전력 R&D SW 표준프레임워크 구조 설계

송재주*, 주용재*
*한국전력공사 전력연구원 소프트웨어센터
e-mail: jyj@kepco.co.kr

Architecture Design of a Standard Framework of R&D Softwares for Electric Power System

Jae-Ju Song*, Yong-Jae Joo*
*Software Center, KEPCO Research Institute

요 약

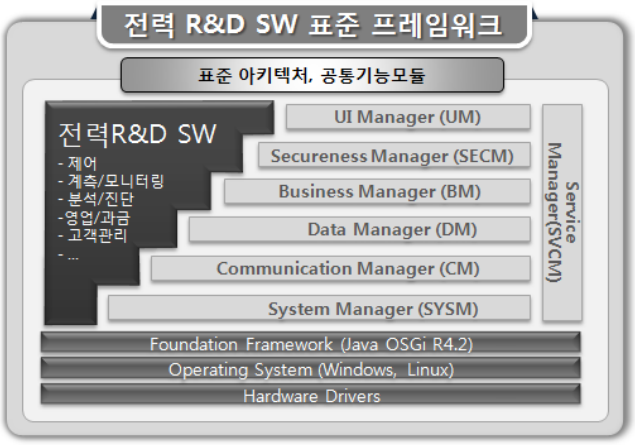
고도의 지능화가 요구되는 디지털 전력시스템의 R&D SW를 개발하기 위해서는 감시, 제어, 최적화 등 다양한 SW 간의 상호 데이터 및 서비스의 더욱 긴밀한 연계가 필요하다. 본 논문에서는 전력분야 R&D SW의 개발 생산성을 높이고 품질을 확보할 수 있을 뿐만 아니라, SW간 데이터 및 서비스의 상호 운용성을 강화할 수 있는 Java OSGi 기반의 SW 표준프레임워크(KSF)를 제안하고, 공통모듈 서비스 및 기초UI를 설계하였다.

1. 서론

전력연구원 소프트웨어센터에서는 연구과제에 포함되어 개발되는 SW(이하 "R&D SW")를 개발하기 위하여 "요구분석 - 설계 - 구현 및 통합 - 시험 - 종료"의 SW 개발 전주기를 수행하고 있다. 전력 R&D SW는 발전-송전-변전-배전-영업-수요자 등 전력 전 분야에 걸쳐 감시, 제어, 계측, 시뮬레이션, 관리, 운영, 최적화 등 다양한 분야의 SW가 존재한다. 과거에는 이러한 다양한 R&D SW들이 서로 다른 SW구조, 통신 프로토콜, 프로그래밍 언어로 구현되었으며, 각기 독립적으로 실행되는 형태로 운영되어 왔다. 그러나 최근 IT의 발달과 더불어 고도화된 지능형 서비스 요구의 증가로 점차 SW간 상호 데이터 교환 및 서비스 연계가 필요해졌으며, 기존의 독립적인 SW개발 방식으로는 효율적이고 충분한 서비스 연계 및 시스템 통합이 어려운 실정이다.

이러한 문제점을 해결하기 위하여 전력분야에서는 표준화 및 시스템 통합 등의 노력이 있어 왔다. 발전분야에서는 화력발전소 통합감시제어시스템[1][2], 수력발전소 통합운영시스템[3], 분산 발전플랜트 이기종 제어시스템의 웹기반 통합 플랫폼[4]에 관한 연구가 있다. 송배전 분야에서는 IEC61850 등의 표준 객체 모델 및 서비스, 통신 프로토콜 등의 표준화 및 이를 활용한 SCADA, DAS, 스마트그리드 통합시스템 등의 연구개발이 활발히 이루어지고 있다. 그러나 이러한 기존 연구들은 기존의 이종 제어 및 응용 소프트웨어의 통신 및 데이터 체계를 표준화하여 연계를 강화하고 서비스를 제공하는 것을 목표로 하고 있으며 실제 SW 개발자에게 필요한 공통 구조, 재사용 가능한 Source Code 등의 프레임워크에 대한 내용은 포함되어 있지 않다.

