SCA 기반 다중모드 SDR 단말기 구조

SCA Based Multi-mode SDR Access Terminal Architecture

이동통신과 방송기술 개발 현황 특집

김준식 (J.S. Kim) 모바일액세스연구팀 선임연구원
김홍숙 (H.S. Kim) 모바일액세스연구팀 선임연구원
박남훈 (N.H. Park) 모바일액세스연구팀 팀장
김진업 (J.U. Kim) SR연구팀 팀장
권오준 (O.J. Kwon) 동의대학교 교수

목 차

* 본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업의 일환으로 수행하였음. [2006-S-012-01, SDR 단말용 미들웨어 플랫폼]

SDR 기술은 다양한 무선통신 환경(다중모드, 다중표준, 다중대역, 다중기능)에 유연하게 대처하기 위하여 하나의 공통 하드웨어에 사용자가 원하는 응용 소프트웨어(무선 프로토콜 규격)로 재구성할 수 있는 개방형 신호처리 기술이다. 그러므로 SDR은 이러한 이동통신시장의 변화에 적극적으로 대처하기 위한 핵심기술로써, 통신시장이 직면하고 있는 문제점을 해결할 수 있는 해결책 중의 하나로 고려되고 있다. 본 고에서는 SCA 기반 SDR 단말기 구현을 위해, 통신 애플리케이션 소프트웨어를 지원하는 하부구조 관점에서 SCA Core Framework의 구조 및 기능을 기술하고, ETRI에서 SCA 2.2 규격을 준수하여 개발한 SCARLET 미들웨어 및 SDR 공용 단말 하드웨어를 소개하고, SCA 기반의 HSDPA/T-DMB 응용 컴포넌트 구현을 통한 실험 내용을 소개한다.

I. 서론

최근 정보통신 발전에 따른 기술의 진화를 특징짓는 요인 중 가장 중요한 변화로 융합이 거론되고 있다. 또한, 음성 서비스 위주의 이동통신 사용자 요구사항이 다양한 형태의 멀티미디어 데이터를 하나의 단말에서 통신모드, 네트워크의 종류 및 사용 장소의 제한 없이 사용하는 방향으로 변화하고 있다. 이러한 요구 사항의 변화는 통신사업자와 단말기 제조업체에게 기존 통신 규격을 포함한 여러 통신 규격의 지원 및 새로운 표준 규격의 신속한 적용 또는 업그레이드에 적용할 새로운 기술, 패러다임의 변화를 요구하고 있다[1].

SDR 기술은 이러한 다양한 무선 통신 환경에 유연하게 대처하기 위하여 하나의 공통 하드웨어에 사용자가 원하는 응용 소프트웨어로 무선 접속 통신 시스템을 재구성할 수 있는 개방형 신호처리 기술이다[2],[3]. 또한 아날로그 변조를 대신할 디지털 변조 기술 및 스마트 안테나의 출현 등의 기술 발전도 SDR 기술의 실용화에 일조하고 있다[4].

SDR 시스템의 기술 개발 촉진을 위하여 1996년 MMITS Forum이 만들어졌으며, 1998년 SDRF으로 개칭되었다. SDRF에서는 SDR 시스템을 기능적 관점에서 정의하는 노력을 통하여 상위 수준 기능 모델인 SDR 소프트웨어 참조 모델을 제시하였으며, 미국방성 JTRS 프로젝트에서는 컴포넌트 기반 프레임워크인 SCA를 설계하였고, SDRF에 의해 SDR 소프트웨어 구조 표준으로 채택되었다[5].

본 고에서는 SDR 기반 통신 애플리케이션 소프트웨어를 지원하는 하부구조 관점에서 SCA Core Framework의 구조 및 기능을 기술하고, ETRI에서 SCA 2.2 규격을 준수하여 구현한 SCARLET 미들웨어 및 SDR 공용 단말 하드웨어를 소개하고, SCA 기반 HSDPA/T-DMB 응용 컴포넌트 구현을 통한 실험 내용을 소개한다.

● 용 어 해 설 ●

SDR: 무선 이동 통신 시스템에서 RF 영역을 포함한 대부분의 기능 블록을 프로그래밍이 가능한 고속의 processing element 상에서 구현된 소프트웨어 블록을 사용함으로써, 하드웨어의 교체 없이 소프트웨어 업그레이드 재구성만을 통하여 다중 무선 접속 규격 및 서비스 기능 등을 실현하는 기술이다.

Ⅱ. SCA 규격

JTRS의 JPEO가 개발하여 공표하고 있는 SCA 규격은 다음과 같은 개발 목표를 가지고 있다.

- SCA 규격을 준수하는 응용 소프트웨어의 이식성 제공
- 상용 표준을 활용한 개발 비용 절감
- 재사용 가능한 설계 모듈을 통한 소프트웨어 개발 기간 절감
- 상용화되고 있는 프레임워크 및 구조 상에서의 구축

SCA는 시스템 사양이 아니라, 위와 같은 목표를 달성하기 위하여 시스템 설계시 준수하여야 할 규칙들을 제공하는 것으로, 이러한 규칙은 특정 구현에 종속되지 않는 일반적인 규칙이다.

SCA 규격의 역할은 통신 시스템상에 존재하는 소프트웨어, 하드웨어 구성 요소에 대한 관리 및 요구사항, 요구 용량들이 적절하게 운용될 수 있도록 하는 공통 하부 구조의 제공이다. 이러한 역할을 수행하기 위하여 Core Framework(CF)라 불리는 인터페이스들의 집합을 정의하고, 이를 시스템에서 운용될 애플리케이션과 하부 하드웨어 사이에 두어서, 애플리케이션은 Core Framework에 정의된 인터페이스에 따라 하부 하드웨어를 이용하는 구조를 가지고 있다.

