SDR을 위한 SCA에서의 소프트웨어 컴포넌트의 실시간 교체 방법

김민준[O]* 류상률** 김승호*
*경북대학교 컴퓨터공학과 **청운대학교 컴퓨터과학과
mjkim[O]@mmlab.knu.ac.kr, rsr@cwunet.ac.kr, shkim@knu.ac.kr

Real-time Swapping Method of Software Components in SCA for SDR

M.J Kim[0]* S.R Ryu** S.H Kim*
*Dept. of Computer Engineering, Kyungpook National University
**Dept. of Computer Science, Chungwoon University

요 약

SDR (Software Defined Radio) 기술은 세계적인 공통 표준이 없는 이동통신 시스템을 하나의 단말을 이용하여 소프트웨어적으로 접근 가능하도록 만들어주는 것이다. SDR에서는 소프트웨어 프레임워크의 표준으로 JTRS JPO에서 정의한 SCA (Software Communications Architecture)를 채택하였다. 본 논문에서는 SCA의 구조에 대해서 간단히 살펴보고 소프트웨어 컴포넌트의 실시간 교체 방법이 필요한 부분임을 프로그램 시뮬레이션을 통해 제시한다.

1. 서 론

현재의 개인 이동통신 시스템은 세계적으로 다양한 표준이 존재하고 각 지역마다 사용하는 무선 프로토콜과 대역이 달라 국제적인 로밍(roaming)이 완전하게 이루어지고 있지 않다. 게다가 국제적인 표준을 제정하고자 했던 IMT-2000 시스템도 유럽 주도의 비동기식 광대역 CDMA 방식인 3GPP 시스템과 북미에서 주도하는 동기식 CDMA 방식인 3GPP2 시스템으로 나뉘어 표준화가 진행되고 있어 사실적인 국제 표준은 구현되기 힘든게 현실이다. 또한 통신 기술의 빠른 진보와 사용자들의 서비스 욕구로 인한 단말이나 기지국의 업그레이드가 자주 필요하며 그 때 마다 하드웨어 장비를 고치거나 사용자에게 불편을 초래하지 않도록 하기 위한 기술이 요구된다.

SDR 기술[1]은 무선 이동 통신 시스템에서 안테나 이후의 RF 영역을 포함한 대부분의 기능 블록이 프로그래밍이 가능한 고속의 처리 소자에 구현된 소프트웨어 모듈에 의해 수행됨으로써 하드웨어의 교체 없이 필요한 소프트웨어의 재구성만으로 다중 무선접속 규격 또는 서비스 기능 등을 지원한다. 이러한 기능을 제공하기 위한 소프트웨어는 개방된 구조를 가져야 하는데, SDR Forum[2]에서는 JTRS (Joint Tactical Radio System) JPO (Joint Program Office)에 의해 정의된 SCA를 소프트웨어 프레임워크의 표준으로 삼고 있다[3].

SCA 는 통상적으로 디바이스 제어 프로그램과 응용 프로그램으로 이루어진 내장형 시스템 소프트웨어의 디자인 패턴을 잘 활용한 프레임워크를 제공하고 있다. SCA 는 분산 객체 모델의 실질적인 산업 표준인 CORBA (Common Object Request Broker Architecture)[4]를 기반 미들웨어로 채택하여 이기종의 하드웨어와 다양한 언어로 작성된 소프트웨어의 유연한 통합 환경을 제공한다. SCA 기반의 모바일 플랫폼은 사용되는 무선 도메인과 관계없이 공통의 개방형 프레임워크를 바탕으로 도메인간의 상호 운용성, 소프트웨어의 이식성, 재사용성, 다양한 범위의 주파수 이용성 등을 제공한다.

본 논문의 구성으로 2장에서는 SCA 소프트웨어 프레임워크의 구조와 기능에 대해서 살펴보도록 한다. 3장에서는 문제점으로 파악된 소프트웨어 컴포넌트에 대한 실시간 교체를 접근하는 방법을 살펴보고 4장에서 프로그램 시뮬레이션에서 도출한 결과를 통해 5장에서 결론을 내리도록 한다.