본 논문에서는 전력 R&D SW의 개발 생산성을 높이고 품질을 확보할 수 있을 뿐만 아니라, SW간 데이터 및 서비스의 상호 운용성을 강화할 수 있는 SW 표준프레임워크(이하 'KSF' : Korea electric power Software Framework)를 제안한다.



(그림 1) 전력 R&D SW 표준프레임워크 아키텍처

2. 전력 R&D SW 표준프레임워크(KSF) 기본 구조

본 논문에서 제안하는 KSF의 기본 아키텍처는 (그림1)과 같다. KSF는 Java의 동적모듈을 지원하는 오픈 프레임워크인 OSGi를 기반으로 하고 있으며, 전력 R&D SW의 개발에 필요한 공통기능 모듈인 7개의 표준모듈(Manager)로 구성되어 있다. 이러한 표준모듈은 OSGi의 번들(Bundle) 형태 또는 Core Spec. Layer의 조합으로 구현된다.

2.1 표준시스템모듈(SYSM : System Manager)

표준시스템모듈은 KSF의 가장 기본이 되는 표준 공통모듈로서 KSF기반 SW 및 타 표준모듈의 라이프 사이클(등록, 업데이트, 시작, 중지 등)을 관리한다. SW 개발자는 SM에서 제공하는 API를 활용하여 개발하고자 하는 SW 번들의 안정적이고 중단 없는 등록·업데이트 등의 작업을 수행할 수 있다.

2.2 표준서비스모듈(SVCM : Service Manager)

표준서비스모듈은 KSF기반 SW를 OSGi Service Repository에 등록시 추가적인 정보를 연결하여 관리한다. KSF기반 SW는 SVCM을 통하여 특정 SW 번들을 검색하고 연결하기 위한 Bundle Context를 획득할 수 있다. 또한 SVCM은 KSF기반 SW들이 제공하는 공유 객체(클래스 인스턴스, DB 레코드, 파일, 공유메모리)와 서비스 정보 및 그에 대한 접근 권한을 관리한다. KSF기반 SW 간 객체·서비스의 공유를 위해서는 먼저 제공하고자하는 SW 번들이 SVCM에 등록하고, 제공받는 SW 번들은 SVCM으로부터 읽기/쓰기 등의 접근 권한을 획득하여야 한다.

2.3 표준통신모듈(CM : Communication Manager)

표준통신모듈은 KSF기반 SW 간 객체데이터·서비스(함수) 요청 및 전달, 공유객체 데이터 값 설정, 공유메모리를 통한 객체공유 등의 기능을 제공한다. 또한 KSF기반 SW는 CM을 통하여 Legacy 시스템과의 이벤트·메시지 전송 및 명령어 전달 등의 통신(MMS, OPC, DNP 프로토콜)을 수행할 수 있다. SW 개발자는 이와 같은 CM의 API를 통해 개발하고자 하는 SW의 데이터·서비스·통신 기능을 빠르게 구현할 수 있다.

2.4 표준데이터모듈(DM : Data Manager)

표준데이터모듈은 KSF기반 SW의 DB 쿼리, KSF기반 SW 간 DB 레코드·파일 요청 및 전송 기능을 제공한다. KSF기반 SW 개발자는 DM의 API를 이용하여 DB 쿼리를 안정적으로 수행할 수 있으며, 시스템 관리자는 DM의 Connection Pool 관리기능을 통해 DBMS를 안정적으로 운영할 수 있다.

2.5 표준비즈니스모듈(BM : Business Manager)

표준비즈니스모듈은 KSF기반 SW의 객체 및 서비스 제공에 대한 계약 당사자간 비즈니스 모델에 따른 권한관리 및 비용처리 등의 기능을 제공한다. KSF기반 SW 개발자는 BM을 통해 개발된 SW 번들의 제공 데이터 또는 서비스의 종류, 빈도, 양에 따라 인센티브를 획득할 수 있다.