● 용 어 해 설 ●

미들웨어(middleware): 분산 컴퓨팅 환경에서 이질성(heterogeneity)을 가진 두 개의 프로그램 사이에서 중개 역할 및 투명한 연결성(transparent connectivity) 등의 상호 운용성(interoperability)에 필요한 기능을 제공하는 소프트웨어 또는 소프트웨어 계층이다.

1. SCA에서의 소프트웨어 구조

일반적으로 애플리케이션은 "하나 이상의 모듈을 포함하는 실행 가능한 소프트웨어 프로그램을 의미한다. SCA 규격에서의 애플리케이션은 규격에서 요구하고 있는 Base Application 인터페이스를 구현하고 SAD 파일을 통해 확인되는 하나 이상의 소프트웨어 모듈들로 구성된 것"으로 제한한다. 애플리케이션이 적재되고 실행되었을 때, 이들 소프트웨어 모듈들은 애플리케이션을 구성하는 컴포넌트들을 생성한다. 공통 하드웨어 상에서 SCA 애플리케이션의 소프트웨어적인 추가, 제거, 수정을 통해 재구성/재배치가 가능한 통신 시스템을 실현한다.

SCA는 계층적인 소프트웨어 구조를 가정한다. SCA 소프트웨어 구조는 크게 운영 환경(operating environment) 계층과 운영 환경 계층에서 제공하는 서비스를 이용하여 구현되는 애플리케이션 리소스 계층으로 구분할 수 있다. 운영 환경은 다시 Core Framework, CORBA 미들웨어, AEP 계층과 POSIX [6] 기반 운영체제로 이루어져 있다.

가. Core Framework

SCA 소프트웨어 구조에서 Core Framework [5],[7],[8]는 CORBA상에서 애플리케이션 프로그램에 컴포넌트 기반 컴퓨팅 능력을 제공한다. Core Framework는 애플리케이션 설계자가 사용할 수 있는 하부 소프트웨어/하드웨어 계층에 대한 추상화(abstraction)를 제공하기 위한 개방형 애플리케이션 계층 인터페이스들과 서비스들의 집합이다.

애플리케이션 프로그램 설계자들은 Core Framework에서 제공하는 인터페이스들을 사용하여 하나의 애플리케이션 프로그램을 구성하는 컴포넌트를 프로세싱 노드에 설치, 시작, 정지, 제거 등의 관리 작업을 수행할 수 있다. 각 인터페이스는 특정 기능에 필요한 오퍼레이션과 속성을 제공하며, 오퍼레이션에 필요한 구체적인 파라미터 등의 내용은 프로파일(profile)이라 불리는 XML 형태의 파일 내에 기술된다.

1) Base Application 인터페이스

Base Application 인터페이스는 모든 애플리케이션 컴포넌트가 구현하여야 하는 인터페이스로 Port, LifeCycle, TestableObject, PropertySet, PortSupplier, Resource와 ResourceFactory 인터페이스로 구성된다.

Port 인터페이스는 컴포넌트 포트간의 연결 관리를 위한 오퍼레이션을 제공한다. 애플리케이션 컴포넌트는 Port 인터페이스를 상속한 인터페이스를 지정하여 특정 포트 타입을 정의하고 이를 사용하여 데이터 또는 제어 신호를 전달하기 위한 오퍼레이션을 정의할 수 있다. 컴포넌트들간의 포트 연결 정보는 SAD 파일과 DCD 파일에 기술한다.

LifeCycle 인터페이스는 생성된 컴포넌트에 따른 데이터 또는 프로세싱 요소(processing elements)를 초기화하거나 해제하는 데 필요한 오퍼레이션들을 제공한다.

TestableObject 인터페이스는 컴포넌트의 구현을 시험하는 데 필요한 오퍼레이션들을 제공한다. 시험항목에 대한 내용은 해당 컴포넌트의 SPD에서 참조하고 있는 Property Descriptor의 test element에서 기술한다.

PropertySet 인터페이스는 컴포넌트의 properties/attributes를 접근하는 configure() 오퍼레이션과 query() 오퍼레이션을 제공한다.

PortSupplier 인터페이스는 포트를 제공하는 컴포넌트에서 특정 포트에 대한 객체 참조를 얻기 위한 getPort() 오퍼레이션을 제공한다.

Resource 인터페이스는 LifeCycle, TestableObject, PropertySet 및 PortSupplier 인터페이스를 상속하여 애플리케이션 컴포넌트 수준에서의 동작 시작 및 정지를 위한 start()/stop() 오퍼레이션을 제공한다.

ResourceFactory 인터페이스는 애플리케이션을 구성하는 리소스를 생성하거나 종료하는 데 필요한 오퍼레이션을 제공한다.

2) Framework Control 인터페이스

SCA 규격 용어집에 따르면, 도메인은 "하나의 도메인 관리자 컴포넌트의 제어 하에 있는 하드웨어 디바이스들과 하드웨어 디바이스상에서 이용 가능한 애플리케이션들의 집합"으로 정의된다.

하나의 도메인 내에서의 Framework Control 인터페이스는 도메인 관리와 디바이스 관리를 위한 인터페이스를 제공한다. Framework Control 관련 인터페이스로는 Application, ApplicationFactory, DomainManager, DeviceManager 인터페이스가 있다. 이들 인터페이스는 애플리케이션, 디바이스, 도메인 내의 디바이스 관리자들의 등록/해제, 도메인 내에서의 애플리케이션의 제어와 관련된 오퍼레이션들을 제공한다.

