UDDI 2.0 레지스트리의 설계 및 구현*

김영선[0] 유수진 이경하 이규철
충남대학교 컴퓨터공학과
{yskim[0], sjryu, bart, kclee}@ce.cnu.ac.kr

Design and Implementation of an UDDI 2.0 Registry

Young-Sun Kim[0], Su-Jin Ryu, Kyong-Ha Lee, Kyu-Chul Lee
Dep't. of Computer Engineering, Chungnam National Unversity

요 약

웹 서비스는 표준화된 XML 메시지를 통해 네트워크상에서 접근 가능한 연산들의 집합을 기술하는 인터페이스로서 많은 업계의 지원과 함께 여러 분야에 빠르게 적용되고 있다. 이러한 웹 서비스는 현재 SOAP, WSDL, UDDI 등의 개방형 표준들로 구성되어 있다. 이 논문에서는 SOAP 기반 XML API 호출을 통해 UDDI와 상호 작용하여 웹 서비스를 등록하고 검색하기 위한 UDDI 2.0 레지스트리 시스템의 설계와 구현을 소개한다.

1. 서 론

웹 서비스는 표준화된 XML 메시지를 통해 네트워크상에서 접근 가능한 연산들의 집합을 기술하는 인터페이스[1]로서 많은 업계의 지원과 함께 여러 분야에 빠르게 적용되고 있다. 이러한 웹 서비스는 현재 SOAP, WSDL, UDDI 등의 개방형 표준들로 구성되어 있다.

UDDI(Universal Description, Discovery and Integration Service)는 웹 서비스에 대한 디렉토리 서비스를 지원하기 위해 개발된 표준으로 웹 서비스를 등록, 검색, 발견하기 위한 메커니즘을 제공한다. 또한 이기종 플랫폼 간의 프로그래밍으로 웹 서비스를 등록하고 발견하기 위해 초기에 제안된 SOAP 1.1 메시지 규격을 적용하고 있다. UDDI 레지스트리는 웹 서비스에 있어서 비즈니스와 각 비즈니스의 서비스에 대한 부가정보 등을 제공해주기 위한 저장소로서, 비즈니스에 대한 정보를 등록하고 검색하며 기업간의 거래를 수행할 수 있도록 지원하는 역할을 한다.

본 논문에서는 SOAP 기반 XML API 호출을 통해 UDDI와 상호 작용하여 웹 서비스를 등록하고 검색하기 위한 UDDI 2.0 레지스트리 시스템의 설계와 구현을 소개한다.

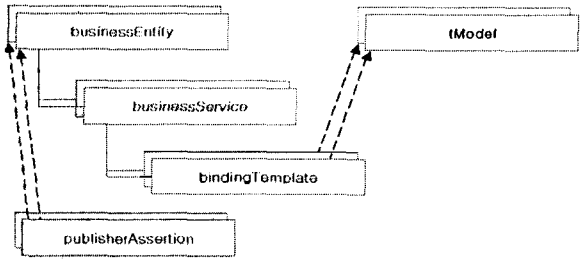
2. UDDI 2.0 레지스트리 설계

UDDI 레지스트리에 대한 질의는 크게 3가지로 나누어진다[7]. 첫 번째는 웹 서비스를 제공하는 서비스 제공자의 입장에서 비즈니스에 대한 이름, 주소, 전화번호, 그리고 접근 정보 등과 같은 내용을 등록하기 위한 Publish, 두 번째는 어떤 서비스와 비즈니스를 할 것인가의 서비스 요청자 입장에서 질의 가능한 Find . 마지막으로 원하는 서비스를 검색했다면 서비스와 상호 거래를 위해 접근하기 위해 사용되는 정보들에 대한 질의인 Bind이다.

* 본 연구는 소프트웨어 연구센터와 BK21 충남대학교 정보통신 인력양성사업단의 지원을 받았음.

2.1 UDDI 2.0 데이터 구조

UDDI 버전 2.0에서는 데이터 구조를 [그림 1]에서 보이는 바와 같이 5가지의 구성 요소로 정의하고 있다[3].