2.6 표준안정화모듈(SECM : Secureness Manager)

표준안정화모듈은 KSF가 운영되는 시스템 및 실행중인 KSF기반 SW의 점유 CPU, HDD, 메모리 자원 및 네트워크 트래픽 발생량 등을 주기적으로 감시하고 KSF 관리자가 사전에 정해놓은 규칙에 의거하여 경고 발생, HDD 용량확보, 네트워크 차단 등의 적절한 조치를 수행한다. KSF 관리자 및 SW 개발자는 SECM을 통해 시스템 및 네트워크의 안정성을 확보할 수 있다.

2.7 표준UI모듈(UM: User Interface Manager)

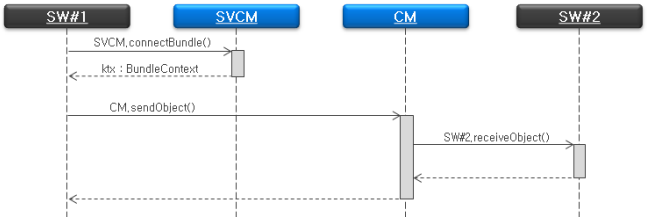
표준UI모듈은 UI의 수행 및 시스템 관리자의 SW 번들 관리를 위한 기능을 제공한다. UI(시스템관리자 모듈 포함)는 UM을 통해 특정 서버에 설치된 KSF기반 SW의 종류 및 상태, 각 SW가 공유한 객체·서비스 정보 등을 확인할 수 있다. SW개발자는 UM을 통해 Spring, WPF, JSF 등 다양한 프레임워크의 UI를 개발·연계할 수 있다.

3. KSF 공통모듈 서비스 설계

본 절에서는 KSF를 구성하고 있는 공통모듈에서 공통 기능을 제공하기 위한 실행 순서를 설계한다.

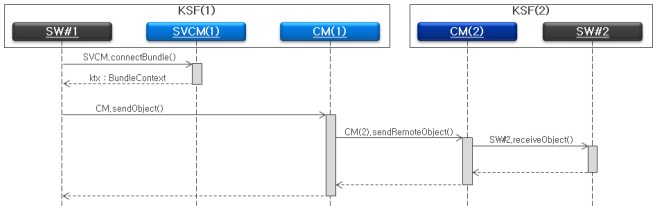
3.1 KSF기반 SW 간 객체 전송

(그림2)는 단일 KSF 내 SW간 객체 전송하는 순서를 표현한다. KSF기반 SW#1은 SW#2에 특정 객체를 전송하기 위해 먼저 SVCM의 connectBundle() 서비스를 호출하여 SW#2에 대한 BundleContext 정



(그림 2) 단일 KSF SW간 객체 전송

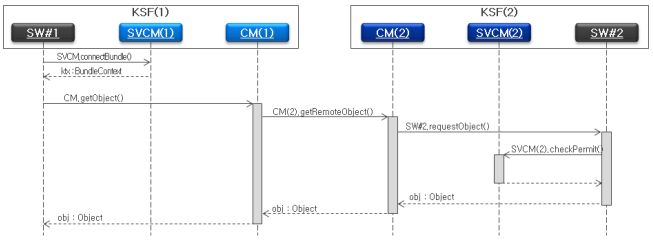
보를 획득하고, CM의 sendObject 서비스를 호출하여 객체전송을 시작한다. CM의 sendObject() 서비스는 목적지 SW가 동일 KSF 내에서의 전송인지 확인한 후 수취자인 SW#2의 receiveObject() 서비스를 호출함으로써 객체를 전달시킨다. SW#2 개발자는 receiveObject내에 필요한 후속 처리를 구현할 수 있다.



(그림 3) 다중 KSF SW간 객체 전송

CM은 동일 KSF 내에서 뿐만 아니라 네트워크로 연결되어 있는 원격 KSF기반 SW로의 객체 전송 기능을 제공한다. (그림3)은 KSF(1)의 SW#1이 네트워크로 연결되어 있는 KSF(2)의 SW#2로 객체를 전송하는 과정을 나타낸다. SW#1은 자신의 SVCM의 서비스를 활용하여 SW#2의 연결정보를 획득한다.

