레거시 시스템을 웹 서비스에 통합하기 위한 연구

김동욱° 국승학, 김현수
충남대학교 전기정보통신공학부 컴퓨터전공
{nivea1, triple888, hskim401}@cnu.ac.kr
이재경, 박성환
한국기계연구원 e-엔지니어링 연구센터
{jkleece, swpark}@kimm.re.kr

A Research on the Integrating Legacy Systems into Web Services

Dong Uk Kim°, Seung Hak Kuk, Hyeon Soo Kim
Dept. of Computer Sc. & Eng., Chungnam National University
Jai-kyung Lee, Seong-whan Park
e-Engineering Research Center, Korea Institute of Machinery & Materials

요 약

업무 프로세스의 개선 및 효율성 증가를 위해 시스템간의 연동을 웹서비스(Web Services)화 하여 시스템 간의 프로세스 및 데이터 연계를 목적으로 하는 통합의 연구가 활발히 진행되고 있다. 본 논문에서는 이기종간의 레거시 시스템을 SOA(Service Oriented Architecture)기반의 웹 서비스와 통합하여 구매/유지보수 비용을 절감하고 신뢰성, 보안성 및 상호 운용성 향상 등의 비즈니스 프로세스의 가치를 향상시키기 위한 방안에 대해 소개한다. 레거시 시스템을 웹서비스와 통합 시 고려해야 하는 비 기능적인 요소를 OASIS, W3C, WS-I등의 단체에서 정해진 웹서비스 명세 표준에 근거하여 선정하고, 이를 준수한 통합 방안 및 구현사례를 제시한다.

1. 서 론

기업의 비용절감을 통한 경쟁력 강화, 비즈니스 신속성의 확보 등을 목적으로 애플리케이션, 저장 데이터 및 업무 프로세스의 통합이 IBM, Oracle, MS, Sun등의 벤더를 중심으로 이루어지고 있다. 레거시 시스템을 가진 기업들은 레거시 시스템에서 가치를 도출하기 위해 SOA를 채택해야 하는 필요성을 느끼고 있다[1]. 일반적으로 레거시 시스템들은 오랜 기간 동안의 구현, 수정, 업데이트로 인해 강결합(Tightly Coupled)되며, 유연성이 떨어지는 단점을 갖는다[2]. 이를 해결하기 위한 방안으로 레거시 시스템을 웹서비스와 통합하여 유연성, 신뢰성, 보안성을 갖춘 IT환경을 구축하기 위한 노력이 벤더들에 의해 진행 되고 있다. 이러한 과정을 통해 기존의 유용한 레거시 시스템을 재사용하여 비용효율성을 기대할 수 있다. 이에 본 논문에서는 레거시 시스템을 웹서비스와 통합할 때 고려해야 하는 요소에 대해 웹서비스(WS-*) 명세를 기준으로 구분하였다. 명세의 구분은 웹서비스 실행 환경을 구성하는 요소들을 기능적 요소로 분류하고 레거시 시스템과 웹서비스의 통합 시 고려해야 하는 보안성, 신뢰성, 유연성, 효율성 등을 비 기능적인 요소로 분류한다. 본 연구에서는 레거시 시스템을 웹서비스와 통합하는 단계에서 고려해야 하는 사항들 중에서 비 기능적 요소에 대해 중점적으로 다루고 있다.

2. 관련 연구

SOA기반의 웹서비스의 이점은 크게 비즈니스 측면과 IT측면으로 분류할 수 있다. 기존의 프레임워크에 관계없이 서비스를 사용할 수 있으며 비즈니스 프로세스를 신속하게 생성하고 애플리케이션을 구축할 수 있다는 장점이 비즈니스 측면에서의 이점이라면 기존의 시스템의 전면적인 교체가 아닌 재활용을 통해 비용 효율성 향상과 업무의 복잡성을 감소시키고 유지보수 비용을 절감할 수 있다는 점은 IT측면에서의 이점이라고 할 수 있다.

2.1 기존 분산처리 시스템과 웹 서비스의 차이점

웹서비스는 SOAP(Simple Object Access Protocol)의

사용으로 인해 CORBA, Java RMI, EJB등의 기존 분산 처리 시스템에서 사용되던 미들웨어가 없으며, 위치정보의 저장이 불필요하기 때문에 간략한 프로토콜이라 할 수 있으며, Port 80을 사용하기 때문에 어디서든 HTML XML을 통하여 명세/요청의 전달이 가능하다.