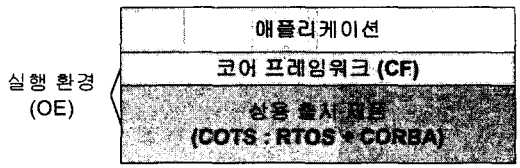

그림 1. SCA 소프트웨어의 기본구조

2. SCA 소프트웨어 프레임워크

2.1 SCA 소프트웨어 기본 구조

SCA 기본 구조는 소프트웨어 컴포넌트 간의 상호 연결 및 관리를 위한 상용 프레임워크 (OS, CORBA) 기반의 개방적, 계층적인 분산 처리 환경을 제공한다. SCA 소프트웨어의 기본 구조가 그림 1에 나타난다. SCA는 크게 실행 환경 (OE : Operating Environment)와 애플리케이션으로 이루어져 있고, 실행 환경은 RTOS (Real-Time Operating System)와 CORBA로 이루어진 상용 출시 제품부(COTS : Commercial Off-the-Shelf)와 코어 프레임워크 (CF : Core Framework)로 나누어진다.

2.2 SCA의 Core Framework

SCA 소프트웨어 구조에서 가장 핵심이 되는 부분이 바로 Core Framework 부분이다. CF는 그 기능과 역할에 따라서 크게 4 부분으로 구성된다.

- 기본 응용 인터페이스
- 프레임워크 제어 인터페이스
- 프레임워크 서비스 인터페이스
- 도메인 프로파일

기본 응용 인터페이스는 소프트웨어 애플리케이션에서 사용될 수 있는 기본이 되는 인터페이스이다. 예를 들면 CORBA의 ORB를 이용한 통신에 사용될 수 있는 포트를 연결하거나 해제하는 등의 오퍼레이션 등이 정의되어 있는 부분이다. 이 인터페이스를 이용하여 각 컴포넌트는 기본적인 생성/제거 작업, 포트 연결 설정/해제 작업 등을 수행할 수 있게 된다.

프레임워크 제어 인터페이스는 CF 내에서도 가장 중요한 도메인 구성요소와 논리 장치들을 제어하는 인터페이스이다. 하나의 도메인 내에서 각종 도메인 구성요소인 애플리케이션, 장치, 장치 관리자, 이벤트 채널 등을 관리하거나 논리 장치를 생성하고 실행시키며 해제하는 등의 주요한 오퍼레이션을 제공하는 부분이다. 프레임워크 제어 인터페이스는 기본 응용 인터페이스를 상속하여 필요한 기능을 추가하는 형식으로 존재한다.

프레임워크 서비스 인터페이스는 코어/Non-코어 애플리케이션에 사용되는 서비스를 제공하는 인터페이스로 지역 파일 시스템뿐만 아니라 원격 파일 시스템도 파일 관리자를 통해서 지역 파일 시스템처럼 접근이 가능하고, 또 파일 시스템에 존재하는 파일을 읽고 쓰는 등의 오퍼레이션을 제공한다.

도메인 프로파일은 SCA 시스템 도메인을 구성하는 모든 요소들의 속성, 능력, 상호 관계 등을 모두 나타내는 파일들의 집합을 일컫는다. 도메인 프로파일 파일들은 XML (eXtensible Markup Langauge)로 구성되고, DTD (Document Type Definitions) 포맷을 따른다[5].

3. 컴포넌트의 실시간 교체 방법

SCA 소프트웨어 프레임워크에서 애플리케이션을 플랫폼 상에 로딩하기 위해서 도메인 프로파일 중 SAD (Software Assembly Descriptor)를 참조한다. SAD는 하나의 애플리케이션을 이루는 소프트웨어 컴포넌트들의 배치와 연결 정보를 기술한다. 시스템에 응용프로그램을 인스톨하는 것은 하나의 SAD 파일을 도메인 영역에 인스톨하는 것으로 볼 수 있다. SAD는 해당 컴포넌트에 대한 배치 정보를 얻기 위해 도메인 프로파일 중 SPD 파일의 인스턴스에 대한 하나 이상의 참조를 가진다.

