자바 카드 기술 발전 동향 분석

Trend of Java Card Technologies

김영진(Y.J. Kim) 생체인식기술연구팀 연구원
반성범(S.B. Pan) 생체인식기술연구팀 선임연구원
정용화(Y.W. Chung) 생체인식기술연구팀 책임연구원, 팀장
정교일(K.I. Chung) 정보보호기반연구부 책임연구원, 부장

현재 IC 카드는 통신, 금융, 교통 등의 여러 응용 서비스에서 널리 사용되고 있는데, 계속적인 하드웨어 기술의 발전으로 인한 메모리 증가, CPU 성능 향상과 통합된 정보 가전을 위한 다양한 형태의 네트워크 연결 인터페이스 구축 노력이 다중 응용 프로그램 사용 요구 및 개방형 운영체제(open-platform operating system)와 맞물려 도약적인 기술 발전을 이루고 있다. 본 고에서는 널리 사용되고 있으며 향후 시장성이 가장 클 것으로 예측되는 자바 카드 플랫폼 탑재 IC 카드의 기술 현황을 H/W 및 S/W 측면에서 살펴보고, 자바 카드 기술의 발전 동향을 조망하고자 한다.

I. 서 론

현재 IC 카드는 통신, 금융, 교통, 신분 확인, 전자화폐 등의 여러 응용 서비스에서 널리 사용되고 있으며 점차로 그 용도가 확대되어 금융면에서는 기존의 신용 카드를 대체하고 다양한 형태의 장비를 통하여 네트워크와 연결되어 사용되는 추세로 바뀌고 있다[1]. 특히 유럽의 GSM 폰에서 사용자 인증 카드에 많은 사용처를 보이고 있으며 향후 2005년 이후에는 신용카드의 IC 카드로의 전이로 인해 그 사용이 획기적으로 확대될 것으로 예상된다.

또한, 이와 더불어 빈번한 사용에 따라 개인 정보의 안전한 저장 및 사용, 사용자 인증을 포함한 정보 보안의 문제도 크게 부각되고 있다. 이러한 조류에 발맞추어 IC 카드의 하드웨어도 발전하고 있다. CPU는 기존의 8비트 위주에서 보다 처리 성능이 뛰어난 32비트로 전환되고, 데이터의 암·복호화를 빠르게 하기 위해 특정 암호 기법에 대한 전용보조 프로세서의 사용이 늘고 있다. 또, 다양한 응용 서비스용 프로그램들을 수용하기 위한 저장 공간과 수행 시 필요한 임시 사용 공간으로 ROM, EEPROM 및 RAM의 메모리 크기를 늘이고 있는 실정이다. 소프트웨어 측면에서도 다수의 응용 서비스용 프로그램 수용을 가능케 하고 하위의 하드웨어에 관계없이 응용 프로그램을 작성하고 수행하기 위한 운영 체제를 구축하는 작업이 진행되었다. 또, 카드가 발급된 이후에도 최종 사용자(end-user)가 원하는 응용 프로그램을 다시 카드에 적재하여 수행할 수 있는 발급 후 적재(post-issuance) 개념을 도입하였다.

결과적으로, IC 카드 기술은 다중 응용 프로그램(multi-application)의 사용과 플랫폼에 독립적인 응용 프로그램의 이식에 대한 완전한 지원이 가능한 개방형 운영체제에 대한 사용의 강화를 강화하고 있는 추세이다. 이에 더하여, 일반 정보 가전과의 호환성을 가지는 네트워크 연결 인터페이스 구축을 통해 간편한 네트워크 접속으로 인한 사용자 편의를 추구

하고 있으며 하드웨어 기술의 발전으로 인한 메모리 및 CPU 성능 향상에 따라 IC 카드 기술 발전의 가속화가 이루어지고 있다.