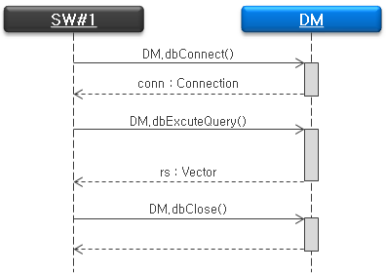
SVCM은 네트워크상에 연결된 KSF SW 번들 저장소를 탐색하여 SW#2가 속한 시스템의 IP 및 연결정보를 획득한다. SW#1은 자신 CM(1)에게 객체전송을 명령하고(sendObject), CM(1)은 전송할 객체 및 sourceBundle(SW#1)과 targetBundle(SW#2) 등의 정보를 포함하여 SW#2가 속한 KSF(2)의 CM(2)의 sendRemoteObject() 서비스를 호출한다.



(그림 4) 다중 KSF SW간 객체 요청 및 전달

3.2 KSF기반 SW 간 객체 요청 및 전달

CM은 단일 또는 다중 KSF에 속한 SW 번들 간에 공유된 객체의 요청 및 전달 기능을 제공한다. (그림 4)는 KSF(1)의 SW#1이 KSF(2)의 SW#2가 공유한 객체를 CM을 통해 요청 및 전달받는 과정을 나타낸다. SW#1은 SVCM(1)을 통해 SW#2의 연결정보를 취득한 뒤 CM(1)의 getObejct() 서비스를 통해 SW#2가 공개한 객체(인스턴스)를 요청한다. CM(1)은 요청받은 객체의 주인번들(ownerBundle)인 SW#2가 속한 CM(2)의 getRemoteObject() 서비스를 호출하고, CM(2)는 SW#2의 requestObject()를 실행한다. SW#2는 객체를 요청한 요청SW(requesterBundle) 및 요청 객체 정보를 포함하여 SVCM을 이용하여 접근권한을 확인(checkPermit)하고, 해당객체에 대한 복사본을 만들어 CM(2)에게 반환한다. CM(2)는 CM(1)에게 CM(1)은 다시 SW#1에게 호출된 함수의 리턴값으로 해당 객체를 전달한다.



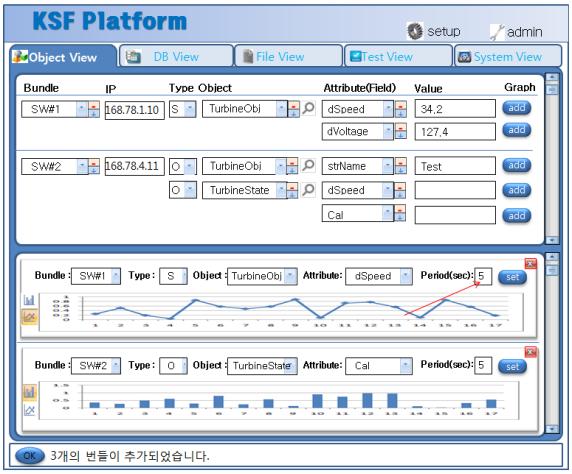
(그림 5) DM의 DB 쿼리 대행

3.3 KSF기반 SW의 DB 쿼리 대행

DM은 KSF기반 SW의 DB 쿼리 수행을 대행할 수 있다. (그림5)는 SW#1이 DM의 쿼리 수행 서비스를 활용하여 DB 접속/종료 및 쿼리 수행을 하는 과정을 나타낸다. SW#1은 우선 접속하고자 하는 DBMS의 IP, 계정, 암호 등의 접속 정보를 포함하여 DM의 dbConnect() 함수를 이용하여 Connection 정보를 획득한다. 그 후 획득한 Connection 정보를 포함하여 접근하고자 하는 DBMS에 쿼리 수행을 요청(DM.dbExcuteQuery)한다. 이와 같이 DM의 쿼리 대행 서비스를 활용함으로써 SW 번들 개발자는 DBMS의 종류에 상관없이 쿼리 수행 기능을 빠르게 개발할 수 있으며, DBMS 관리자 입장에서는 DM을 통해 Connection Pool을 관리함으로써 DB의 안정적 운영을 꾀할 수 있다.

