특 집

# SDR 소프트웨어 구조 및 다운로드*

이 헌 우, 정 상 국, 김 한 경
창원대학교 컴퓨터 공학과

## 요 약

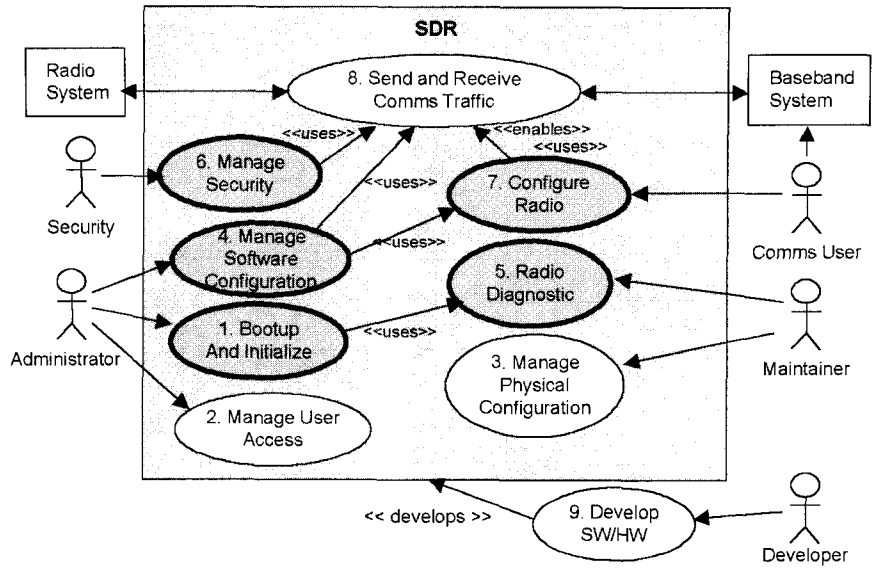
이동통신서비스 기술이 서비스 속도 측면에서의 차별화로 인하여 이동 통신 단말기 구축 기술 스펙트럼이 다양하게 확대되어감에 따라 이를 지원하는 기술 사이에 호환성에 대한 필요성이 대두되고 있으며, 이를 지원할 기술로 소프트웨어에 의한 단말기 형상의 재구성을 고려하게 되었다. 이를 Software Defined Radio(SDR)라고 하며, SDR의 핵심을 이루는 기술 중의 하나로 소프트웨어 다운로드 기술이 요구된다. SDR Forum에서 소프트웨어 다운로드 기술을 표준화하기 위한 작업이 진행되고 있으나 아직 확정된 바가 없다. 소프트웨어 다운로드 중에서도 global roaming의 기반이 되는 Over-the-air(OTA)에 의한 형상 재구성 방안은 각 단말기가 갖는 다양성과 다운로드 과정의 복잡성으로 인하여 요구사항을 정하는 것이 쉽지 않다. 이에 다운로드 프로토콜의 진화와 변경을 고려하여, 소프트웨어의 구조와 다운로드 프로토콜을 제시한다.

## I. 서 론

이동통신 시장의 성장과 통신기술의 발달은 네트워크에 대한 단말기의 유연한 적응성을 요구하고 있으며, 다중모드(multi-mode), 다중대역(multi-band), 다중기능(multi-function) 단말기의 특성을 지원할 수 있는 해법으로 SDR (Software Defined Radio) 기술이 대두되고 있다. 이에 대한 연구를 위하여 SDR Forum이 발족되었고, SDR Forum에서는 WAP(Wireless Application Protocol) Forum과 ETSI SMG4-MExE(Mobile Execution Environment)와 함께 프로토콜 스택의 상부로부터 소프트웨어 다운로드, 보안과 암호화를 위한 표준화 작업을 하고 있으며, SDR은 프로토콜 스택의 하부구조로부터 소프트웨어 다운로드와 재구성력을 표준화하기 위한 일을 진행중이다.

SDR 기술에 대한 추세는 기술 표준을 결정하기 위한 노력 위주의 작업이 진행 중인데 대학 및 벤처 수준의 기업에서 Radio 하드웨어 분야에 대한 개발이 활발한 상황이다. 2001년 9월에는 ACT(Advanced Communications Technologies)사의 기술연구소에서 소프트웨어 기술인 SDR 기술을 사용한 기지국과 기존 CDMA 이동전화기를 이용해 IS-95B CDMA 프로토콜 방식의 이동전화 통화를 성공하였다. 또한, 영국에서는 현재 Surrey 대학교 이동통신시스템연구센터에서 종합적인 연구가 활발하게 진행되고 있으며, 소프트웨어 아키텍처, 프레임워크를 자체적으로 설계하였으며, 이에 따라 다운로드 기능을 실험실 수준으로 개발하는 추세이다. 한편, 일본의 NTT에서는 PHS 프로세스를 바탕으로 SDR용 멀티프로세서 프로토타입을 실험실 수준에서 개발하였으며, 교또 대학, 요코하마 대학 등지에서도 하드웨어 관련 분야 연구를 진행 중이다. 그리고 대만은 정부 출연연구소인 ITRI의

* 2002년도 한국전자통신연구원과 창원대학교의 연구비 지원에 의한 연구 결과의 일부임




〈그림 1〉 SDR use case 다이어그램

CCL에서 Java EJB(Enterprise Java Beans) 기반의 소프트웨어 개발을 진행 중이며, CORBA 및 SDR 아키텍처를 아직 채택하고 있지는 않은 것으로 판단되어지고 있다.