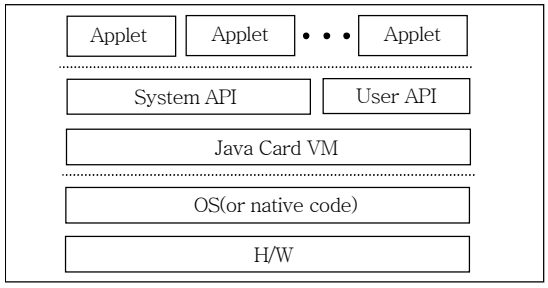
본 고에서는 다중 응용 프로그램용 IC 카드 중에서도 근래에 널리 사용되고 있는 개방형 운영체제의 대표적인 플랫폼인 자바 카드 플랫폼을 탑재한 IC 카드(이하 자바 카드로 지칭함)의 기술 현황을 살펴보고 발전 동향에 대해 기술하고자 한다. 먼저, II장에서는 자바 카드의 개요를 살펴보고, III장에서는 자바 카드 H/W 및 칩 현황을 분석한다. IV장에서는 자바 카드 플랫폼 기술 현황을 살펴보고 V장에서는 자바 카드 기술 발전 동향을 기술한다. 마지막으로 결론을 맺는다.

II. 자바 카드 개요

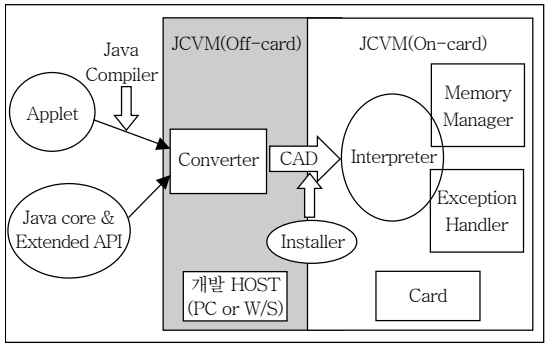
자바 카드는 자바 기술을 IC 카드에 대해 최적화하여 구현하고 있다. 일반적인 IC 카드에 적용되는 모든 표준을 따르는 전형적인 IC 카드인데 하위의 운영체제 위에 존재하는 자바 카드 가상 기계(java card virtual machine)가 자바 카드 애플릿(java card applet)의 바이트 코드(bytecode)를 수행하고 메모리, I/O 같은 카드 내의 모든 자원에 대한 접근을 제어한다는 점에서 차이가 난다.

자바 카드 기술은 플랫폼간에 이진 코드의 이식성(portability) 즉, 상호 운용성(inter-operability)이 뛰어나고 타입 검사 등에 의해 악의적 코드에 대한 보안성(security)을 지닌 자바 언어를 IC 카드의 실시간 환경에 대해 최적화하고 있다. 자바 카드에서의 가상 기계 이용에 의한 장점은 응용 프로그램과 운영체제를 분리하는 개방형(open-platform) 운영체제를 가져오며 하드웨어 의존적인 어셈블리 코드가 아닌 상위 언어인 자바 언어로 쉽게 응용 프로그램을 작성 및 수행할 수 있으며 카드가 최종 사용자에게 발급된 이후에도 필요한 응용 서비스에 따른 응용 프로그램을 IC 카드에 적재할 수 있다는 것이다[2]. 이에 따라 다양한 다수의 응용 프로그램(multi-application)을 수용할 수 있는 유연성(flexibility)을 가지게 한다. 또한, 자바 카드에서는 자바 언어 자체의 보안 특성 이외에 응용 프로그램간의 방화벽을 제공함으로써 엄격한 보안성을 보장한다. 자바 카드 플랫폼은 위에서 기술한 혁신적인 IC 카드 기술을 충분히 안정적으로 제공하고 있으며 자바 언어 자체의 폭 넓은 프로그래밍 자원 및 대중성에 힘입어 1999년에 ETSI(European Telecommunications Standards Institute)에 의해 GSM (Global System for Mobile telecommunication) 휴대폰 내에 SIM(Subscriber Identity Module) 카드의 표준으로 채택되는 등 IC 카드 응용 서비스 시장에서의 점진적인 점유율 증가를 나타내고 있다.

