2000 년 한국정보처리학회 추계 학술발표논문집 제 7 권 제 2 호

# 안전한 전자상거래를 위한 JavaCard Toolkit 의 설계 및 구현

하영국*
*한국전자통신연구원 전자상거래연구부
e-mail : ygha@econos.etri.ac.kr

# Design and Implementation of JavaCard Toolkit for Secure Electronic Commerce Application

Young-Guk Ha*
* Electronic Commerce Department, ETRI

요 약

최근 인터넷 전자상거래 시스템상에서 전송되는 개인 정보들을 안전하게 관리하기 위한 방법으로서 휴대가 가능한 스마트카드 시스템이 주목을 받고 있다. 현재 다양한 COS 를 탑재한 스마트카드 시스템들이 존재하고 있으나 강력한 보안 기능 및 다중 응용프로그램 환경을 제공하는 MULTOS 기반 시스템과 JVM 기반 JavaCard 시스템으로 서서히 양분되어 가고 있는 추세이다. 본 논문에서는 JavaCard 시스템을 대상으로 하는 Java Toolkit 의 설계 및 구현에 대하여 설명한다. 개발된 Toolkit 은 PKI 및 암호 처리 기술을 바탕으로 스마트카드, 사용자 시스템 및 서비스 제공자 시스템 간의 정보 교환을 위한 Java API 를 제공함으로써 안전한 인터넷 전자상거래 응용 및 다양한 정보보호 시스템 개발을 위한 기반을 제공한다.

## 1. 서론

최근 들어 일반인들을 대상으로 하는 인터넷 서비스가 보편화 되면서 B2C 기반의 전자상거래의 수요가 급속히 증가하고 있다. 뿐만 아니라 기업과 기업간(B2B) 혹은 기업과 정부간(B2G)의 전자상거래를 위한 포탈 사이트나 허브 사이트 등의 구축도 매우 활발히 이루어 지고 있다. 이와 같은 인터넷 기반의 전자상거래 시스템은 사용자들에게 기존 상거래에서는 경험해 보지 못했던 다양하고 방대한 정보와 편리함 등을 제공하고 있는 반면, 지불 정보나 구매 정보 혹은 개인 신상 정보 등의 유출과 같은 보안상의 문제점들을 해결해야 하는 어려움을 내포하고 있다.

최근에는 이러한 보안의 대상이 되는 정보들을 안전하게 관리하기 위한 방법으로서 휴대가 가능한 스마트카드 시스템이 주목을 받고 있다. 현재 다양한 COS(Card Operating System)를 탑재한 스마트카드 시스템들이 존재하고 있으나 강력한 보안 기능 및 다중 응용프로그램 환경을 제공하는 MULTOS 기반 시스템과 JVM 기반의 JavaCard 시스템으로 서서히 양분되어 가고 있는 추세이다.

이에 본 고에서는 JavaCard 시스템을 대상으로 하는 Java Toolkit 의 설계 및 구현에 대하여 설명한다. 개발된 JavaCard Toolkit 은 PKI(Public Key Infrastructure), 암호 처리 기술 및 Java 보안 기술을 바탕으로 스마트카드, 사용자 및 서비스 제공자 간의 정보 교환을 위한 Java API 를 제공함으로써 안전한 인터넷 전자상거래 응용 개발을 위한 기반을 제공한다. 본 논문의 구성은 다음과 같다. 2 장에서는 스마트카드 및 JavaCard 기술의 개요와 관련 표준들에 대해서 살펴보고, 3 장에서는 JavaCard Toolkit 의 설계 및 구현에 대해서 설명한다. 마지막으로 4 장에서는 향후 연구 방향에 대하여 논하고 결론을 맺는다.

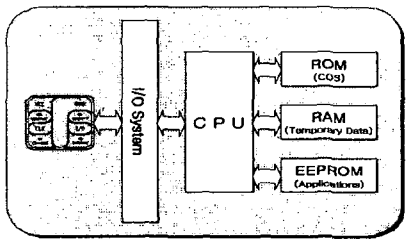
## 2. 관련 연구

### 2.1 스마트카드 기술

스마트카드란 마이크로프로세서와 메모리를 내장하고 있어서 카드 내에서 정보의 저장과 처리가 가능한 신용카드 형태의 플라스틱 카드이다. 스마트카드의 종

류는 마이크로프로세서의 내장 유무에 따라 메모리 카드와 마이크로프로세서 카드로 나뉘어지며 일반적으로 스마트카드라 함은 마이크로프로세서 카드를 뜻한다. 또한 카드 판독기와의 인터페이스 방식에 따라 접촉식, 비접촉식 및 혼합형 카드로 분류되기도 한다[1, 3]. 컴퓨터의 축소판이라고 할 수 있는 스마트카드(마이크로프로세서 카드)의 구조는 보통 CPU, ROM, RAM, EEPROM 및 I/O 시스템 등으로 구성되어 지는데, 스마트카드를 구동하기 위한 카드 운영체제(COS)는 ROM 에 탑재되며 EEPROM 은 응용프로그램을 위한 저장 공간으로 사용된다(<그림 1> 참조).

