. PJAE 명세

PersonalJava 응용 환경은 Java 언어 명세 및 Java VM 명세에서 정의하고 있는 모든 사양을 지원한다[4].

나. PJAE API

PJAE API 명세는 기본적으로 JDK 1.1 API를 기반으로 하고 있는데 추가로 마우스가 없는 장치를 위한 인터페이스, PJAE 관련 Exception 및 Timer Event 관련 Class 등을 정의하고 있다. <표 2>는 JDK 1.1 기반의 PersonalJava API 패키지 목록을 보여준다. 지원 수준은 required의 경우 모든 기능이 구현되어야 하고, optional의 경우는 선택적으로 구현이 가능하다는 의미이며, modified는 해당 패키지 중 일부가 optional인 경우, unsupported는 PJAE에서 지원되지 않는 경우이다[4].

<표 2> PersonalJava API

패키지 명	지원 수준
java.applet	required
java.awt.	modified
java.awt..datatranster	required
java.awt..event	required
java.awt..image	required
java.awt..peer	modified
java.beans	required
java.io	modified
java.lang	required
java.lang.reflect	required
java.math	optional
java.net	required
java.rmi	optional
java.rmi.dgc	optional
java.rmi.registry	optional
java.rmi.server	optional
java.security	optional
java.security.acl	unsupported
java.security.interfaces	optional
java.sql	optional
java.text	required
java.text.resources	modified
java.util	required
java.util.zip	modified

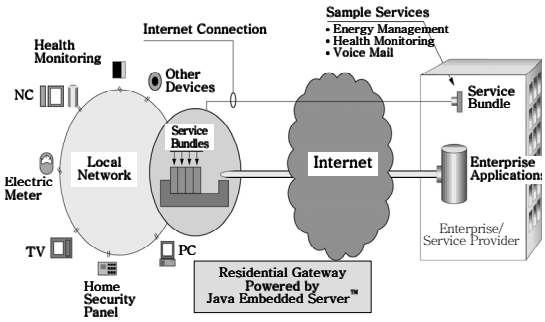
다. Java Embedded Server

Java Embedded Server™(JES)는 홈 서버 및 홈 게이트웨이 시스템 등을 위한 PersonalJava 기반의 응용 서버 프레임워크이다[5]. (그림 3)은 JES의 운영 환경을 보여준다.

JES 프레임워크는 크게 Service와 ServiceSpace의 핵심 구성요소로 나뉘어 지는데 Service는 컴포넌트화된 어플리케이션을 의미하며 ServiceSpace는 Service가 실행되는 환경을 제공하는 컨테이너의 역할을 수행한다[5].

1) Service

Service는 특정한 역할을 수행하기 위한 Class들의 집합으로 이루어진다. 예를 들어, 가정의 전기 사용량을 모니터링하는 시스템을 위하여 A/D 컨버터를 통해 전기계량기의 값을 읽어낼 수 있는 Method를 포함하는 Class들을 구현하는 것을 생각해 볼 수 있다. 이러한 Service는 Bundle이라는 형태로 네트워크상에서 전송되는데, 일반적으로 하나의 Service Bundle은 특정 Service를 위한 Class들을 포함하는 JAR(Java Archive) 파일 형태로 이루어진다[5].



(그림 3) Java Embedded Server

2) ServiceSpace

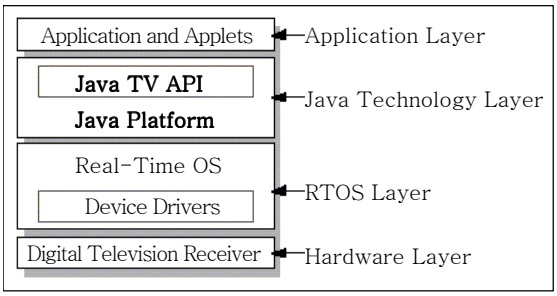
ServiceSpace는 Service가 수행되기 위한 실행 환경을 제공하는 프레임워크로서 Service의 Lifecycle 관리(Installation, Instantiation, Execution, Termination, 및 Unloading) 기능을 제공한다. 또한 Service들간의 Dependency 문제를 해결하고, Service를 위한 API를 제공한다. JES는 기본적으로 다음과 같은 Core Service를 포함하며 처음으로 서버가 시동될 때 ServiceSpace로부터 수행할 수 있도록 하고 있다[5].