2.2 웹서비스를 적용한 레거시 시스템의 기존 연구방향

XML기반의 웹서비스가 기업의 내외부적인 통합 서비스의 문제점을 해결할 수 있는 새로운 방안으로 부상하고 있다. Sun microsystems의 Sun One Integration Platform은 EAI, B2B 부분의 메시지 지향 미들웨어 제품을 공급하고 있으며[3], IBM의 WebSphere 애플리케이션 서버는 트랜잭션 관리, 클러스터링, 보안, 연결성 및 확장성에 이르는 애플리케이션 서비스를 구축하고, 기업 전반의 애플리케이션에 대한 관리와 통합을 지원하고 있다.[4] 그 외에도 PeopleSoft, SAP, Oracle등과 국내의 TmaxSoft등의 ERP(Enterprise Resource Planning) 및 SCM (Supply Chain management)벤더들도 웹서비스를 이용한 애플리케이션의 통합을 시도하고 있다.

3. 레거시 시스템에 웹서비스의 통합

레거시 시스템을 웹서비스에 의해 통합하여 시스템들간 상호연동이 가능하기 위해서는 표준화된 기술을 사용하는 것이 바람직하다. 그림 1의 웹서비스 명세(WS-*)들이 표준 기술에 해당된다. 웹 서비스 명세는 SOA, 그리드 등의 차세대 웹 기술을 위한 배경기술로 넓게 인용되고 일부는 이미 특정 응용분야에서 이용되고 있다. 따라서 OASIS[5], W3C[6], WS-I[7]등의 표준기구에서는 중요성이 인정된 웹서비스 명세들을 국제표준으로 개발완료하였거나 개발 중에 있다[8]. 이러한 웹서비스 표준의 구분은 서비스에 관련된 메타데이터 관리에 필요한 "메타데이터(Metadata)", 메시지의 생성, 전송, 해석, 처리와 관련된 "메시징(Messaging)", 서비스의 트랜잭션 처리 및 비즈니스 프로세스 실행에 필요한 "트랜잭션/비즈니스 프로세스(Transactions and Business Process)", 서비스에서 제공된 정보를 화면에 표현하는데 필요한 "포탈/프리젠테이션(Portal and Presentation)", 다양한 보안 처리를 위한 "보안(Security)", 서비스의 분산관리 및 프로비저닝에 필요한 "관리(Management)" 영역으로 구분된다[9]. 다음 그림1은 WS-*명세 스택을 나타내며 명세 스택에서 굵은 선으로 표기된 부분은 본 논문에서 논하게 될 레거시 시스템과 웹서비스의 통합에 있어서의 비 기능적인 요소이다. 이는 그림 1의 Assurances 부분이 이에 해당된다. 그 외의 WS-* 명세스택에 명시되어 있는 XML, SOAP, WSDL등의 웹서비스 실행 환경 구축을 위한 표준에 대해서는 4장의 사례연구에서 이를 적용하여 상호보완적으로 구현한 엔지니어링 프레임워크를 통해 소개하고자 한다.