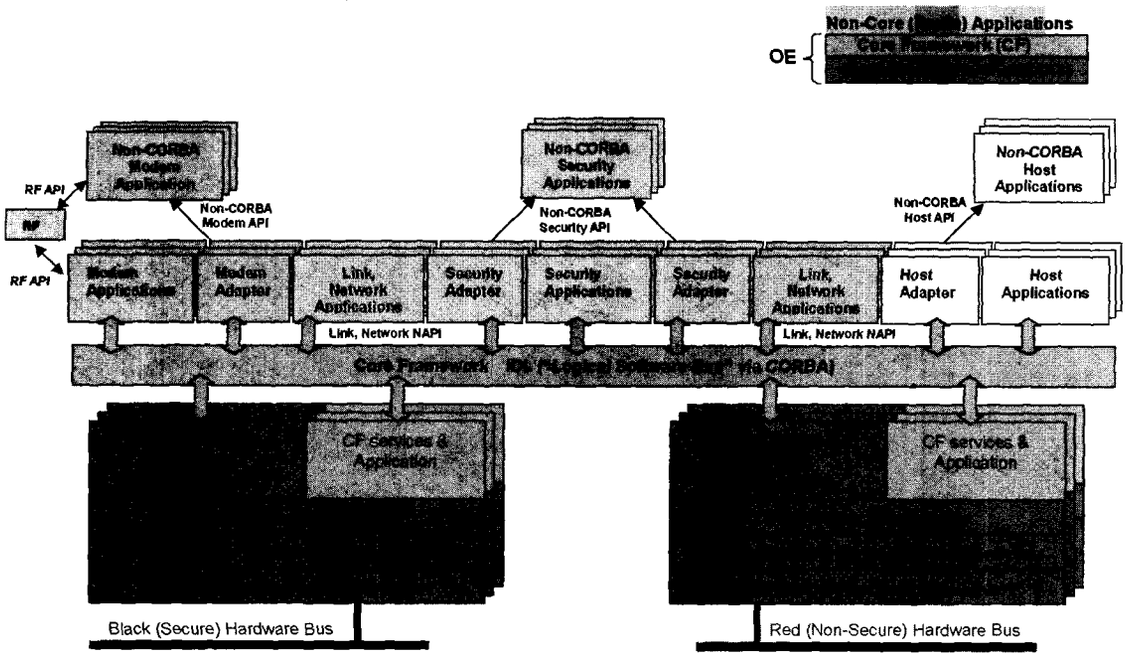
SDR Forum에서는 SDR 단말기를 규격화하는 기법으로 객체지향 방법론을 도입하여, 기술 표준화 작업을 UML을 이용하여 표준화를 추진하고 있다. 특히 미들웨어로는 OMG(Object Management Group)의 CORBA(Common Object Request Broker Architecture)를 채택한 바가 있으며 이들의 기반 위에서 단말기 형상 재구성 응용 소프트웨어가 수행되는 시스템 계층 구조를 제시하였다. 본 고에서는 SDR Forum에서 제시하는 각종의 기술 규격과 draft를 통해 SDR 다운로드 소프트웨어를 중심으로 소프트웨어 구조 및 다운로드 기술에 대하여 고찰한다.

〈그림 1〉에 단말기 시스템이 제공하여야 할 어플리케이션 범위를 SDR 단말기에 대한 유스 케이스(use case) 뷰(view)에 의해 보여준다. 기본적으로 8개의 유스 케이스를 가지고 있는데, 이중에서 유스 케이스 8번으로 표시된 트래픽 서비스는 어플리케이션 소프트웨어에 의해 제공되는데, 트래픽 서비스 어플리케이션 소프트웨어의 정상적인 작동을 온라인으로 제공해 주기 위하여 나머지 유스 케이스들이 필요하며, 이들 모두와 이들이 운용될 플랫폼이 개발의 대상이 된다.

## II. 소프트웨어 구조

SDR 아키텍처가 가져야 하는 기본 개념은 개방성, 분산성, 객체지향성, 소프트웨어 제어성이다. 즉, SDR 소프트웨어 구조는 개방성을 유도함으로써, 상용(COTS, Commercial Off-The-Shelf) 소프트웨어의 사용으로 야기될 수 있는 소프트웨어 자원 사이의 상호 연결성을 보장할 수 있는 체제를 가져야 한다. 이러한 개념을 표현한 것이 〈그림 2〉의 SDR 아키텍처 모델이다.

이러한 아키텍처로 구성되는 SDR 시스템은 내장형 무선 응용 시스템이고 자원 제약성을 가지고 있으며 실시간 작업 처리 및 동시 처리를 필요로 한다. SDR의 특징이 하드웨어가 수행하는 기능을 소프트웨어로 대체하므로 이를 위해서는 SDR 소프트웨어를 크게 SDR 무선 응용 기능을 담당하는 응용 소프트웨어와 이를 지원하는 소프트웨어 플랫폼 계층으로 나눌 수 있다.




〈그림 2〉 SDR 아키텍처

## 1. SDR 소프트웨어 플랫폼

SDR은 소프트웨어의 재구성(recon-figuration)이 핵심이므로, SDR을 지원하기 위해 시스템 소프트웨어는 첫 번째로 SDR에서 사용되는 다양한 하드웨어를 추상화시켜야 하며, SDR 응용을 언어에 무관하게 개발할 수 있게 지원하여야 한다. 하드웨어의 추상화는 응용 소프트웨어를 통하여 달성하고 응용 소프트웨어는 API를 통하여 운영체제와 다른 응용 소프트웨어와 인터페이스되게 한다. 특히 SDR Forum에서 CORBA를 미들웨어로 적용하여 분산 어플리케이션 환경을 지원하기로 결정한 바가 있다 이는 SDR 응용이 재구성될 수 있도록 컴포넌트 기반 컴퓨팅을 지원하여야 하는데 이를 위해서 응용 소프트웨어와 RTOS(Real Time OS) 사이에 위치하는 내장형 미들웨어가 그 기능을 담당한다. 미들웨어는 클라이언트-서버 형태의 분산처리 동작을 표준화하는 방안으로 CORBA 프레임워크를 사용하여 분산처리를 지원한다. 이들 시스템 소프트웨어들은 실시간 제약 조건이 만족될 수 있도록 SDR 무선 응용을 처리하여야 하며, SDR 내장형 시스템에 사용될 수 있도록 유연성 및 경량성을 갖추어야 한다.

SDR 단말기의 소프트웨어를 표준화하기 위해서는 구현에 독립적인 시스템 프레임워크를 제시하고, 이를 기준으로 시스템에 대한 기준 요구사항을 제시하는 것이 바람직하다. 이러한 요구사항은 인터페이스 규격, API 및 동작 규격 등으로 나타난다. 먼저 플랫폼을 구성하기 위해서 플랫폼의 핵심인 OE(Operating Environment)를 보면, 〈그림 2〉에서 볼 수 있듯이 하드웨어 플랫폼 위에 운영체제, CORBA 미들웨어, 코어 프레임워크, AP(Application)와 같은 요소들로 이루어진다.

