developer Works

유니버설 미들웨어 프레임워크 – OSGi

OSGi 서비스와 활용 사례

지난 시간에는 OSGi의 발전 배경과 기본 개념, 그리고 구조 등을 차례로 살펴봤다. 유니버설 미들웨어(Universal Middleware)라 불리는 OSGi는 초창기에 가정용 홈 게이트웨이의 임베디드 환경에서 출발해 지금은 모바일과 자동차, 엔터프라이즈 환경에 이르기까지 매우 폭 넓게 활용되고 있다. 이번 호에서는 OSGi의 다양한 활용 사례와 OSGi의 핵심인 서비스에 대해 집중적으로 소개한다.

2

김석우 suhgoo.kim@samsung.com | Polytech 전산학 석사. 뉴욕의 IBM 연구소를 거쳐 현재 삼성전자 선행개발팀에 근무 중이다. 빌딩 컨트롤 네트워크(Building Control Network)와 임베디드 시스템이 주요 개발 분야로, 현재는 OSGi 기반의 WSN 컨트롤러를 구상하고 있다.

OSGi의 현재 가장 큰 이슈라면 RCP와 엔터프라이즈 프레임워크(Enterprise Framework)로의 확장일 것이다. 오픈소스로 출발해서 자바 업계뿐 아니라 소프트웨어 개발자들의 기본 개발 도구인 이클립스에까지 OSGi가 탑재된다는 것은 그 자체가 큰 이슈였다. 또한 지금까지 엔터프라이즈 서버 환경에서의 적용을 감히 생각하지 못하던 차에, 스프링(Spring) 프레임워크와의 통합이 이뤄져 IT 업계에 신선한 충격을 주기도 했다. 이처럼 단순한 임베디드 또는 홈 게이트웨이 정도로만 치부되어 왔던 OSGi가 데스크탑 리치 클라이언트 플랫폼으로, 그리고 서버환경에서의 프레임워크로 발전하게 된 이유는 무엇일까? 여러 가지 이유가 있겠지만 가장 큰 이유로는 역시 가볍고(Light-Weight) 동적인(Dynamic) 모듈(Module) 시스템 구조와 번들의 라이프 사이클 관리시스템을 들 수 있다. 이 외에도 버전 업을 통해 임베디드 환경에 적용되면서 점차 기능별로 세분화된 스펙(SPEC)과 다양한 서비스 컨텐츠들을 제공한 점도 빼놓을 수 없는 이유일 것이다. 이 글에서는 OSGi의 핵심 기능인 서비스와 실제 활용 사례를 살펴봄으로써 OSGi가 더 이상 이론상으로만 존재하는(또는 임베디드에만 국한된) 시스템이 아니라 실제적이고 파워풀한 유니버설 미들웨어임을 확인하게 될 것이다.

OSGi의 가장 핵심적인 임무는 '다양하고 새로운 서비스의 제공' 이다. 과거 OSGi가 처음 형성됐을 때는 전 세계를 잇는 네트워크를 홈 네트워크에 어떻게 연결하고 어떤 서비스를 제공할지를 도출하는 것이 OSGi 멤버들의 최대 관심사였다.

OSGi의 핵심, 서비스

OSGi가 오픈 서비스 게이트웨이를 구성하는 아키텍처의 소프트웨어를 개발하는 것도 이 때문이다. OSGi 참여업체들은 이를 통해 '서비스 기반으로 제공할 수 있는 새로운 비즈니스 모델' 을 찾고 있다. 이는 단순히 제품만을 판매하는 20세기 사업모델에서 탈피해 다양하고 새로운 서비스를 판매하는 21세기형 사업모델을 만드는 것이 OSGi의 주요 업무였기 때문이다.

특히 OSGi는 네트워크 환경이 구축된 일반 가정이 차세대 프론티어 역할을 할 것으로 판단함으로써 인터넷과 신기술이 일반 가정에 새로운 서비스를 제공할 수 있을 것으로 기대하고 있었다. 이를 통해 새로운 가치사슬도 형성된다는 판단인데, 이를 위해서는 반드시 관련 업계(컴퓨터, 가전, 통신 등)가 모두 수긍하는 표준화 작업이 선행되어야 한다. 특정 업체만 지원하는 서비스나 프로토콜은 오래 지속되지 않을 뿐더러, 대규모의 시장을

창출할 수 없기 때문이다. OSGi 서비스 제공업체들은 네트워크로 연결된 가정 내 각종 기기에 접속해 기기의 이상 유무에서부터 원격 제어 및 원격 수리 등의 다양한 부가서비스를 제공하고, 이를 통해 새로운 수익을 창출할 수 있다.