SPD (Software Package Descriptor) 는 특정 소프트웨어 컴포넌트에 대한 구현 정보를 기술하는 파일이다. 이름, 저자, 속성파일, 구현 코드정보 그리고 하드웨어와 소프트웨어의 의존관계와 같은 소프트웨어 패키지에 대한 일반적인 정보를 포함한다. SCD (Software Component Descriptor) 의 인스턴스에 대하여 참조를 가질 수 있다. SCD는 특정 컴포넌트 (Resource, Resource-Factory, Device)에 대한 CORBA 인터페이스 정보를 기술한다. 컴포넌트가 상속/지원하는 인터페이스와 지원하는 포트도 포함된다.

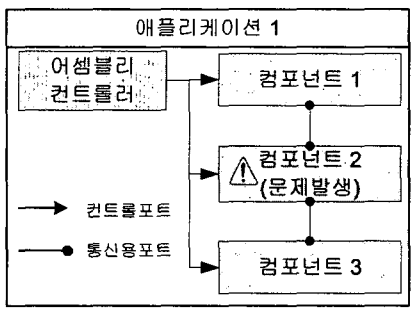

그림 2. 애플리케이션의 구성 예

이러한 기술 구조를 바탕으로 프레임워크 제어 인터페이스의 ApplicationFactory는 SAD를 이용하여 도메인 영역에 애플리케이션을 인스톨할 수 있다. 그림 2와 같은 구조를 갖는 애플리케이션을 인스톨하였다고 가정하자. SAD에는 어셈블리 컨트롤러와 컴포넌트 1~3의 SPD를 참조하고 있다. 그리고 각 컴포넌트간의 연결 포트를 지정하여 어셈블리 컨트롤러에서 컴포넌트들을 조절할 수 있고, 컴포넌트들 간에는 통신용 포트가 설정되어 서로간에 통신이 가능하다.

그림 2와 같은 애플리케이션에서 구성하는 소프트웨어 컴포넌트 2에서 문제가 발생하였을 경우를 가정하자. 현

재의 SCA v3.0 규격에서 제공하는 오퍼레이션 중에는 문제가 발생한 컴포넌트에 대한 교체 방법이 존재하지 않는다. 또, CRC (Communication Research Center)[6]의 RMSC (The Military Satellite Communication) 그룹에서 SCA 규격을 참조 구현한 SCARI 프로젝트와 Virginia Tech의 MPRG (The Mobile and Portable Radio Research Group)의 OSSIE (The Open Source SCA Implementation::Embedded) 프로젝트[7]에서도 이러한 교체 방법에 대한 구현이 존재하지 않는다.

SCA 소프트웨어 프레임워크가 현재 문제가 발생한 컴포넌트를 교체할 수 있는 오퍼레이션을 제공하지 않으므로 교체하기 위해서는 우선 해당 애플리케이션을 언인스톨해야 한다. 다음으로 교체할 컴포넌트에 대한 SPD를 새로운 컴포넌트에 대한 SPD로 교체한 후 SAD에도 변경된 컴포넌트 SPD에 대한 정보를 변경해 주어야 한다. 그런 후에 애플리케이션을 인스톨하는 과정을 거쳐야 교체 작업을 완료할 수 있게 된다.

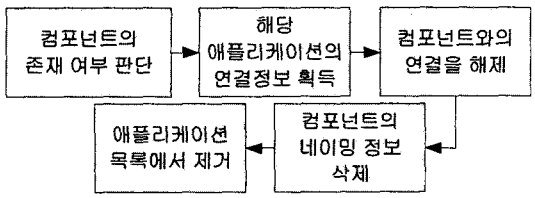

그림 3. 컴포넌트 제거 오퍼레이션

이러한 복잡한 과정을 거쳐야 하는 문제를 해결하기 위해서 본 논문에서는 교체할 컴포넌트만 도메인 영역에서 제거하는 오퍼레이션을 CF의 프레임워크 제어 인터페이스 중에서 Application 인터페이스에 추가하였다. 추가된 오퍼레이션의 작업 절차는 그림 3과 같다.

