웹서비스를 위한 WSDL 리포지토리 설계

최유순* · 박종구*

Web Service Method using WSDL Repository

Yue-Soon Choi* · Jong-Goo Park*

이 논문은 2006년도 원광대학교 연구비를 지원받았음

요 약

웹 서비스는 분산 컴퓨팅의 차세대 주자로 인터넷 상에서 표준 기술을 통해서 이용될 수 있는 모든 비즈니스 처리를 의미하는 분산 솔루션이다. 웹 서비스는 웹 인터페이스를 통해서 기능을 수행한다. 본 논문에서는 이러한 웹 서비스 절차를 단축시켰다. WSDL을 저장하기 위한 데이터베이스로 WSDL Repository를 이용하였다. 서비스 제공자가 서비스를 등록할 때 서비스에 대한 정보를 UDDI Registry에 저장하게 하고, 이 때 WSDL을 같이 전송하도록 구현했다. WSDL Repository는 WSDL 뿐만 아니라 웹 서비스에 대한 서비스 정보를 갖고 있는 UDDI도 저장한다. UDDI Registry에 저장된 서비스에 대한 정보와 WSDL의 연결은 UDDI에서 데이터 필드로 구성했다.

ABSTRACT

Web service, the next generation of distributed computing, is a distributed solution that handles all businesses through standard techniques in the internet. Web service performs its function using web interface. The goal of this thesis is to reduce network overloading, to manage WSDL efficiently, and to provide convenience to service users by simplifying the web service procedure. Web service system proposed in this thesis is based on WSDL Repository that can include UDDI and store WSDL. WSDL Repository manages WSDL by file system and has UDDI Registry embedded within it. Because this system is based on WSDL Repository, Web service supplier must register WSDL when he registers services. Then, users can receive WSDL too when he searches for services.

키워드

웹서비스 UDDI WSDL

I. 서 론

웹 서비스는 인터넷 상에서 표준기술을 통해서 이용될 수 있는 모든 비즈니스 처리를 의미한다. 웹 서비스는 웹 혁명의 다음 단계로서, 분산 컴퓨팅의 차세대 주자라고 할 수 있다. 웹 서비스는 웹 인터페이스를 통해서 기능을 수행한다. 기존의 클라이언트와 서버가 어떤 기술, 어떤 언어, 어떤 장비라고 하더라도 실행이 가능하다. 개발자에게는 자신이 선호하는 기술을 선택할 수 있도록 해준다[1,2]. 각 서비스에서 정의되는 것은 인터페이스

* 원광대학교 전기전자정보공학부

접수일자 : 2006. 11. 28

일 뿐이다. 그러므로 클라이언트가 자바 서블릿이든 Visual Basic .NET이든 Perl CGI 또는 WAP이 가능한 이동전화이든 간에 이러한 것들은 별로 중요하지 않게 되었다. 이 클라이언트 기술들이 웹 서비스에 접속하는 방법은 단지 하나, HTTP 위에 있는 SOAP이다[3]. 인터넷에서 서버 어플리케이션에 접속하는데 필요한 것은 SOAP 메시지로 인코딩된 XML 텍스트 프로세스 뿐이다[4,5].

웹 서비스는 기능 재사용성을 제공하고 있다. 기능 재사용성은 어플리케이션에서 사용하는 특정 기능을 수행하는데 다른 시스템을 사용할 수 있는 능력을 제공한다. 개발자가 기능을 찾을 때 어떤 특정한 기술 사이에서 고민하는 대신에 웹 서비스는 정확한 기능을 제공하지만 특정 기술에 얽매이지 않게 한다[6]. 이에 따라 웹 서비스에 대한 관심도 고조되고 있다.

이상과 같은 맥락에서 본 논문은 웹 서비스를 효율적으로 지원하기 위한 방안에 대하여 연구하고자 한다. 현재 웹 서비스를 지원하기 위한 여러 가지 표준이 지정되었다. 서비스 제공자는 요구되는 기능을 연관시키는 웹 서비스들을 보관하기 위해서 UDDI(Universal Description, Discovery and Integration) 레지스트리를 사용한다[7]. UDDI 레지스트리는 개발자들이 정확하게 자신의 일을 처리하는데 도움을 주고 있다.

UDDI는 비즈니스 정보와 서비스의 내용을 저장하는데 사용되는 레지스트리이다. 웹 서비스 사용자는 UDDI로 등록된 비즈니스를 통해서 웹 서비스를 검색한다. UDDI 레지스트리 자체는 여러 웹 서비스를 외부로 드러내어 어떤 클라이언트라도 표준 SOAP 메시지를 사용하는 레지스트리를 찾을 수 있다.