3.4 KSF기반 SW 간 DB 레코드 요청 및 전달

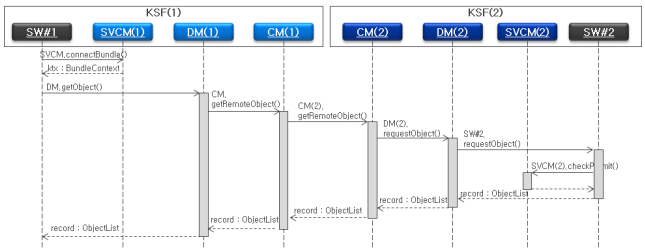
DM은 단일 또는 다중 KSF기반 SW 간 공유된 DB Table의 레코드를 요청/전달하는 기능을 제공한다. (그림6)은 다중 KSF에 속한 SW 번들 간의 DB 레코드 요청 및 전달하는 과정을 나타낸다. KSF(1)에 속한 SW#1은 DM의 getObject() 서비스를 호출하여 KSF(2)에 속한 SW#2가 공개한 DB Table객체의 전송을 요청한다. 이때 requesterBundle(요청자)는 DB 쿼리의 조건정보를 포함하여 요청하며, ownerBundle(제공자)은 해당 조건문을 포함한 쿼리문을 만들어 DM에게 DB 쿼리 대행을 요청하고 그 결과를 반환하여 전달한다.



(그림 6) 기초UI : 공개객체 조회 화면

4. 기초UI 설계

KSF기반 SW 번들의 개발자는 KSF가 제공하는 UM 및 기초UI를 활용함으로써 개발 과정중 단위 및 통합시험 등을 수행할 수 있다.



(그림 7) 다중 KSF SW간 DB 레코드 요청 및 전달

기초UI는 공개된 객체의 정보를 조회할 수 있는 객체조회(Object View), DB Table 데이터 조회할 수 있는 DB조회(DB View), 공개된 File의 다운로드를 수행할 수 있는 파일조회(File View), 공개객체의 데이터 변경 및 요청・전송 시험 수행하는 시험수행(Test View), 시스템 및 SW 번들의 상태를 조회할 수 있는 시스템조회(System View) 화면 등으로 구성된다. (그림7)은 기초UI를 통해서 여러 IP주소의 시스템에 설치된 KSF기반 SW 번들의 공개된 객체 데이터를 텍스트 및 그래프의 형태로 조회하기 위해 설계된 화면이다.

5. 결론 및 향후 연구개발 계획

본 논문에서는 전력 R&D SW의 개발 생산성을 높이고 품질을 확보할 수 있을 뿐만 아니라, SW간 데이터 및 서비스의 상호 운용성을 강화할 수 있는 Java OSGi 기반의 SW 표준프레임워크(KSF)를 제안하고, 공통모듈 서비스 구현 방안 및 기초UI의 화면을 설계하였다. 향후에는 제안한 “KSF”와 “전력 R&D SW 엔지니어링 플랫폼(R&D SW의 개발/품질 관리 업무의 효율적 수행을 위한 프로세스지원 통합 시스템[5])”과의 연계 방안, IEC61850 등의 전력분야 표준 정보 및 서비스 규격 수용 방안 등을 연구・개발할 예정이다.

참고문헌

[1] 황재필 외, “화력발전용 통합감시제어시스템의 개발 및 검증”, 정보 및 제어 학술대회 논문집, 213-214, Oct. 2010

[2] 김승민 외, “화력발전용 통합감시제어시스템의 개발 및 기능 검증”, 대한기계학회 논문집, 1164-1169, Nov. 2010

[3] 옥연호, 박지군, 곽원구, 이재흥, “발전통합운영 시스템 최적제어 및 시스템 구성에 관한 연구”, 대한전기학회 논문집, 1852-1853, Jul. 2010

[4] Mark Strembeck, Otto U. Plhal, "VIDIHIP - A Web Service based Integration Platform for Power plant Control Systems", Proceedings of IEEE International Conference on Service-Oriented Computing and Applications(SOCA’07), 207-214, Jun. 2007

[5] 주용재 외, “전력 R&D SW 엔지니어링 플랫폼 개발”, 한국정보처리학회 논문집, Nov. 2011(게재 예정)