이렇듯 초기의 OSGi 개발 및 참여업체들은 다양한 서비스가 가능한 게이트웨이의 표준 소프트웨어 사양 개발 작업에 착수해 현재 그 버전이 4.0대에 이르고 있다. 따라서 오늘날 OSGi에 참여해 개발을 주도한 업체들은 이미 다수의 구체적인 성과물을 제시하고 있다. OSGi가 차세대 IT 산업과 가전산업을 주도할 태풍의 핵으로 부상하고 있는 것도 바로 이 때문이다.

JES와 OSGi

OSGi는 자바 환경에서 구동되는 미들웨어이므로 자바와 매우 밀접한 관계를 나타낸다. 임베디드 자바 솔루션 중에서 자바 임베디드 서버(Java Embedded Server, 이하 JES)는 임베디드 디바이스를 위한 퍼스널 자바 기반의 런타임 환경이다. 그리고 JES는 네트워크를 통한 서비스의 제공과 서비스 라이프 사이클의 관리 기능을 제공하는 소용량 애플리케이션 서버라고 할 수 있다. 그렇다면, OSGi와 JES는 어떤 관계일까? OSGi의 표준화를 주도한 업체가 썬 마이크로시스템즈(이하 썬)라는 사실을 안다면 그 해답을 쉽게 찾을 수 있다. 즉 JES는 OSGi 표준을 구현한 임베디드 서버 솔루션 중의 하나인 것이다. JES의 구조 역시 프레임워크와 서비스로 구성되어 있다. 서비스는 JES 서버에서 동작하는 컴포넌트화된 애플리케이션이고, ServiceSpace 프레임워크는 서비스의 컨테이너이다. 프레임워크의 기본적인 기능은 서비스들의 설치와 업그레이드, 삭제 등과 같은 라이프사이클에 대한 제어이고, 서비스는 서비스 인터페이스에 구현되는 형태로 구성된다. 그리고 JAR(Java ARchive) 파일로 패키지화된 서비스를 '번들'이라고 부른다. 이런 개념들은 이미 우리가 살펴보았던 OSGi와 동일하다. JES 서버는 네트워크를 통한 서비스의 설치와 업그레이드, 삭제 등을 할 수 있지만, 처음 기동되는 경우에 몇 가지 기본적인 서비스를 포함하고 있다. 결국 OSGi의 개념과 구조는 새롭게 탄생된 것이 아니라 탄생을 주도한 썬의 영향력 아래에서 차분하게 성장한 것임을 알 수 있다. 다음은 JES에서 제공하는 기본 서비스 기능이다.

- HTTP 서비스 : 웹 서버 기능
- 로그 서비스 : 에러와 이벤트 등의 원격 로깅 기능
- 날짜 서비스
- 연결 관리 서비스 : 네트워크 서비스, 소켓 바인딩, 연결 요청 처리 등의 기능
- 스레드 매니저 서비스 : 스레드 풀의 최대치 등에 대한 스레드 관리 기능
- 스케줄러 서비스 : 향후의 이벤트에 대한 스케줄링 기능
- RMI(Remote Method Invocation) 서비스
- SNMP(Simple Network Management Protocol) 서비스
- 콘솔 서비스 : 애플릿을 통한 원격 관리 기능

이 중에서 OSGi 1.0 규격에 포함된 표준 서비스는 HTTP 서비스, 로그 서비스, 연결 관리 서비스(OSGi에서는 디바이스 액세스 서비스)이며 계속해서 다양한 분야의 스펙으로 확장해 가고 있다. 발표된 1.0 스펙은 주로 게이트웨이에 탑재되는 응용프로그램 인터페이스(API)에 대한 규정을 담고 있으며 자바(JAVA)용 API에 집중되어 있다. 이는 자바가 현재 가장 널리 사용되는 프로그래밍 언어인터라 개방형 서비스 게이트웨이에 유연하게 적용될 수 있었기 때문이다.