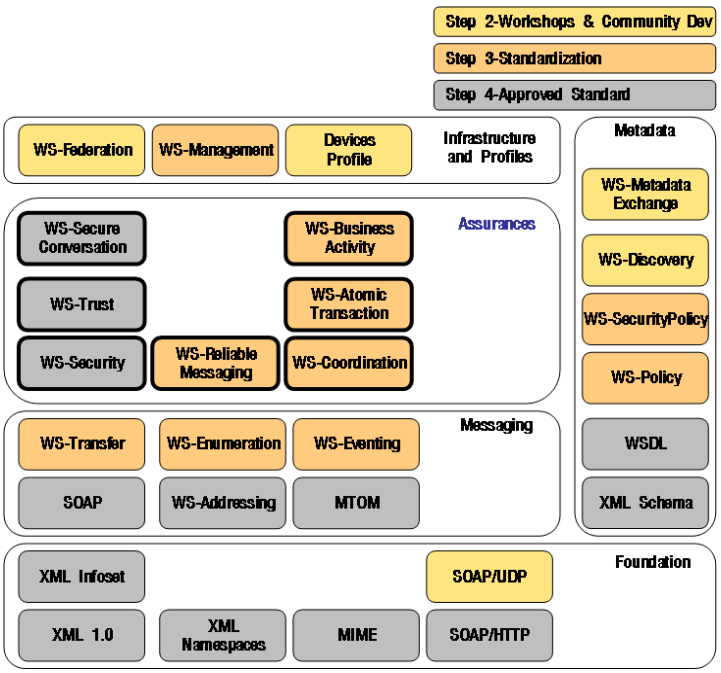

그림 1 WS-* 명세 스택

3.1 레거시 시스템의 웹서비스화 단계에서의 고려사항

벤더가 제안한 정책들은 OASIS, W3C등의 표준 단체에 의해 표준화 단계를 거치게 되며 WS-*에 명세 된 여러 표준들은 상호의존적으로 서로간의 역할을 수행하게 된다. 예를 들어 WS-Policy는 WS-PolicyAssertions를 사용하며, WS-Security는 서비스 제공자와 서비스 사용자 간의 메시지 암호화에 대해 설명하고 있지만 WS-Trust는 상호간의 메시지 교환 이전에 신뢰할 수 있는 전달 방법에 대해서 WS-Security에 기반하여 사용된다. 따라서 기능적 요소와 비 기능적 요소는 구현 단계에서 별개로 진행되는 것이 아니라 기능적 요소에 비 기능적 요소를 추가하여 구현하는 형태로 진행이 이루어진다고 할 수 있다.

다음의 6가지 Assurances 요소는 레거시 시스템을 웹서비스와 통합할 때 고려해야 하는 웹서비스 표준이다.

● **웹 서비스 보안 (WS-Security)**

웹서비스 상호 운용성 기관(WS-Organization)이 제안한 메시지의 비밀성보호와 인증을 목적으로 SOAP 메시징을 확장한 규정인 WS-Security는 메시지 전달과정에서의 인증과 권한, 무결성, 기밀성, 부인방지를 명시하고 있다.

● **웹서비스 트러스트 (WS-Trust)**

웹서비스 트러스트는 사용자 ID정보를 신뢰성 있게 송

수신하는데 있어서 발생되는 문제를 해결하고자 하며, 이를 위해 WS-SecurityPolicy와 WS-MetadataExchange 기술과 함께 사용된다.

● **웹서비스 신뢰 메시징 (WS-ReliableMessaging)**
웹서비스의 확장 규약인 WS*(Webservic Extension) Reliable messaging 기능을 제공하고 있으며, 실패 없는 메시지의 전달을 보장하는 메시징 도구들이 활용된다.

● **웹서비스 트랜잭션 (WS-Transactions)**
웹서비스 트랜잭션은 분산된 시스템에서 인터넷을 통해 비즈니스 로직을 서비스하므로 물리적 위치가 상이하다. 이로 인해 중앙 집중적인 트랜잭션의 조정과 리소스의 관리가 어려우며 단일 트랜잭션의 보장과 트랜잭션의 처리에 소요시간의 낭비가 초래될 수 있다. 이러한 문제의 해결을 위해 WS-Transaction이 정의되어 있으며 비즈니스 프로세스의 트랜잭션 관리를 WS-Transaction 표준을 정의하고 트랜잭션 처리를 위한 프로토콜을 정의하고 있다.

● **웹서비스 조정 (WS-Coordination)**
웹서비스 조정은 웹서비스 트랜잭션과 더불어 신뢰성있는 애플리케이션에 필요한 매커니즘을 정의하고 있다. 웹 서비스 조정은 기존의 표준 트랜잭션 매커니즘을 지원하며, 확장이 가능한 프레임워크와 트랜잭션을 조정한다.

● **웹서비스 보안대화 (WS-SecureConversation)**
웹서비스 보안대화는 웹서비스 보안과 웹서비스 트러스트의 보완적 스펙으로 활용된다. 상호간의 보안에 필요한 공유에 대해 기술하며 메시지의 서명과 암호화에 사용되는 공유 비밀키에 대한 SCT(Secure Context Token) 요소를 정의하고 있다.

3.2 레거시 시스템을 웹서비스에 통합하기 위한 방안

그림 2의 스택은 레거시 시스템을 웹서비스화 하기 위한 래퍼(Wrapper)에 해당한다. 웹 서비스 실행 환경과 이를 구성하게 되는 SOAP, UDDI, WSDL, XML등의 웹서비스 실행 환경 구성요소들은 기능적 요소에 해당되며, 기능적 요소는 시스템 구축 환경, 규모, 구현 언어에 따라 다양하게 변경 및 적용될 수 있다. 보안모듈, 에러 핸들링 그리고 작업 관리 모듈은 비 기능적 요소에 해당한다.