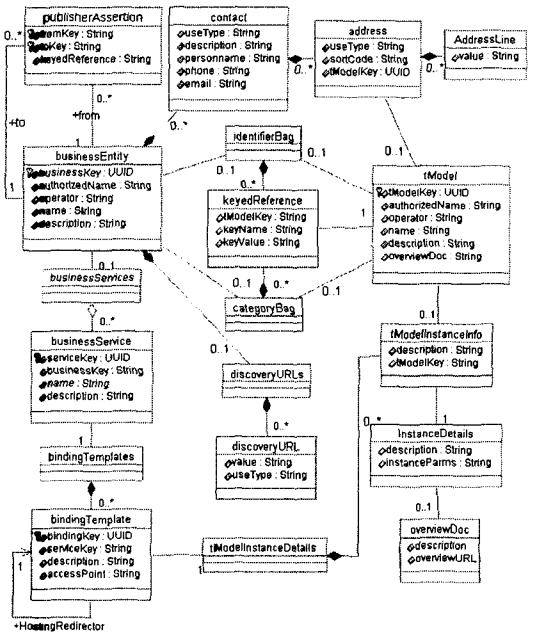


[그림 1] UDDI 2.0 데이터 구조

businessEntity 데이터 구조는 특정 서비스를 UDDI 레지스트리에 등록하고자 하는 비즈니스 개체에 대한 정보를 표현한다. 이 businessEntity는 비즈니스 개체에 대한 정보를 표현하기 위한 최상위 구조이며, 비즈니스 개체가 제공하는 서비스는 이 businessEntity의 하위에 존재한다. businessService는 businessEntity의 하위 데이터 구조로 등록되는 서비스에 대한 논리적 정보를 표현하는데 이용된다. 하나의 businessEntity에는 여러 개의 businessService가 존재할 수 있으며, businessEntity와는 별도의 UUID 키를 가진다. bindingTemplate은 웹 서비스에 대한 기술적 정보를 표현하기 위해 사용되는 데이터 구조로써, 실제 웹 서비스의 실행에 필요한 기술 정보들을 담고 있다. tModel 데이터 구조는 UDDI에서 표현되어지는 모든 객체들에서 사용되어지는 메타 데이터의 기술을 위해 이용된다. UDDI 레지스트리에서 tModel을 사용하는 주요 목적은 웹 서비스를 이용하기 위해 필요한 기술 지문(technical fingerprint)들을 표현하는데 있다. 이러한 기술 표준으로는 통신 프로토콜이나, 교환되어지는 메시지의 형식, 서비스 이용규칙 등이 이에 해당된다. UDDI 버전 2.0에서는

기업간의 관련정보를 저장하기 위한 데이터 구조로 publish erAssertion 타입이 추가되었다.

다음의 [그림 2]는 UDDI 2.0 데이터 구조를 UML 클래스 다이어그램으로 작성한 것이다. 이 다이어그램은 서로 다른 기능을 가진 기본적인 데이터 엘리먼트들 사이의 전체적인 관계(relationship)를 표현한다.