자바 카드는 (그림 1)에서 보는 것처럼, 자바 카드 Application Programming Interface(API)와 가상기계(Virtual Machine: VM)로 이루어진 자바 카드 수행 환경(Java Card Runtime Environment: JCRE), Chip Operating System(COS), H/W 및 응용 프로그램인 애플릿으로 구성된다. (그림 2)는 분리된 가상 기계 구조를 가진 자바 카드에서, 카드 밖



(그림 1) 자바 카드의 구조



(그림 2) 자바 카드 수행 과정

에서 애플릿을 컴파일하고 변환한 뒤, 카드리더를 통해 카드로 적재하여 수행하는 과정을 보인 것이다.

III. 자바 카드 H/W 칩 현황

근래에 칩 공정 기술의 발달로 인한 집적화의 증가로 0.5k~1k바이트 RAM, 16k 바이트 EEPROM, 16k 바이트 ROM을 장착한 자바 카드가 4k 이상의 RAM(최고 8k에 이르고 있음), 64k 이상의 EEPROM, 128k 이상의 ROM을 사용하는 형태로 H/W 성능이 향상되고 있다. 이로 인해 코드 크기가 크고 실시간 수행 메모리가 많이 필요한 라이브러리 및 응용 프로그램 탑재가 용이해지고 있는 실정이다.

CPU는 초기의 8비트 CPU에서 32 비트 CPU로 바뀌고 있으며, 클럭수가 수십 MHz 대에서 100 MHz 이상의 고성능을 보이고 있다. 또한, ARM사의 SC200 시리즈와 같이 자바 가상 기계의 수행 속도 개선을 위한 자바 가속기 하드웨어를 내장하고 있는 형태가 나타나고 있다[3].

주변 모듈들은, 공개키 기반의 메시지 또는 데이터 암·복호화의 성능 개선을 위한 암호 프로세서를 수용하고 있으며, 비밀키 생성을 위한 난수 발생기(Random Number Generator: RNG)를 내장하고 있다. 또한, 메모리의 효율적인 관리를 위한 메모리 관리 장치(Memory Management Unit: MMU)를 장착하고 있다.

<표 1>은 대표적인 IC 카드 업체 또는 IC 카드용 칩 생산업체의 최근 IC 카드 칩 사양을 정리한 것이다[3]-[7].

IV. 자바 카드 플랫폼 기술 현황

자바 카드 플랫폼은 1996년 SUN에 의해 자바 카드 표준 1.0이 발표된 이후 지속적인 버전업(version-up)을 통해 2.1.2가 공표되어 사용되고 있는 상태이며, 2.2 베타 버전이 근래에 발표되었다.

자바 카드 플랫폼은 제한된 자원 및 성능 제한과 ISO 7816 표준 준수로 인해 자바 플랫폼의 일부 내용을 취하면서 IC 카드 환경에 최적화되어 설계, 구현되고 있어 별도의 수행 환경 특징들을 내포하고 있다. 이러한 내용들을 자바 플랫폼에 대해 간략히 정리하면 다음과 같다.

- 자바 카드 언어 특성[8]
  - string 지원 않음
  - char, long, double, float 타입 지원 않음
  - 쓰레드(threads), 다중 차원 배열 지원 않음
  - 패키지의 클래스 수 제한
  - 배열 크기의 제한
  - int의 선택적 사용
  - 가비지 콜렉터(garbage collector) 지원 않음
- 분리된 자바 카드 가상 기계 구조 및 동작
  - pre-loading 및 dynamic allocation
  - converter(off-card) 및 installer(on-card) 사용

<표 1> 업체별 IC 카드 칩 사양

업체(칩)	RAM/ROM/EEPROM	CPU bit/clock	Crypto	MMU	RNG	Java H/W 가속기
Atmel (AT91SC321RC)	5k/96k/64k	32/133MHz	RSA(2048)/3 DES	있음	있음	없음
ARM (SC210)	Totally 4G	32/110MHz	RSA(2048)	있음	있음	있음(Jazelle)
Hitachi (AE46C)	6k/160k/68k	16/-	RSA(1024)/DES	있음	있음	없음
Infineon (SLE88CX720P)	8k/240k/80k	32/66MHz	RSA(1024)/ECC/3 DES	있음	-	부분적 지원
ST (ST22WJ64)	8k/224k/64k	32/-	RSA(-)/ECC/3 DES	있음	있음	있음