Application 인터페이스는 Resource 인터페이스를 상속하며, 도메인 내의 생성된 애플리케이션 인스턴스의 제어, 구성, 상태와 관련된 오퍼레이션들을 제공한다.

ApplicationFactory 인터페이스는 도메인 내의 지정된 타입의 애플리케이션 생성을 요청하는 데 필요한 create() 오퍼레이션을 제공한다. create() 오퍼레이션은 SAD 파일과 SPD 파일을 참조하여 애플리케이션 인스턴스를 생성한다. DomainManager 인터페이스를 구현한 도메인 관리자는 도메인 내에 설치된 SAD별로 각각의 ApplicationFactory 클래스의 인스턴스를 생성한다.

DomainManager 인터페이스는 시스템 도메인의 제어와 구성을 위한 오퍼레이션을 제공한다. DomainManager 인터페이스에서 제공되는 오퍼레이션은 크게 HCI, 등록(registration), CF 관리(CF management)로 구분할 수 있으며, HCI 관련 오퍼레이션은 도메인을 구성하고 도메인의 디바이스, 서비스, 애플리케이션의 역량(capabilities)에 관련된 정보를 접근하고, 유지 관리 기능을 시작하는 데 사용한다. 등록 관련 오퍼레이션은 시작 또는 동작중에 디바이스 관리자, 디바이스 관리자에 등록된 디바이스 및 서비스, 애플리케이션의 등록/등록 해제와 관련된 작업에 사용한다. CF 관리 관련 오퍼레이션은 등록된 디바이스 관리자와 도메인 관리자의 파일 관리자에 대한 인터페이스 접근을 제공한다.

DeviceManager 인터페이스는 일련의 디바이스들과 서비스들을 관리하는 데 필요한 속성들과 오퍼레이션들을 제공한다. DeviceManager 인터페이스의 deviceConfigurationProfile 속성은 디바이스와 서비스가 전개되어 배치될 물리적 디바이스의 위치 및 논리적 명칭에 대한 매핑 정보가 기록된 DCD 파일에 대한 정보가 기록된다. DeviceManager 인터페이스의 fileSys 속성에는 디바이스 관리자와 연관된 파일 시스템에 대한 정보를 유지한다. 또한 DeviceManager 인터페이스의 registeredDevices 속성에는 해당 디바이스 관리자에 등록되고 관리되는 디바이스들의 리스트가 유지되며, DeviceManager 인터페이스의 registeredServices 속성에는 해당 디바이스 관리자에 등록되고 관리되는 서비스들의 리스트가 유지된다.

3) Base Device 인터페이스

시스템 내의 하드웨어 장치들을 해당 소프트웨어 인터페이스를 통하여 관리, 제어하기 위한 인터페이스로 Device, LoadableDevice, ExecutableDevice, AggregateDevice 인터페이스가 있다.

Device 인터페이스는 Resource 인터페이스를 상속하며, 물리적인 하드웨어 디바이스에 대한 소프트웨어 추상화를 제공하기 위한 추가적인 속성과 오퍼레이션을 제공한다.

Device::softwareProfile 속성은 디바이스의 ports, query/configure properties, capacity properties 및 status properties에 대한 정의를 포함하고 있는 SPD 파일에 대한 참조를 포함하고 있다. 또한, 상태 관리를 위한 usageState, adminState, operationalState 속성을 유지하고 있으며, 디바이스로부터 메모리 등의 특정 용량을 요구하거나 반납하기 위한 allocateCapacity 오퍼레이션과 deallocateCapacity 오퍼레이션을 제공한다.

LoadableDevice 인터페이스는 Device 인터페이스를 확장하여 소프트웨어의 적재(loading) 및 적

재 해제(unloading)와 관련된 오퍼레이션을 추가한 것이다.

ExecutableDevice 인터페이스는 LoadableDevice 인터페이스를 확장하여 디바이스 상에서 소프트웨어 프로세스 또는 스레드의 실행을 시작하거나 종료하는 오퍼레이션을 추가한 것이다.

AggregatedDevice 인터페이스는 부모 디바이스로부터 자식 디바이스를 추가하거나 제거하는 데 필요한 오퍼레이션을 제공하도록 확장한 인터페이스이다.

4) Framework Service 인터페이스

기타 지원 기능 및 서비스를 제공하기 위하여 File, FileSystem, FileManager 인터페이스가 있다. File 인터페이스는 파일을 읽고, 쓰는 데 필요한 오퍼레이션을 제공한다. FileSystem 인터페이스는 물리적 파일 시스템에 대한 원격 접근을 제공한다. FileManager 인터페이스는 다른 파일 시스템들을 논리적으로 하나의 파일 시스템처럼 다룰 수 있는 기능을 제공한다.

나. AEP 및 CORBA 미들웨어 계층

Core Framework 서비스에서 사용하는 분산 컴포넌트 지원에 필요한 명명법(naming and typing), 오류 관리(error management), 서비스 중개 관리(service brokering management) 및 통신 링크 제어(communication link control) 등의 하드웨어, 소프트웨어, 운영체제, 네트워크 프로토콜, 프로그래밍 언어의 차이에 따른 이기종 분산 컴퓨팅 관련 순수 미들웨어 기능은 OMG Minimum CORBA[9]를 사용한다.