### 1) 운영체제

응용 프로그램의 이식성과 상호 운용성을 보장하기 위해서는 먼저 OS의 표준화가 요구된다. OS의 표준화는 OS 인터페이스의 표준화가 필요하며 이를 위해 IEEE에서 POSIX(Portable Operating System Interface)에 의한 산업계 표준화가 이루어진 바가 있다. SDR에서는 POSIX 1003.13 API를 사용하도록 권고하고 있으며, 여

〈표 1〉 POSIX 1003.13 API 실시간 시스템 프로파일

| 종 류 | 특 징 |
|---|---|
| Minimal Realtime System Profile(PSE51) | 기본적으로 로컬 메모리를 갖는 단일 프로세서 구조를 갖는 하드웨어상에서 동작하며, MMU 및 파일 시스템은 지원되지 않는다. 멀티태스킹 기능은 단일 프로세스에 멀티 쓰레딩으로서 지원된다. |
| Realtime Controller System Profile(PSE52) | PSE51의 확장으로 PSE51에 파일 관리 기능과 비동기 I/O 처리기능이 추가 지원된다. SDR 응용 환경 프로파일의 기초를 두고 있다. |
| Dedicated Realtime System Profile(PSE53) | PSE52와 마찬가지로 PSE51의 확장이지만 다중 프로세서 구조와 공통 인터페이스, 메모리 관리기능이 지원된다. |
| Multi-Purpose Realtime System Profile(PSE54) | 상위 프로파일의 모든 기능 위에 고속의 외부 기억장치, 네트워크 기능, 디스플레이 기능 등이 추가된다. |

기에는 실시간 처리 서비스를 위한 멀티태스킹, 메모리 관리, 파일 시스템 등의 기능 제공 여부와 정도에서 차이를 나타낸다. POSIX 1003.13 API 실시간 시스템 프로파일은 총 4개(PSE51, PSE 52, PSE53, PSE54)로 구성된다(참고 〈표 1〉).

2) 미들웨어

응용 객체의 분산처리를 위하여 CORBA를 사용하는데, SDR의 소프트웨어 표준인 SCA (Software Communication Architecture)에서는 다양한 CORBA 표준 중에서도 minimum CORBA를 선정하였다. CORBA는 그 근간이 되는 Object Broker와 어플리케이션에 서비스를 제공해 주기 위한 코어 프레임워크(CF, Core Framework)로 구성된다. 용어가 암시하듯이 응용 소프트웨어를 위한 핵심 부분인 CF는 SDR 기능들을 구축하기 위해서 필수적으로 포함되어야 한다. 또한 응용 소프트웨어가 기능을 발휘할 수 있도록 객체 사이의 통신을 지원하고, 객체를 관리하며, 객체 상호 관계와 객체의 서비스 유무 상태를 처리한다. CF는 시스템에 포함되는 컴포넌트들 즉, 하드웨어와 소프트웨어 자원에 대한 인식 및 설정을 위하여 XML(Extensible Markup Language)을 사용하여 표기한다. CF가 외부로는 인터페이스와 프로파일에 의해 추상화 되는데, 인터페이스는 OMG의 IDL(Interface Definition Language)을 이용하여 명세화한다. CF 인터페이스는 〈표 2〉와 같이 분류된다.

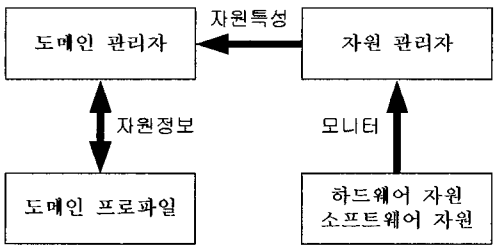
〈표 2〉 CF 인터페이스의 종류와 의미

| 인터페이스 종류 | 관련 객체 | 의 미 |
|---|---|---|
| CORBA 기본 인터페이스 | LifeCycle, StateManagement, Message, Resource, ResourceFactory, PropertySet, MessageRegistration, | 소프트웨어 응용 객체 사이의 정보 교환을 위하여 공통적으로 사용되는 일련의 인터페이스를 제공 |
| 프레임워크 제어 인터페이스 | Application, ApplicationFactory, DomainManager, ResourceManager | CORBA 인터페이스를 통하여 시스템 시동, 제어, 응용 소프트웨어 부품의 제거, 하드웨어 할당 및 제어를 위한 인터페이스를 제공 |
| 프레임워크 서비스 인터페이스 | File, FileSystem, FileManager, StringConsumer, Logger, Installer, Timer | 응용 소프트웨어 객체에게 이벤트 로깅 서비스를 제공하고 또 분산된 파일의 접근을 지원하는 인터페이스를 제공 |
| 선택적인 프레임워크 인터페이스 | Factory, Adapters | core 및 non-core 응용 기능들의 life span을 제어하는 인터페이스 |

## 2. 어플리케이션 소프트웨어

하드웨어 자원들을 제어하기 위한 비핵심(non-core) 어플리케이션은 자원 인터페이스나 NAPI 인터페이스로 구체화 된다. 어플리케이션 개발자들은 그 어플리케이션을 위한 별도의 자원 인터페이스를 생성할 수 있다. 현재 SDR에서 정의한 자원은 모뎀 자원, 링크 자원, 네트워크 자원, 보안 자원, 유틸리티 자원이다. 이러한 자원의 내부 동작은 어플리케이션 개발자가 임의로 정하여 구현할 수 있지만 인터페이스는 〈표 2〉에 보여주는 객체에 맞추어 정의된다. 즉, CORBA 미들웨어와 미들웨어에서 제공해 주는 서비스와 퍼실리티들을 CORBA IDL을 통하여 어플리케이션에 접근하게 된다. 〈그림 2〉에 보여지는 것처럼, CORBA와 연동되지 않는 어플리케이션을 위하여 관련 Adapter를 추가할 수도 있다.

CF를 구성하는 핵심적인 객체는 프레임워크 제어 인터페이스(Framework Control Interfaces)의 도메인 관리자(Domain Manager) 객체와 자원 관리자(Resource Manager) 객체, 소프트웨어 자원(Software Resource) 객체이다. 여기에서 소프트웨어 자원이라 함은 분산처리 환경에서 구축된 각종 소프트웨어 부품 즉, 소프트웨어 객체, 프로세스, 스레드 등을 지칭하는데 2가지 종류로 구분할 수 있다. 하나는 하드웨어를 제어하는 소프트웨어 자원으로써 무선 장비 내의 모뎀, 접속장치 등과 같은 각종 하드웨어 요소를 제어하고, 다른 하나는 사용자들에게 서비스를 제공하기 위한 것들로써 데이터 링크 계층이나 네트워크 계층의 소프트웨어들이 여기에 속한다.




〈그림 3〉 서비스 제어 구조

하드웨어 및 소프트웨어 자원들의 상호 운용성을 제공하기 위해서는 자원들을 관리하는 시스템 메커니즘이 필요하다. 자원들이 일반적인 형태를 갖추도록 하여 분산 환경에서 그 위치에 영향을 받지 않도록 구현하고, 자원을 구현한 모듈 또는 객체가 변경이 되어도 CF 코드의 변경을 초래하지 않도록 한다. 이러한 요구사항을 충족시키기 위해 〈그림 3〉과 같이 도메인 관리자 객체와 자원 관리자 객체를 두어 자원에 대한 형상관리 및 통제가 이루어지도록 한다.

