제21회 한국정보처리학회 춘계학술발표대회 논문집 제11권 제1호 (2004. 5)

자바카드를 이용한 의료정보 시스템 구현

최재희*, 권경희**
*단국대학교 전자계산학과
e-mail : opgaja@cs.dankook.ac.kr

Implementation of Medical Information System Using Java Card

Jae-Hee Choi*, Kyung-Hee Kwon**
*Dept. of Computer Science, Dan-Kook University

요 약

개인의 신상 정보와 같은 민감한 정보를 다루는 의료기관의 정보시스템을 스마트 카드를 이용하여 보다 안전하고 효율적인 시스템을 설계 및 구현하고자 한다. 현재 빠르게 보급되고 있는 자바 카드 기술을 이용하여 설계되었으며, 기존의 시스템에 사용자와 의료기관사이의 인터페이스를 스마트 카드로 대체함으로써 보안적 측면의 강화 및 사용자의 중심의 편의성을 높이도록 설계하였다.

1. 서론

경제활동의 다양화와 정보통신의 발달로 인해 카드라는 매체는 전자화된 개인 정보의 저장 및 인증의 전달 매체로써 각광받고 있다. 그러나 현재 카드의 상당수를 차지하는 기존의 엠보싱(Embossing)과 자기 띠(Magnetic Strip) 방식의 카드들은 온라인과 오프라인에서 다양화 되는 서비스 및 정보의 처리를 뒷받침하기에 역부족이다. 따라서 이러한 카드는 보안성의 요구, 저장성에 대한 요구 그리고 컴퓨터 시스템 외부 매체로서의 역할에 대한 요구를 받고 있으며 이는 스마트 카드로의 전환을 가속화 시켰다. 더욱이 우리나라에서는 금융감독원의 주도하에 현금카드를 2005 년까지, 신용카드(현금카드겸용 포함)는 2008 년까지 전환 하도록 “IC 카드 도입을 위한 세부 추진방안”을 추진하고 있다.

전통적으로 IC 카드의 개발은 칩 카드 벤더들에 의해 제공되는 운영체제와 시뮬레이터를 통해 개발되어 왔다. 따라서 칩 의존도가 높으며 제 삼자가 어플리케이션을 독립적으로 개발하여 카드 발행자에게 판매하기란 사실상 불가능하다. 이러한 이유로 인해 사용되는 대부분의 스마트 카드 어플리케이션은 철저하게 카드 발행자 위주의 맞춤 형식이 될 수 밖에 없다. 더욱이 이러한 어플리케이션의 상호 운영성의 결핍과 제한된 카드 기능은 스마트 카드 어플리케이션의 개발시간의 소요, 비용의 증가 그리고 사실상 업그레이드의 불가능을 초래한다.

그러나 자바 카드 기술(Java Card Technology)은 하드웨어 독립적인 COS(Chip Operating System)을 제공함으로써 이러한 문제를 극복할 수 있게 하였다. 더욱이 자바카드 기술은 다중 어플리케이션을 지원할 수 있어 카드가 제조된 후에도 부가적인 어플리케이션을 카드에 이식할 수 있고 동일 카드내의 어플리케이션간, 민감한 정보에 대해서 Applet Firewall Mechanism 과 SOI(Sharable Object Interface)를 통해 상호 데이터의 보안과 공유를 제어 할 수 있게 한다.

본 논문에서는 이러한 자바 카드의 특성을 살려 병원 예약, 진료, 수납, 결재, 그리고 처방전에 이르기까지 한 장의 카드를 통해 사용자의 편의를 도모하는 의료정보 시스템을 구현하고자 한다. 또한 현실화 방안으로 신용카드의 제휴카드로 혹은 부가서비스와 같은 보안적 측면이 강한 유형의 상품 개발에 그 초점을 두고 있다.

