멀티 애플리케이션 스마트카드를 위한 애플릿 추천 시스템

An Applet Recommendation System for Multi-Application Smart Cards

은나래*, 조동섭**
Narae Eun, Dong-sub Cho

Abstract - As multi-application smart cards have become very attractive mobile devices, card users are able to add and to remove card-applets after card issuance. However, because of constrained memory on a smart card, it is necessary to manage card-resident applets. In this paper, we propose an adaptive applet management algorithm in order to recommend card-resident applets which can be removed. This algorithm's goal is to select card-resident applets in a way minimizes the number of applet downloads. To serve this purpose, our algorithm identifies the applets that are most likely to be executed again, and based on that, decides which should be kept in the memory and which can be discarded.

Key Words : Multi-application smart card, Recommendation, Applet management algorithm

1. 서론

스마트카드가 유비쿼터스 환경에서 빠르게 진화하고, 기술적인 발전을 하고 있다. 안전하고, 비교적 쉽게 인증할 수 있는 장점을 가지고 있는 스마트카드가 많이 사용되면서 생활을 편리하게 도와주고 있다. 기존 스마트카드에서는 한 개의 카드에 한 가지 서비스만 처리할 수 있었는데 최근에 스마트카드에 들어가는 메모리 공간이 늘어가면서 여러 개의 애플리케이션을 담을 수 있게 되었다. 멀티 애플리케이션 스마트카드는 카드 사용자에게 카드가 전달되기 이전에 설치되어 있는 프로그램뿐만 아니라, 카드가 사용자에게 전달된 후에도 사용자가 원하는 프로그램을 인터넷을 통한 다운로드를 통하여 카드에 저장할 수 있기 때문에 원하는 서비스를 받을 수 있다. 또한 카드에 저장되어 있는 프로그램 중 더 이상 필요하지 않거나, 서비스 기간이 끝난 프로그램은 제거할 수 있게 되었다.

멀티애플리케이션 스마트카드에 여러 프로그램이 저장될 수 있게 됨에 따라서 스마트카드에 저장되는 애플리케이션에 대한 관리가 필요하게 되었다. 스마트카드 메모리는 ROM (Read Only Memory), RAM (Random Access Memory), EEPROM (Electrically Erasable Programmable Read Only Memory)으로 구성되어 있다. ROM에는 스마트카드의 운영체제, 가상 기계, 기본 프로그램이 내장되어 있다. RAM, EEPROM 모두 읽고 쓸 수 있지만, RAM에 저장되는 정보는 전원이 나가면 내용이 모두 사라져 버리기 때문에 임시로 저장하는 공간으로 쓰인다. 그래서 카드가 사용자에게 전달된 후에 다운로드하는 애플릿은 EEPROM에 저장이 된다.

그러나 EEPROM에 쓰기 연산 속도는 RAM보다 30배나 느리고, 쓰기 연산 횟수도 제한되어 있다 [1]. 그렇기 때문에 EEPROM에 저장되는 애플릿의 설치를 최소화해야한다. 그래서 본 연구에서는 모바일 기기에서 메모리관리를 효율적으로 하기 위해 애플릿 제거 추천 알고리즘을 구현한 스마트카드 애플릿 추천 시스템을 제안한다. 2장에서는 관련 연구로 스마트카드 관리 시스템에 대하여 알아보고, 3장에서는 본 연구에서 제안하는 애플릿 제거 추천 알고리즘을 사용한 애플릿 추천 시스템에 대해 설명한다. 4장에서는 알고리즘을 구현하여 비교분석하였고, 마지막 5장에서는 결론을 기술한다.

2. 관련 연구

스마트카드 관리 시스템의 가장 중요한 목적은 카드 수명주기(life-cycle)와 애플리케이션 수명 주기를 관리하는 것이다. 특히 멀티애플리케이션 스마트카드 관리 시스템은 그 외에도 서로 다른 표준, 플랫폼, 데이터베이스, 인터페이스를 가진 카드에 대해서도 다루어야 하기 때문에 복잡해진다. 멀티 애플리케이션 스마트카드 관리 시스템은 카드 타입, 카드 버전, 애플리케이션 타입, 애플리케이션 버전이 다양하므로 이에 대해 총괄적으로 관리해야 한다. [그림 1]은 스마트카드 관리 시스템의 전체적인 구성을 나타낸 것이다.