### 1) 도메인 관리자 객체

도메인 관리자 객체는 다중 모드 응용에서 각각의 응용에서 필요한 자원들을 할당하는 일을 책임지고 수행한다. 이를 위해 도메인 관리자 객체는 도메인 프로파일을 이용하는데, 여기에는 운용 모드에 따라 필요한 하드웨어 및 소프트웨어의 자원 목록과 각각에 대한 특성을 유지한다. 또한, 이 객체는 호스트(Host) 기능과 등록(Registration) 기능으로 분류가 가능하다. 호스트 기능은 주로 Radio 형상관리, Radio 능력관리, 소프트웨어 자원의 관리, Radio 상태 정보 제공에 관한 것이다. 등록 기능은 자원 관리자 객체에 관계되는 기능을 제공한다. 예를 들면 시스템의 능력에 대한 정보를 요구하거나 보고하는 일 등이다.

사용자는 HCI 인터페이스를 통하여 Radio 응용 서비스를 시작, 종료, 재구성 및 관리할 수 있다. 사용자가 이 인터페이스를 통해 요구한 응용 서비스 기능이 인식되면 도메인 관리자 객체는 도메인 프로파일에 있는 정보를 이용하여, 필요한 장비들을 할당하여 구성(configure)함으로써 해당 응용 서비스가 이용할 수 있도록 한다. 만일 요구한 장비들이 사용 가능한 상태이면서 동작 가능 상태이면, 도메인 관리자 객체는 소프트웨어 자원을 수행시켜 동작 모드에 대한 서비스가 제공되도록 한다. 소프트웨어 자원이 수행되기 위해서 자원 관리자 객체의 인터페이스를 이용하여 소프트웨어 자원을 해당 프로세서에 로딩하고, 그 결과를 보고 받아 사용자가 요구한 응

용 서비스가 제공 가능한지를 결정한다.

한편으로 도메인 관리자 객체는 자원 객체의 상태 변화를 관리한다. 이는 상태 관리(State Management) 객체의 인터페이스를 이용하여 수행하는데, 상태 정보가 도메인 프로파일에 기록된다. 도메인 프로파일에는 생성된 자원과 각 자원의 현재 상태와 다음 전이될 상태 등에 대한 정보가 제공된다.

2) 도메인 프로파일(Domain Profile)

도메인 프로파일은 도메인 관리 객체가 정보를 저장해 놓고 각종의 다양한 기능, 예를 들면 Radio 시스템 시동, CF 부품 및 소프트웨어 자원의 초기화, 시스템의 재시동, 자원의 할당 및 회수와 같은 일을 수행할 때 참조한다. 또한 도메인 프로파일에는 소프트웨어 자원이 파일 시스템에 저장이 되는데 현재 저장되어 있는 소프트웨어 자원의 형상번호 및 버전번호와 그에 필요한 하드웨어 및 소프트웨어 자원 정보, 파라미터 등이 보관되기도 한다. 도메인 프로파일에는 응용 서비스에 대한 정보도 포함된다. 응용 서비스란 Waveform, 네트워크, 라우팅에 관련되는 기능, 사용자 서비스 등을 지칭한다. 이런 응용 서비스가 여러 형태의 SDR 제품 사이에 이식성을 갖도록 개발하여 개발 및 유지보수 비용을 절감할 수 있게 한다. 도메인 프로파일에 있는 정보를 이용하여 도메인 관리 객체가 네이밍(Naming) 또는 트레이딩(Trading) 기능을 수행하기도 한다. 이는 자원을 획득하는 일로써, 이것은 Factory 객체를 통해 이루어진다.

3) 자원 관리자 객체

자원 관리자 객체는 하드웨어 장치들을 부팅하거나 초기화 시키고 또 상태를 도메인 관리자 객체에 보고하기 위하여 SDR에 구현되는 CF 객체이다. 하드웨어 장치들의 관리를 일반화시키기 위하여 공통되는 특성을 정의하였으며, 각 모듈별로 추가 특성을 선언할 수 있다. 자원 관리자 객체의 또 다른 역할은 도메인 관리자 객체에 의해 사용할 자원이 통보되면, 그 하드웨어 자원을 구동할 수 있는 소프트웨어 자원을 로딩하고 수행시키는 기능이다. 시스템을 시동할 때에, 자원 관리자 객체는 도메인 관리자 객체가 도메인 프로파일에 있는 정보를 보고 지시해 주는 대로 Logger, FileManager, FileSystem 및 기타 자원에 대한 객체를 생성한다.

## Ⅲ. 소프트웨어 다운로드 기능

SDR 기술을 적용한 통신 시스템은 이전의 하드웨어에 고정되어 제공되던 기능들이 소프트웨어의 제어에 의해 기능 선택이 가능하게 됨에 따라, 네트워크의 다양한 통신 방식이나 부가 서비스에 맞추어 스스로의 능력을 재구성할 수 있다. SDR 단말기의 형상을 재구성하기 위한 새로운 응용들을 적용하는 소프트웨어 다운로드 기능은 SDR의 성공적인 전개를 위해 필수적이다. 특히 이를 무선 인터페이스를 통하여 자동으로 단말기 재구성을 가능하게 하는 OTA 다운로드 기능에 대해서 알아본다.

### 1. 다운로드 환경

SDR 기술은 소프트웨어로 구현된 대부분의 기능을 쉽게 고칠 수 있어 다양한 통신 방식이나 부가 서비스의 수행을 용이하게 할 수 있다. SDR Forum에서 정의하는 "소프트웨어 다운로드"란 SDR 장치의 동작이나 성능의 특성을 바꾸기 위한 목적으로 새로운 프로그램 코드나 데이터를 로딩하는 프로세스이다. SDR 터미널은 소프트웨어 다운로드를 통하여 각기 다른 시간에 각기 다른 기능, 프로토콜, 서비스 등을 수행할 수 있는데, 다운로드 기능을 통하여 SDR 단말기에 재구성할 수 있는 소프트웨어를 분류하는 것은 여러 가지 방안이 있을 수 있지만 SDR Forum의 소프트웨어 다운로드 분야에서는 기본적으로 〈그림 4〉와 같이 무선용 소프트웨어와 비무선용 소프트웨어, 또 기본적인 무선 소프트웨어와 보조적인 소프트웨어, 그리고 실행코드와

데이터로 분류한다.