WSDL(Web Service Description Language)은 웹 서비스가 표현하는 메소드와 각각의 메소드에 사용되는 파라미터들이 무엇인지 정의한다. WSDL 파일은 어떤 SOAP 메시지가 오고가는지 제약을 가한다. SOAP 메시지는 그 자체로서 XML로 인코딩되고, 서버에 요청되는 메소드와 클라이언트에 반환되는 데이터를 가지고 있다.

기존의 웹 서비스 구조에서는 서비스 제공자가 WSDL을 보유하고 있다. 서비스 사용자가 UDDI에서 서비스에 대한 위치정보를 확인한 뒤에 서비스를 이용하는데 필요한 정보를 얻기 위하여 서비스 제공자에게 WSDL을 요청하고 있다.

본 연구는 서비스를 UDDI에 등록할 때, 등록된 서비스의 위치정보와 WSDL을 관리하는 별도의 리포지토리를 두고자 한다. 리포지토리에서는 UDDI와 WSDL 문서를 보관하고 있어, 웹 서비스 검색 요청이 있을 때, 서비스 위치 정보와 WSDL 문서를 같이 전송하고자 한다. 리포지토리를 이용하므로서 요청된 서비스의 이용 절차를 간소화하기 위한 방법을 제시하는데 주된 목표를 두고 있다.

II. 관 련 연 구

본 장에서는 웹 서비스의 정의와 장점에 대하여 기술하고, 웹 서비스를 이용하는 절차 및 방법에 대해 고찰한다. 그리고 웹 서비스 제작시 사용되는 기술들인 XML, SOAP, WSDL, UDDI 등의 개념과 기존 UDDI 서비스의 문제점에 대해서 검토하고 살폈다.

2.1. 웹 서비스 개요

웹 서비스는 본격적인 비즈니스의 장이 되었다. 서비스 제공자는 웹사이트에 좀더 많은 사용자들이 접속할 수 있게 하는데 많은 노력을 기울였다. 로드밸런싱 같은 기술적인 진보를 통해서 동시 접속자의 수를 증가시키고 서비스의 안정성을 높였다[8,9]. 그러나 근본적인 문제인 컨텐츠를 관리/배포하는 일은 여전히 시간과 노력을 많이 소모하는 일이었다. 그러므로 이를 체계적이고 총괄적으로 관리해주는 시스템이 필요했다.

그리하여 웹에 출판하기 위한 컨텐츠를 일정한 포맷에 의해서 관리하고 데이터베이스에 저장, 빠르게 검색하는 기능을 갖춘 시스템이 요구되었다. 이러한 일들은 XML이 있었기에 가능했다. XML은 모든 페이지를 직접 만들고 출판하는 비효율적인 방식에서 벗어나 정형화된 템플릿을 미리 정의하고 여기에 사용자가 컨텐츠의 내용을 지정하면 바로 양식에 맞춘 웹페이지를 생성하여 배포할 수 있게 되었다[10]. 이는 XML의 가능성을 보여준 것이지만 XML이 할 수 있는 일은 이것이 전부는 아니다. XML은 이 모든 차세대 웹의 핵심적인 요소이다.

이렇게 웹이 모든 곳에서 사용되면서 관련 기술도 급속도로 발전했다.

2.2. SOAP

SOAP은 플랫폼과 무관하게 서비스, 객체 그리고 서버를 접근하기 위한 XML과 HTML의 사용 방법에 대하여 정의하고 있다. 달리 말하면, SOAP은 이질적인 소프트웨어 컴포넌트간의 접착제 역할을 하는 프로토콜이다. HTTP와 XML의 동시 사용을 원할 경우, 표준적인 방법으로 경쟁적인 기술들을 연계하기 위한 메커니즘을 SOAP은 제공한다. 이것은 HTTP와 XML을 단일 솔루션으로 결합하여 전혀 새로운 수준의 상호 운용성을 제공한다. 예를 들면, 윈도우 환경에서 비쥬얼 베이직으로 작성된 단말 프로그램으로부터 UNIX 환경에서 작동하는 CORBA(Common Object Request Broker Architecture) 서비스를 손쉽게 작동시킬 수 있으며, 자바 스크립트로 작성된 단말 프로그램으로부터 대형 컴퓨터에서 작동 중인 서비스를 작동시킬 수도 있다[11].

SOAP은 표준 분산 환경인 CORBA의 복잡한 자체 내의 구조와 사용법의 어려움을 해결하기 위한 방안으로 개발되었다. CORBA가 아무리 복잡한 구조를 가지고 있더라도 기본적인 개념은 서버에서 구현해 둔 서비스를 호출해서 처리된 결과 값만 받아서 이용하는 것이기 때문에 SOAP에서 호출방법 또한 XML을 기반으로 객체의 개념을 도입하여 호출/응답 서비스가 상호 독립적이면서 상호 운용성이 뛰어나다는 장점을 가지고 있다. SOAP은 SOAP envelope와 body를 가지며 선택적으로 header를 가질 수 있다. [그림 1]에 SOAP의 구조를 나타내고 있다.