SCA CF 및 Minimum CORBA 미들웨어가 실행되도록 지원하는 하부의 운영체제는 그 위에서 실행되는 애플리케이션 프로그램 또는 컴포넌트들이 무선 통신 프로토콜의 기능을 구현하는 경우, 주어진 종료시한 내에 실행을 완료하는 실시간 처리 특성을 가지고 있으므로 실시간 운영체제(RTOS)이어야 한다. 이를 위해 SCA에서는 IEEE std. 1003.13에 정의된 POSIX Real-time Application Support[6]를 준수하는 운영체제를 사용하도록 명시하고 있다.

먼저, 애플리케이션 컴포넌트는 CF의 Framework Control 인터페이스를 통하여 제어된다. 또한 애플리케이션 컴포넌트는 Port 인터페이스를 사용하여 다른 애플리케이션 컴포넌트와 통신하거나 시스템이 제공하는 시스템 컴포넌트(서비스 또는 디바이스)와 통신하게 된다. 애플리케이션 컴포넌트와 Framework Control 또는 Framework Services 인터페이스간의 통신은 CORBA 미들웨어를 통하여 이루어져야 한다. 이러한 통신 제약 사항의 의도는 서비스 또는 디바이스와 같은 시스템 컴포넌트에 대한 API들을 특정 시스템 또는 도메인에 맞게 표준화함으로써 프레임워크와 함께 애플리케이션과 시스템간의 일관된 통신 메커니즘을 제공하는 데 있다.

애플리케이션 컴포넌트는 운영체제가 제공하는 기능 중 POSIX 규격의 부분 집합인 AEP에서 제공하는 기능만을 사용하도록 제한된다.

디바이스와 서비스 같은 시스템 컴포넌트는 하부 시스템에 의존적일 수 있으므로 운영체제가 제공하는 기능의 사용에 대한 제약 사항은 없으며, Base Device 인터페이스를 통하여 Framework Control 인터페이스에 의해 관리된다.

2. 도메인 프로파일

컴포넌트들을 배치하고 연결하여 애플리케이션이 동적으로 구성될 수 있도록 시스템 내의 하드웨어 장치 및 소프트웨어 컴포넌트 특성을 기술하는 XML 파일을 통칭하여 도메인 프로파일(domain profile)이라 한다. 도메인 프로파일 내에는 하드웨어 및 소프트웨어 컴포넌트의 인터페이스, 기능적인 면에서의 역량, 논리적 위치, 상호 의존성 및 관련 파라미터 등의 내용이 포함된다.

도메인 프로파일은 파일 내에 기술되는 내용에 따라 CF에 애플리케이션 프로그램, 애플리케이션 프로그램 구성 컴포넌트 등의 소프트웨어적인 정보를 제공하는 소프트웨어 프로파일, 프로세싱 노드

및 노드의 특성과 같은 하드웨어적인 정보를 제공하는 디바이스 프로파일과 도메인 관리자용 구성 파일로 구분할 수 있다.

가. 소프트웨어 프로파일

소프트웨어 측면에서의 컴포넌트 및 애플리케이션 정보를 기술하는 소프트웨어 프로파일은 다음과 같다.

- SAD: 하나의 애플리케이션 프로그램을 구성하는 애플리케이션 컴포넌트들과 이들 간의 연결 관계를 기술한다. Application Factory는 SAD 파일을 이용하여 지정된 애플리케이션을 생성한다.
- SPD: 소프트웨어 컴포넌트의 구현들에 대한 정보를 유지한다. 소프트웨어 패키지의 이름, 작성자, property 파일 정보 및 구현 코드의 정보, 하드웨어/소프트웨어 의존성(실행 가능한 플랫폼, 실행 환경, 실행에 필요한 최소 메모리, CPU 요구량) 등의 정보를 기술한다.
- SCD: 특정 SCA 소프트웨어 컴포넌트(Resource, ResourceFactory, Device) 구현에서 제공하거나 사용하는 인터페이스에 대한 정보를 포함한다. Device에 대한 SCD인 경우 DPD 파일에 대한 참조를 포함한다.
- PRF: 컴포넌트 수준에서 적용 가능한 형상(configuration), 검사(test), 실행(execute) 및 할당(allocation) 관련 properties에 대한 정보를 기술한다.

나. 디바이스 프로파일

하드웨어 측면에서의 구성 정보를 제공하는 디바이스 프로파일은 다음과 같다.

- DCD: 특정 디바이스 관리자와 연관된 디바이스에 대한 정보, 도메인 관리자를 찾는 방법, 개별 디바이스에 대한 구성 정보(Log, FileSystem 등)를 기술한다.
- DPD: 개별 노드 수준에서의 하드웨어 장치 정보(제작자, 모델번호, 하드웨어 타입, 일련번호, 메모리 크기 등)를 기술한다.
- PRF: 디바이스 패키지에 적용 가능한 구성(configuration), 검사, 실행 및 할당 관련 properties에 대한 정보를 기술한다.

다. 도메인 관리자 구성 파일

도메인 관리자는 애플리케이션 프로그램 수준에서의 설치, 제거 및 하드웨어 디바이스의 등록, 해지를 위한 최상위 인터페이스를 제공한다. 도메인 관리자의 실행위치 및 도메인 관리자에 의해 사용되는 CF 인터페이스 또는 서비스 등의 내용은 DMD에 기술한다[10].

Ⅲ. SCA 기반 애플리케이션 개발 절차

SDR용 애플리케이션은 일반적으로 신호처리, 패킷 입출력, 보안, 라우팅 등의 다양한 기능을 가진 모듈들이 모여서 구성된다. SCA 기반 애플리케이션을 구성하는 컴포넌트들은 각각의 기능 모듈에 해당하는 기능을 수행하는 단위인 컴포넌트로 구현되며, Core Framework에 의해 제어될 수 있도록 Base Application 인터페이스를 구현하여야 한다. 그러나, 기존에 개발된 바이너리 코드만 있는 경우에는 소스코드가 없으므로 컴포넌트화를 할 수 없다. 이때에는 기존 바이너리 코드에 adaptor 코드와 연동하여 다른 컴포넌트와의 통신 및 Core Framework 구조에 호환되도록 구성하여야 한다.