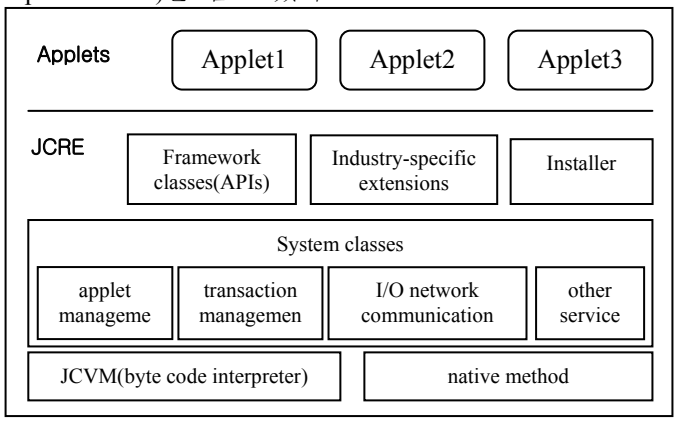
2. 관련기술

2.1 자바 카드 기술(Java Card Technology)

자바 카드 기술은 자바 프로그래밍 언어로 작성된 프로그램이 스마트 카드와 다른 자원 제약적 디바이스 상에서 실행하게 하는 기술이다. 자바 카드는 상호 운영성(interoperability)를 제공하여 COS 와 상관 없이 개발자가 다른 실리콘 구조와 운영체제에서 실행할

수 있도록 어플리케이션을 개발할 수 있는 특징을 가진다. 이는 벤더가 새로운 어플리케이션을 개발 시, 카드 내의 다른 어플리케이션의 변경 혹은 영향 없이 간단히 다운로드를 받기만 하면 된다. 현재 Java Card 2.2.1 환경을 지원하고 있다.

자바카드는 JCVM(Java Card Virtual Machine), JCRE(Java Card Runtime Environment), 그리고 APIs(Application Programming Interfaces)로 구성된다. JCVM 은 자원 제약적 특성으로 인해 카드 소프트웨어를 플랫폼 독립적인 분산 시스템으로 구성시킨다. 카드 상에서 실행되는 Java Card Byte Code Interpreter 를 가지는 on-card VM 측과 class 파일을 압축한 형태인 CAP 파일로 전환하는 converter 를 가지는 off-card VM 으로 구성된다 CAP 파일은 자바 패키지에서 실행할 수 있는 클래스들의 바이너리 표현(executable binary representation)을 담고 있다.



[그림 1] On-card system architecture

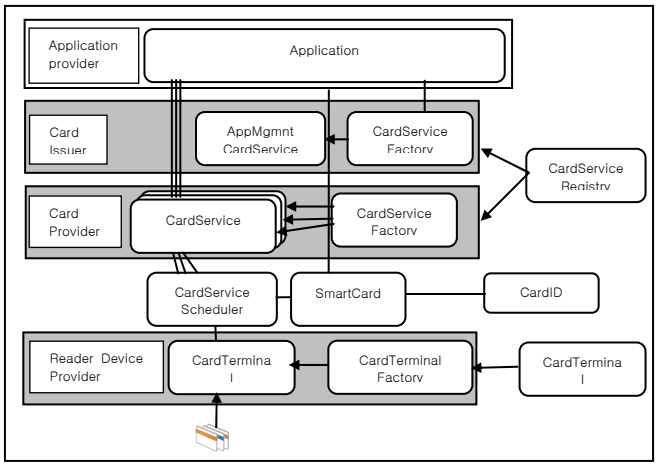
JCRE 는 스마트 카드 상에서 실행되는 자바 카드 시스템 컴포넌트들로 구성되며, 가장 큰 역할은 카드 자원관리, 네트워크 통신, 애플릿(applet) 실행 그리고 애플릿 보안에 관여한다. 특히 JCRE 는 애플릿 인스턴스가 생성될 때, context 라 불리 우는 보호된 객체 공간으로 방화벽을 형성하여 카드 내의 애플릿간의 보안을 위한 중요한 역할을 한다.

자바 카드 APIs 는 ISO/IEC 7816 모델에 따라 스마트 카드 어플리케이션을 프로그래밍 하기 위한 클래스의 집합으로 구성된다.

2.2 OCF(Open Card Framework)

스마트 카드 시스템은 카드 측 소프트웨어와 호스트 측 소프트웨어를 통합함으로써 개발이 된다. 본 논문에서 호스트 측 어플리케이션은 OCF 를 사용한다. OCF 는 카드 운영체제, 터미널, 개발자간의 통합된 인터페이스를 제공할 필요성 따라 1997 년 OCF 컨소시엄에서 제정한 것으로, 카드 터미널 상에서 실행되는 어플리케이션에 대한 API 를 제공하는 객체지향 소프트웨어 프레임워크이다. OCF 는 ISO/IEC 7816, PC/SC, EMV 등과 같은 표준 규격을 참조하여 설계되었다.