현재 스마트카드 기술의 수준은 기존의 단일 응용처리만을 지원하는 고정기능 스마트카드의 단계를 넘어서, 개방형 API 를 기반으로 다중 응용의 동적 로딩을 지원하는 MULTOS 및 JVM 기반의 스마트카드 기술이 보편화 되어가고 있다. 가까운 미래에는 16 비트 이상의 보다 강력한 CPU 및 대용량의 메모리를 장착하고 네트워크를 통한 서비스 접속을 지원하는 네트워크 스마트카드 기술의 개발이 이루어질 것으로 예상된다[1].




<그림 1> 접촉식 스마트카드의 구조

2.2 관련 기술 동향

본 절에서는 스마트카드와 관련된 국제 표준 단체나 업체 주도의 표준 명세와 기술 동향에 대해서 간략하게 살펴보기로 한다.

1) ISO(International Standards Organization) 표준

스마트카드를 위한 ISO 표준으로는 <표 1>과 같은 것 들이 있다. ISO 표준에서는 주로 스마트카드의 물리적, 전기적 특성과 판독기와의 통신에 필요한 프로토콜 명세 등이 제공된다[3, 6, 7].

<표 1> 스마트카드 관련 ISO 표준

| 형 태 | 표 준 | 내 용 |
|---|---|---|
| 접촉식 카드 | ISO7816-1 | Physical characteristics |
| | ISO7816-2 | *Dimensions and location of contacts* |
| | ISO7816-3 | Electronic signals and transmission protocols |
| | ISO7816-4 | Inter-industry commands for interchange |
| | ISO7816-5 | Numbering system and registration procedure for application identifiers |
| | ISO7816-6 | Inter-industry data elements |
| 비접촉식 카드 | ISO10536 | Dose coupling |
| | ISO/IEC14443 | Remote coupling |
| | ISO/IEC15693 | |

2) PC/SC(Personal Computer/Smart Card) 표준

Gemplus 와 Microsoft 등이 주축이 되어 활동하고 있는 PC/SC Workgroup 은 PC 환경에서 스마트카드와 판독기간의 상호 운용 및 관련 자원의 관리를 위한 표준 명세(Interoperability Specification for ICCs and Personal Computer Systems)를 정의하였다[3, 6, 7].

3) OCF(Open Card Framework)

IBM 을 주축으로 Gemplus, Visa, SUN 등과 같은 스마트카드 관련 기술을 보유한 업체들로 이루어진 OpenCard 컨소시엄의 OCF 는 스마트카드 및 판독기를 위한 Java 기반의 응용 프레임워크를 제공한다. OCF 환경하에서 개발자는 개방형 API 를 사용하여 스마트카드 호스트 시스템을 개발할 수 있다[3, 7].

4) OP(Open Platform)

Visa 에서 개발된 OP 은 다중 응용기반 스마트카드 시스템을 위한 개발 및 운용 환경을 정의한다. OP 은 카드 명세(Card specifications)와 터미널 명세(Terminal specifications) 등으로 구성되어 있으며 Open Platform 카드 구현을 위한 요구사항을 정의하고 있다[3, 7].

5) EMV(Europay, Mastercard, Visa) 표준

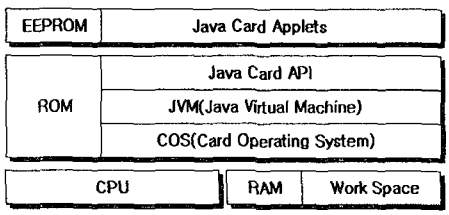
EMV 는 Europay, Mastercard 및 Visa 에 의해 제정된 표준으로서 ISO7816 표준을 기반으로 금융이나 지불 시스템을 위한 IC 카드, 터미널, 응용 등에 대한 명세를 정의하고 있다[3, 6, 7].

6) MULTOS(MULTi-application Operating System)

Mondex 에 의해 처음 개발된 MULTOS 는 높은 보안성 및 다중 응용프로그램의 사용을 지원하는 스마트카드용 운영체제 이다. MULTOS 는 응용프로그램 개발자를 위하여 스마트카드에 최적화된 언어인 MEL(MULTOS Enabling Language)과 MULTOS-API 를 제공한다[7].