1. 기능에 따른 모듈화

SCA 기반 애플리케이션 개발은 먼저 해당 애플리케이션의 기능을 정의하고, 각 기능을 분할하여 모듈별로 분리하는 작업이 선행되어야 한다. 이때 모듈화는 재사용성이 높으면서 독립적으로 배포가 가능하도록 하여야 한다.

2. 외부에 제공할 인터페이스 정의

기능의 모듈별 분리가 끝나면, 각 모듈 기능이 구현될 컴포넌트는 외부에 제공할 인터페이스를 정의하여야 한다. 각 컴포넌트가 제공할 인터페이스는 CORBA IDL을 사용하여 정의한다. 컴포넌트가 외부에 제공하여야 할 인터페이스는 SCA Core Framework를 위한 Base Application 인터페이스와 사용자 지정 인터페이스가 있다. IDL로 정의한 인터페이스 명세에 따라 IDL 컴파일러는 클라이언트 stub 코드와 서버용 skeleton 코드를 생성한다. 컴포넌트 내부에서만 사용하는 기능을 구현하는 코드의 경우에는 컴포넌트 외부에서 호출할 경우가 없으므로 인터페이스에 포함시키지 않는다.

3. 인터페이스 정의에 따른 컴포넌트 구현

컴포넌트 개발자는 IDL로 정의한 Base Application 인터페이스와 사용자 정의 인터페이스에 따라 생성된 stub/skeleton 코드를 사용하여 컴포넌트가 외부에 제공하는 오퍼레이션들과 속성에 대한 구현 코드를 작성한다. SCA 컴포넌트는 컴포넌트간 통신을 위하여 포트 개념을 사용한다. 따라서, 컴포넌트간 통신 기능이 필요한 컴포넌트는 데이터 및 제어 신호의 송수신을 위한 Use Port와 Provide Port를 구현하여야 한다. 데이터 및 제어신호는 소스 컴포넌트의 Use Port에서 대상 컴포넌트의 Provide Port로 전송된다. 컴포넌트간 통신을 위한 포트 구현을 완료한 후, 해당 컴포넌트의 고유 기능인 신호 처리 기능을 구현하여 추가한다.

애플리케이션 수준에서의 시작과 정지 같은 제어 흐름에 따라 애플리케이션 구성 컴포넌트들이 동작하도록 제어하기 위하여 애플리케이션의 대리역할을 수행하는 assembly controller 컴포넌트가 있어야 한다. Assembly controller 컴포넌트는 Core Framework로부터의 명령을 애플리케이션을 대리하여 수신 후, 받은 명령에 따라 다른 컴포넌트의 동작을 제어하도록 한다.

4. 컴포넌트 디스크립터 파일 작성

컴포넌트별로 PRF, SCD, SPD 파일을 작성한다. PRF 파일에는 컴포넌트의 properties 정보를 기재하고, SCD 파일에는 컴포넌트의 인터페이스 정보 및 컴포넌트 타입 정보를 기재하며, SPD 파일에는 컴포넌트의 SCD 파일 및 PRF 파일 정보, 구현별 바이너리 파일의 위치 등의 구현 패키지 정보를 기록한다.

5. 애플리케이션 디스크립터 파일 작성

SCA 애플리케이션은 한 개 이상의 컴포넌트들로 구성되므로, 애플리케이션을 구성하는 컴포넌트들의 결합과 형상 정보를 SAD 파일에 기술하여 제공하여야 한다. SAD에는 애플리케이션에 필요한 컴포넌트들의 SPD 파일 리스트, 컴포넌트간 포트 및 인터페이스 연결 정보, 디바이스 및 서비스로의 연결 정보, 디바이스와 서비스 찾는 방법, 배치에 따른 종속성, 애플리케이션의 assembly controller 역할을 하는 컴포넌트의 지정 등이 포함된다.

6. 패키징 및 배포

도메인 프로파일 및 도메인 프로파일 내에서 참조하고 있는 모든 파일은 배포 가능한 하나의 패키지로 묶어진다. jar 또는 tar 형태의 패키징이 주로 사용된다. 배포와 관련된 내용은 구현에 종속적인 부분으로 SCA 규격에서는 다루지 않는다[11].

Ⅳ. HSDPA/T-DMB 통신 컴포넌트 구현

SCARLET은 SCA 규격 2.2를 준수하는 SCA Core Framework를 구현한 미들웨어 플랫폼으로서 소프트웨어와 하드웨어의 이식성 및 재사용성을 보장하고, SCA를 사용하여 개발된 제품들간 상호 연동성 보장을 원칙으로 한다. 특히, 기지국에 비하

여 컴퓨팅 리소스가 한정되는 단말기를 목표로 경량화에 주안점을 두고 개발되었다.

1. SDR 단말용 공용 단말 하드웨어

SCARLET 시스템이 탑재된 공용 단말 하드웨어는 T-DMB, HSDPA 이외에 현재 개발이 진행중인 Mobile WiMAX까지 수용 가능하도록 설계하였다. 미들웨어 플랫폼을 수용하기 적합한 구조를 가지도록 하였고, RF 모듈은 커넥터를 통해서 독립적으로 연결되도록 하였다.