[그림 2]에서처럼 OCF 는 크게 CardTerminal, CardService, ApplicationManagement 레이어로 크게 구분



[그림 2] Open Card Architecture

이 된다. CardTerminal 은 물리적인 카드 터미널과 스마트 카드에 대한 접근을 제공한다. CardService 레이어는 스마트 카드의 운영에 대한 다양한 기능을 API 로 구현하고 그 기능과 APDU 통신을 담당한다. ApplicationManagement 는 멀티 어플리케이션 환경하에서 카드에 탑재된 애플릿들에 대한 선택, 해제, 설치 삭제, 갱신 등과 같은 역할을 한다.

3. 자바카드를 이용한 병원 정보 시스템 설계

본 논문은 APDU(Application protocol data units) 통신을 기반으로 한 병원정보 시스템을 구현한다. 어플리케이션의 구성은 가장 크게 off-card 와 on-card 로 구분된다. 그리고 카드 사용자의 이동 경로 즉 카드의 발급, 접수, 진료, 수납, 처방전 발행 그리고 약 조제까지 각 단계별로 수행 목적에 따라 데이터의 저장과 보안이 다른 시스템으로 이루어진다. 이 시스템의 가장 큰 특징은 기존의 종이 문서대신 대신 카드를 사용하는 점이며, 특히 병원과 불특정 약국간에 네트워크 공유 없이 독립화된 정보를 자바 카드를 통해 전달 하며, 이러한 과정에서 보안적 문제와 결재 문제까지 자동화 하고자 한다. 따라서 기존의 병원 시스템과 제안한 시스템을 비교하여 보다 나은 시스템의 설계를 목적으로 한다.

3.1 기존 시스템과의 비교.

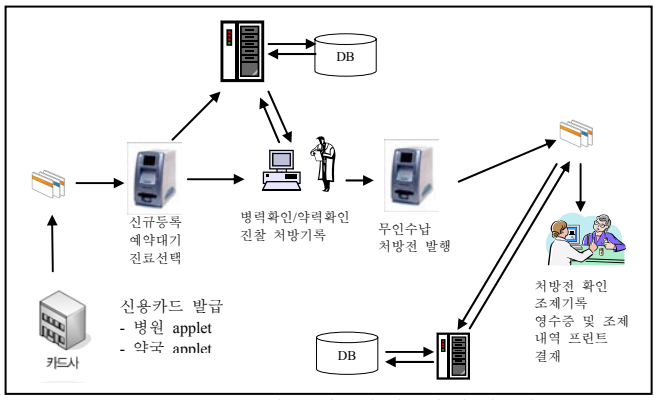
기존의 시스템은 병원, 약국 그리고 결재 부분이 연계 없는 독립적 시스템으로 데이터를 처리하고 있다. 결재라는 부분을 제외하고 환자에 대한 기록은 각 의료기관 DB 서버에 기록되고 전자화 된 데이터의 연속성이 없는 종이문서를 사용하여 약국으로 이동하게 된다. 이때 병원의 DB 서버에 접속하기 위한 방법으로 마그네틱 카드를 통하거나 아니면 엠보싱 된 시리얼 번호를 입력해 사용하게 된다.

또한 종이 처방전은 환자를 통해 불특정 약국으로 이동하게 되는데 종이라는 물리적 한계성으로 환자의 병력, 부작용 약, 현재 복용 약에 대한 정보는 사실상 이동 될 수 없고, 의료기관의 DB 에 저장되며, 처방될 약품과 복용량만을 전달한다. 이는 의료행위의 참가자

인 환자에게 제한된 정보만을 제공하는 것이며 환자의 알 권리 및 환자의 건강 상태를 위협하는 것이다. 이런 정보의 은폐성은 의료 정보를 잘 알지 못하는 환자 구두에 의존하게 되며, 또한 종이 문서의 특성상 훼손 및 분실을 통해 각 의료기관의 전달과정에서 인증이 보장되지 않음으로 인해 미연에 생길 수 있는 약학 사고에 대한 명백한 책임 소지도 미비할 수 밖에 없으며 종이 문서 위조를 통한 의료기관의 보험금의 허위 청구를 막을 수 있다.