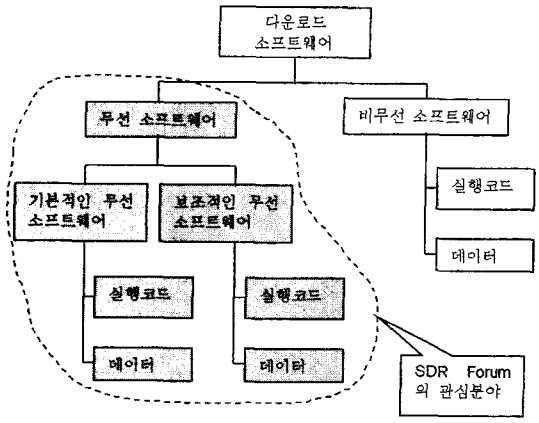
SDR Forum에서는 소프트웨어 다운로드의 연구 대상으로 무선용 소프트웨어에 국한하고 있으나, 보안 소프트웨어, 인증 및 검증 소프트웨어 등이 무선용 소프트웨어를 다운로드하기 위해서는 필요한 소프트웨어들이기 때문에 이들 비무선 소프트웨어에 대하여서도 관심을 가져야 한다.

## 2. 다운로드 프로토콜

SDR Forum에서 WAP Forum과 MExE의 연구 내용을 종합하여 나름대로 마련한 다운로드 프로토콜의 일반적인 절차는 〈그림 5〉와 같다. 여기에 참여하는 엔티티는 SDR 서버와 SDR 단말기, 인증 서버가 있다.

### 1) 초기화 단계

다운로드 절차의 시작을 시도하는 주체는 SDR 장비나 네트워크가 될 수도 있고, 또는 SDR 장비를 사용하는 여러 유형의 사용자, 서비스를 관리하는 기술자가 될 수도 있다. 소프트웨어 다운로드 절차를 시작하기 위하여 SDR 장비는 현재 지원할 수 없는 서비스를 SDR 장비 사용자가 요구하는 경우에 대한 대처 능력이 필요하다. 예를 들어, 음성 송수신 능력 밖에 없는 SDR 장비를 이용하여 패킷 데이터의 송수신 서비스를 요청하는 경우에 "형상 재구성 관리 엔티티(reconfiguration management entity)"는 사

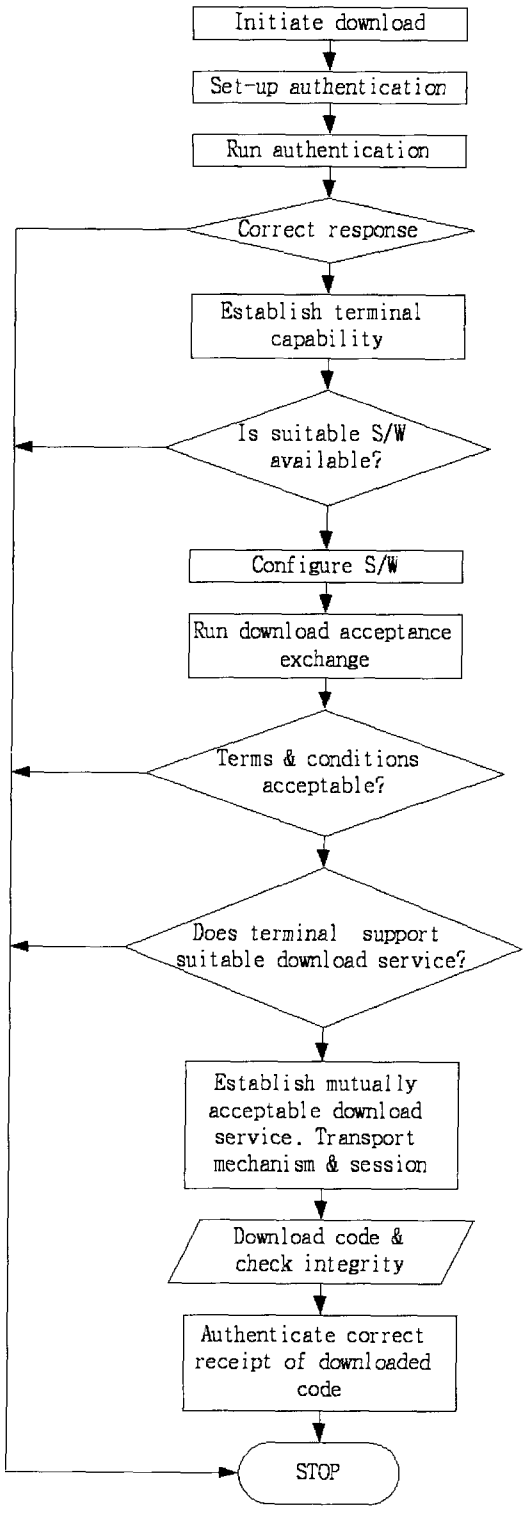



〈그림 4〉 다운로드할 소프트웨어의 분류

용자의 요구를 처리하기 위하여 다운로드 절차의 시작을 사용자로부터 확인을 받아야 한다. 만일 사용자가 확인을 해주면 다운로드 절차를 시작한다.

네트워크가 주체가 되어 소프트웨어를 다운로드 하게 되는 경우는, 네트워크에서 SDR 장비에 새로운 버전 또는 업그레이드 된 소프트웨어의 적용해야 하는 것으로, 이러한 필요성이 발생하면, 소프트웨어 다운로드 계획을 수립한다. 이러한 경우의 발생은 SDR 장비에서 수행되고 있는 소프트웨어의 버그를 수정하기 위한 것일 수도 있고, 수행 중인 소프트웨어를 개선한 새로운 버전을 적용하기 위한 것일 수도 있으며, 아니면 SDR 장비에게 새로운 서비스나 어플리케이션 능력을 가질 수 있도록 소프트웨어를 제공하기 위한 것일 수도 있다.

소프트웨어 다운로드 절차가 사용자에 의해 수동적으로 시도될 수도 있다. SDR 장비 사용자는 유선을 이용하든 무선을 이용하든 네트워크 서버와 연결을 설정하여 다운로드 절차를 수동적인 방법으로 시작할 수도 있다. 따라서 다운로드 절차는 서버나 클라이언트 양쪽 모두에서 시도할 수 있다. 이러한 시도는 get_SDRF_id() 메시지를 발송함으로써 절차가 시작되며, 이 메시지에 의해 원격 장치는 자신이 SDRF 다운로드 API를 지원할 수 있는지 확인해 준다. get_SDRF_id() 메시지를 송신한 다음, 2가지의 경우를 생각해 볼 수 있는데 첫째는, SDR-ID 메시지를 접수하게 되는 경우로써 다운로드를 계속할 것인지 여부를 나타내는 정보와 함께 상태(status) 정보를 반환 받는다. 둘째는 상대방 장치가 SDRF 능력을 가지고 있지 않아서 SDR-ID 메시지에 대한 응답이 없는 경우이다. 그러므로 get_SDRF_id() 메시지를 송신할 때 time-out을 걸어야 하며, 응답 메시지가 없으면 재송신을 몇 번까지 시도할 것인가를 결정하여야 한다. enable() 메시지는 get_SDRF_id() 메시지의 전후에 주고 받을 수 있지만 권고사항은 get_SDRF_id() 메시지에 의해 다운로드 가능 여부를 확인한 다음에 다운로드를 위한 자원 할당을 요청하는 의도로




〈그림 5〉 다운로드 흐름도

사용하도록 하고 있다. 즉, enable() 메시지에 의해 다운로드 세션이 시작된다.