앞의 그림 2에서 살펴본 애플리케이션에서 컴포넌트 2가 문제가 발생한 경우를 다시 한번 가정하자.. 새로 추가한 컴포넌트 제거 오퍼레이션을 통하여 문제가 발생한 컴포넌트 2의 연결 정보를 SAD를 통해서 얻어온다. 얻어온 연결 정보를 바탕으로 하여 일단 컴포넌트 2의 연결을 해제한다. 그리고 네이밍 서비스에 등록된 컴포넌트 2의 네이밍 정보를 삭제하고 애플리케이션의 컴포넌트 목록에서 컴포넌트 2를 제거하는 것으로 제거 오퍼레이션 작업은 완료된다.

4. 실험 결과

프로그램 시뮬레이션을 위해 Fedora Core 3 Linux에서 작업하였고 Virginia Tech의 MPRG의 OSSIE를 기반으로 하여 SCA v3.0의 참조 구현을 진행하였다. 프레임워크 제어 인터페이스에 추가된 컴포넌트 제거 오퍼레이션은 교체해야 할 컴포넌트를 제거한다. 그 후에 변경된 컴포넌트를 포함하는 새로운 애플리케이션을 인스톨하여 교체 작업을 그림 4와 같이 진행하였다.

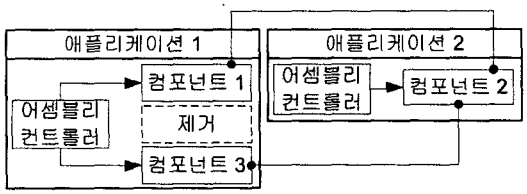

그림 4. 컴포넌트 교체 방법

시뮬레이션 결과, 교체를 결정한 컴포넌트가 컴포넌트 제거 오퍼레이션을 통하여 도메인 영역과 네이밍 서비스 목록에서 제거되었고, 새로운 애플리케이션을 인스톨하였을 때, 처음과 같은 연결을 유지하는 것을 확인하였다.

5. 결 론

다양한 이동 통신 환경 시스템 통합을 고려할 때 SDR은 좋은 선택이 될 수 있다. 이 SDR에서 개방형 소프트웨어 프레임워크로 JTRS JPO에서 정의한 SCA를 채택하였다. SDR이 실제 환경에서 사용되기 위해선 SCA에 필요한 컴포넌트 교체 방법을 논문에서 제안하였다. 실시간 교체 방법을 채택하면 SCA는 지금보다 더욱 유연한 구조를 가질 수 있었다.

제안한 방법이 기존 애플리케이션에서의 연결 정보 확인을 위해 SAD를 조사한다. 이 때 확인한 연결 정보를 바탕으로 새로운 애플리케이션을 설치하는 것이 아니라 필요한 부분, 즉 교체되어야 할 컴포넌트만 교체할 수 있는 방법이 연구되어야 할 것이다.

참고 문헌

1. 황경호, 조동호, " Software Defined Redio 기술," Telecommunication. Review, 제10권, 제1호, 2000. 1.~2., pp.130-143.
2. SDR forum. http://www.sdrforum.org/
3. Joint Tactical Radio System (JTRS). http://www.jtrs.saalt.army.mil/
4. The Common Object Request Broker: Architecture and Specification, Version 3.0, Object Management Group, June 2002
5. Software Communications Architecture (SCA) Specification JTRS-5000 SCA V3.0 Aug. 27, 2004
6. Communication Research Center (CRC). http://www.crc.ca/
7. The Open Source SCA Implementation::Embedded (OSSIE). http://ossie.mprg.org