- HTTP Service
- Log/Remote Log Service
- Date/Time Service
- Connection Manager Service
- Thread Manager Service
- Scheduler Service
- RMI(Remote Method Invocation) Service
- SNMP(Simple Network Management Pro-

tocol) Service
• Console/Administration Service

라. Java TV API

Java TV™ API는 디지털 TV를 위한 Java 플랫폼 Extension으로서 Sun Microsystems 및 디지털 TV 업체들을 중심으로 개발되었으며, 세계의 주요 가전 업체에서는 이를 디지털 TV 국제 표준으로 채택되도록 지원할 것임을 발표한 바 있다. (그림 4)는 Java TV API를 적용한 디지털 TV Application의 S/W 스택을 보여준다[6].



(그림 4) Java TV API 환경

Java TV API는 디지털 TV(Interactive TV, 인터넷 TV 등) 수신기를 위한 다음과 같은 기능들을 제공한다[6].

• Audio/Video Streaming
• Conditional Access
• In-band and Out-of-band Data Channels Access
• Service Information Data Access
• Tuner Control for Channel Changing
• On-screen Graphics Control
• Media Synchronization
• Application Lifecycle Control

이러한 Java TV API를 이용하여 TV 컨텐츠 개발자들은 VOD(Video on Demand), EPGs(Electronic Programming Guides) 또는 대화형 스포츠 중계(Interactive Sporting Events) 등의 다양한 서비스를 개발할 수 있다.

마. JavaPhone API

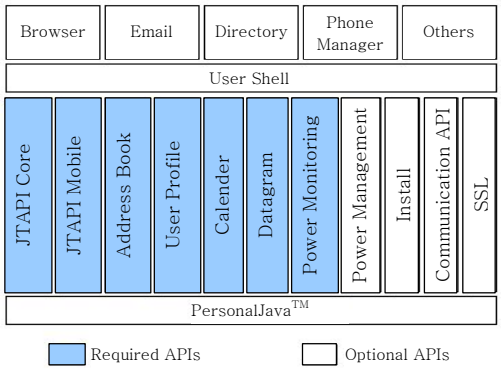
JavaPhone™ API는 디지털 전화를 위한 PersonalJava 플랫폼 Extension으로서 Sun Microsystems 및 세계의 주요 Telecommunication 업체들을 중심으로 개발되었으며, 디지털 전화기상의 다양한 정보 교환을 위한 안전한 환경을 제공한다[7].

JavaPhone API는 디지털 전화기(Wireless Smartphone, 인터넷 Screenphone 등)를 위한 다음과 같은 기능들을 제공한다[7].

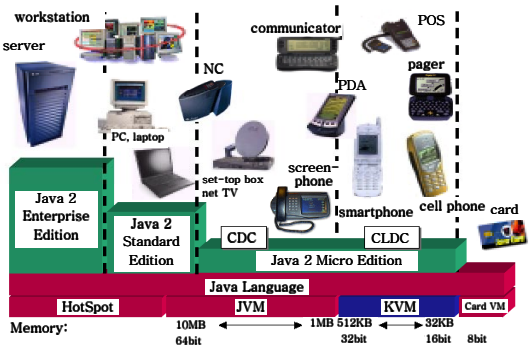
• Direct Telephony Control
(JTAPI Core, JTAPI Mobile APIs)
• Datagram Messaging
(Network Datagram API)
• Address Book Information Access
(Address Book API)
• Calendar Information Access
(Calendar API)
• User Information Access
(User Profile API)
• Power Management
(Power Management, Power Monitor APIs)
• Application Installation Mechanisms
(Installation API)
• Serial Communications
(Communication API, SSL API)

(그림 5)는 무선 전화기(Wireless Smartphone) 어플리케이션을 위한 JavaPhone API 패키지의 구성도를 보여준다.

이러한 JavaPhone API 컴포넌트를 이용하여 전화 장비 제조업체는 개발 시간 단축 및 제품의 품질을 높일 수 있으며, 컨텐츠 제공자들은 H/W 플랫폼에 독립적인 컨텐츠의 개발을 통해 서비스 시장을 넓힐 수 있고, 인터넷 망 사업자들은 새로운 형태의 어플리케이션 및 부가 서비스를 다양한 장비를 통해 제공할 수 있는 기회를 얻게 된다.