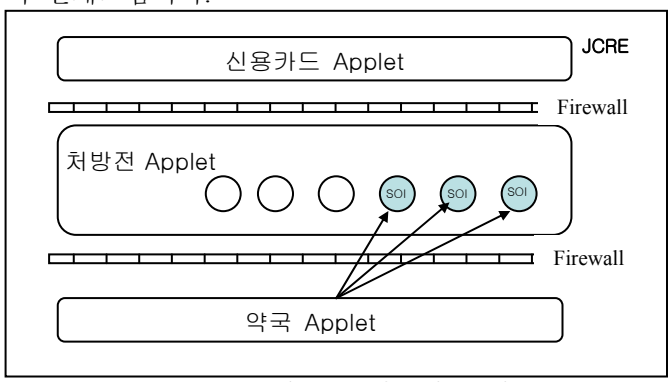
또한 병원에서는 접수 및 수납 그리고 처방전에 발행 과정에서의 혼잡성과 진료카드의 발급에 대한 비용의 부담을 가지고 있으며, 약국 측에서도 또한 종이 처방전의 대한 재입력전산화에 대한 비용과 시간이 걸린다.

본 논문에서 제안한 시스템은 불연속적인 데이터 이동 문제를 사용자 측면에서 자바 카드로 묶음으로써 위와 같은 문제들을 해결하고자 한다.



[그림 3] Java Card 를 이용한 병원 처방전 시스템

제안된 시스템의 구성은 [그림 3]과 같이 하였다. 우선 카드 측 어플리케이션은 개인의 신상 정보에 대한 보안을 유지하기 위해 카드에 접근하는 각 단계마다 접근 할 수 있는 권한을 다르게 주도록 하였다. 이를 위해 JCRE 에 의해 관리되는 자바 카드 내의 context 라 불리는 애플릿간의 방화벽 정책과 이러한 context 를 통한 객체 공유로 카드상의 서로 다른 서비스를 하는 애플릿 객체에 대한 보안을 유지한다. [그림 4]는 이러한 보안 정책을 위한 카드 내의 애플릿의 설계그림이다.



[그림 4] 카드 애플릿의 구성

각 기능을 요약하면 다음과 같다. 첫 번째로 신용카드 애플릿은 전적으로 독립적으로 운영되어야 한다. 병원이나 약국에서의 지불 수단으로써만 쓰이기 때문에 다른 카드 관리를 담당하는 JCRE 를 제외한 어떠한 객체도 접근될 수 없다. 두 번째, 처방전 애플릿은 병원에서 사용되어야 하며 이 애플릿은 처방전 관련 정보, 병원 DB 에 연결하기 위한 개인 정보, 카드 사용자의 병력, 부작용 등 환자의 응급 시 사용할 수 있는 정보를 가진다. 그렇게 하기 위해 이러한 자료는 전역 변수 혹은 공유 인터페이스(SOI)를 사용하여 설정한다.

사용자의 신용카드에 진료카드 및 처방카드를 발급받음으로써 접수 및 수납 그리고 처방전 발급을 자동화 함으로써 사용자의 대기시간을 줄이고 혼잡을 예방함과 동시에 비용을 감소 시킨다.

마지막으로 약국 애플릿은 병원에서 처방한 지정된 자료에 대한 접근 권한을 갖고 의사가 처방한 결과에 대한 약사의 서명을 저장한다.

3.2 제안한 시스템의 흐름도

신용카드사는 자바 카드를 내장한 신용카드를 발급하며 병원을 위한 처방전 애플릿과 약국을 위한 애플릿을 내장하여 사용자에게 발급한다. 또한 신용카드사는 제휴된 각 병원에게 애플릿에 접근하기 위한 고유의 AID(Application Identification)를 할당하며 약국을 위해 서도 AID 를 발행한다. 발급받은 사용자는 제휴된 병원에 가서 진료카드 및 처방전 카드를 활성화하고 이때 개인 정보 및 비밀번호인 PIN(Personal Identification Number) 을 입력한다.