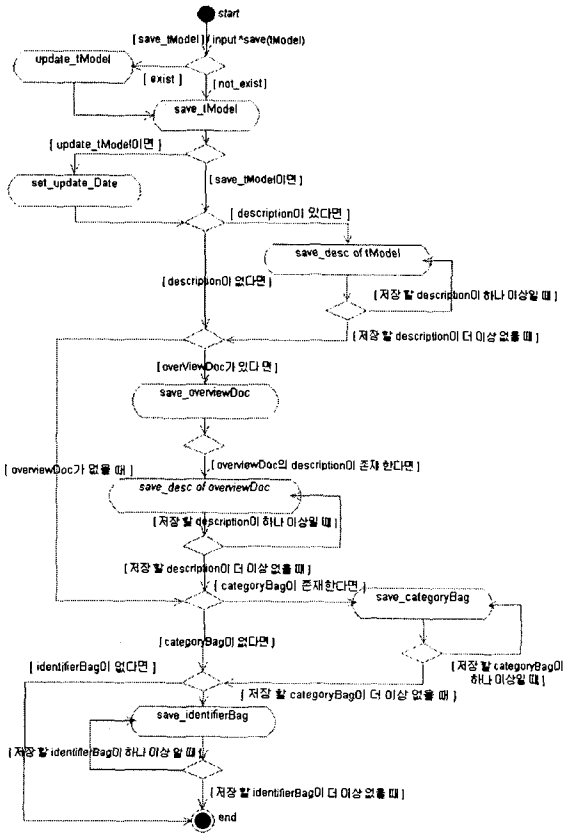
[그림 2] UDDI 2.0 데이터 구조

2.2 UDDI API

UDDI API는 서비스 요청자 또는 서비스 제공자가 UDDI 레지스트리를 이용하는데 사용된다. UDDI 2.0 API는 크게 publishing API와 inquiry API로 나뉜다. Publishing API는 서비스 제공자가 비즈니스 개체 또는 비즈니스 서비스, 기술모델을 UDDI 레지스트리에 등록하거나 삭제하는데 이용되며, inquiry API는 UDDI 레지스트리에 등록된 비즈니스 관련 정보를 검색하는데 사용된다. 예를 들면 일반적으로 inquiry API는 find_xxx, get_xxx 형식의 이름을 가지며, publishing API는 save_xxx, delete_xxx 형식의 이름을 가진다[4].

각각의 API 들의 동작 및 데이터 흐름을 정의하기 위해 또한 UML을 사용하였다. 본 논문에서는 지면제약으로 아래의 [그림 3]에서 tModel을 저장하기 위해 호출되는 API인 save_tModel의 내부 알고리즘만을 보여준다.

save_tModel의 예와 같이 프로그래머를 위한 API구현에 있어서 빠른 이해를 돕고 알고리즘 구현에 용이하도록 UML 활동 다이어그램을 사용하여 UDDI 2.0 의 API들을 설계하였다.

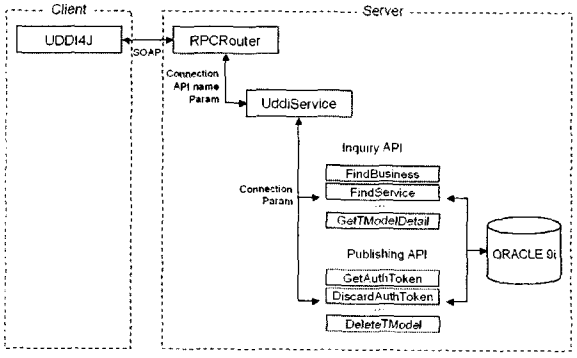


[그림 3] save_tModel의 UML 활동 다이어그램

각각의 입력 속성들의 유무나 하나 이상의 하위 엘리먼트가 반복되는 경우의 처리에 따른 처리과정을 명시하였으며, API 입력요소들에 따른 데이터의 처리에 발생되는 SQL문장들을 명시하였다.(위의 그림에서는 생략되었음)

2.3 UDDI 2.0 레지스트리 시스템 구성

[그림 4]는 UDDI 2.0 레지스트리 서버의 프로그램 수행에 관한 전체적인 시스템 구조이다.



[그림 4] UDDI 2.0 레지스트리의 시스템 구조

먼저 레지스트리 서버에 접근하기 위해 클라이언트에서는 호출할 비즈니스 프로세스에 관련된 요청(request)메시지를 작성해야 한다. 메시지 작성은 직접 메시지를 작성하거나 UDDI4J와 같은 패키지를 사용해서 서버쪽 서블릿 RPCRouter로 SOAP 메시지를 전달할 수 있다.

다음으로 RPCRouter에서 클라이언트로부터 받은 SOAP 메시지에서 API 이름과 API에 따른 파라미터 정보들을 추출해 UddiService로 전달하게 된다. 또한 각각의 API에서 데이터베이스 접근에 사용할 데이터베이스 커넥션을 미리 생성을 해 UddiService로 보내어지게 된다. 즉 RPCRouter는 호출될 UDDI API의 종류를 분별하기 위한 선처리 프로세서의 역할을 한다. 또한 요청 서비스의 응답메시지를 생성해 다시 클라이언트로 보내는 일종의 라우터(router) 역할을 하게 된다.

UddiService의 기능으로는 RPCRouter에서 전달 받은 API 이름으로 해당 서비스요청에 따른 API를 호출한다. 이때 전달 받은 파라미터 정보와 데이터베이스 커넥션을 해당 API에 인자로 넘겨주게 된다.

3. UDDI 레지스트리 구현

UDDI 2.0 레지스트리의 구현 플랫폼으로는 운영체제는 윈도우 2000을 사용하였으며 자바 플랫폼을 기반으로 컴파일러는 JDK 1.4, 웹 서버로는 Tomcat 4.0.4를 기반으로 시스템을 구현하였다. 클라이언트와 서버간의 SOAP 메시지 통신에서 클라이언트의 요청메시지 작성은 IBM에서 개발된 UDDI4J를 사용하여 메시지 작성을 하였고, 클라이언트와 서버간의 SOAP 메시지 전송은 아파치 SOAP을 사용하였으며 전송과정에서 XML문서의 파싱을 위해 Xerces를 사용하였다. 또한 서비스에 관련된 데이터의 저장과 관리는 관계형 데이터베이스는 오라클9i를 사용하였다.