SOAP은 무상태성이므로 어플리케이션이 저장해야 할 것을 다룬다면 문제가 발생한다. 이때 세션ID를 이용해서 RPC 호출이 끝날 때까지 프로그램 상태를 기억하도록 한다. [그림 2]는 클라이언트와 서버간의 SOAP을 통한 메시지 교환을 나타냈다.

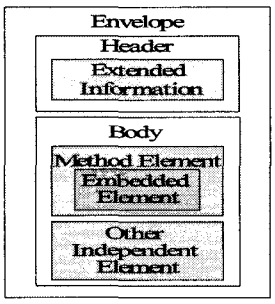

그림 1. SOAP의 기본 구조
Fig. 1. SOAP Structure

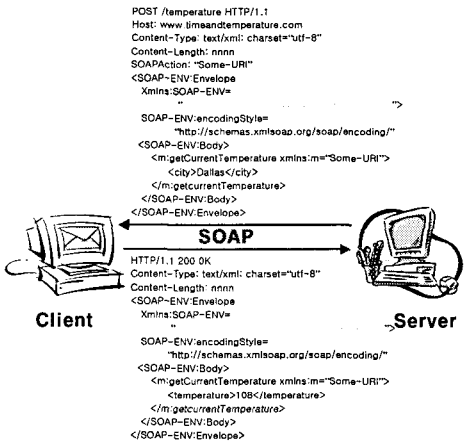

그림 2. SOAP의 메시지 전달 형태
Fig. 2. SOAP Message Transmission Form

클라이언트는 필요에 의해서 http를 경유, 객체를 생성하고 이의 메소드를 사용할 수 있다. 이런 과정이 복잡한 RPC를 거치지 않고 단지 http 상의 코멘트에 의해서 구동된다. SOAP을 쓰게 되면 단지 서버의 주소와 객체의 이름 그리고 전달되는 파라미터들만 명기하면 클라이언트의 운영체제나 환경에 상관없이 언제라도 객체를 사용할 수 있다는 장점이 있다.

XML이 어플리케이션간에 전달되는 데이터를 표준화 했다면 SOAP은 서버의 객체를 호출하는 RPC 규약을 표준화함으로써 XML을 데이터 전달 도구로 사용할 때와 유사한 이점을 개발자들에게 줄 수 있다.

SOAP의 장점은 HTTP로 보내진다는 것이다. 대부분의 방화벽은 인터넷을 통한 HTTP 트래픽을 최종 사용자(EndUser)에게 전송하는 것을 허용한다. 웹 서비스는 방화벽에서 같은 포트를 사용하면서 작동하며 여러 기능들이 사용될 수 있도록 하면서 서버 어플리케이션들이 보호될 수 있도록 해야 한다.

2.3. UDDI

UDDI는 마이크로소프트, IBM 그리고 Ariba에 의해서 제안된 프로젝트의 결과로서 기업들에게 어느 누구나 사용할 수 있는 Universal Business Registry에 그들이 제공하는 웹 서비스에 대한 정보를 알리는 것을 가능하게 한다. 선두 기업들은 UDDI가 모든 회사들의 전세계적인 기회를 디지털 혁명에 의해서 제공받게 될 B2B 온라인 상거래의 성장을 가속화 시키는데 앞장설 것이라

고 말하고 있다[12].

UDDI Business Registry를 사용하면 협력업체에 의해서 제공된 프로그래밍이 가능한 웹 서비스의 정보를 찾을 수 있다. 또한 이 레지스트리를 사용하면 모든 조직에서 제공할 웹 서비스에 대한 정보를 게시할 수 있다.

프로그래머의 인터페이스는 XML, SOAP 및 HTTP에 능숙한 프로그래머가 프로그램에서 직접 레지스트리와 상호작용할 수 있게 하는 UDDI 레지스트리 사양의 일부로 정의되어 있다.

[그림 3]은 간단한 구성요소 집합과 UDDI에 대한 이들의 관계를 보여준다. 프로그램에서 요청 서식 설정, 요청 관리자로 요청 전송 및 응답 수신을 가능하게 하는 세 개의 기본 개체가 나타나 있다. 요청 관리자는 모든 XML 및 SOAP 세부 정보, 인증 및 오류 관리를 전부 숨긴다.

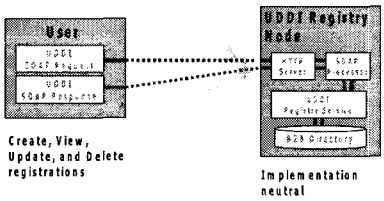

그림 3. UDDI와 SOAP
Fig. 3. UDDI and SOAP

2.4. WSDL

WSDL은 XML 포맷으로 구성되고 HTTP를 통해서 전달될 수 있다. 다시 말한다면 인터페이스를 정의하는 IDL에 해당한다. 즉, 이 서비스가 어떤 메소드, 어떤 속성을 가지며, 어떤 인수로 호출해야 하고, 어떤 방식의 리턴값을 제공하는지 알려주고 있다. 이 내용을 파악하면 클라이언트에서는 파악된 인터페이스 규약에 맞추어 호출하고 서비스를 사용할 수 있다[13]].