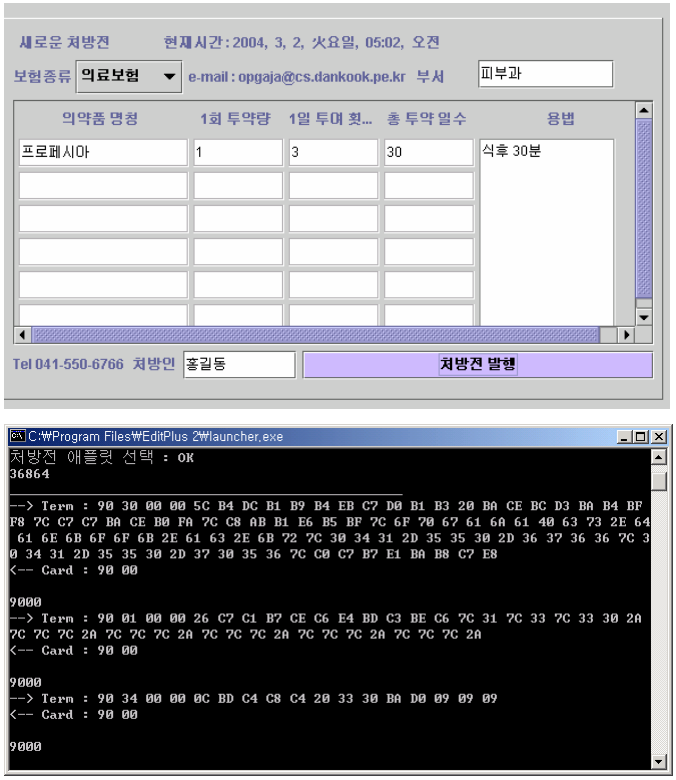
환자는 병원에 방문하여 무인 수납기를 통해 병원 서버에 접속하여 진료의사와 대기 시간을 선택하고 자신의 차례를 기다리어 의사로부터 진료를 받는다. 의사는 터미널을 통해 병원 자체의 서버에 환자의 처방 내역을 입력한다. 이때 환자가 원한다면 다른 의사로부터 데이터를 격리 시키기 위해 자신이 처방한 내역에 PIN 을 활성화 시키도록 입력한다. 이렇게 함으로써 같은 진료기관 혹은 약국에서 환자가 밝히길 꺼려하는 정보에 대해 제약을 둘 수 있기 있다. 그렇기 때문에 이 정보에 대한 접근은 항상 접근하려는 누구이던 간에 환자의 동의 없이 접근이 불가능 하다. 진료가 끝난 환자는 결재 시 무인 수납기 혹은 수납 창고에서 신용카드를 통한 결제를 하고 병원 서버로부터 단말기를 통해 처방전을 발급 받는다. 이때 병원에서 발급한 처방 내용은 SOI(Sharable Interface Object)를 통해 카드 상의 약국을 위한 애플릿에서 접근 할 수 있는 상태이다.

약국을 방문하여 처방전이 든 카드를 약사에게 제출한다. 이때 약사는 의사가 처방한 처방 내용이 PIN 에 의해 묶여 있다면 환자에게 PIN 번호를 요구할 수 있다. 처방전 카드로부터 출력된 내용은 자동으로 약국 시스템의 서버에 저장된다. 조제 후 그 결과에 대한 서명 카드에 입력하게 되는데 약사 법에 따라 처방 내역이 의사의 동의 하에 변경이 되었을 경우 그 변경된 내용을 같이 입력하게 된다. 환자는 신용카드로 결제를 처리한다.