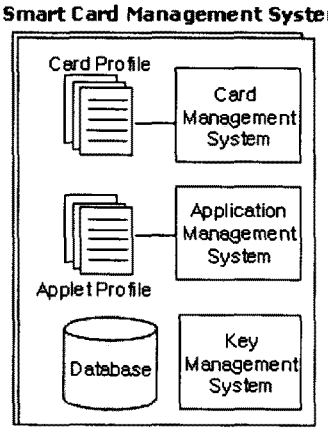
스마트카드 관리 시스템은 크게 키 관리 시스템, 카드 관리 시스템, 애플리케이션 관리 시스템 세 가지로 나눌 수 있다 [2].

카드 관리 시스템은 멀티 환경에서 멀티 애플리케이션 스마트카드를 관리한다. 카드 관리 시스템에서 가장 중요한 기능은 각각의 카드의 상태, 즉 카드의 수명주기를 기록하고 카드 프로파일에 이런 정보들을 저장하여 관리한다.

저자 소개
*이화여자대학교 컴퓨터학과 석사과정
**이화여자대학교 컴퓨터학과 교수

키 관리 시스템은 스마트카드의 암호를 관리하고, 키의 생성, 키의 안전한 분배, 보안정책에 관한 책임을 가지고 있다. 키는 스마트카드 내에서만 사용되는 것이 아니고, 스마트카드와 관련된 처리가 이루어지는 모든 시스템에서 필요하다.

애플리케이션 관리 시스템은 멀티 애플리케이션 스마트카드의 애플리케이션을 관리하는 기능을 가지고 있다. 그리고 스마트카드 애플리케이션 정보를 가지고 있는 애플리케이션 프로파일을 관리한다. 카드 발급 후 인터넷으로 연결된 터미널에서 카드 사용자가 원하는 애플리케이션을 선택하여 다운로드하고, 더 이상 필요하지 않은 애플리케이션은 제거할 수 있도록 해주는 기능을 가지고 있다.



[그림 1] 스마트카드 관리 시스템

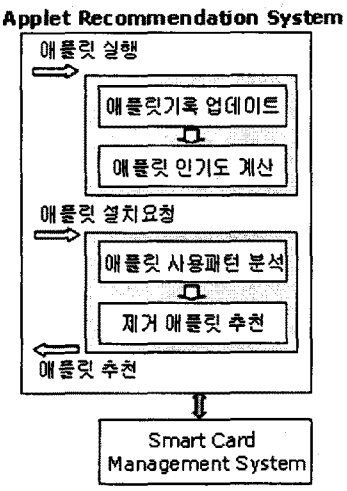
애플리케이션의 기본적인 수명주기는 애플리케이션이 특정 카드 안에 저장되어 있지 않은 초기상태, 카드에 애플리케이션이 로딩되어 있는 로딩 상태, 카드에 설치가 되어있는 설치상태, 애플리케이션이 카드 사용자 각각의 데이터로 개인화된 개인화상태, 사용자에게 전달되어 애플리케이션을 사용하는 사용상태, 더 이상 애플리케이션을 사용할 수 없는 상태 순으로 진행된다.

3 장. 애플릿 추천 시스템

본 연구에서는 모바일 기기에서 메모리관리를 효율적으로 하기 위해 제안된 코드 콜렉션 알고리즘[3]을 기반으로 스마트카드에 저장되어 있는 애플릿 제거 추천 알고리즘을 구현한 스마트카드 애플릿 추천 시스템을 제안한다. 제안하는 시스템의 전체적인 구성은 [그림 2]와 같다.