(그림 1)은 검증용 단말 하드웨어 보드의 사진이고, <표 1>은 검증용 단말 하드웨어 제원을 정리한 것이다.

(그림 1) 검증용 단말 하드웨어

〈표 1〉 SCARLET 검증용 하드웨어 제원

CPU	- PXA320(Intel) 806MHz - Memory: Flash NAND 256MB, DDR SDRAM 256MB - User I/O: Ethernet 2 Port USB 2.0(Full Speed Host), USB Memory SD/MMC Interface UART(Serial Port, 38,400bps)
Control Logic	- XC4VLX15_10SF363C(XILINX) - CPLD
DSP	- TMS320DM642(TI) - 600MHz, 4800 MMACs
FPGA	- XC4VLX200-FFG1513-10C(XILINX) - 200,000 Logic Cell

<자료>: ETRI, 2007.

2. 통신 애플리케이션

단말기에서의 SCARLET과 SDR 통신 애플리케이션의 동작을 확인하기 위하여 공용 단말 하드웨어를 제작하고, SCA 기반의 컴포넌트로 HSDPA[12], T-DMB[13]용 애플리케이션을 개발하여 시험하였다. 각각의 애플리케이션을 구성하는 컴포넌트들은 SCARLET 시스템에 의해 GPP/DSP/FPGA 디바이스로 배치되어 구동되게 되며, 이러한 정보는 XML 형태의 도메인 프로파일에 기재된다.

T-DMB와 HSDPA 모드의 애플리케이션 컴포넌트들은 각각 하나의 애플리케이션 단위로 패키징되어 SCARLET 시스템에 의해 관리되고 실행되며 각각의 애플리케이션은 별도의 SAD 파일에 기술된다. 또한 CORBA의 naming service와 event service를 받기 위해 CORBA와 연동한다. 컴포넌트간 통신은 CORBA의 IDL을 이용한 SCARLET 시스템의 SCA Port 인터페이스를 이용한다.

SDR 단말용 SCA 기반 재구성 가능한 미들웨어 플랫폼 시스템 SCARLET 관리 및 진단 기능을 위하여, GUI 기반의 클라이언트 프로그램인 Waveform Manager를 MS Windows 상에서 개발하였다. 공용 단말 하드웨어와 Waveform Manager는 LAN으로 연결된다. Waveform Manager는 CORBA 기반 메시지 전달을 통해 단말 하드웨어 상의 SCA Core Framework에 접근하여 SCA 기반 애플리케이션 동작을 제어하고 관리한다.

가. HSDPA 애플리케이션

HSDPA용 애플리케이션을 구성하는 컴포넌트는 모두 6개로 구성하였다. HSDPA용 애플리케이션 컴포넌트는 기존 소스를 기능별로 재분할하여 직접 SCA 컴포넌트화 하였다.

HSDPA 어셈블리 제어 블록(HACONB)은 assembly controller의 기능을 수행하는 블록으로서

SCARLET 도메인 상에 설치된 HSDPA 애플리케이션의 컴포넌트들인 HLMACAB, HPHYAB, HL23B 리소스들의 start/stop/property 제어 기능을 수행하고, 다른 컴포넌트들이 자신에게 접근할 수 있도록 하기 위해 naming service에 자신을 등록(bind)한다.

HSDPA L2/L3 블록(HL23B)은 HSDPA의 MAC 이상 계층의 프로토콜을 담당하는 SCA 기반 응용 컴포넌트이다. 또한 SCA 기반 오퍼레이션의 수행을 위하여 상부로 HACONB, SCARLET CF와 인터페이스 되어 있다.

HSDPA LMAC 리소스 적응 블록(HLMACAB)은 HSDPA의 Lower MAC 프로토콜 기능을 수행하는 HLMACB의 SCA 인터페이스 기능을 담당한다. HLMACAB는 통신 프로토콜 계층 소프트웨어를 SCA의 규격에 부합하는 응용 컴포넌트로 바꾸어 SCARLET 환경 하에서 원활히 동작할 수 있도록 해준다.

HSDPA PHY 리소스 적응 블록(HPHYAB)은 HPHYAB는 HSDPA의 PHY 기능을 담당하는 HPHYB의 SCA 인터페이스 기능을 담당한다. HPHYAB는 HSDPA PHY를 SCA의 규격에 부합하는 응용 컴포넌트로 바꾸어 SCARLET 환경 하에서 원활히 동작할 수 있도록 해준다.

HSDPA LMAC 블록(HLMACB)은 HSDPA의 Lower MAC 기능을 담당하는 Non-SCA 기반 블록이다.

HSDPA PHY 블록(HPHYB)은 HSDPA의 PHY 기능을 담당하는 Non-SCA 기반 블록이다[11].

나. T-DMB 애플리케이션

T-DMB 응용컴포넌트는 아래와 같이 총 7개의 블록으로 구성된다.

T-DMB 어셈블리 제어 블록(TACB)은 assembly controller로서 SCARLET 도메인 상에 설치된 T-DMB 애플리케이션의 컴포넌트들인 TRMB, TAMB, TDCWB, TFCWB, TDCB, TFCB들의 start/stop/property 제어 기능을 수행하고, 다른 컴포넌트들이 자신에게 접근할 수 있도록 하기 위해 naming service에 자신을 등록한다.

T-DMB DSP 제어 래퍼 블록(TDCWB)은 T-DMB의 DSP 처리 기능을 담당하는 TDCB의 SCA 인터페이스 기능을 담당한다. TDCWB는 TDCB를 SCA의 규격에 부합하는 응용 컴포넌트로 바꾸어 SCARLET 환경 하에서 원활히 동작할 수 있도록 해준다.