인터넷에서 전자상거래 뿐만 아니라 다양한 웹 서비스를 지원하기 위해서는 자료전송을 담당하는 미들웨어로 CORBA보다 훨씬 간편한 SOAP을 이용하는 추세이다. SOAP을 이용하여 자료를 전송하고 서비스를 제공하는 측은 WSDL로 UDDI에 SOAP을 통하여 서비스를 등록한다. [그림 4]는 기본 웹 서비스 구조를 보여주고 있다. 이 구조에서는 자동적으로 자료 전송이 가능하게 하고 있다.

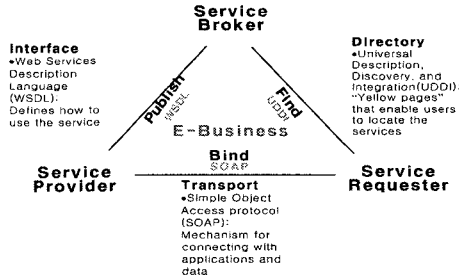

그림 4. SOAP과 UDDI, WSDL의 관계
Fig. 4. SOAP and UDDI, WSDL Relation

1) Service Provider는 서비스를 제공하고, WSDL을 통해서 서비스 브로커에게 서비스를 전달한다.
2) Service Broker는 UDDI 레지스트리에 접근 한다. Service Provider가 레지스트리로 알리게 되면, Service Requestor는 레지스트리로 요청된 서비스를 갖고 있거나, 검색하게 된다.
3) Service Requestor는 UDDI를 통해 Service Broker에서 서비스를 검색하고, SOAP을 통해서 Service Requestor에게 서비스를 바인딩한다.

UDDI 레지스트리는 비즈니스 이름과 바인딩 정보, 어플리케이션 인터페이스 지점 또는 WSDL을 포함하고 있다. CORBA나 RMI 같은 분산된 객체구조와는 달리, SOAP은 단지 Header와 Body를 가지고 있는 프로토콜이다. Service Requestor가 적당한 웹 서비스를 찾게 되면, Service Requestor로 SOAP 바인딩을 효율적으로 하기 위해서 Service Broker의 레지스트리에서 정보를 사용한다. WSDL은 웹 브라우저와 웹사이트 사이에 서로 활동을 묘사하기 위해서 HTTP GET이나 POST 방식을 사용한다.

모든 서비스들이 웹 서비스 형태로 변모하면서 웹 서비스 제공자는 서비스 목록 보관소인 UDDI에 제공할 서비스를 등록한다. 산재되어 있는 모든 서비스들은 특정한 곳에 집중되어 있기 때문에 서비스 사용자는 원하는 서비스를 쉽게 찾을 수 있다. [그림 5]는 웹 서비스 기본 구조이다.

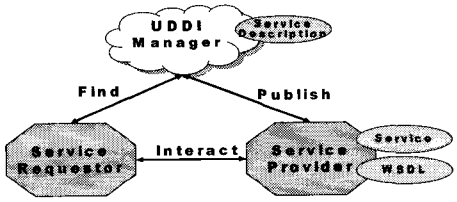

그림 5. 웹 서비스의 구조
Fig 5. Oriented Web Service Architecture

현재의 웹 서비스 시스템은 서비스 사용자가 자신이 이용하려 하는 정보를 UDDI에 접속하여 서비스의 위치를 검색하게 된다. UDDI는 해당 서비스의 주소를 서비스 사용자에게 전달하면 사용자는 위치정보를 가지고 서비스의 규격을 의뢰한다. 서비스 제공자는 요청된 서비스에 대한 문서인 WSDL을 서비스 사용자에게 전달한다.

Ⅲ. WSDL Repository 기반 Architecture 설계

현재의 웹 서비스는 UDDI의 UBR(UDDI Business Registry)가 서비스의 위치정보를 가지고 있다. 현재의 추세라면 앞으로 서비스의 양은 많아질 것으로 예측된다. 따라서 UBR은 현재보다 더욱 확대되어야 한다. 본 논문은 UBR의 기능을 확대시킬 수 있는 저장소로서 WSDL Repository를 이용한 서비스 구조를 제안하기 위한 연구이다.

WSDL Repository를 이용한 웹 서비스 아키텍처는 서비스 제공자가 서비스를 등록하면서 WSDL도 같이 등록하도록 하고 있다. 여기에는 서비스 제공자가 등록한 WSDL이 WSDL Repository에 포함되어 있다.