애플릿 제거 추천 알고리즘의 목표는 애플릿의 제거, 설치 용량을 최소화하는 것이다. 제거, 설치를 최소화하게 되면 애플릿을 다운로드해야 하는 크기도 작아지기 때문에 다운로드 시간도 줄일 수 있다. 이를 위하여 애플릿 추천 알고리즘은 미래에 다시 실행될 가능성이 있는 애플릿은 그대로 두고, 미래에 실행될 가능성이 가장 낮은 애플릿을 추천하는 것이다.

애플릿 제거 추천 알고리즘은 실행단계, 추천단계 두 단계로 나누어진다.



[그림 2] 애플릿 추천 시스템의 구성

3.1. 실행 단계 (Execution Phase)

이 단계에서는 카드 내에 애플릿이 수행되는 단계이다. 이 때 애플릿 추천 시스템은 수행되는 애플릿의 기록을 업데이트하고, 이 애플릿의 PI(Popularity Index)를 계산하여 데이터베이스에 저장한다. PI는 애플릿의 인기도와 관련이 있다. 애플릿의 인기도는 애플릿이 미래에 다시 실행되는 가능성을 나타내어 준다. 애플릿이 실행될 때마다 애플릿의 인기도는 다음 식에 따라 계산되어 저장된다.

$$PI = PI + Increment + InheritedPI$$

여기서 Increment는 애플릿이 실행되는 수를 나타내는 상수이다. 즉 애플릿이 실행될 때 마다 더해지는 값을 의미한다. 그리고 Inherited PI는 라이브러리일 경우에 더해진다. 보통 라이브러리는 카드가 카드 사용자에게 전달되기 이전에 ROM에 저장되어 있다. 그러나 라이브러리가 추후에 만들어지거나, 어떤 특정 애플릿을 수행할 때 필요하거나, 애플릿 크기를 줄이기 위하여 라이브러리로 만들어졌을 경우 애플릿처럼 라이브러리도 다운로드 받을 수 있다. 라이브러리를 통해 애플릿 간 공유도 이루어지기 때문에 효율적으로 메모리를 사용할 수 있다. 애플릿이 수행되면서 필요한 라이브러리가 실행되면서 라이브러리의 인기도에 라이브러리를 호출한 애플릿의 인기도를 더해줌으로써 라이브러리의 인기도를 높여준다. 이는 자주 사용되는 애플릿이 자주 사용되지 않는 라이브러리를 호출했을 경우 라이브러리의 인기도가 올라갈 수 있다. Inherited PI는 다음 식으로 구한다.

$$InheritdPI = (appletPI - libraryPI) \times \sigma(appletPI - libraryPI)$$

$\sigma(x)$는 x값이 0보다 작을 때는 0을 나타내고, x값이 0보다 크거나 같을 때는 1을 나타낸다. 이것은 애플릿 PI가 라이

브러리 PI보다 높을 때에만 Inherited PI를 사용하기 위한 함수이다.

3.2. 추천 단계 (Recommendation Phase)

이 단계에는 카드 내에 제거될 애플릿을 추천하는 단계이다. 애플릿을 요청했을 때 카드의 용량이 충분하다면 애플릿을 다운로드 받아 설치할 수 있다. 그러나 원하는 애플릿을 다운로드하고자 하는데 원하는 애플릿의 크기가 카드에 남아있는 용량보다 클 때 실행되는 단계이다. 이 때 카드에 저장되어 있는 각각의 애플릿의 인기도를 읽어와 여러 변수들을 이용해서 계산하여 미래에 수행될 가능성이 가장 낮은 애플릿을 추천한다. 이 때 계산되는 값이 RI(Recommendation Index)이고 RI값이 가장 적은 애플릿이 제거될 애플릿으로 추천된다. RI에는 PI와 애플릿의 크기 변수와 유효기간 변수를 추가하여 계산한다. RI는 다음 식으로 계산 된다.

$$RI = PI + size\_factor + expiraton\_factor$$

size_factor는 카드에 있는 애플릿 중에서 각각의 애플릿이 얼마만큼 차지하는지 나타낸다. 그리고 expiration_factor는 유효기간이 얼마나 남았는지를 나타낸다. 스마트카드 애플릿에서는 서비스 받을 수 있는 유효기간이 중요하다. 특히 인증할 때 많이 사용되기 때문에 유효기간에 대한 변수를 RI를 계산하는 변수로 추가하였다. 유효기간 변수는 현재 시간에서 유효기간이 얼마나 남아있는지에 대한 값으로 정하였다. 다음은 size_factor를 구하는 식이다.