OSGi – SPEC

OSGi는 1.0부터 4.0대에 이르기까지 버전을 업그레이드하면서 스펙 표준안을 마련하고, 마련된 스펙이 플랫폼이나 응용소프트웨어 등에 전혀 구애받지 않는다고 발표했다. 또한 보안 기능이 우수할 뿐 아니라 다양한 서비스 제공업체들이 전달해주는 멀티 서비스를 각기 다른 장치나 설비에 제공하는 기능을 항상 염두에 두고 표준안을 마련하고 있다. 이러한 업계의 표준화 노력으로 OSGi의 스펙 표준안은 특히 블루투스(Bluetooth)와 HAVi, HomePNA, HomeRF, IEEE-1394, LonWorks, USB, VESA 등을 다양한 유무선 네트워크 기술을 수용할 수 있어 가장 포괄적인 개방형 네트워크 기술로 인정받고 있다. 특히 OSGi는 완전히 새로운 개념의 장비들이 등장할 것에 대비해 JINI와 HAVi 등이 제공하는 기능도 전폭적으로 수용한다. 이는 셋톱박스와 케이블모뎀, 라우터, 경보시스템, 전력관리시스템, 가전제품, PC 등 모든 제품에 대한 관리 및 연결 기능을 제공하기 위한 것이다.

OSGi의 주요 서비스

OSGi에서는 1.0부터 4.0에 이르기까지 수많은 서비스와 스펙들이 발표되어 사용되고 있다. 여기서는 그 가운데 핵심 서비스를 골라 살펴본다. OSGi를 구성하는 중요 구성요소는 다음과 같다.

- 서비스 : 특정 기능을 수행하는 자바 인터페이스와 실제 구현객체
- 번들 : 서비스를 제공하기 위한 기능적 배포 단위
- 프레임워크 : 번들의 라이프 사이클을 관리하는 번들 실행 환경

컨트롤 네트워크 기술

컨트롤 네트워크 기술에는 유선 형태인 IEEE1394, HomePNA와 무선 형태인 HomeRF, 블루투스 등이 있다. 또한 미들웨어로는 HAVi, JINI, UPnP 등이 있다.

● **IEEE1394** : 애플과 텍사스 인스트루먼트가 주도한 A/V 기기간의 데이터 전송을 위한 A/V 기기용 High Performance Serial Bus 기술 표준

● HomePNA(Home Powerline Network Alliance) : 기존의 전화 배선을 사용해 고속 댁내망(home networking)을 구축하는 기술 표준. 가장 저렴한 홈네트워킹 방법

● **블루투스** : 2.4GHz 주파수 대역에서 1Mbps 전송속도로 약 10m 거리 이내의 각종 컴퓨터 및 통신 단말기를 무선으로 연결하는 무선 접속 기술의 표준

● **HomeRF(Home Radio Frequency)** : 2.4GHz 주파수 대역에서 1Mbps/2Mbps의 전송속도로 약 50m 거리 이내의 각종 컴퓨터 및 가전기기들을 무선으로 연결하는 무선 접속 기술의 표준

● **HAVi(Home Audio/Video interoperability)** : 가정에 있는 네트워크를 통해 연결된 다양한 벤더들의 디지털 오디오와 비디오 장치들 사이의 상호 운용성을 제공해주는 소비자 가전 산업의 국제 표준

● JINI : 자바를 기반으로 네트워크에 접속된 디지털 디바이스간의 동적인 상호 작용을 가능하게 하는 기술 표준

● **UPnP(Universal Plug and Play)** : 기기들간의 상호접속을 위해 이뤄지는 복잡한 설정작업이나 환경설정 작업을 생략하고 각종 디바이스들이 네트워크에 접속하기만 하면 자동으로 이 디바이스들을 찾아 사용할 수 있는 기술 표준

서비스는 미리 정의된 서비스 인터페이스를 통해 접근이 가능한 컴포넌트이다. 하나의 애플리케이션은 여러 개 서비스의 협동 작업을 통해 구성되고 런타임시에 필요한 서비스를 요청할 수도 있다. 프레임워크는 각 서비스와 그 서비스에 해당하는 실제 구현에 대한 매핑을 가지고 있고, 간단한 쿼리 메커니즘을 통해 서비스의 실제 구현을 찾을 수 있다. 또한 프레임워크는 각 서비스간의 상호 의존관계를 관리한다. 번들은 여러 서비스의 구현을 하나의 패키지로 묶은 JAR 파일의 형태로 존재한다. JAR 파일에는 하나 이상 서비스의 구현 객체와 리소스 파일, 그리고 매니페스트(manifest) 파일이 포함되어 있다. 번들 컨텍스트는 프레임워크 내부의 번들 실행 환경이며 번들의 설치와 실행, 정지, 삭제 등의 번들 라이프 사이클을 관리한다. 이렇듯 서비스, 번들, 프레임워크는 항상 상호 작용하면서 OSGi를 구성하게 된다. OSGi 1.0 규격에서 명시한 패키지들은 〈표 1〉과 같다.