T-DMB FPGA 제어 래퍼 블록(TFCWB)은 T-DMB의 FPGA 처리 기능을 담당하는 TFCB의 SCA 인터페이스 기능을 담당한다. TFCWB는 TFCB를 SCA의 규격에 부합하는 응용 컴포넌트로 바꾸어 SCARLET 환경 하에서 원활히 동작할 수 있도록 해준다.

T-DMB DSP 제어 블록(TDCB)은 T-DMB의 DSP에서 처리할 기능들을 담당하는 Non-SCA 기반 응용 컴포넌트이다.

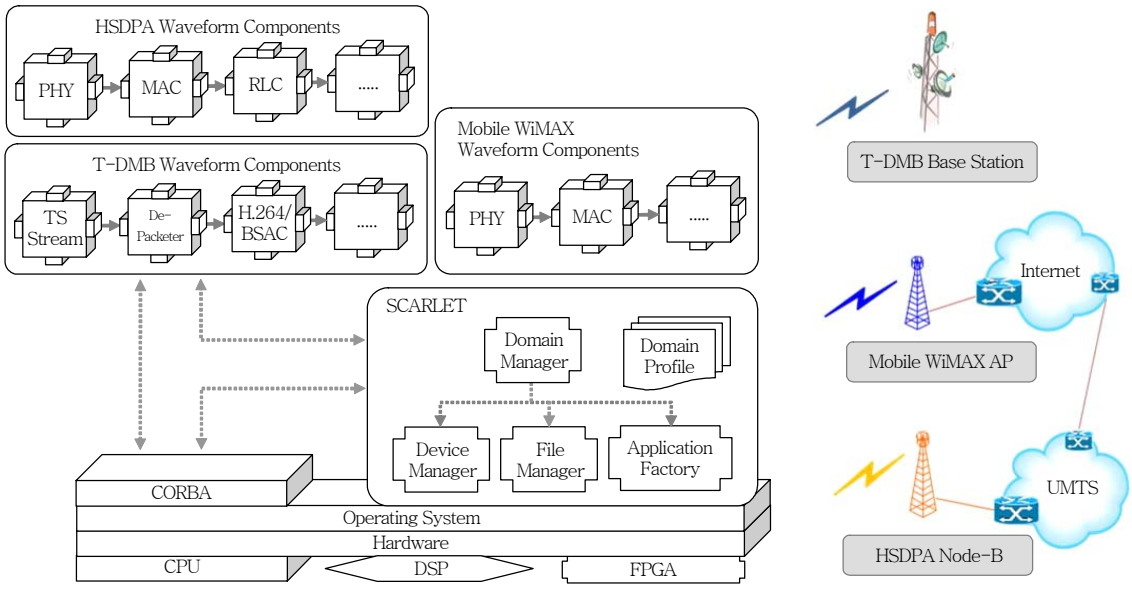
T-DMB FPGA 제어 블록(TFCB)은 T-DMB의 FPGA에서 처리할 기능들을 담당하는 Non-SCA 기반 응용 컴포넌트이다.

T-DMB RF 관리 블록(TRMB)은 T-DMB의 GPP에서 처리할 기능들 중, 채널변경 등의 제어를 담당하는 SCA 기반 응용 컴포넌트이다.

T-DMB 오디오 관리 블록(TAMB)은 T-DMB의 DSP에서 처리할 기능들 중, 음량변경 등의 제어를 담당하는 SCA 기반 응용 컴포넌트이다[11].

다. 공용 단말 하드웨어 상에서의 동작 실험

동작 실험은 (그림 2)와 같은 환경에서 실시하였다. HSDPA 서비스 시험을 위한 동영상 플레이어와 같은 애플리케이션과 호 설정 및 해제용 GUI는 TE에서 실행되며 Waveform Manager에 의해 단말 하드웨어에 실장된 HSDPA 모드의 응용컴포넌트와 연동하여 발신호 처리 절차를 거친 후 streaming server를 이용한 동영상 서비스 시험을 수행하였다. HSDPA Node-B와 UMTS 망 기능은 일본 Anritsu사의 MD8480C 시험장비를 이용하여 실시하였다.



(그림 2) SCARLET 상에서의 단말 애플리케이션 동작 실험 환경

T-DMB 서비스 시험을 위한 방송수신 플레이어와 같은 애플리케이션은 TE에서 실행되며 Waveform Manager에 의해 단말 하드웨어에 실장된 T-DMB 모드의 응용컴포넌트와 연동하여 T-DMB 기지국에서 방송중인 T-DMB 방송 동영상을 SDR AT의 T-DMB 수신용 안테나를 거쳐 TE에 위치한 T-DMB 방송 수신용 플레이어를 통해 수신하는 시험을 수행하였다.

실험 결과에 따르면, 재구성 시간(단말기가 새로운 모드로 전환하기 위해 통신모드 애플리케이션을 재구성 가능한 하드웨어상에 설치하고 시작될 때까지 소요되는 시간)은 19초 정도가 소요되었다. T-DMB 서비스는 모든 채널이 정상적으로 수신되어 동작하였으며, HSDPA는 계측 장비를 통해 프로토콜 규격에 적합한 호처리를 확인하였으며, 외부 망과 연동한 데이터 서비스를 제공하였다.

단말기는 기지국에 비하여 컴퓨팅 리소스가 제한적일 수 밖에 없으므로 SCARLET이 사용하는 CORBA 미들웨어, XML 파서 등의 추가적인 경량화, 최적화 개발을 통해서 재구성 시간을 좀 더 합리적인 시간 범위 내로 단축하려는 노력이 필요한 것으로 판단된다[11].