2.3 JavaCard 기술

JavaCard 는 SUN Microsystems 에 의해서 개발된 Java 기술을 기반으로 하는 스마트카드 시스템이다. JavaCard 의 구조는 <그림 2>와 같으며 일반적인 스마트카드와의 차이점은 응용프로그램이 JVM(Java Virtual Machine)상에서 동작한다는 것이다[3, 4].




<그림 2> JavaCard 의 구조

JavaCard 의 특징은 플랫폼 독립성, 보안성 및 객체지향성 등과 같은 Java 의 특성을 대부분 그대로 이어받고 있다는 점이다. 또한 개발자의 측면에서 Java 컴

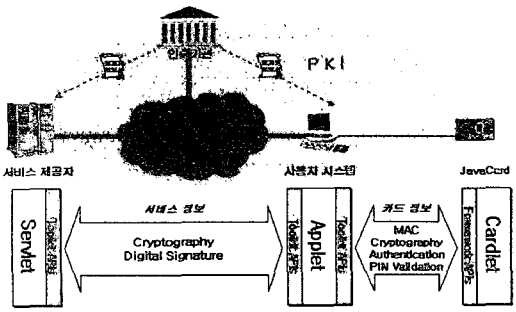
파일러 및 상용 IDE 와 같은 기존의 개발 도구를 그대로 사용할 수 있으며 개방형 API 인 JavaCard API 를 제공하고 있는 장점이 있다. 특히 스마트카드 기술적인 측면에서 볼 때 JavaCard 시스템은 다중 응용프로그램의 지원, 응용프로그램의 후발행(Post-issuance) 기능, ISO7816 과 같은 국제 표준과의 호환성 제공 등의 장점을 가지고 있다[2, 4, 7].

현재 대부분의 스마트카드 솔루션 업체에서 JavaCard 및 관련 S/W 패키지를 제공하고 있으며 대표적인 것으로는 SUN Microsystems 의 JavaCard Development Kit 을 비롯하여 Gemplus 의 GemXpresso, Schlumberger 의 Cyberflex, De La Rue 의 GalactIC, Bull 의 Odyssey 등이 있다.

## 3. JavaCard Toolkit 의 설계 및 구현

### 3.1 시스템 구조

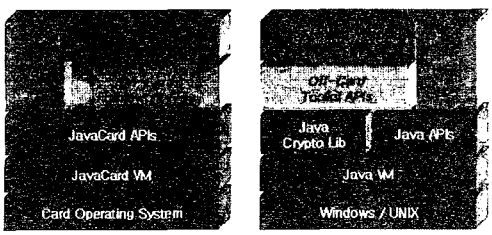
JavaCard Toolkit 과 이를 이용한 서비스 시스템의 구조는 <그림 3>과 같다. 그림에서 보는 것과 같이 서비스 제공자, 사용자 시스템 및 JavaCard 상에서는 각각 Servlet, Applet 및 JavaCard Applet(이하 Cardlet 이라 함)이 실행되며 JavaCard Toolkit API 가 제공하는 안전한 보안 채널을 통하여 필요한 정보를 서로 주고 받는다.




<그림 3> JavaCard 시스템의 구조

### 3.2 Toolkit APIs

JavaCard Toolkit 은 두 종류의 API 그룹으로 구성되어 있는데 JavaCard 상의 EEPROM 에 탑재되어 JCRE(JavaCard Runtime Environment)상에서 동작하는 On-Card Framework API 와 사용자 시스템 및 서비스 제공자 시스템에 탑재되어 JRE(Java Runtime Environment)상에서 동작하는 Off-Card Toolkit API 가 있으며 <그림 4>는 이들 API 의 구조를 보여준다.




<그림 4> Toolkit API 의 구조

각각의 API 들은 Java 패키지와 여기에 포함된 클래스의 형태로 제공되며 주요 API 및 기능은 <표 2>와 <표 3>에 정리되어 있다.