$$size\_factor = card\_applet\_size / applet\_size$$

애플릿 사이즈가 클수록 RI값은 작아지고, 유효기간이 많이 남아있을 수록 RI값은 커진다. 인기도 값이 크면 클수록 RI값도 커진다. 카드 안에 저장되어 있는 각각의 애플릿의 RI를 계산하여 가장 작은 RI값을 가지는 애플릿을 제거하도록 추천해준다.

4. 구현 및 성능평가

3장에서 제안한 애플릿 제거 추천 알고리즘을 구현하였다. 언어는 자바를 사용하였고, 데이터베이스는 MS SQL 2000 Server를 이용하였다. 입력 파일은 카드와 수행되는 애플릿으로 구성하였고, 카드의 개수는 10개, 애플릿의 개수는 20개로 지정하여 2000번 수행되는 로그를 랜덤으로 생성하여 만들었다.

성능평가는 애플릿 제거 추천 알고리즘의 기반이 된 코드 콜렉션 알고리즘과 비교하였다. 애플릿 설치 용량, 제거 용량, 유효기간이 지난 애플릿의 요청 용량에 대해 비교하였다. 여기서 카드의 용량은 EEPROM의 크기로 하였고, 애플릿 프로그램의 크기를 카드에서 차지하고 있는 애플릿 크기로 가정하였다.

	애플릿 추천 알고리즘	코드 콜렉션 알고리즘	
애플릿 설치	4411 KB	4418 KB	-7 KB
애플릿 제거	4287 KB	4308 KB	-21 KB
유효기간 지난 애플릿	1268 KB	1427 KB	-159 KB
유효기간 지난 애플릿 요청수	449 번	544 번	-95 번

기존 코드 콜렉션 알고리즘을 사용했을 때보다 유효기간 지난 애플릿의 요청 횟수가 많이 줄어들었고, 이에 따라 유효기간을 지난 애플릿을 제거하고 추가하는 별도의 시간이 줄어들었다. 그리고 애플릿을 제거하고 설치하는 용량의 크기도 조금 줄었음을 볼 수 있다.

5. 결 론

본 논문에서 스마트카드에서 메모리 관리와 애플릿 관리를 효율적으로 하기 위해 애플릿 제거 추천 알고리즘을 이용한 스마트카드 애플릿 추천 시스템을 제안하였다. 애플릿 제거 추천 알고리즘을 실행 단계, 추천 단계 두 단계로 나누어서 구성하였다. 실행 단계를 통하여 수행되는 애플릿의 PI를 계산하였고, 애플릿의 다운로드 요청 시 카드의 용량이 부족할 때 카드에 저장되어 있는 각각의 애플릿의 RI값을 구하여 가장 적은 값을 가지는 애플릿을 미래에 실행 가능성이 가장 낮게 보고 이를 제거하도록 추천하였다.

향후 연구로는 교체 알고리즘으로 사용되는 기본적인 캐싱 알고리즘과 비교분석하고, 또한 애플릿 제거 추천 알고리즘을 구하는데 사용한 식에서 상수를 두어서 상수 간 변화에 따라 결과가 어떻게 변하는지에 대한 연구가 계속 진행되어야 할 것이다.

참 고 문 헌

[1] Marcus Oestreicher, Ksheerabadhi Krishna. "Object Lifetimes in Java Card", USENIX Workshop on Smart Card Technology. May 1999.

[2] Uwe Hansmann, Martin S. Nicklous, Thomas Schack, Achim Schneider, Frank Seliger, "Smart Card Application Development Using Java"

[3] Lucian Popa, Costin Raiciu, Radu Teodorescu, "Using Code Collection to Support Large Applications on Mobile Devices", Mobicom'04 ACM. Oct, 2004.