레지스트리에 비즈니스 개체, 서비스, 분류정보등의 정보를 저장하기 위해서 선행되어야 할 것은 사용자에 대한 인증과정을 수행해야 한다. 그러기 위해서는 사용자 ID와 패스워드 정보를 담고 있는 SOAP메시지를 작성한 후 서버(RPCRouter)에 보낸다. 만약 사용자 인증에서 실패를 했을 경우 응답메시지는 실패를 나타내는 실패(fault) 코드 메시지를 클라이언트에 보낸다. 다음의 [그림 5]은 레지스트리 시스템의 get_authToken API를 호출하는 요청(request)메시지의 예이다.

```
<s:Envelope
   xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
   <get_authToken
            generic="2.0" xmlns="urn:uddi-org:api_v2"
            userID="uddiuser"
            cred="uddipw">
   </get_authToken>
  </s:Body>
</s:Envelope>
```

[그림 5] get_authToken API 요청(request) 메시지

위의 예에서 사용자 ID가 "uddiuser"이고 패스워드가 "uddipw"인 사용자에 대한 인증정보를 서버에 요청을 하게 되면 서버의 구현된 get_authToken API에서는 데이터 베이스의 사용자 인증테이블에서 사용자를 확인하고, 인증토큰이 담긴 응답메시지를 작성해 다시 클라이언트에 메지시를 전송하게 된다.

다음의 [그림 6]은 인증토큰 요청에 대한 응답메시지 예이다.

```
<?xml version='1.0' encoding='UTF-8'?>
<Envelope
   xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
   <authToken generic="2.0">
    <authInfo>uddiuser62107</authInfo></authToken>
  </Body>
</Envelope>
```

[그림 6] 응답(response) 메시지

본 논문에서 구현한 레지스트리에서는 한명의 사용자 ID에 다중 사용자 접근을 방지하기 위한 방안으로 인증토큰의 생성은 사용자 ID에 임의의 5자리 10진수를 추가한 스트링값으로 생성을 한다.

4. 결론 및 향후 과제

본 논문에서는 기업간의 전자거래에서 요구되는 웹 서비스의 등록, 검색을 위해 필요한 UDDI 2.0 표준안에 충실한 UDDI 2.0 레지스트리의 설계 및 구현을 수행하였다. UDDI 레지스트리는 웹 서비스에 대한 정보를 일관성 있고 안정적으로 검색할 수 있어야 한다. 그러기 위해서는 국제적인 웹 서비스 표준 규약에 적합해야 하며, 한국의 실정에 맞는 UDDI 레지스트리의 신속한 구축과 웹 서비스의 인프라를 구성해야 하는 문제가 과제로 남는다.

향후 계획으로는 2002년 7월에 Open Draft로 공개되어 있는 UDDI 버전 3.0 표준안에 대한 레지스트리 개발을 지속할 것이다.

참 고 문 헌

[1] Header Kreger, IBM Software Group, "Web Services Conceptual Architecture(WSCA 1.0)", http://www-4.ibm.com/software/solutions/webservices/pdf/WSCA.pdf, May. 2001.

[2] 이경하, 이규철 "웹 서비스의 향후 발전 방향", 한국정보처리학회지 7월호, 2002년.

[3] "UDDI Version 2.0 Data Structure Reference", http://www.uddi.org/pubs/DataStructure-V2.00-Open-20010608.pdf, uddi.org, June 2001

[4] "UDDI Version 2.0 API Specification", http://www.uddi.org/pubs/ProgrammersAPI-V2.00-Open-20010608.pdf, uddi.org, June 2001

[5] Ariba Inc., Microsoft Corp. "UDDI Technical White Paper", http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf, uddi.org, September 2000.

[6] 김영선, 유수진, 김미혜, 이경하, 이규철, "UDDI 등을 이용한 공공부문 전자거래 구현 방안 연구" 중간 보고서, 한국전산원, 충남대학교, 2002.8

[7] Venu Vasudevan, "A Web Services Primer", http://www.xml.com/pub/a/2001/04/04/webservices/, xml.com, 2001.4