V. 결론

본 고에서는 무선 데이터 통신 서비스인 HSDPA와 T-DMB 프로토콜 규격을 만족하는 통신 애플리케이션 컴포넌트를 설계 및 구현하였다. 범용 프로세서 기반의 공용 단말 하드웨어 상에서, SDR 기술의 핵심 중의 하나라고 할 수 있는 단말 미들웨어 플랫폼과 함께 통신 애플리케이션 재구성 기능에 대한 구현을 통해 상용화의 가능성을 검증하였다. 종래의 ASIC 기반 무선 통신 단말은 규격이 개정되거나 새로운 규격이 출현하는 경우 새로운 칩을 출시하기까지 많은 시간과 비용이 요구되는 어려움이 있었으나, 본 SDR 단말에서는 통신 애플리케이션 소프트웨어의 변경만으로 언급한 문제가 해결되는 장점을 가진다.

본 고에서는 SDR 단말 소프트웨어를 구성하고, 미들웨어 및 통신 애플리케이션의 컴포넌트화 구현과 관련하여 다음과 같이 접근하였다.

SCA에서 제공하는 규격을 근거로 구현된 미들웨어 플랫폼 상에서 SDR 단말용 통신 애플리케이션 컴포넌트를 설계 및 구현하기 위해, 우선 현존하는 다양한 통신 서비스들을 분석하여 이들 중 현실성,

활용 가능성 그리고 개발 용이성 등을 고려하여 양방향 통신 서비스를 포함한 HSDPA, T-DMB와 Mobile WiMAX의 조합을 선정하였다. 선정된 서비스들을 수행하기 위하여 기존의 프로토콜들을 SCA wrapper와 SCA adapter를 사용하여 SCA의 기준에 맞도록 설계 및 변경하였고, 이들과 미들웨어 간의 연동을 위하여 assembly controller를 설계 및 구현하였다. 이에 따라, 현재 HSDPA와 T-DMB는 개발이 완료된 상태이고, Mobile WiMAX는 개발이 진행중이다. 또한 세 가지 모드의 통신 애플리케이션 컴포넌트들이 모두 수행될 수 있는 SDR 공용 단말 하드웨어도 구현하였다. 개발된 통신 애플리케이션 컴포넌트들은 공용 단말 하드웨어와 미들웨어 상에서 원활히 수행됨을 확인하였으며, Core Framework의 지시에 따라 재구성 동작이 원활히 수행됨을 확인함으로써 상용화 가능성을 검증하였다.

약 어 정 리

AEP	Application Environment Profile
AT	Access Terminal
CORBA	Common Object Request Broker Architecture
DCD	Device Configuration Descriptor
DMD	Domain Manager Descriptor
DPD	Device Package Descriptor
DSP	Digital Signal Processor
FPGA	Field Programmable Array
GPP	General Purpose Processor
GUI	Graphic User Interface
HCI	Human Computer Interaction
HSDPA	High-Speed Downlink Packet Access
IDL	Interface Definition Language
JPEO	Joint Program Executive Office
JTRS	Joint Tactical Radio System
MMITS	Modular Multifunction Information Transfer System
POSIX	Portable Operating System Interface
PRF	Properties Descriptor
RTOS	Real-Time Operating System
SAD	Software Assembly Descriptor
SCA	Software Communication Architecture
SCARLET	SCA ReconfigurabLe middleware of ETRI
SCD	Software Component Descriptor
SDR	Software Defined Radio
SDRF	SDR Forum
SPD	Software Package Descriptor
T-DMB	Terrestrial - Digital Multimedia Broadcasting
TE	Terminal Equipment
UMTS	Universal Mobile Telecommunications System

참 고 문 헌

[1] Bill Schilit, Norman Adams, and Roy Want, "Context-aware Computing Applications," *In Proc. of IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, California, Dec. 1994, pp. 85-90.

[2] Jeffrey H. Reed, Software Radio: A Modern Approach to Radio Engineering, Prentice Hall Ptr., 2002.

[3] 김지연, 김진업, "차세대 이동통신시스템을 위한 SDR 기술," *IT Standard Weekly*, 2002. 5.

[4] 김덕배, 김진업, "SDR 기지국 구조/기술," 대한전자공학회 학회지, 제33권 제2호, 2006. 2., pp.29-39.

[5] Walter Tuttlebee et al., Software Defined Radio: Origins, Drivers and International Perspectives, John Wiley & Sons, 2002.

[6] ANSI/IEEE, POSIX 1003.13: Standardized Application Environment Profile - POSIX Realtime Application Support(AEP), IEEE Std. 1003.13, 1998.

[7] JTRS, Software Communication Architecture Specification(Ver. 2.2.2), May 2006.

[8] 김세화, 유종훈, 홍성수, "SDR(Software Defined Radio)의 소프트웨어 구조," 대한전자공학회 학회지, 제33권 제2호, 2006. 2., pp.17-28.

[9] OMG, Minimum CORBA Specification Version 1.0, Aug. 2002.

[10] 김홍숙, 오상철, 박남훈, 김진업, "SDR용 미들웨어 구조: SCA 코어 프레임워크," 전자통신동향분석, 제21권 제3호, 2006. 6., pp.71-78.

[11] 김홍숙, 김준식, 오상철, 박남훈, 김진업, "SCA 기반 SDR 단말기용 통신 컴포넌트 구현," 한국통신학회지, 제24권 제7호, 2007. 7., pp.47-58.

[12] Harri Holma and Antti Toskala, HSDPA/HSUPA for UMTS, John Wiley & Sons, June 2006.