<표 2> On-Card Framework API(ver 0.9b)

| Package Name | |
|---|---|
| com.mcard.javacard.framework | |
| Classes | Description |
| APDUUtil | Application Protocol Utility API |
| ApplicationKey | Application Authentication API |
| Cardlet | JavaCard Applet Framework(3.3 참조) |
| CardString | Byte String Utility API |
| KeyGenerator | Secret Key Generator API |
| OPObject | Terminal Authentication API |
| OPUtil | Card Manager Utility API |
| SecurePIN | Personalization API |
| Version | Cardlet Version Information API |
| Exceptions | Description |
| CardStringException | Byte String Exception |

<표 3> Off-Card Toolkit API(ver 0.9b)

| Package Name | |
|---|---|
| com.mcard.javacard.framework | |
| Interfaces | Description |
| Monitor | Information Monitor Interface |
| Classes | Description |
| ASNUtil | ASN.1 API |
| Base64Encoder | Base64 Encoding API |
| CardChecker | Card Insertion Notification API |
| CardletLoader | Secure Cardlet Loader(3.4 참조) |
| CardRemover | Card Removal Notification API |
| FTPListResult | File Transfer Protocol Result |
| FTPTool | File Transfer Protocol API |
| Gauger | Progress Gauger API |
| GaugerStream | Gauger Information Writer API |
| IAAPDUDataSpec | Application Key Spec |
| IAInitParameterSpec | Cardlet Installation Parameter Spec |
| JarFileEntry | JAR(Java ARchive) File Entry API |
| JarUtil | JAR Utility API |
| KeyStoreTool | Certificate & Key Repository API |
| ManifestFile | JAR Manifest File API |
| MonitorStream | Monitor Information Writer API |
| OCFUtil | OpenCard Framework Utility API |
| OPUtil | Open Platform Utility API |
| PackageInstallInfo | Cardlet Installation Information API |
| PackageLoadInfo | Cardlet Loading Information API |
| SignatureBlock | JAR Signature Block API |
| SignatureFile | JAR Signature File API |
| SystemUtil | System Utility API |
| Tracer | Progress Tracer API |
| TracerStream | Tracer Information Writer API |
| Exceptions | Description |
| CardletLoaderException | CardletLoader Exception |
| FTPToolException | FTP Exception |
| SignatureBlockException | Signature Block Exception |

### 3.3 Cardlet 클래스

JavaCard Applet 의 기본 프레임워크를 제공하는

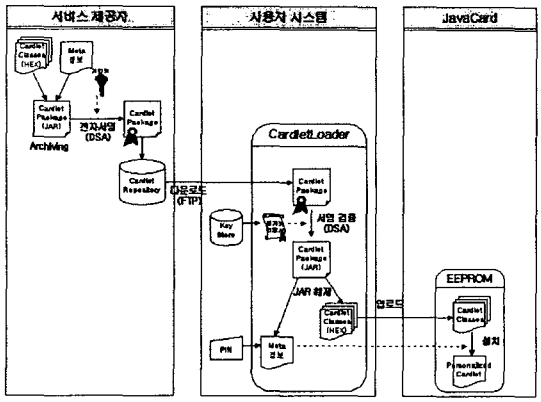
Cardlet 클래스는 javacard.framework.Applet 클래스의 서브클래스로서 <표 4>와 같은 메쏘드를 제공한다.

<표 4> Java Cardlet API(ver 0.9b)

| | |
|---|---|
| public byte | OPT_APP_AUTH, OPT_PIN_VAL, OPT_EXT_AUTH, OPT_ALL(모든 옵션 선택), OPT_NONE(옵션 없음)<br>→ Cardlet instance 보안 설정 옵션 |
| public void | jcInit(byte opt, byte[] instparam, short offset, byte length, Version ver, byte[] atr) throws CardRuntimeException<br>→ Cardlet instance 의 초기화 수행 및 옵션에 따른 각종 보안 객체 설정 |
| public void | jcSelect() → Cardlet 이 select 되는 경우에 수행됨 |
| public void | jcDeselect() → Cardlet 이 deselect 되는 경우에 수행됨 |
| public void | jcProcess(APDU apdu) throws ISOException<br>→ 수신된 APDU(Application Protocol Data Unit)를 처리 |
| public boolean | jcIsPINValidated() → PIN 검증 여부를 반환<br>(OPT_PIN_VAL 이 설정되지 않으면 항상 true 반환) |
| public boolean | jcIsKeyValidated() → Application key 인증 여부를 반환<br>(OPT_APP_AUTH 가 설정되지 않으면 항상 true 반환) |
| public boolean | jcIsOPAuthenticated() → Terminal 인증 여부를 반환<br>(OPT_EXP_AUTH 가 설정되지 않으면 항상 true 반환) |

### 3.4 CardletLoader 클래스

CardletLoader 클래스는 서비스 제공자의 서명이 첨부된 Cardlet 패키지를 인터넷을 통해 다운로드하여 서명을 검증하고 JavaCard 상에 업로드 및 설치를 해주는 Toolkit 의 핵심 모듈이다. <그림 5>는 CardletLoader 클래스의 동작 과정을 보여준다.