패키지명	설명
org.osgi.framework	OSGi 자바 서비스 프레임워크
org.osgi.service.device	OSGi 디바이스 액세스 서비스 명세
org.osgi.service.http	OSGi HttpService 명세
org.osgi.service.log	OSGi LogService 명세

〈표 1〉 OSGi 프레임워크 패키지

org.osgi.framework

OSGi 프레임워크는 소용량 메모리 디바이스에서 프로그래밍하는 개발자들이 연속적으로 동작하는 애플리케이션을 작성할 수 있는 컨텍스트를 제공하는 것이 목적이다. 애플리케이션의 swap-in과 swap-out이 런타임시에 발생하는 이러한 환경에서는 런타임시에 다른 애플리케이션과 구조적이고 의존적인 방식으로 커뮤니케이션을 수행해야 할 필요가 있다. OSGi 프레임워크는 자바 프로그래밍 언어가 가진 코드의 네트워크 이동성을 이용해 컴포넌트 기반의 개발환경을 제공함으로써 보다 풍부하고 구조적인 서비스 개발을 가능하게 한다. org.osgi.framework 패키지는 10개의 인터페이스와 6개의 클래스, 2개의 예외 클래스로 구성되어 있다. OSGi 프레임워크가 제공하고자 하는 환경의 목표는 다음과 같다.

- 애플리케이션이 실행 중에도 동적 다운로드 및 업그레이드 가능
- 제한된 메모리 디바이스 사용 가능
- 효율적이고 통합된 컴포넌트 개발 환경 제공
- 애플리케이션간의 의존성에 대한 관리 기능 제공
- 확장 가능성(scalable)

org.osgi.service.device

디바이스 매니저는 OSGi 프레임워크에서 하나의 서비스 리스너로 등록(register)된다. 디바이스 매니저는 새롭게 추가되는 디바이스의 서비스를 탐지하고, 새로 추가된 디바이스의 드라이버 번들을 설치한다. 다음은 디바이스 매니저의 알고리즘이다.

1. 디바이스 탐지(Device Detection) : 모든 새로운 디바이스(org.osgi.service.device.Device 인터페이스 객체)를 검색
2. 위치 단계(Location Phase) : 새로 추가된 모든 드라이버를 Driver Locator를 이용해 위치시킴
3. 경선 단계(Bidding phase) : 각 드라이버가 디바이스 상에서 경선에 참여
4. 추가 단계(Attach Phase) : 경선 단계에서의 최상위 드라이버 추가
5. 청소 단계(Cleanup Phase) : idle 드라이버를 청소

각 드라이버 번들은 세계적으로 유일한 스트링 아이디를 가지

고 있어야 하고, 이 아이디는 드라이버 매니저가 드라이버의 revision을 해석해 프레임워크의 위치 아이디로 사용된다. org.osgi.service.device 패키지는 3개의 인터페이스로 구성되어 있다.

org.osgi.service.http

HttpService는 프레임워크 내의 다른 번들이 리소스를 등록하고 HTTP를 통해 서블릿에 접근할 수 있게 한다. HttpService를 통해 등록할 수 있는 엔터티는 서블릿과 리소스이다. 서블릿은 Java Servlet API를 구현한 객체이며 서블릿의 등록은 URI 네임스페이스에 대한 서블릿 제어권을 부여한다. 리소스는 HTML 파일, GIF 파일, 클래스 파일 등을 포함하며 리소스의 등록은 이러한 리소스들을 URI 네임스페이스에서 사용 가능하게 한다. org.osgi.service.http 패키지는 2개의 인터페이스와 하나의 예외 클래스로 구성되어 있다

org.osgi.service.log

LogService는 번들로부터의 로그 요청을 받아들이고 LogReaderService는 다른 번들이 로그 항목을 읽을 수 있게 한다. LogService는 이벤트와 에러 상황에 대한 리포트를 주목적으로 하지만, 다른 용도로도 활용할 수 있다. org.osgi.service.log 패키지는 4개의 인터페이스로 구성되어 있다.