• 보안 모델 특성
- 오프 카드 검증기(off-card verificator)
- 카드상의 애플릿간의 방화벽(firewall)
- 데이터 유출 방지를 위한 일시적 객체(transient object) 사용
- EEPROM 데이터 무결성 보장을 위한 트랜잭션(transaction)

• OP 사용
- OP(Open Platform)는 Visa에 의해 개발된, IC 카드상의 다수의 응용 프로그램의 안전한 수행 및 관리 프로그램임
- 암호를 이용한 카드 리더 또는 호스트의 인증을 통해 믿을 수 있는 대상 또는 통신 선로로부터 카드에 변환된 애플릿(Converted Applet: CAP) 파일을 적재, 설치할 수 있게 함
- 카드 외부 또는 카드 내부의 다른 애플릿에서의 해킹 등에 대해 안전하게, 원하는 애플릿의 정보를 관리하게 함

V. 자바 카드 기술 발전 동향

슐럼버저, 젬플러스 등 유수의 자바 카드 관련 업체들로 구성된 자바 카드 포럼(Java Card Forum: JCF)은 자바 카드 플랫폼을 지속적으로 발전하기 위한 노력을 해오고 있다. 그 결과로서 자바 카드 2.2에 대해서는 이전 자바 카드 플랫폼에서는 지원이 되지 않았던 가비지 콜렉터나 네트워크 연계 기능을 지원할 예정이다. 또한, 사용자 인증을 위한 정보로서 개인 식별 번호(Personal Identification Number: PIN)가 아닌 생체 정보를 사용하여 카드 내에서의 정합 및 인증 처리를 할 수 있는 매치 온 카드(match-on-card)용 API 개발을 추진하고 있으며, OP를 보완하여 카드 관리 시스템 라이브러리 개발도 추진하고 있다. 자바 카드 포럼에서 추진하고 있는 기술 개발 상황을 정리하면 다음과 같다.

• Java Card Management System(JCMS)[9]
→ OP의 기능 보완 및 확장
- 기본 카드 내용 관리(생성, 삭제 등) + 카드 감사, 애플릿 공급자에 대한 블랙리스트 제공
- 안전한 발급 후 적재
- 서비스 확장을 위한 응용 프로그램 API 제공
- 메모리 관리 기능 강화(가비지 콜렉션 포함)

• Java Card Remote Method Invocation(JCRMI) 인터페이스[10],[11]
→ 자바 RMI의 부분 집합을 재구성 및 이용
- 호스트상에서 구동되는 클라이언트 응용 프로그램이 카드상의 원격 객체(remote object)의 메소드를 호출할 수 있도록 함
- 정의된 사용자 인터페이스로부터 off-card측의 응용 프로그램 프록시와 on-card측의 카드 애플릿 스켈레톤(skeleton) 생성 후 이용함
- Off-card측에서 카드와의 통신은 Open Card Framework(OCF)가 수행하여 이용하며 프록시는 OCF를 이용함
- 카드 애플릿은 스켈레톤과 내부 통신을 하며, 응용 프로그램 프록시와 애플릿 스켈레톤은 ISO 7816에 규정된, 카드와 카드리더간의 통신 프로토콜 Application Protocol Data Unit (APDU)으로 통신함
- 프록시를 이용하여 Jini와 같은 네트워크 환경 구성 서비스에 이용 가능

• 제한된 가비지 콜렉터[10],[12]
- 애플릿에서 미리 정의된 API의 가비지 콜렉션 메소드를 호출함으로써 기동함
- 가상기계에서는 현재 선택된 애플릿에 의해 소유되어 있으며 참조되지 않는 EEPROM상의 객체를 수집하며 참조되지 않는 일시적 객체를 수집함
- 가비지 콜렉션의 수행은 카드와 카드 리더간의 통신 프로토콜인 APDU 하나가 처리될 동안 이루어져야 함

• 자바 카드 Biometry API[9]
- 개인의 기본 생체 정보를 카드에 등록하고 개인

인증 수행시 카드상의 정합을 가능하도록 함
- 여러 종류의 생체 정보 및 종류별 다수의 기본 생체 정보 이용이 가능하도록 API를 설계함

자바 카드 포럼 외에도 자바 카드 기술을 개선 및 보완하기 위한 노력은 여러 곳에서 계속되고 있는데, 대표적인 것이 온 카드 바이트 코드 검증기와 가상기계 수행 속도의 개선에 관한 것이다.