(그림 5) 무선 전화용 JavaPhone API 구성



(그림 6) Java 2 플랫폼

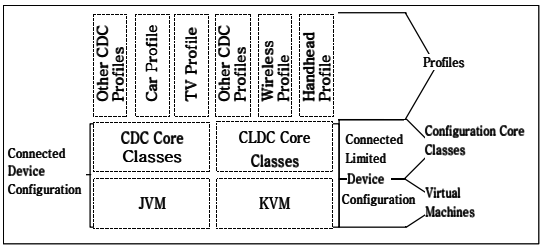
IV. Java 2 ME

Sun Microsystems는 Java 2를 발표하면서 이를 세 가지 플랫폼으로 분류하였는데 Java 2 Enterprise Edition(J2EE™), Java 2 Standard Edition(J2SE™) 및 Java 2 Micro Edition(J2ME™)이 그것이다. 이들의 명칭에서 알 수 있듯이 J2EE는 기업 환경에 적합한 Java 플랫폼을, J2SE는 개인용 컴퓨팅 환경을 위한 Java 플랫폼을, J2ME는 정보가전 및 내장형 기기를 위한 Java 플랫폼을 제공한다[8]. (그림 6)은 각각의 Java 2 플랫폼 환경을 도식화한 것이다.

J2ME는 다양한 내장형 시스템에 적합한 맞춤형 실행 환경을 제공하기 위한 Java Virtual Machine 및 API 세트로 구성되어 있으며 Configuration과 Profile이라는 두 가지 중요 요소를 가지고 있다[3]. (그림 7)은 J2ME의 구조를 나타낸다. 현재 J2ME 기반의 다양한 Configuration 및 Profile들이 JCP 프로그램을 통하여 공개적으로 개발되었거나 개발 중에 있다. 이 장에서는 J2ME 기반의 Configuration 및 Profile들에 대해서 설명한다.

1. Configurations

J2ME 플랫폼은 두 가지 설계 기준(Design Center)을 설정하고 있는데 그것은 휴대할 수 있는 이동형 장치(PDA, 무선 단말기 등)와 플러그에 연결된 고정형 장치(셋탑박스, 인터넷 TV, 디지털 냉장고 등)이다. 이러한 구분 하에 각각의 설계 기준은 그에 최적화된 VM 및 하위수준의 라이브러리를 필요로 하게 된다[8].



(그림 7) Java 2 ME Architecture

Configuration은 이러한 VM 및 하위 수준의 라이브러리로 구성되어 있으며 설계 기준에 따라 128K~512K의 응용 환경을 제공하는 것과 512K 이상의 응용 환경을 제공하는 것으로 나뉘어 진다. 또한 Configuration은 중첩가능(Nestable) 하여 하위 수준의 Configuration에서 수행되는 S/W는 상위 수준의 Configuration 상에서도 수행이 가능하다[8]. <표 3>은 J2ME Configuration(CLDC, CDC)을 정리한 내용이다.

가. CLDC

J2ME CLDC(Connected Limited Device Con-figuration)는 매우 제한적인 자원을 갖는 휴대용 장치를 위한 Virtual Machine인 KVM(Kilobyte Virtual Machine) 및 Core API로 구성되어 있다[9]. 이러한 장치들의 특성은 배터리로 동작하며 제

<표 3> J2ME Configurations

Configuration / Feature	CLDC	CDC
Virtual Machine	KVM	CVM
Processor	16~32bit	32~64bit
Memory Footprint	128K~512K	512K 이상
Design Center	휴대 가능한 이동형 장치	플러그인 된 고정형 장치
Target Devices	PDA, Wire-less Phone 등	STB, Inter-net TV 등

<표 4> CLDC API

구 분	기 능	패키지 명
J2SE 기반	System & Data types	java.lang
	I/O	java.io
	Collection, Time & Utilities	java.util
CLDC 전용	Generic connection	javax.microedition.io

한적인 메모리, 적은 연산 능력 및 Low bandwid-th, High-latency 네트워크 연결을 제공한다는 점이다.

CLDC의 핵심은 KVM이다. KVM은 그 이름에서도 알 수 있듯이 수십 Kbyte의 크기를 갖는 경량의 Java VM으로서 16~32bit RISC/CISC 프로세서를 장착한 휴대용 기기에 적합하다. KVM은 주어진 메모리 제한선 안에서 Java VM 명세를 지원하지만, 다음과 같은 차이점을 갖는다[10].