2) 상호인증 단계

다운로드 절차가 시작된 다음에 가장 중요한 것은 SDR 장비(하드웨어와 소프트웨어), 사용자, 서비스 제공자, 네트워크 사업자를 인식하는 것이다. 이것을 상호 인증이라고 하며, 다운로드 프로토콜을 불법(fraudulent)으로 사용하는 것을 방지하기 위해서 절대적으로 필요한 사항이다. SDR 장비로 다운로드 되는 소프트웨어는 대개 지적소유권을 가진 경우가 있을 수 있으며. 이런 소프트웨어에 대한 접근 권한이 부여되지 않은 자에게는 접근할 수 없도록 보호하는 것이 매우 중요하다. 이와 같은 인증은 자격을 갖추지 못한 사용자에게는 서비스가 활성화되지 않도록 하는 서비스 활성화 방지 기능에도 적용할 수 있다.

다운로드를 시작하기 위해 양방향 상호 인증 절차를 거쳐야 한다. 인증 과정에 의해 각각은 상호 상대방의 구체적인 다운로드 능력을 확인하고 이 정보를 표(table)로 작성한다. 상대방의 다운로드 능력 정보는 소프트웨어를 다운로드 받아 설치하고 수행이 성공적으로 이루어지도록 하는데 필요하다. 여기에서 제시하는 인증 절차는 일반적인 프로세스를 나타내며 다운로드 받는 측에 대한 인증은 이미 이루어진 것으로 간주한다. 즉, 이는 사용자가 SIM card를 사용할 것인지 또는 다른 어떤 매체를 이용할 것인지, 또는 사전 등록에 의해 서비스를 제공할 것인지 등에 따라 방법이 틀려진다.

주어진 절차에 따라 능력 정보와 형상 정보를 교환하고, 교환된 정보를 이용하여 알고리즘과 키를 협상에 의해 결정한다. 결정된 내용을 authenticate() 메시지를 이용하여 원격 장치에 통보하여, 원격 장치에게 새로운 알고리즘을 적용하여 수행할 것을 요구한다. 원격 장치는 이 메시지에 대한 결과와 상태를 응답 메시지로 반환한다. 장치 A가 장치 B의 형상을 인증하기 위하여 필요한 정보를 요청하고 또 새로운 형상으로 설정하기 위하여 get_config() 및 set_config()

API가 준비되어 있다. 이들은 선택적(optional)으로 사용할 수도 있다.

인증 절차가 완료된 다음에 'in progress'라는 상태 정보를 가지면서 응답 메시지를 보내지 않을 수도 있다. 이는 나중에 비동기적으로 USI (Unsolicited Status Information) 메시지를 보내어 그 결과가 포함되어 있는 형상표의 ID를 제시하게 되는데, 이에 의해 장치 A는 get_config() 메시지에 의해 장치 B를 접근할 수 있게 된다. 이렇게 되면 장치 A도 장치 B의 접근을 허용하게 된다. 능력과 형상 정보 교환을 통하여 협상된 알고리즘과 키를 정하고 난 다음 authenticate() 메시지에 의해 원격 장치에서 알고리즘이 수행된다.

3) 능력확인 단계

SDR 장비는 여러 형태와 유형으로 만들어질 것이다. 다양한 형태의 핸드셋들이 무선 장비 상용 시장에 나타날 수도 있다. 서로 다른 유형의 SDR 장비들은 각기 다른 소프트웨어와 파라미터 값을 가지고 자신의 능력을 나타내게 될 것이다. 따라서 네트워크가 SDR 장비로부터 소프트웨어 다운로드 요구를 접수하면, 맨 먼저 SDR 장비의 능력에 대한 정보를 요구하여야 한다. 이러한 능력 정보를 교환하는 것은 네트워크가 SDR 장비로부터 요구 받은 소프트웨어의 다운로드를 위해 그 장비에 해당하는 올바른 소프트웨어 엔티티와 파라미터 값을 설정하는데 필요하다. 이는 SDR 장비가 다운로드 받은 소프트웨어를 수용하고, 설치하고, 성공적으로 수행이 되는지를 확인하는 데에도 필요하다.

능력정보 교환은 2 단계로 이루어진다. 첫번째 단계는 네트워크에서 SDR 장비로 능력 정보를 요구하는 것이고 두 번째 단계는 첫번째의 요구에 의해 SDR 장비로부터 네트워크에게로 능력 정보를 전달하는 것이다. 능력 정보의 내용은 〈표 3〉과 같다

네트워크는 〈표 3〉의 정보를 접수하면, SDR 장비의 능력에 맞는 소프트웨어 엔티티와 파라미터를 선정하고, 이들을 SDR 장비에게로 다운로드 한다. 만일 네트워크 서버가 능력 정보에 맞는 소프트웨어 엔티티나 파라미터를 찾지 못하면, 다운로드 절차는 종료된다.

〈표 3〉 능력 정보

| |
|---|
| 현재의 SDR 장비 형상 정보 |
| 승인된 유형 데이터 |
| 적용할 API 형상번호 정보 |
| 사용중인 하드웨어 자원 정보 |
| 사용중인 소프트웨어의 프로파일 |
| 사용중인 컴파일러와 운영체제 정보 |
| 보유중인 라이선스 정보 |
| 기타 |

다운로드 서버는 터미널의 능력을 2가지 관점에서 설정한다. 첫째, 터미널의 능력에 맞추어 어떤 소프트웨어 모듈과 파라미터를 다운로드하여 줄 것인가를 결정하고, 둘째, 다운로드 서버와 터미널 사이의 다운로드 채널을 설정하는 것과 같은 터미널에서 지원되는 다운로드 모드와 애트리뷰트를 파악하기 위하여 터미널의 능력을 설정한다. 이러한 2가지 관점의 터미널의 능력 정보는 capability_exchange() 메시지에 의해 수집 가능하다. 선택된 다운로드 애트리뷰트와 파라미터는 set_config() 메시지에 의해서 설정되고, 새로운 정보는 select_config() 메시지에 의해 선택할 수 있다.