<그림 5> CardletLoader 클래스의 동작 과정

### 3.5 Toolkit 응용프로그램

이번 절에서는 JavaCard Toolkit 을 이용하여 개발한 전자지갑(JavaCard Wallet) 응용프로그램의 예를 설명한다. 개발 환경은 아래와 같다.

- PC / OS : Pentium III / Windows NT
- Java 플랫폼 : JDK 1.2.2 및 JavaCard 2.1
- 카드 / 터미널 : GemXpresso 211 / Gemplus GCR410

사용자 시스템의 웹브라우져 상에서 동작하는 서명된 전자지갑 Applet 은 보안 채널(Cipher, MAC)을 통하여 전자지갑 Cardlet 과 통신하여 JavaCard 내에 저장된 잔액의 현재 액수를 얻어오거나 일정 금액을 입금 또는 인출하는 기능을 수행한다. 전자지갑 Cardlet 에 접근하기 위해서 사용자는 우선 Terminal 및 Application (Applet)에 대한 인증을 받아야 하며 PIN 을 입력하여 카드 소유자임을 인증 받아야 한다. 다음의 프로그램은 전자지갑 Cardlet 원시 코드의 주요 부분으로서 JavaCard Toolkit 의 사용예를 보여준다[3, 5].

```
public class Wallet extends Cardlet
{
    private static final byte CLA_WALLET        = (byte)0xFA;
    private static final byte INS_GET_BALANCE                = (byte)0x30;
    private static final byte INS_DEBIT                      = (byte)0x31;
    private static final byte INS_CREDIT        = (byte)0x32;

    protected Wallet(byte[] instparam, short offset, byte length) {
        jcInit(OPT_ALL, instparam, offset, length, null, null);
    }

    public static void install(byte[] instparam, short offset, byte length)
    throws ISOException {
        new Wallet(instparam, offset, length);
    }

    public void jcSelect() { }
    public void jcDeselect() { }

    public void jcProcess(APDU apdu) throws ISOException {
        byte[] apduBuffer = apdu.getBuffer();

        switch (APDUUtil.getCLA(apduBuffer)) {
            case CLA_WALLET:
                switch (APDUUtil.getINS(apduBuffer)) {
                    case INS_GET_BALANCE:   doGetBalance(apdu);    break;
                    case INS_DEBIT:         doDebit(apdu);         break;
                    case INS_CREDIT:                doCredit(apdu);
break;
                    default:
ISOException.throwIt(ISO7816.SW_INS_NOT_SUPPORTED);
                }
                break;
            default: ISOException.throwIt(ISO7816.SW_CLA_NOT_SUPPORTED);
        }
    }

    private void doGetBalance(APDU apdu) throws ISOException {
        if                                              (OPUtil.isBlocked())
ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);

        APDUUtil.sendShort(apdu, (short)5, balance);
    }

    private void doDebit(APDU apdu) throws ISOException {
        if                                              (OPUtil.isBlocked())
ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);

        if (!jcIsPINValidated() || !jcIsKeyValidated() || !jcIsOPAuthenticated())
            ISOException.throwIt(ISO7816.SW_SECURITY_STATUS_NOT_SATIS
FIED);

        // debit routine here ...
    }

    private void doCredit(APDU apdu) throws ISOException {
        if                                              (OPUtil.isBlocked())
ISOException.throwIt(ISO7816.SW_COMMAND_NOT_ALLOWED);
```

## 4. 결론

본 논문에서는 안전한 JavaCard 응용시스템의 개발을 위한 Toolkit 의 설계 및 구현에 대하여 논하였다. 개발된 JavaCard Toolkit 은 전자상거래용 지불시스템 및 전자화폐, PKI 인증서 및 키 관리, 정보가전 시스템 등 다양한 정보 기술 분야에 활용이 가능하다. 향후에는 Cryptoki(PKCS #11) 표준 지원 모듈 및 분산환경을 지원하는 Java RMI 또는 JINI Technology 와의 연동을 고려한 Toolkit API 를 개발할 계획이다.

## 참고문헌

[1] 임영이, 이윤철, 강희일, 이동일, "스마트카드 기술동향", 한국전자통신연구원, 2000
[2] Gary McGraw, Edward W. Falten, "Securing Java", Wiley Computer Publishing, 1999
[3] Zhiqun Chen, "JavaCard Technology for Smart Cards", Addison-Wesley, 2000
[4] SUN Microsystems, "JavaCard 2.1 Specification", 1999
[5] SUN Microsystems, "JavaCard Developer's Guide", 1999
[6] http://www.smartcardbasics.com/overview.html, "Smart Cards and Security Overview"
[7] http://www.smartcardcentral.com/technical, "Smart Card Technical Information"