● **Security Module**은 WS-Security에 해당되며 웹서비스는 약결합(Loosely Coupled에 의해 특정 구현에 구애받지 않는 중립적인 인터페이스를 가짐과 동시에 보안과 관리에 취약점을 드러낸다. 이로 인해 'Security Module'에는 인증, 사용자 권한 부여 및 SSL(Secured Socket Layer)등의 보안이 포함되면 데이터의 프라이버시를 보장할 수 있다. 이는 WS-Security의 모든 메시지 또는 일부 메시지의 인증과 암호화 단계에 해당되며 WS-SecureConversation과 보완적으로 활용된다.

● **Job Manager**는 다수의 서비스 사용자의 요청을 관리 하는 기능을 담당한다. 이러한 작업 관리를 위한 예로써 큐를 이용한 관리자 모듈을 이용해 별도의 큐를 관리하게 되면 요청된 서비스에 대한 순차적인 관리가 가능하며, 유연한 트랜잭션 처리를 위한 트랜잭션 큐를 통해 메시지 컨텍스트에 따른 다양한 기준에 따라 큐의 우선순위를 정할 수 있다.

● **Error Handling**은 서비스 요청 및 수행 결과를 로그를 기록하는 데이터베이스에 저장하는 기능을 담당한다. 에러 핸들링을 위한 로깅 모듈의 구현을 통해 시스템 구성 요소에 장애가 발생할 경우, 장애 발생이전에 데이터베이스에 저장 된 로그의 정보를 판단하여 서비스를 다시 시작한다. 이를 위해 서비스 사용자의 서비스 요청을 처리하기 위한 큐(Queue) 관리 모듈을 두게 되면 다수의 사용자의 요청을 순차적으로 처리할 수 있게 된다. 서비스 사용자의 요청은 서비스가 요청된 시점부터 종료되는 시점까지 일련의 단계들을 데이터베이스에 저장하고 시스템에 Fault 또는 Crash가 발생되면 데이터베이스에 저장된 로그를 사용하여 장애 발생 시점 이전의 완료된 단계부터 재시작하도록 구현되면, 로그기반의 회복기법을 통해 메시지 전달과정의 신뢰성을 향상시킬 수 있게 된다. 또한, 회복기법을 위해 로그를 저장하게 되는 데이터베이스의 커밋(Commit) 프로토콜의 관리를 통해 트랜잭션 커밋과 롤백(Rollback)이 적절하게 처리되면 트랜잭션을 최적화 할 수 있다. 위의 과정들은 WS-Transactions 및 WS-Coordination에 해당된다.

● **UDDI Publish Module**은 서비스 제공자가 UDDI에 서비스하는 정보를 퍼플리쉬하고 업데이트 및 수정관리가 가능한 모듈을 구성한다. 서비스 저장소(UDDI)의 접근방식은 HTTP 프로토콜을 사용해서 이루어지며 SOAP 형식으로 메시지가 전송된다.

● **WSDL Generator**는 제공하는 서비스에 대한 WSDL 파일의 생성과 검증 기능을 갖는다. Generator의 구현을 통해 WSDL의 자동적인 구현이 가능하도록 한다.

● **SOAP Engine**은 SOAP 메시지를 생성 및 전송하는 기능을 담당하게 되며 UDDI4J는 UDDI Publish Module

을 지원하는 라이브러리에 해당된다. SOAP 프로토콜을 통해 서비스 제공자와 요청자간의 메시징 프로토콜의 정의가 가능하다. 또한 XML기반이므로 구현 언어와 운영체제에 상관없이 원격의 서비스를 호출할 수 있는 장점을 지닌다.

● **Web Service Runtime Environment**은 웹서비스를 위한 실행 환경을 제공한다.

● **Java Runtime Environment**은 자바로 구현된 실행 환경을 제공한다.