서비스 사용자가 서비스를 검색할 때, UDDI Manager는 서비스에 대한 정보와 서비스 사양서인 WSDL도 같이 전송한다. 그러므로 기존의 구조에서 서비스 사용자가 서비스 제공자에게 별도로 WSDL을 요청했던 절차를 한 단계 줄이게 된다.

기존 웹 서비스 구조와 WSDL Repository를 이용한 구조에서 성능 평가는 제5장에서 기술한다.

WSDL Repository는 방대한 양의 자원을 저장할 수 있다. 또한 저장공간이 커진 리포지토리는 WSDL을 보관할 수 있도록 구성되었다. 이는 서비스 사용자가 요청한 서비스의 주소를 가지고 제공자에게 WSDL을 요청해야 하는 절차를 줄이도록 하기 위한 것이다. 따라서 서비스 사용자는 UDDI에서 검색된 서비스의 위치정보와 WSDL을 동시에 제공받는다. [그림 6]은 본 연구에서 제안하는 서비스 구성도이다.

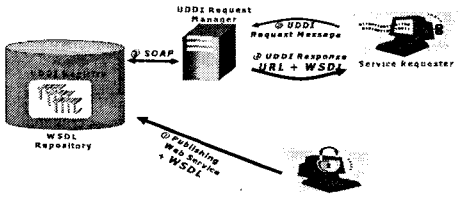

그림 6. WSDL Repository를 이용한 서비스
Fig. 6. USing WSDL Repository Service

구성도에 따른 서비스 절차를 보면, 먼저 웹 서비스 제공자는 제공할 서비스를 등록시킨다. 이때 서비스에 관련된 정보와 WSDL 파일을 전송한다. WSDL 파일은 XML 형식으로 HTTP를 통하여 전송된다. HTTP는 방화벽에 관계없이 통과할 수 있으므로 전송에 문제가 없다. WSDL 파일은 서비스를 정의할 때 자동으로 생성된다.

다수의 사용자가 서비스를 이용할 때마다 서비스 위치를 요청하고, 그 주소로 옮겨 WSDL을 요청하는 방법은 시간의 낭비와 절차의 복잡성이라는 문제점을 가지고 있다. 이와 같은 측면에서 서비스를 이용하는 사용자에게 편리함과 유용성을 제공하기 위해서 WSDL을 서비스 위치정보와 함께 WSDL Repository에 저장할 필요가 있다.

WSDL repository에는 WSDL 파일과 UDDI Registry가 저장되어 있다. 서비스 사용자는 UDDI 관리자에게 서비스 검색을 요청하고, 검색된 서비스는 서비스의 위치정보와 함께 서비스의 사양을 보관하고 있는 WSDL 파일을 같이 전송한다.

이 때에도 URL 주소가 서비스 요청자에게 전달되는 방식은 기존의 방식과 같고, 다만 WSDL 파일이 첨부되어 요청자에게 전송된다. 서비스 요청자는 WSDL을 받고서 원하는 서비스의 사양과 서비스를 이용하는 방법을 파악할 수 있게 된다.

웹 서비스를 지원하기 위한 방법으로 WSDL Repository를 이용하려면 서비스 제공자의 서비스 등록 절차가 변경되어야 한다. 서비스 제공자는 서비스 이름과 서비스 위치 외에 서비스 사양에 관한 WSDL 파일을

UDDI에 등록한다. UDDI는 비즈니스 정보와 서비스 정보, 바인딩 정보 그리고 서비스를 위한 명세를 가지고 있다. WSDL 파일은 많은 서비스를 저장하기 위하여 데이터베이스를 이용하도록 되어있다. 데이터베이스는 서비스 이름과 함께 WSDL Repository에 저장된다.

WSDL Repository는 WSDL 파일을 처리하기 위한 데이터베이스이다. [그림 7]은 데이터베이스로 되어있는 WSDL Repository 구조를 나타내고 있다. WSDL Repository는 UDDI Registry를 포함하고 있으면서 각 서비스별 WSDL 파일을 저장하고 있다. WSDL 파일은 WSDL Manager에 의하여 파일 시스템으로 관리된다. WSDL Manager는 UDDI Registry에서 가지고 있는 WSDL 파일 연결 정보를 가지고 있으면서 서비스 사용자로부터 서비스 검색요청이 있을 때, WSDL 파일과 함께 UDDI Registry에 있는 위치정보가 사용자에게 전송된다.