4. 구현

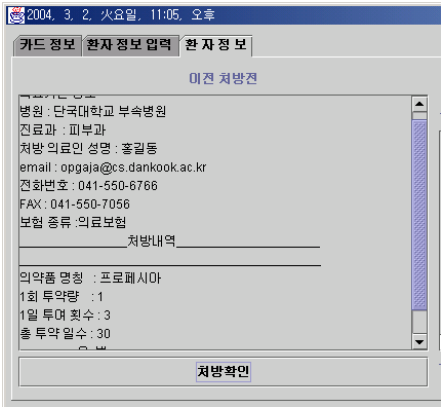
본 논문의 시뮬레이션을 위한 카드와 CAD(Card Accept Device)는 각각 GemXpresso 211PK_IS 와 GCR410P 를 사용하였으며 카드로의 업로드를 위해 GemXpresso Rad III 를 사용하였다. 애플릿 프로그래밍을 위해 Java Card Platform 2.2.1 API 명세서를 따랐고, 호스트 측의 프로그래밍은 OCF 1.2 와 JDK1.3.1 을 사용하였다.

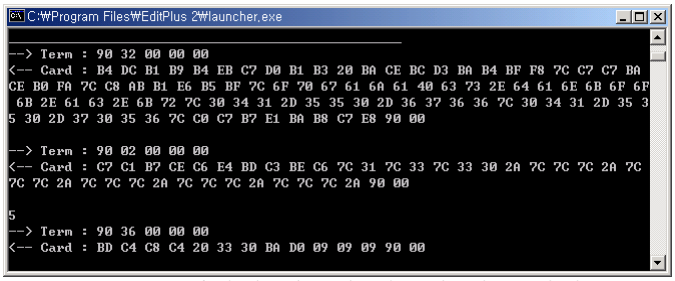
병원에 제공해야 할 AID 는 임으로 "A0 00 00 00 18 FF 00 00 00 00 00 00 00 00 01 02"로 설정하였으며, 이 AID 를 통해 카드상의 애플릿 인스턴스에 대한 접근을 하게 된다. 또한 사용자가 수납, 진료, 처방전 발행 등과 같은 절차에 있어서 카드와 CAD 의 세션간에 항상 보안 인증을 받도록 하였다. 인증은 세션 키를 이용하였고 필요에 따라 사용자가 원치 않는 대상이 데이터에 대한 접근을 막기 위해 PIN 을 사용하였다.



[그림 5] 처방전 입력과 전송과정

본 논문의 유효성을 증명하기 위한 예로, [그림 5]는 처방전 발행을 위한 어플리케이션과 카드 애플릿 사이의 command APDU 와 response APDU 에 대한 byte 배열 통신 내용을 보여주고 있다. 병원에서 이렇게 발행된 처방전은 사용자를 통해 약국으로 이동하여 그 내용을 확인하는 과정 혹은 다른 의사에게 이전의 약력을 보여주는 그림은 [그림 6]과 같다.





[그림 6] 처방전 내용의 획득과 전송 과정

response APDU 는 처리 결과에 대한 상태 워드(status word)나 결과 바이트 배열을 반환하게 된다.

5. 결론 및 향후 과제

본 논문은 개인과 의료기관, 혹은 의료기관과 의료기관 사이의 비효율적인 데이터 형태, 이동 시간 그리고 방법을 자바 카드를 통해 온라인화 시킴으로써 효율적인 병원 정보화 시스템을 구성할 수 있도록 제안하였다. 제시된 시스템은 기존 의료기관의 전산화 시스템에서 단지 사용자 이동에 따른 전산화된 보안적 데이터의 사용에 중심을 두었기 기존의 시스템의 큰 변동 없이 사용자의 편리성과 보안, 특히 전자화된 문서로 인해 업무상의 자동화를 통하여 경제적인 이득을 취할 수 있을 것이다.

향후 연구 과제로는 자바 카드를 이용하여 인터넷을 통한 보안적인 예약 시스템을 구현하여 보다 다양한 시스템으로 확장 가능하도록 하겠다.

참고문헌

[1] Zhiqun Chen, "Java Card Technology for Smart Cards : Architecture and programmer's Guide Foreword by Patrice Peyret", Addison Wesley, 2000.
[2] Vesna Hassler, Martin Manninger, Mikhail Gordeev and Christoph Muller, "Java Card for E-Payment Application", Artech house, 2001.
[3] Uwe Hansmann, Martin S. Nicklous, Thomas Schack, Achim Schneider, Frank Seliger, "Smart Card Application Development Using Java", Sec. Ed. Springer, 2002.
[4] http://java.sun.com
[5] http://www.opencard.org
[6] http://www.javaworld.com
[7] http://www.globalplatform.org