기존의 바이트 코드 검증기는 카드의 제한된 H/W 성능으로 인해, 인증된 호스트 상에서 애플릿이 작성, 컴파일 및 변환되어 생성된 자바 카드 바이트코드에 대해, 애플릿이 카드에 적재되기 전에 이루어졌다(off-card verifier). 그러나, 현재는 호스트와 무관하게 애플릿을 카드에 적재하면서 검증이 일어나는 온 카드 바이트코드 검증기(on-card verifier)가 구현, 시험되고 있다[13]. 이것은 상위 플랫폼인 J2ME(Java 2 Micro Edition) 이상에서 사용되고 있는 검증 형태를 자바 카드 플랫폼에서 이용하는 것으로, 보다 강화된 사용자 편의적 보안 모델을 도입하고 있으며 향후 고성능의 카드에서 채택할 것으로 예상된다.

가상기계의 속도 개선 노력은 <표 1>에서 보는 바와 같이 자바 카드용 칩 업체에서의 자바 카드 바이트 코드 H/W 수행 엔진 구현 또는 가속기의 장착으로 잘 나타나고 있다. S/W적인 가상 기계 성능 최적화 노력은 [14]에서 보는 바와 같이, 동일한 바이트 코드 집합을 매크로 명령어로 대체하고 가상기계를 수정함으로써, 메모리를 주된 제약으로 하고 수행 시간도 고려하였다. 앞으로는 대용량, 고성능 자바 카드 칩을 겨냥한 가상 기계의 속도 최적화가 우선시 될 것으로 보인다. 또한, 상위 자바 플랫폼에서 이용되는 Just-in-Time(JIT) 컴파일러 또는 이의 변형 결과의 자바 카드 적용도 가능하리라 생각된다.

VI. 결 론

본 고에서는 자바 카드 기술의 현황을 간단히 살펴 보았으며, 카드 H/W의 발전에 따른 자바 카드 플랫폼 기술의 발전 동향을 기술하였다.

자바 카드는 분리된 가상기계 구조에 따라 코드 변환이 많이 이루어지고, 카드의 제한된 자원을 이용하도록 최적화 작업이 이루어지며, 카드 표준인 ISO 7816을 준수하도록 설계 및 구현되고 있다. 하지만, H/W 기술의 발전으로 인해 메모리가 대용량이고 연산 성능이 뛰어난 카드에 대해서 상위 자바 플랫폼의 내용을 수용하는 방향으로 발전하고 있다. JCRMI나 온 카드 검증기 등이 바로 이러한 예이다.

참 고 문 헌

[1] 한국전자통신연구원, 스마트 카드 기술 시장 보고서, 2001. 12.

[2] B. Michael, B. Peter, E. Thomas, H. Frank, O. Marcus, "JavaCard-From Hype to Realty," *IEEE Concurrency*, Oct.-Dec. 1999.

[3] http://www.arm.com.

[4] http://www.atmel.com.

[5] http://www.hitachi.com.

[6] http://www.infineon.com.

[7] http://us.st.com/stonline/index.shtml.

[8] Z. Chen, *Java Card™ Technology for Smart Cards: Architecture and Programmer's Guide*, ADDISON-WESLEY, 2000.

[9] http://www.javacardforum.org.

[10] *Java Card™ 2.2 Virtual Machine Specification*, Sun Microsystems, Inc., Early Access, 2001.

[11] E. Vetillard, "Tools for Integrating the Java Card™ API into Jini™ Connection Technology," javaone conf., 2000.

[12] *Java Card™ 2.2 API Specification*, Sun Microsystems, Inc., Early Access, 2001.

[13] X. Leroy, "On-Card Bytecode Verification for Java Card," E-smart 2001, LNCS 2140, 2001, pp. 150 - 164,

[14] L. Clausen et al., "Java Bytecode Compression for Low-End Embedded Systems," ACM Transactions on Programming Languages and Systems, Vol. 22, No. 3, May 2000, pp. 471 - 489.