실행 환경의 표준 권고, OSGi - R3

2003년 4월에는 OSGi 서비스 플랫폼 릴리즈 3(OSGi R3)가 발표되었다. 필히 구현되어야 하는 표준 명세에는 가장 중심이 되는 프레임워크 명세를 포함해 모두 19개의 명세가 있고 선택적 구현이 가능하다. 또한 피드백을 위한 권고명세에는 총 4개가 포함되어 있다. OSGi R3의 가장 큰 특징은 권고명세에 포함되어 있는 JINI와 UPnP 지원일 것이다. 오디오/비디오 쪽 미들웨어 표준인 HAVi나 전력선 제어 쪽의 미들웨어와 OSGi가 연계된다면 OSGi의 활용성이 더욱 높아질 것으로 예상되었으므로, 실제 이런 시도들이 LonWorks와 같은 전력선 통신 관련 업체에서나 또는 HAVi 측과의 협력에 의해 가시화됐다. 또한 기존의 OSGi R2까지는 OSGi 서비스 플랫폼들간의 호환성이 문제였다. OSGi R3에서는 포함된 실행 환경 표준으로 인해 보다 엄격한 호환성 제공을 권고해 다양한 환경과 버전 가운데서도 유연한 실행 환경을 제공한다. OSGi R3 표준 명세(Normative Specification)의 구성 요소와 새롭게 추가된 핵심 서비스는 다음과 같다.

- URL Handlers Service Spec 1.0 : 자바에 관련된 명세로 새로운 URL 방식에 대한 URL 처리기를 등록 가능하도록 지원
- User Admin Service Spec 1.0 : 사용자의 인증(Authentication)을 처리하며 역할 정보(Role Repository)를 이용해 접근 권한에 대한 확인(Authorization) 지원
- IO Connector Service Spec 1.0 : J2ME의 Connector 프레임워크를 채용. URI(Uniform Resource Indicator) 형태로 임의의 리소스에 접근할 수 있게 해주며 리소스 접근 후에는 동일한 IO Connector Service API를 통해 제어
- Preferences Service Spec 1.0 : 데이터를 지속적으로 저장/조회/변경하는 기능 제공
- Wire Admin Service Spec 1.0 : 임의의 두 서비스를 서로 연결시켜 데이터를 주고받을 수 있게 함. Producer-Consumer 모델 형태로 서비스간의 통신을 지원
- XML Parser Service Spec 1.0 : OSGi 서비스 등록기(Registry)에 JAXP를 준수하는 XML Parser를 등록할 수 있게 함. 다른 번들에서는 서비스 등록기를 통해 등록된 XML Parser를 사용
- Metatype Spec 1.0 : LDAP과 같이 데이터를 담고 있는 통의 역할을 하는 서비스를 정의. 데이터 구조 또한 LDAP과 흡사하고 객체가 있어 그 밑에 여러 속성들을 담을 수 있다. 한편 LDAP처럼 다양한 쿼리(Query)는 불가능하고 ID로 찾는 방법만을 지원
- Measurement and State Spec 1.0 : 나라와 문화에 따라 상이한 측정 및 상태 단위에 대한 표준을 제공
- Position Spec 1.0 : GPS 시스템을 지원하기 위해 제안되었으며 WGS(World Geodetic System) 84를 지원
- Execution Environment Spec 1.0 : 여기에는 두 가지 실행 환경이 기술되어 있다. 하나는 Foundation Profile의 부분집합과 Framework/표준 명세 구현 환경을 포함하는 최소한의 실행 환경이고, 다른 하나는 CDC와 Foundation Profile 환경을 더한 것. 이 명세가 OSGi R3에 추가됨으로써 OSGi 서비스 플랫폼간의 호환성이 획기적으로 높아졌다.

앞의 서비스에 나타난 것처럼 OSGi는 R3에 이르러서 임베디드, 모바일, 데스크탑은 물론이고 엔터프라이즈 서버 환경에서 적응하기 위한 새로운 시도들을 발견하게 된다. Preferences Service, XML Parser Service, Metatype Spec, Execution Environment Spec이 바로 대표적인 핵심 서비스들이다.

활용 사례 소개

대부분의 IT 기술은 미국에서 태동했고 그 분야의 리더 역시 대개 미국기업이다. OSGi의 경우도 이를 이끌고 주도하는 업체는 미국기업인 IBM이지만, 유럽 업체들도 미국기업 못지않게 활발한 활동을 펼치고 있다는 점은 주목할 만하다.