- Floating Point Data Type(float 및 double) 지원 안함
- Java Native Interface(JNI) 지원 안함
- User-defined, Java-level Class Loader 지원 안함
- Reflection 기능 지원 안함
- Thread Group 및 Daemon Thread 지원 안함
- Finalization 및 Class Instance 지원 안함
- Weak Reference 지원 안함
- Error Handling 기능 제한

CLDC API는 Java 2 SE로부터 직접적으로 상속된 것과 CLDC 전용으로 정의된 API로 구성된다. 전자의 경우 각각의 Class는 해당 Java 2 Class의 일부 Method 및 Field가 생략될 수 있는 Subset 형태로 구현되며, 후자의 경우 하부의 H/W에 종속적으로 구현된다[9]. <표 4>는 CLDC API 목록을 나타낸다.

나. CDC

CDC(Connected Device Configuration)는 CLDC 보다 상위 수준(Higher-end)의 Config-uration으로서 정보가전 기기와 같이 플러그에 연결된 고정형 장치를 위한 Java Virtual Machine인 CVM 및 Core API로 구성되어 있다[11].

CLDC의 핵심을 이루는 CVM은 Java 2 VM 명세의 모든 기능을 지원하며 32~64bit 프로세서를 장착한 정보가전 기기에 적합하다[12]. CDC API는 CLDC API의 Superset으로서 <표 5>는 CDC API 목록을 나타낸다.

<표 5> CDC API

구 분	기 능	패키지 명
J2SE 기반	System & Data types	java.lang java.lang.ref java.lang.reflect
	I/O & Network	java.io java.net
	Arithmetic	java.math
	Security & Certification	java.security java.security.cert
	Text handling	java.text
	Collection, Time & Utilities	java.util java.util.jar java.util.zip
CDC 전용	Generic connection	javax.microedition.io

2. Profiles

Profile은 하부의 Configuration을 기반으로 구현되며 특정한 장치를 위한 완전한 실행 환경을 제공하는 Java API이다. Profile은 완전성(Complet-

eness)을 가져야 하는데, 다시 말해서 해당 Profile을 기반으로 구현된 Application은 별도의 선택 패키지(Extension 패키지)의 추가 없이도 완전하게 실행되어야 한다는 것이다[9].

이러한 Profile들은 특정 산업 분야(Device 제조사, ISP, 통신 사업자 등)의 요구사항을 만족시키기 위한 방향으로 설계되고 통합되는데, 이는 Configuration 및 Profile 명세를 개발하는 JCP 프로그램 자체가 해당 업계의 전문가들로 구성된 그룹에 의해서 주도되기 때문이다.

현재 J2ME Profile은 CLDC를 기반으로 하는 MID Profile, PDA Profile과 CDC를 기반으로 하는 Foundation Profile, Personal Profile, RMI Profile 등이 개발 되었거나 개발중에 있으며, 새로운 profile들이 계속 제안되고 있다.

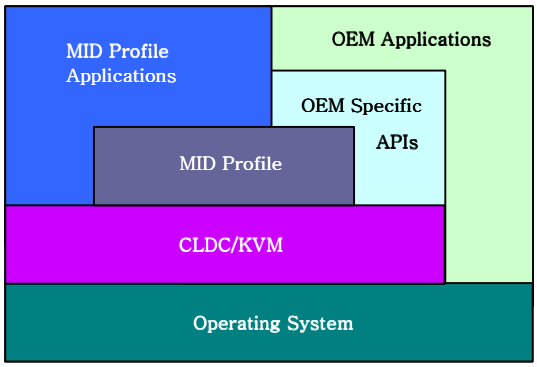
가. MIDP

MIDP(Mobile Information Device Profile)는 MIDPEG(MIDP Expert Group)에 의해 개발되었으며 휴대용 장치(Pager, Cellular Phone, PDA 등)에 탑재되는 Application 실행 환경을 제공하기 위한 CLDC 기반의 Profile로서 MIDP 명세에서는 다음과 같은 사항을 정의하고 있다[13].

- System Functions
- User Interface(LCD-based UI API)
- Persistence Storage(Record Management System)
- Networking(HTTP Connection)
- Application Model(MIDlet Suite)

(그림 8)은 MIDP 기반의 Application 환경을 보여준다.

MIDP는 휴대용 장치의 제한된 자원을 MIDP 응용(MIDlet)들간에 공유할 수 있도록 하는 Application 모델을 정의한다. MIDlet Application 모델에서는 Suite을 형성하는 다중 MIDlet들의 패키징 방법 및 단일 JVM 컨텍트 상에서의 자원 공유 방법 등을 정의하고 있다[13].