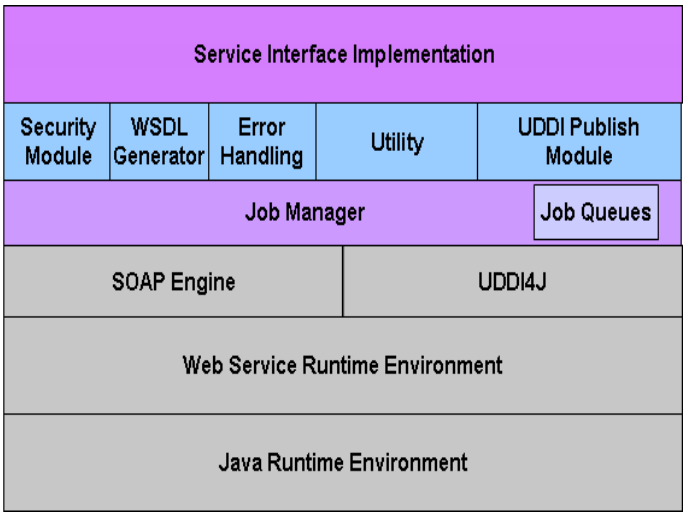

그림 2 레거시 시스템을 웹서비스화 하기 위한 래퍼

4. 사례연구

SOA기반의 웹서비스는 XML기반 언어인 WSDL을 사용하여 인터페이스를 설명한다. 이로 인해 기존의 CORBA, DCOM보다 동적이고 유연한 인터페이스 시스템을 구축할 수 있다. 본 연구에서는 e-엔지니어링 프레임워크의 레거시 시스템을 웹서비스에 통합하는 과정에서 비기능적 요소에 해당하는 보안성, 신뢰성, 상호운용성, 유연성 향상에 중점을 두고 구현하였다.

그림 3은 다수의 에이전트간의 상호작용과 의사소통을 위해 e-엔지니어링 프레임워크를 확장하여 웹서비스를 적용한 통합 시스템 구조도이다. e-엔지니어링 프레임워크는 웹 상의 여러 권한을 가지고 있는 각각의 사용자를 위한 인터페이스 및 프리젠테이션 계층, 공학적 문제를 분할하고 수행하기 위한 에이전트 그룹 계층, 그리고 분산된 엔지니어링 자원을 웹 서비스로 통합하기 위한 계층으로 구성하였다[10]. 엔지니어링 프레임워크 아키텍처에서 PAS Process와 UDDI Registry, 엔지니어링 서비스 서버(ESS: Engineering Service Server)는 서비스 지향 아키텍처에서 서비스 사용자, 서비스 저장소, 서비스 제공자에 대응된다.

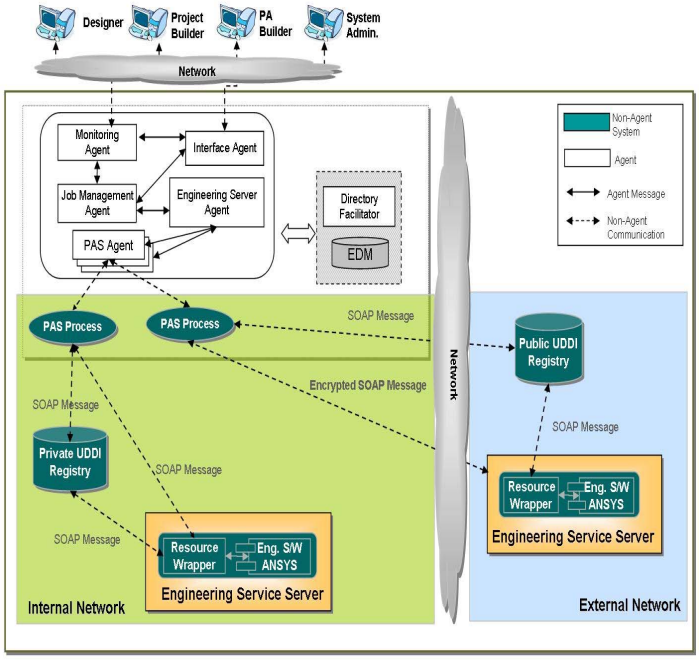

그림 3 e-엔지니어링 프레임워크의 시스템 아키텍처

표 1은 SOA와 e-엔지니어링 프레임워크의 연관 관계를 나타낸다.