과거 IBM은 OSGi 분야에서 자사 솔루션인 SMF(Service

Management Framework)를 시작으로 해서 임베디드/모바일 자바 환경인 J2ME와 오픈소스 개발 툴에서 플랫폼으로 확장하고 있는 이클립스, 그룹웨어 협업 솔루션 Expeditor에 이르기까지 지속적으로 OSGi 솔루션에 대한 영향력을 확대해 왔다. 특히 IBM은 이클립스의 오픈소스화뿐만 아니라 이클립스 3.0대에 이르러 OSGi가 이클립스 플랫폼에 탑재되자 SMF를 지원함으로써 Equinox 솔루션이 탄생하기까지 많은 지원을 아끼지 않았다. 이런 J2ME(J9), Expeditor, Equinox 등의 솔루션 라인업 덕분에 IBM은 많은 업체들이 OSGi 솔루션을 써서 제품을 개발하거나 탑재하고자 할 때 가장 먼저 찾게 되는 OSGi 선두업체로 자리매김하게 된다.

한편 유럽의 경우는 기업들이 크게 솔루션 업체와 OSGi 개발 벤더로 양분되었는데 대표적인 업체들로는 지멘스, 노키아, 필립스, BMW 등의 전통적인 제조업체와 Knopflerfish, Prosyst 같은 전문 벤더를 꼽을 수 있다. 지멘스-노키아는 모바일 솔루션 분야에서, 필립스는 홈 네트워크 분야에서, BMW는 자동차 정밀 제어 분야에서 각각 OSGi를 자사의 핵심 미들웨어 솔루션으로 탑재해 제품을 개발하고 있다. 또한 Knopflerfish, Prosyst 등의 전문벤더들은 전통적인 제조업체들이 OSGi 솔루션을 신뢰성 있는 제품 형태로 사용할 수 있도록 표준화 스펙을 상용화해 공급하고 있다. OSGi가 R4에 이르러서 OSGi 커뮤니티 전문 기술 그룹들이 모바일, 자동차, 임베디드, 엔터프라이즈 서버 환경으로 재편된 것도 이러한 유럽 업체들의 영향력을 무시할 수 없었기 때문이다. 미국의 경우에는 IBM을 필두로 다양한 상용화 솔루션과 오픈소스 진영(Eclipse, Apache)으로의 영향력을 확대하고 있다면, 유럽은 오히려 다양한 기업들이 자동차, 모바일, 홈 네트워크 분야의 상용화 솔루션에서 두각을 나타내고 있다. 그렇다면 우리나라의 상황은 어떠할까? 국내에서도 많은 연구개발이 이뤄지고 있는 것은 사실이나 상용화 제품으로 이어져 지속적인 사업 매출로 연결되는 경우는 많지 않다. 다만 최근 들어서 가정용 세톱박스와 텔레매틱스 분야(내비게이션 등)에서 눈에 띄는 성과를 거두고 있다.

김영수 책임
삼성전자 생활가전총괄 시스템가전 사업부
공조디지털제어그룹

interview

"효율적인 원격 중앙관리가 필요"

DMS 개발 배경은? _ 삼성전자는 시스템 에어컨을 빌딩 내에 구축해 이를 관리하고 있다. 그러나 최근 들어 시스템 에어컨 관리 분야에 고도의 IT 신기술이 접목되기 시작했고 학교, 병원, 기업 등 빌딩의 특성 및 규모별로 각기 다른 관리시스템이 등장하고 있다. 따라서 삼성전자도 보다 효율적인 원격 중앙관리가 가능한 솔루션이 필요해졌고, 더 효율적인 시스템 에어컨 관리를 위해 DMS 솔루션을 개발하게 된 것이다.

기존 시스템의 문제점은? _ 기존의 시스템 에어컨 관리는 PC를 통한 원격 관리가 이뤄지고 있었다. 그러나 PC를 통한 원격 관리는 안정성, 확장성이 낮은 문제점을 지니고 있다. 또 통신이 원활하지 않고 365일 내내 지속적인 관리가 이뤄지지 못하는 한계점을 지니고 있었다.

IBM 제품을 도입하게 된 배경은? _ 여러 제품을 놓고 BMT를 진행한 결과 IBM의 J9, SMF, DB2e, WSDD 제품이 선정됐다. 개발의 편의성과 성능, 그리고 속도 면에서도 만족할 만한 수준이라 이를 채택한 것이다.

DMS의 개발 효과는? _ DMS를 적용할 경우에 시스템 에어컨의 더 편리한 관리가 가능해져 고객 만족도가 크게 높아질 것으로 보인다. 그런 면에 있어서 삼성전자는 강력한 솔루션을 갖게 된 것이다. 2005년 7월에 호서대학교에서 필드 테스트를 수행한 바 있는데 이 당시 무척 만족할 만한 결과를 얻을 수 있었다.