(그림 8) MIDP 응용환경

나. Foundation Profile

Foundation Profile은 CDC 기반의 Profile로서 2001년 2월 21일 현재 Final Draft가 승인되어 배포된 상태이다. Foundation Profile은 다양한 네트워크 연결(Socket, Datagram, HTTP 등)을 필요로 하지만 사용자 인터페이스를 필요로 하지 않는 장치를 대상으로 한다. 다시 말해서 GUI(Graphical User Interface) 기능 또는 다른 추가 기능을 지원하며 네트워크 연결을 필요로 하는 다른 Profile들을 위한 Base Profile로서의 기능을 수행하도록 설계되어 있다[14].

다. 기타 Profile

현재 JCP 프로그램을 통하여 개발이 진행중에 있는 J2ME Profile 명세로는 Personal Profile, RMI Profile, PDA Profile 명세 등이 있으며, <표 6>은 이와 관련된 개발 진행 상황에 대하여 정리하고 있다.

<표 6> 기타 J2ME Profiles

JSR No.	Profile Name	Base Cfg.	Status	Expert Group
62	Personal Profile	CDC	Expert group formed	Motorola, Nokia, Philips 등
66	RMI Profile	CDC	Proposed Final Draft	Epson, Siemens, SUN 등
75	PDA Profile	CLDC	Expert group formed	IBM, Palm, Sony 등

V. 결 론

본 논문에서는 정보가전 및 내장형 시스템을 위한 Java 플랫폼인 EmbeddedJava, PersonalJava 및 Java 2 ME 플랫폼 기술에 대하여 살펴보았다.

최근 WAP 또는 Mobile Explorer와 같은 무선 인터넷 브라우징 기능을 탑재한 이동 전화나 PDA 등의 보급이 급속히 증가되면서 내장형 Java 기술에 대한 관심이 모아지고 있다. 세계적인 시장분석 기관인 IDC는 오는 2004년까지 정보가전 분야의 세계시장 규모가 약 179억 달러에 이를 것이며, 무선 인터넷 서비스 가입자는 약 7억 5,000만 명에 이를 것이라고 예측하고 있다. 이러한 자료로 미루어 볼 때 가까운 미래에 초고속 무선 인터넷, 정보가전 및 Automotive 컴퓨팅 기기 등이 대중화되는 시점에서는 내장형 Java 플랫폼에 대한 기술 수요는 급속도로 증가할 것으로 전망된다.

그러나 아직까지 이러한 내장형 플랫폼 및 응용서비스에 대한 정보보호 기술의 개발이 활발히 이루어지고 있지 않은 상태이다. 따라서 조속한 관련 표준 확립과 내장형 경량 암호처리 기술 및 인증 기술, 무선기반의 종단간 보안 기술(Wireless End-to-End Security) 등과 같은 보다 체계적이고 적극적인 정보보호 기술에 대한 연구개발이 이루어져야 할 것으로 생각된다.

참 고 문 헌

[1] Sun Microsystems, "EmbeddedJava Application Environment Specification, Version 1.1(Final)," Jan. 1999.

[2] James Gosling *et al.*, "The Java Language Specification, 2nd Edition," Addison Wesley, Jun. 2000.

[3] Tim Lindholm *et al.*, "The Java Virtual Machine Specification, 2nd Edition," Addison Wesley, Apr. 1999.

[4] Sun Microsystems, "PersonalJava Application Environment Specification, Version 1.1.2(Final)," Aug. 1999.

[5] Anne Thomas *et al.*, "On-demand Embedded Applications," JES White Paper, Oct. 1998.

[6] Bart Calder *et al.*, "Java TV API Technical Overview, Version 1.0," Sun Microsystems Inc., Jul. 2000.

[7] Sun Microsystems, "JavaPhone API Specification, Version 1.0," 2000.

[8] Eric Giguere, "Java 2 Micro Edition Professional Developer's Guide," John Wiley & Sons, Inc., Nov. 2000.

[9] Sun Microsystems, "Connected Limited Device Configuration Specification, Version 1.0," May 2000.

[10] Sun Microsystems, "Java 2 Platform Micro Edition Technology for Creating Mobile Devices," KVM White Paper, May 2000.

[11] Sun Microsystems, "Connected Device Configuration and the C Virtual Machine," http://java. sun.com/products/cdc/

[12] Sun Microsystems, "The C Virtual Machine," http://java.sun.com/products/cdc/cvm/

[13] Sun Microsystems, "Mobile Information Device Profile(JSR-37) Version 1.0," JCP Specification, Sep. 2000.

[14] Sun Microsystems, "JSR #000046: J2ME Foundation Profile," http://java.sun.com/about-Java/community-process/jsr/jsr_04-6_j2mefnd.html.