표 1 SOA와 e-엔지니어링 프레임워크의 연관 관계

e-Engineering	SOA	기능
PAS Process	Service Consumer	• 엔지니어링 문제를 처리하고 해석하기 위한 프로세스로 PAS 에이전트에 의해 생성되고 PAS 에이전트에 의해 전달받은 서비스를 수행
UDDI Registry	Service Registry	• 서비스 등록 및 발견 기능을 수행하며 PAS 프로세스와 Resource Wrapper 사이에서 서비스 중계자 역할을 담당 • UDDI 사용
엔지니어링 소프트웨어 및 Resource Wrappers	Service Provider	• 다양한 엔지니어 소프트웨어들이 서비스로 모델링 • 랩퍼(Wrapper)를 사용하여 소프트웨어의 상태를 모니터링하고 PAS 프로세스의 요청에 응답하는 역할 담당

그림 4는 레거시 시스템과 웹서비스간의 작업 절차를 나타낸다. 레거시 시스템이 특정 인터페이스를 사용하던 문제점은 HTTP(Port 80)를 통해 SOAP 메시지를 교환하는 인터페이스를 포함하여 독립적인 운영이 가능하게 하였다. 또한, 보안성 향상을 위해 사용자 권한 부여 및 암/복호화를 통한 메시지 보안을 강화하였다. 로그를 통

한 에러 핸들링을 위해 데이터베이스에 로그를 기록하여 정전과 같은 외부적인 문제 및 네트워크로 인한 내부적인 문제에 대해 대처할 수 있도록 장애 발생 시점 이전의 단계부터 시스템이 재시작되도록 구현되었다. 작업 관리자를 통해 다수의 서비스 요청들이 순차적으로 처리되도록 큐 관리 모듈을 두었으며 이러한 일련의 과정들은 엔지니어링 소프트웨어 드라이버를 통해 레거시 시스템과 연동된다.

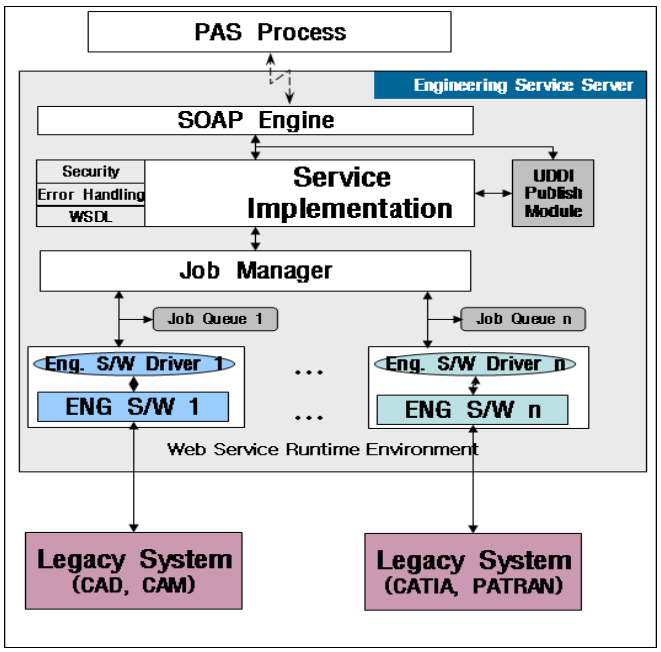

그림 4 레거시 시스템과 웹서비스간의 작업 절차

본 사례연구에서는 그림 4에서 간략하게 설명되었던 비기능적 요소를 e-엔지니어링 프레임워크의 상황에 맞게 적용한 구현 사례를 소개한다.

4.1 웹 서비스 보안성 향상을 위한 통합 방안

SOA의 대표적인 특징인 약 결합(Loosely Coupled)으로 인해 레거시 시스템과의 통합 시 기존 보안 구현의 약점들까지 외부로 노출될 수 있다. 이를 해결하기 위해 WS-Security 및 WS-Trust 표준을 활용하여 웹 서비스 보안과 시스템에 Crash 또는 Fault 발생 시 이를 신속하게 복구할 수 있도록 웹서비스의 신뢰성 향상을 위한 회복 기법을 구현했다. 로그를 기반의 신뢰성 향상을 위한 회복 기법은 [11]에 자세히 설명되어 있다. 웹서비스의 보안성 향상을 위해 구현한 개별 인증방법과 사용자 권한별 서비스 제공방법을 적용한 Security Module의 구현을 소개한다.

개별 인증을 통한 접근 제어방법은 다음 그림 5의 형태로 개별 인증과정을 거친다.