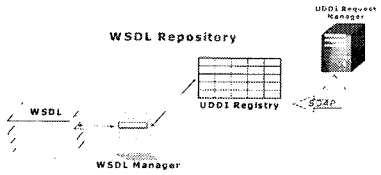

그림 7. WSDL Repository 구조
Fig. 7. WSDL Repository Structure

UDDI Registry에는 서비스 제공자에 의하여 사용자의 정보와 서비스 정보가 저장된다. 서비스 제공자는 서비스를 등록하기 위한 권한이 있어야 한다. 사용자 아이디와 암호, 이름, 전자우편 등 제공자에 관한 정보와 제공자가 속한 조직체의 정보를 입력해야 한다. 입력된 서비스 제공자 정보는 UDDI Registry에 저장되어 관리되고, 서비스에 관한 정보와 서로 연결된다. 서비스 사용자에 의하여 요청된 서비스 정보는 UDDI Manager에 의하여 UDDI Registry에서 검색을 하는데 검색요청이나, 검색결과 전송은 SOAP 형태로 전송된다.

또한 서비스 정보는 wsdlName에 의하여 WSDL 파일에 연결된다. [그림 8], [그림9]는 서비스 제공자와 서비스 정보 테이블과 테이블 관계를 나타낸 것이다. WSDL Repository로 전송된 WSDL 파일은 wsdlName에 의하여 서비스 정보와 연결된다.

서비스 제공자 테이블은 서비스를 제공하는 측의 정보를 가지고 있고, 서비스 정보 테이블은 서비스 이름을 포함하여 서비스 위치, 서비스 키 등 서비스의 일반적인 정보를 저장하고 있다. 서비스 제공자 테이블과 서비스 정보 테이블 간에는 사용자 아이디로 관계를 설정한다.

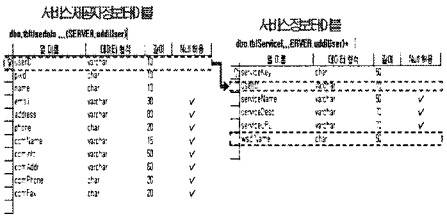

그림 8. UDDI 테이블 설계
Fig. 8. UDDI Table Design

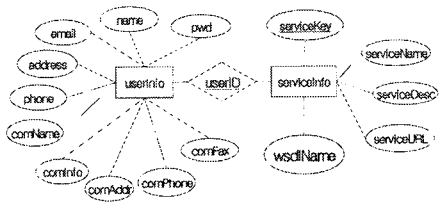

그림 9. UDDI 테이블 관계
Fig. 9. UDDI Table Relation

```
string loginID = Session["UserID"].ToString();
string ConStr = "server=Firebird;uid=uddiSA;pwd=aha2505;database=uddiUser";

SqlConnection dbConn = new SqlConnection(ConStr);
dbConn.Open();

string strUserInsertQry = "INSERT INTO tblServiceInfo";
strUserInsertQry += "(userID, serviceName, serviceDesc, serviceURL, wsdlName)";
strUserInsertQry += " VALUES";
strUserInsertQry += "('"+loginID+ "'," + "'" + txtSvcName.Text + "'," + "'" +
        txtSvcDesc.Text + "'," + "'" + txtSvcURL.Text + "'," + "'" +
        txtWsdlName.Text + "')";

SqlCommand dbCmd = new SqlCommand(strUserInsertQry, dbConn);
dbCmd.ExecuteNonQuery();
dbConn.Close();
Response.Redirect("serviceInfo.aspx");
```

그림 10. WSDL Repository 테이블 쿼리
Fig. 10. WSDL Repository Table Query

WSDL Repository에서 서비스에 대한 정보를 저장하기 위하여 데이터베이스를 이용한다. 데이터베이스에 저장할 정보는 SQL(structured Query Language)로 전달이 되며, SQL에 의하여 설정된 테이블에 해당하는 필드값들이 채워진다. [그림 10]은 WSDL Repository의 테이블 Query이다.

Ⅳ. 구 현

서비스 제공자는 WSDL Repository 및 UDDI Registry에 접근하기 위한 인증을 받아야 한다. PublisherRegist는 인증을 받기 위한 절차로 미리 인적사항을 등록하도록 하고 있다. 새로운 서비스 제공자가 등록될 때마다 사용자 아이디와 패스워드, 이름, E-mail, 전화번호, 회사이름, 회사 소개, 회사 주소, 회사 전화번호, 회사 팩스 등의 정보가 UDDI Registry에 저장된다. [그림 11]은 서비스 사용자 등록을 위한 코드를 나타내고 있다.

```
string ConStr = "server=server;uid=uddiSA;pwd=aha3595;database=uddiUser";
SqlConnection dbConn = new SqlConnection(ConStr);
dbConn.Open();

string strUserInsertQry = "INSERT INTO tblUserInfo";
  strUserInsertQry += "(userID, pwd, name, email, address, phone, comName,
    comInfo, comAddr, comPhone, comFax)";
  strUserInsertQry += " VALUES";
  strUserInsertQry += "(" + "'" + txtID.Text + "'," + "'" + txtPwd1.Text + "',"
    + "'" + txtName.Text + "'," + "'" + txtEmail.Text + "'," + "'" +
    txtAddr.Text + "'," + "'" + txtPhone.Text + "'," + "'" + txtCom.Text + "',"
    + "'" + txtComInfo.Text + "'," + "'" + txtComAddr.Text + "'," + "'" +
    txtComPhone.Text + "'," + "'" + txtComFax.Text + "'" + ")";

SqlCommand dbCmd = new SqlCommand(strUserInsertQry, dbConn);

dbCmd.ExecuteNonQuery();

dbConn.Close();

Response.Redirect("userRegOK.aspx");
```