삼성전자 빌딩용 공조시스템 컨트롤러 DMS

필자가 근무하는 삼성전자 생활가전총괄은 사람들에게 꼭 필요한 의식주 관련 제품을 연구 개발하는 곳으로 이른바 'Life Style Innovator'의 역할을 수행하기 위해 네 개의 사업영역에서 삶의 질적 향상을 꾀하기 위해 노력하고 있다. 이 총괄에서는 더 효율적인 시스템 에어컨 관리를 위해 임베디드 컨트롤 서버인 'DMS(Data Management Server)' 개발을 추진해 최근 완료했다. 여기서는 이를 활용 사례로 소개해 본다.

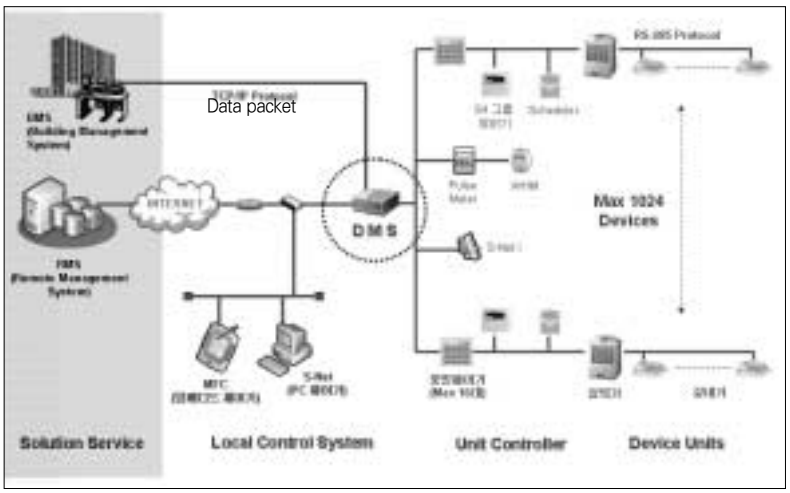
● 사업 확장에 따른 신규 시스템 필요

시스템 에어컨의 관리 체계에는 최근 들어 고도의 기술들이 적용되기 시작했다. 즉 웹 기반으로 환경이 바뀔 뿐 아니라 오픈 프로토콜이나 임베디드 디바이스 같은 최신 IT 기술이 공조시장에 접목되기 시작한 것이다. 아울러 전문화도 이뤄지기 시작했다. 학교, 병원, 사무실 빌딩 등 사이트 특성이나 규모에 맞는 관리 시스템이 사이트에 알맞게 출시되기 시작했다. 서비스 대응 수준도 높아지고 있다. 일본에서는 원격 진단 서비스가 보편화되는 시기를 맞이했다. BMS 시장도 변화되고 있다. 과거와 달리 원격 관리 시스템들도 오픈 프로토콜 기반으로 전환되고 있는 것이다. 오픈 프로토콜의 적용 사례는 이미 국내의 신규 대형 빌딩에서 찾아볼 수 있다.

삼성전자는 기업, 대학, 병원 등의 빌딩에 시스템 에어컨(DVM)을 공급 및 구축해 운영해 오고 있다. 따라서 삼성전자도 시스템 에어컨 사업 확장에 따른 신규 시스템 아키텍처가 필요하

게 됐다. 또한 원격 관리 시장 진입을 위한 체계 구축도 필요했다. 삼성전자는 그동안 공급한 시스템 에어컨에 대해서는 PC를 활용해 원격 관리를 해왔으나 이는 많은 문제점을 발생시켰다. PC를 통한 시스템 에어컨 관리는 범용이 아닌 탓에 활용도가 낮을 뿐 아니라 안정성과 확장성 면에서도 불리했다. 또 통신이 원활하게 이뤄지지 않고 보안상에도 문제가 있는 것으로 지적되어 왔다. 기존 시스템과 연동이 쉽게 이뤄지지 않고 24시간 365일 운영되지 못한다는 점도 한계로 작용했다.