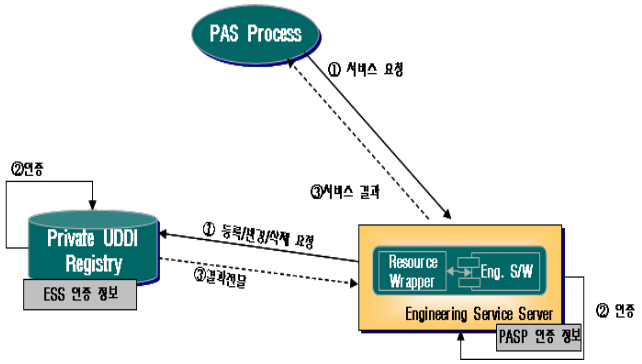

그림 5 개별 인증을 통한 인증 방법

PAS(Process/Analysis Server) Process에서 Engineering Service Server에 서비스 요청 시 임의의 사용자가 접근할 수 없도록 하기 위해 엔지니어링 서비스 서버에 ID와 패스워드의 인증정보를 암호화하여 서비스를 이용하는 사용자의 인증을 수행할 수 있도록 시스템을 확장하였다. 개별 인증을 위해서 데이터베이스를 추가하였으며 데이터베이스는 사용자의 ID, 패스워드 및 서비스 사용 권한 레벨 등의 각종 정보를 포함하고 있다.

그림 6은 사용자 레벨, 제공 서비스 사이에 관계 정보를 이용하여 사용자가 서비스를 요청하였을 때 권한을 검증하여 서비스를 제공/거절하는 방법을 도식화한 것이다. 이를 통해 서비스 사용자의 서비스 요청 시 서비스 제공자는 해당 사용자의 레벨에 맞는 응답을 해주게 되며 제공하는 서비스의 신뢰성을 향상시킬 수 있다.

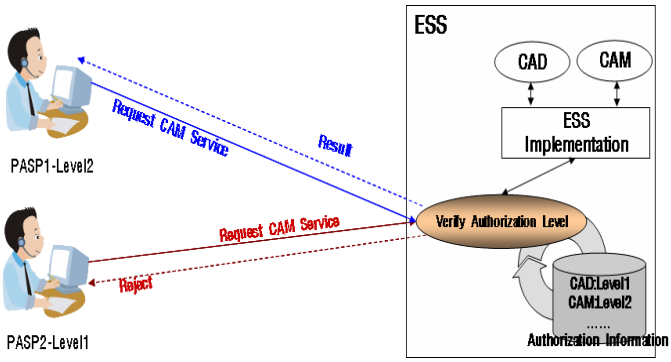

그림 6 사용자 권한 별 서비스 제공 방법

사용자의 권한 별 서비스를 제공하기 위해 데이터베이스의 사용자 테이블에 레벨 항목을 추가하여 사용자의 레벨을 분류하고 레벨에 따라 제공 서비스에 차이를 두게 된다. 웹 서비스 서버에 접속한 서비스 사용자의 정보를 저장하고 저장된 정보를 통해 사용자의 가입 여부를 판별한다. 또한 사용자 개인의 권한을 부여하여 레벨에 따른 서비스를 제공 받도록 구현한다.

그림 7은 데이터베이스에 저장된 사용자의 정보 및 레벨을 관리자용 사용자 인터페이스에 연동하여 나타낸 그림이다.

Userid	Password	UserIP	GroupName	Official	Level	No
kimm	1223	203.247.63.41	CNU	kimm	A	1
park	0803	198.188.46.198	기계연구원	웹서비스	B	2
root	0803	198.188.46.198	CNU	kimm	C	3

UI 와 데이터베이스에 저장

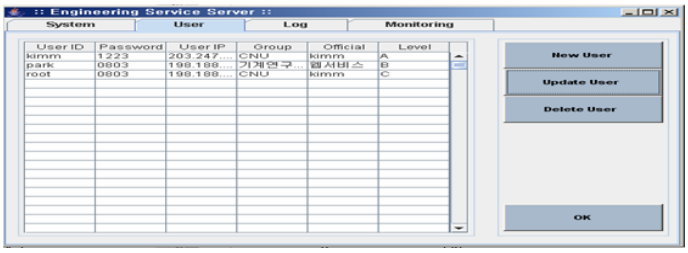

그림 7 사용자의 정보 및 레벨 관리 인터페이스

4.2 웹 서비스 신뢰성 향상을 위한 통합 방안

엔지니어링 프레임워크의 신뢰성 향상을 위해 각 시스템 구성 요소에 장애가 발생할 경우 로그 기반의 회복 기법을 적용하여 장애발생 이전의 상태로 정상적인 복구를 하기 위한 Logging Module을 구현하였다. 이는 PAS 프로세스와 엔지니어링 서비스 서버의 상호 통신 중에 내외부적인 문제가 발생하게 될 경우, 이를 신속하게 복구하기 위해 데이터베이스에 로그를 기록하고 이를 모니터링한다.