그림 11. 서비스 제공자 등록
Fig. 11. Service Provider Publishing

ServiceRegist는 서비스 제공자가 만든 서비스를 등록하는 컴포넌트이다. 서비스 제공자는 제공할 서비스를 사용자들이 쉽게 검색할 수 있도록 검색 디렉토리인 UDDI에 서비스할 정보와 서비스가 존재하고 있는 위치를 등록하고, 아울러 WSDL Repository에 WSDL을 같이 등록한다. WSDL은 서비스의 메소드, 속성, 인수 호출 방법, 리턴 방식 등을 정의해 놓은 서비스에 대한 사양서이므로 사용자에게 전달된 WSDL에 따라서 사용자는 서비스를 요청할 수 있다. WSDL을 서비스 제공자에게 별도로 요청해야 하는 이전의 서비스 방식보다는 서비스 사용자와 제공자간의 통신 단계를 줄임으로써 시간을 단축시키는 효과가 있다.

UDDIManage 컴포넌트는 서비스 제공자로부터 서비스 등록 업무를 담당하고 있다. 서비스 사용자로부터 서비스 등록요청이 오면 등록 권한에 대한 인증절차를 거쳐 서비스를 등록 할 수 있도록 한다. [그림 12]는 UDDI에서 서비스 등록에 대한 인증절차를 확인하는 과정이다.

UDDIDiscover는 서비스 사용을 위한 컴포넌트이다. 서비스 사용자는 이용할 서비스가 어디에 있는지 검색하기 위해서 UDDI에 접속하여 위치정보를 요청한다. 이때 UDDIManage는 서비스를 검색하여 서비스에 대한 정보와 위치정보 그리고 WSDL을 SOAP 형태로 서비스 사용자에게 전송한다.

본 논문에서 구현한 WSDL Repository를 이용하기 위해서 문서 공유 시스템 서비스를 기존의 서비스 방식과 본 연구에서 제안한 방식에서 실행시켰다.

[그림 13]은 도큐먼트 공유 시스템의 클라이언트의 메인화면이다. 이 화면에서 서비스 이용자는 서비스 분야를 지정하며, 서비스 되는 파일형식을 선택할 수 있다. 본 시스템에서는 PPT, DOC, HWP, PDF 등 여러 문서 파일을 서비스할 수 있도록 구현했다. 파일형식을 지정하지 않을 경우 디폴트로 전체 문서가 지정된다.

```
if(!IsPostBack)
{
        if(Session["UserID"] == null)
        {
                Response.Redirect("NoUser.aspx");
        }
        else
        {
                string loginID = Session["UserID"].ToString();

                string strSvcSearchQry = "SELECT * ";
                strSvcSearchQry += "FROM tblServiceInfo ";
                strSvcSearchQry += "WHERE userID='"+loginID+"' ";

                dbConn = new SqlConnection(globalVars.ConStr);

                dbConn.Open();

                dbCmd = new SqlCommand(strSvcSearchQry, dbConn);

                myReader = dbCmd.ExecuteReader();

                DataGrid.DataSource = myReader;
                DataGrid.DataBind();

                myReader.Close();
                dbConn.Close();

        }
}
```

그림 12. 웹 서비스 등록을 위한 제공자 인증과정
Fig. 12. Web Service Provider Authentication

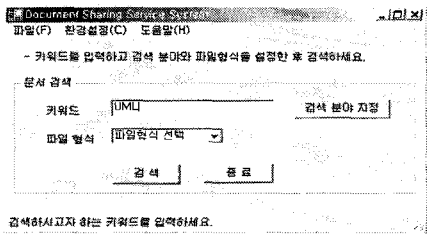

그림 13. 도큐먼트 공유 시스템 메인화면
Fig. 13. Document Sharing System

[그림 14]은 클라이언트 어플리케이션에서 서비스 분야를 지정하는 화면이다. 이 화면에 나타난 서비스 분야들은 문서가 저장된 리포지토리의 성격에 따라서 달라질 수 있다. 여기서 서비스 분야는 프로그래밍, 논문, 원격교육, 어학, 정보통신, 정보화, 강의 관련 등을 예로 설정했다. 그리고 리포지토리도 이에 따라 구성하였다. 또

한 문서가 서비스될 분야는 다중 지정이 가능하게 구현하였다. 서비스 분야를 지정하지 않을 경우에는 디폴트로 전체 분야가 설정된다.