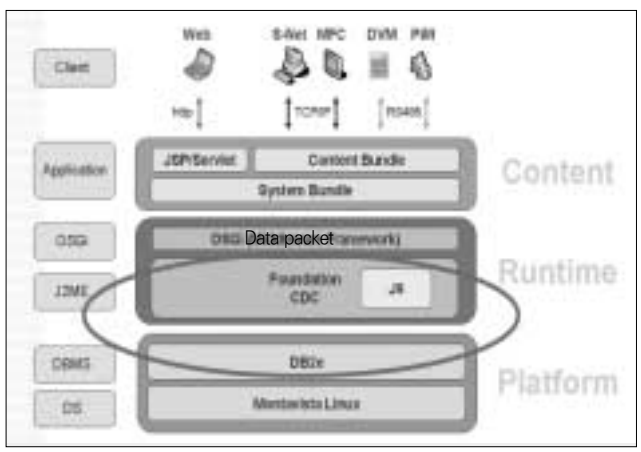
이에 따라 생활가전총괄에서는 중앙 관리가 필요한 학교, 기업, 호텔의 빌딩 등을 주요 타깃으로 정하고 365일 내내 시스템 에어컨의 운영 통제 관리를 가능케 하는 솔루션을 개발하기로 했다. 이러한 결정으로 시스템 에어컨 중앙관리를 위한 각 사이트의 운영을 총괄하는 임베디드 컨트롤 서버(DMS)의 개발이 시작된 것이다.



〈그림 1〉 DVM 시스템 아키텍처

● J2ME(J9), SMF(OSGi), WSDD, DB2e 적용해 DMS 개발

DMS의 개발은 지난 2004년 3월부터 진행됐다. 이번 개발을 통해 제품이 등장하면 이 제품은 시스템 에어컨 기술 로드맵 상의 가장 근간이 되는 핵심 디바이스가 된다. 이DMS의 주요 기능에는 통신 오픈 프로토콜 지원(RS485, TCP/IP, Lon Works, BACnet), DMS 1대당 최대 16대의 중앙 제어기와 256대의 실내외기 지원, 웹서버 자체구동 및 웹 제어 등이 있다. 삼성전자가 개발하는 DMS는 임베디드 기반의 운영체제를 가진 다비이스로 시스템 에어컨 데이터를 485통신으로 수집하고 이더넷(Ethernet)을 통해 DVM 전용 제어기나 서비스 서버로 전송한다. 한편 DMS는 임베디드 DB 시스템인 'DB2e'를 탑재해 더 효율적으로 시스템 에어컨 데이터를 관리한다. 자체 메모리 용량을 고려해 에어컨의 상태 데이터를 일정 기간 저장할 수 있으며 Web Configuration & Management, DI/DO 입출력 Port 내장, 년간 및 주간 일정 기능을 저장 및 수행할 수 있다. 또한 Console Port(RS232C)를 이용해 구성되어 있고 디바이스 모니터링 기능으로 고장 정보, 상태 정보, 운영 정보를 모니터링하고 전력관리 기능으로 전력분배, 피크 전력 제어 등을 수행할 수 있다. 임베디드 컨트롤러에서 과거 이러한 기능은 RTOS 환경을 통해 C/C++로 개발하는 경우가 대부분이었으나, OSGi 탑재로 모든 컨텐츠들을 번들로 처리해 컨텐츠의 추가, 수정, 삭제, 업데이트 등의 관리를 원격에서 실시하고 확장성이 뛰어난 구조로 설계할 수 있다. DMS 개발 환경은 애플리케이션 툴로는 WSDD(WebSphere Studio Device Developer) 등이 사용됐다. 플랫폼



〈그림 2〉 DMS 디자인 레이아웃

으로 운영체제는 Montavista 임베디드 리눅스가, DB로는 임베디드 DB인 DB2e, 미들웨어 프레임워크는 OSGi가 각각 채택되었으며 개발 환경에는 J2ME - J9 & CDC-F/P가 적용됐다.

● DMS로 고객 만족 높여 매출 확대

삼성전자는 이번 프로젝트를 통해 시스템 에어컨 중앙관리를 위한 각 사이트별 운영을 총괄하는 임베디드 컨트롤 서버 솔루션을 보유하게 됐다. 이를 통해 삼성전자는 시스템 에어컨 관리의 효율성이 증대될 것으로 전망하고 있다. 또 새로 개발된 DMS 솔루션을 적용할 경우에 기존 관리보다 편의성이 높아져 고객 만족도가 증대될 것으로 보인다.

제어 솔루션 부문에서도 경쟁사와 비교해 다소 열세였던 부분을 만회함으로써 공조 중앙관리가 필요한 시장에서의 경쟁력을 한층 강화할 수 있을 것으로 예측된다. 이와 더불어 이번 솔루션 개발을 통해 온라인 서비스 체제 구축을 위한 기반도 마련하게 됐다.