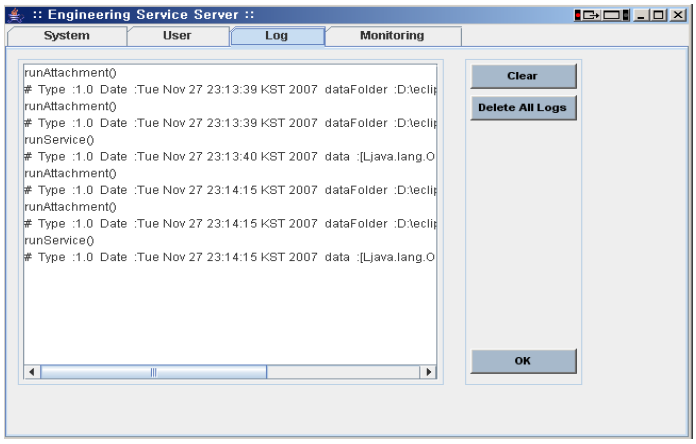

그림 8 작업로그 및 모니터링을 위한 인터페이스

5. 결론 및 향후연구

본 연구에서 제안한 통합 방안은 OASIS, W3C, WS-I 등의 표준 기관에서 정해진 웹서비스 명세를 준수하였다. 웹서비스는 표준 인터넷 프로토콜의 이용, 약 결합, SOAP프로토콜의 사용, WSDL 명세 등의 장점을 기반으로 레거시 시스템과 통합되기 때문에 복잡성을 줄일 수 있을 것으로 판단된다. 또한, 레거시 시스템에 웹서비스를 적용함으로써 고가의 레거시 시스템을 공동 활용하고 이로 인해 구매/유지비용의 감소 및 보안성, 신뢰성, 유연성, 상호운용성의 향상을 가져올 것으로 기대된다. 향후에는 최근 기업 간 통합 시스템 구축과 관련되어 주목받고 있는 시맨틱 웹(Semantic Web)을 적용한 차세대 통합 환경에 대한 연구를 시행할 예정이다.

Acknowledgement

본 연구는 한국기계연구원 기본사업의 지원으로 이루어진 결과임

참 고 문 헌

[1] Shantanu Bhattacharya, "http://www.ibm.com/developerworks/webservices/library/ws-soa-legacyapps/?S_TACT=105AGX55&S_CMP=EDU" 2007.12

[2] Calvin Lawrence, "http://www.ibm.com/developerworks/webservices/library/ws-soa-adaptleg/?S_TACT=105AGX55&S_CMP=content" 2007.6

[3] http://www.sun.com

[4] http://www.ibm.com

[5] http://www.oasis-open.org

[6] http://www.w3.org

[7] http://ws-i.org

[8] 백종현, 김형석, 김영호, 한상인, "SOA 플랫폼 분석과 시장전망", 2007 한국정보과학회지, 제25권 제1호, pp40-46,2007

[9] 김수형, 진승헌, "Identity Metasystem 기술 및 동향",전자통신동향분석, 제22권 제3호, pp144-153, 2007

[10] Seung Hak Kuk, Il Noh Oh, Hyeon Soo Kim, Jai-Kyung Lee, and Seong-Whan Park", An e-Engineering Framework Based on Service-Oriented Architecture and Agent Technologies", The Proceedings of the 2007 11th International Conference on Computer Supported Cooperative Work in Design", volume one, pp.429-434, 2007

[11] 김동욱, 국승학, 김현수, 이재경, "웹 서비스 기반 e-엔지니어링 프레임워크의 신뢰성 향상을 위한 회복 기법", 정보과학회 논문지: 컴퓨팅의 실제 및 레터 제14권 제1호, pp.76-80, 2008

[12] 오라클, "웹서비스 보안: SOA 보안을 위해 필요한 요소,"오라클 기술백서, http://www.oracle.com/technology/global/kr/tech/standards/pdf/security_kor.pdf, 2006, 10

[13] Andréa Matsunaga, Maurício Tsugawa and José A.B. Fortes,"Integration of Text-based Applications to Service-Oriented Architectures for Transnational Digital Government",The Proceedings of the 8th Annual International Digital Government Research Conference