4) 다운로드 수락정보 교환 단계

실제적인 소프트웨어 다운로드를 시작하기 전에, 다운로드 서버와 SDR 장비 사이에 소프트웨어 다운로드를 위한 단계 설정을 위하여 다운로드 수락정보 교환 단계를 거친다. 다운로드 수락정보 교환 단계에서는, 코드에 대한 승인된 유형, 다운로드 절차와 스케줄, 설치 절차, 요금과 라이선스 옵션, 절차 등의 정보를 "다운로드 설치 프로파일(Download Installation Profile)" 형태로 다운로드 서버가 SDR 장비측으로 제공한다. 다운로드 설치 프로파일에 포함되는 내용은 〈표 4〉와 같다.

다운로드 설치 프로파일을 접수하면, SDR 장

〈표 4〉 설치 프로파일 정보

| |
|---|
| 다운로드가 필수적인가 옵션인가 여부 |
| 다운로드 절차(점진적 또는 전체적(complete)인 다운로드) 다운로드 스케줄 |
| 설치 옵션 |
| 라이선스 및 요금 정보와 옵션 |

비나 SDR 장비 사용자들은 설치 옵션을 선택할 수 있으며, 소프트웨어 다운로드에 대한 조건을 수락 또는 거절할 수 있다. 다운로드 수락정보 교환의 마지막 단계는, 다운로드 서버가 SDR 장비나 SDR 장비 사용자들이 선택한 옵션을 검증하는 것이다. 검증 결과 옵션의 선택이 부적절하거나, SDR 장비 또는 장비 사용자들이 소프트웨어 다운로드 조건을 거절한 경우에는 다운로드 절차를 종료한다.

다운로드 수락정보 교환 단계에서 수행하는 일은 서버가 터미널에 보내준 다운로드 설치 프로파일의 빈 항목을 기입하여 서버에게 되돌려 주는 것이다. 이와 같은 사항을 결정하는 동안 서버가 비동기적으로 동작하도록 한다. 이 과정에 사용되는 API는 이전에 이미 사용되었던 인터페이스를 활용한다.

5) 다운로드 및 무결성 시험 단계

지금까지의 작업이 성공적으로 수행이 되면, 합의된 다운로드 스케줄에 따라 다운로드 서버로부터 SDR 장비의 버퍼로 소프트웨어를 다운로드하게 된다. 다운로드하는 과정에서 소프트웨어를 다운로드하는 데 이용되는 매체에 따라 전송에러를 수정하거나 재전송하여 다운로드 되는 소프트웨어의 무결성을 보장해 주어야 한다. 다운로드가 완료되면, 다운로드된 소프트웨어에 대한 최종 통합시험(integrity test)이 수행되어야 한다. 통합시험의 예로는 CRC 등이 있을 수 있다. 통합시험이 실패하면, SDR 장비는 다운로드 서버에게 소프트웨어 재전송을 요구한다. 소프트웨어의 일부에 대해서 재전송을 요구할 수도 있고 소프트웨어 전체에 대해서 재전송을 요구할 수도 있다.

다운로드되는 소프트웨어는 코드 모듈(code modules), 능력 테이블(capability table), 설치 정보(delivery wrapper)의 3가지 부분으로 구성된다. 코드 모듈은 하나 이상의 코드 세그먼트(응용 프로그램, 프로토콜 엔티티, 기능 엔티티)와 현재의 소프트웨어를 재구성하기 위한 일련의 파라미터들을 포함한다. 능력 테이블은 다운로드 되는 소프트웨어의 운용에 필요한 SDR 장비의 자원 형상에 대한 요구사항을 기술한다. 설치 정보는 컴파일 하는데 필요한 요구사항, 실시간과 관련된 제약조건, 설치 정보 등을 포함한다. 플랫폼에 무관한 소프트웨어 다운로드를 위해서는 컴파일러가 각 장비에 기본적으로 상주하고 있어야 한다.

소프트웨어를 성공적으로 다운로드 받게 되면, SDR 장비는 안전하게 수신하였음을 확인해 주는 Acknowledge Safe Receipt 메시지를 다운로드 서버에게로 전송한다. 다운로드 서버가 이 메시지를 접수하면 소프트웨어 다운로드 절차를 종료한다. 다운로드 과정은 start() 메시지에 의해 시작되며, 다운로드의 중간 결과 또는 완료된 것을 USI status 메시지로 통보한다.

6) 설치 단계

소프트웨어가 SDR 장비에 다운로드 완료되면, 소프트웨어가 SDR 장비에 의해 액세스가 가능한 로컬 버퍼에 존재하게 된다. 다운로드 받은 소프트웨어의 설치 절차는 다운로드 서버나 SDR 장비에 의해 시작이 될 수 있다. 그러나 소프트웨어의 다운로드와 소프트웨어의 설치는 시간적인 간격을 두고 수행될 수도 있다. 또한 소프트웨어를 설치하기 전에 SDR 장비의 하드웨어와 소프트웨어 형상이 변경될 수도 있다. 이러한 여러 가지 가능성이 발생할 수 있기 때문에 소프트웨어를 설치할 당시에 SDR 장비의 형상이 설치할 소프트웨어를 수용할 수 있는지 그 가능성 여부를 확인하는 과정이 "내부 능력정보 교환(Internal Capability Exchange)" 절차이며, 이 절차에 의한 확인 작업이 SDR 장비 자체에서 이루어지도록 한다. 만일 내부적인 점검을 통해 다운로드

받은 소프트웨어를 설치할 수 없다고 판단이 되면 설치 절차는 종료된다.

내부 능력정보 교환이 성공적으로 마무리 되고 하자가 없다면, SDR 장비는 다운로드 서버에게 설치용 키(Key)를 요청한다. 다운로드 서버는 설치용 키를 넘겨주기 전에 SDR 장비와 과금 문제와 라이선스에 대하여 협상을 한다. 협상에서 SDR 장비가 요금과 라이선스 조건에 동의하면, 다운로드 서버는 설치용 키를 통보해주고, 동의하지 않는 경우에는 설치용 키를 제공하지 않으며 이에 따라 설치 절차를 종료한다. SDR 장비가 설치용 키를 통보 받으면, 다운로드 받은 소프트웨어를 설치하고 그에 따라 내부 능력 정보를 변경한다.

7) In-situ Testing

이 과정은 SDR 장비의 플랫폼에서 다운로드 받아 설치한 소프트웨어를 시험해보기 위한 것이다. 이는 소프트웨어와 함께 다운로드 받은 테스트 벡터를 이용하여 설치된 소프트웨어를 검증하고 또 수정이 된 SDR 장비가 기대한 바대로 잘 동작하는지 검증한다. In-situ testing의 책임은 장비 제조회사에 있는 것으로 예상된다.