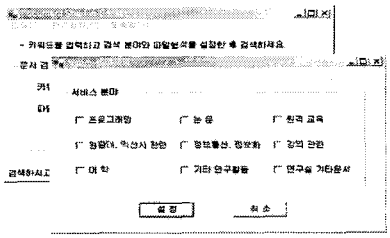

그림 14. 서비스 분야지정
Fig. 14. Service Part Selection

SOAP 요청 메시지는 파라미터로 string 타입의 키워드, string 타입의 파일 타입 그리고 XML Schema 형태의 DataSet 타입인 서비스 분야들을 넘겨준다.

[그림 15]는 실제 사용자에게 브라우징되는 도큐먼트 서비스 요청에 대한 결과이다.

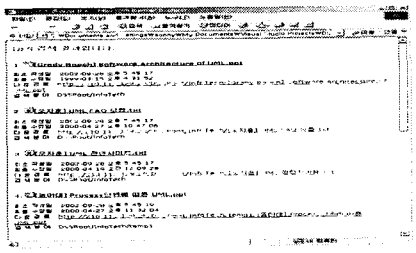

그림 15. 서비스 결과 브라우징
Fig. 15. Service Result Browsing

V. 결 론

본 논문에서는 이러한 웹 서비스 절차를 간소화시켰다. 서비스 제공자가 가지고 있는 WSDL을 서비스 등록시에 WSDL Repository에 저장하도록 구현했고 서비스 사용자가 서비스 검색과 동시에 WSDL을 전송받도록 했다.

추후 연구과제로 WSDL을 효율적으로 관리할 수 있는 새로운 방법이 제시되어야 한다. 웹 서비스가 많아지면 UDDI에서 검색되는 서비스에 대한 한계가 예상되기 때문이다. 많은 양의 서비스에 대한 또 다른 검색방법의 연구도 지속되어야 할 과제이다.

참고문헌

[1] IBM, Living in an on demand world, IBM Corporation(2002), http://www-106.ibm.com/e-business/doc/content/feature/offers/whitepaper.pdf

[2] J. Nick, OGSA - Framework for Grid Service Evolution, Presentation at OGSA Early Adopters Workshop, Argonne National Lab, CA (May 29 - 31, 2002), http://www.globus.org/ogsa/events/JeffNickOGSAFramework.pdf

[3] D. Box et al, SOAP 1.1, http://www.w3.org/TR/SOAP, C. Boyens and O. Guenther, Trust is not enough: privacy and security in ASP and Web services environments, Proc. ADBIS 2002-6th East-European COnference in Advances in Databases and Information Systems (September 8-11, 2002, Bratislava, Slovakia).

[4] Ch. Kaler (ed), Web Services Security, http://www-106.ibm.com/developerworks/web-services/library/ws-secure

[5] Skonnard, A., "SOAP:The Simple Object Accessprotocol", http://www.microsoft.com/mind/0100/soap/soap.asp/, Microsoft Internet Developer, January 2000.

[6] M. Champion, Ch. Ferris, E. Newcomer and D. Orchard, Web Services Architecture, http://www.w3.org/TR/ws-arch/

[7] Siemeon Simeonov, "Deeper into UDDI", XML Journal, volume:2 Issue:11, 2001.

[8] A. Kropp, Ch. Leue and R. Thompson (editors), Web Service for Remote Portlets, Working draft 0.85 (OASIS, 26 November 2002), http://www.oasisopen.org/committees/wsrp

[9] F. Cabrera, G. Copeland, T. Freund, J. Llein, T. Storey and S. Thatte, Web Services Transactions, BEA Systems & IBM Coporation & Microsoft Corporation, 2002, http://www-106.ibm.com/developerworks/library/ws-coor/

[10] K. Ballinger, D. Ehnebuske, M. Gudgin, M. Nottingham and P. Yendluri, Basic Profile Version 1.0, http://www.ws-i.org/Profiles/BasicProfile-1.0-WGD.htm

[11] Y.-S. Tan, B. Topol, V. Vellanki and J. Xing,

Implementing service Grids with the service domain toolkit, IBM Corporation, 2002

[12] T. Belwood et al, UDDI Version 3.0, http://uddi.org/pubs/uddi-v3.00-published-2002/0719.htm

[13] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana,WSDL1.1, http://www.w3.org/TR/WSDL

저자소개

최 유 순(Yue-Soon Choi)

2004년 원광대학교 컴퓨터공학과 공학박사
2005년~현재 원광대학교 전기전자정보공학부 강의전담교수

박 종 구(Jong-Goo Park)

1999년 동국대학교 통계학과 이학박사
1981년~현재 원광대학교 전기전자정보공학부 교수