8) 서비스 요청(Non-repudiation Exchange)

다운로드 받은 소프트웨어에 대한 설치와 설치된 소프트웨어의 동작 시험이 성공적으로 마무리 되면, SDR 장비는 "설치 성공(Successful Installation)" 메시지를 다운로드 서버에게 발송한다. 이것은 소프트웨어를 설치한 후에 최종적으로 서비스를 활성화시키기 위한 승인이 필요하거나 요금 부과가 시작되도록 하기 위하여 승인하는 절차이다.

## Ⅳ. 결 론

객체지향 개념에서 객체의 특성은 위치에 대한 구속성이 없다는 점 때문에 분산처리 환경을 구축하는 경우에 특히 효과적이다. 분산처리 환경에서 객체의 분산 처리를 위하여 OMG에서 표준으로 제시한 CORBA를 SDR Forum에서 채택하였다. 분산처리 환경에서 객체지향 분석, 설계 및 구현을 지원하는 모델링 언어인 UML (Unified Modeling Language)을 OMG에서 표준으로 채택하였으며, SDR Forum에서도 문서화 작업에 UML을 이용하기로 결정하였다.

SDR 단말기의 소프트웨어를 설명하기 위해서, SDR의 아키텍처를 하드웨어, 운영체제, CORBA 미들웨어, 어플리케이션으로 크게 4가지의 구성요소 또는 계층 구조로 구별하였으며, 이들을 구현하기 위한 시스템 개발 방법론으로 객체지향 기법을 도입하고 있다. 즉 하드웨어의 사용을 인터페이스만 맞추면 어떠한 용도로든 사용할 수 있게 유연성을 부여하는 것이다. 이러한 특성은 "플러그 앤 플레이"가 가능한 제품 능력을 갖게 한다. SDR 소프트웨어의 어플리케이션 계층은 다시 2 부분으로 분리가 되는데 첫째는 코어 어플리케이션이고, 다른 하나는 비핵심 어플리케이션이다. 비핵심 어플리케이션이라 하여 코어 어플리케이션에 비해 덜 중요하다는 의미가 아니라 어플리케이션 분야임을 나타내기 위한 의미이며, 코어 어플리케이션은 어플리케이션을 제어하거나 하위 계층에 대한 접근 방식을 표준화 시키기 위한 것이다. 코어 어플리케이션이란 비핵심 어플리케이션을 관리하기 위한 일종의 제어 객체라고 할 수 있으며, 이들에 의해 어플리케이션의 다운로딩, 상태관리 및 형상 관리가 이루어진다. 코어 어플리케이션이 가져야 할 다운로드 프로토콜에 대하여 제시 하였지만, 보안 즉, 시스템 운영 정보의 보호 방법, 서비스에 대한 지불 수단과 과금 방안 등의 문제에 대한 집중적인 검토가 계속 요구된다. 비핵심 어플리케이션은 일단 로딩되고 나면 무선 통신 하드웨어와 연동이 되면서 단말기 기능이 수행된다. 전반적인 SDR 표준화 과정에서 SDR 단말기의 프레임워크가 도출되었으며, 프레임워크를 지원하기 위하여 각 계층간의 인터페이스를 표준화 시키는 작업이 이루어졌다. 이 인터페이스는 CORBA IDL로 정의하였고, 이에

따라 SDR 단말기 벤더들의 구현 노력이 경주되고 있다.

앞으로 추진하여야 할 일은, SDR 단말기의 프레임워크 소프트웨어를 향상된 성능으로 구현하기 위한 기술적인 문제 도출과, 또 지속적으로 변화가 이루어질 사용자의 요구사항을 유연하게 수용할 수 있는 소프트웨어 다운로더 구현 및 보안 기술 개발, 제3자 소프트웨어를 수용하면서 무결성을 유지할 수 있는 소프트웨어 표준화 기술에 대한 연구가 필요할 것으로 생각된다.

## 참 고 문 헌


〔1〕 SDR Forum Techincal Document, Distributed-Object Computing Software Radio Architecture, V1.2, Sep. 16, 1999

〔2〕 황경호, 조동호, "Software Defined Radio 기술", Telecommunications Review, 제10권 1호, 2월, 2000년

〔3〕 Enrico Buracchini, "The Software Radio Concept", IEEE Communications Magazine, Sep., 2000

〔4〕 Annamarie Miller, Byron Tarver, Eric Christensen, "Architecture Foundation for Software Defined Radios", SDRF Contribution Document, Mar., 2000

〔5〕 JTRS Technical document, "Software Communications Architecture Specification", MSRC-5000SCA V1.0, May 17, 2000

〔6〕 홍성호, "SDR을 위한 RTOS와 내장형 미들웨어의 설계", 통신학회지, 제19권 11호, 11월, 2002년

〔7〕 SDRF Technical Report, Chapter 6 Software Download Implementation

〔8〕 Karl Davis, SCA 1. 0 Summary Paper, SDRF Contribution Report, Aug. 1, 2000

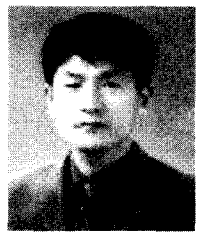
〔9〕 Klaus Moessner, Rahim Tafazolli, Software Radio Integration and Reconfiguration Management, SDRF Contribution Document(SDRF-01-I-0064w-V0.00), Oct. 27, 2001

〔10〕 Stephen M Blust, The Software Defined Radio Forum: An Overview, SDRF Contribution Document(SDRF-00-P-0000-V0.00), Nov. 7, 2000


## 저 자 소 개


**李 憲 雨**

2002년 창원대학교 컴퓨터공학과(공학사), 2002. 3~현재 : 창원대학교 공과대학 컴퓨터공학과 석사과정, 〈주관심 분야 : 객체지향 개발 방법론, UML, 멀티 에이전트, SDR〉



**鄭 尙 國**

1996년 창원대학교 전자계산학과(이학사), 1998년 창원대학교 대학원 전자계산학과(이학석사), 2002. 3~현재 : 창원대학교 공과대학 컴퓨터공학과 박사과정, 〈주관심 분야 : SDR, 이동통신, 프로토콜 공학〉

**金 漢 慶**

1973년 서울대학교 원자력공학과(공학사), 1987년 충북대학교 대학원 전산통계학과(이학석사), 1992년 충북대학교 대학원 전자계산학과(이학박사), 1983. 3~1997. 8 : 한국전자통신연구원, 책임연구원, 1997. 9~현재 : 창원대학교 공과대학 컴퓨터공학과 교수, 〈주관심 분야 : ATM, MPLS, SDR, 프로토콜 공학, 소프트웨어 공학〉