# 차세대 인터넷서비스 기반으로서의 웹서비스에 대한 고찰

## A Review on the Web Services as the Infrastructure for the Next Generation Internet Services

안 현 수*

**Hyunsoo Ahn**

차 례



초 록

웹의 이용이 급속하게 증가함에 따라, 웹 기반의 어플리케이션 대 어플리케이션 상호작용을 체계적으로 지원하는 적절한 모형과 이러한 새로운 분산 전산 플랫폼을 기존 환경들과 효율적으로 통합할 수 있는 방안에 대한 요구가 발생했으며 이에 대한 해결책으로 웹 서비스가 등장하였다.

본 고에서는 먼저 웹 서비스의 발생 배경과 정의, 특징 등에 대해 언급한다. 그리고 웹 서비스가 작동하는 원리에 대해 상세히 설명을 한 후, 웹 서비스를 구현하는데 필요한 핵심 기술들과 실제 구현 사례들을 살펴본다. 마지막으로 웹 서비스의 향후 발전 방향에 대해 언급한다.

키 워 드

웹 서비스, 차세대 인터넷 서비스

* 한국통신 연구개발본부 선임연구원
(Senior Researcher, KT R&D Group, hsan62@kt.co.kr)

**ABSTRACT**

According to the rapid increasement of web usage, an appropriate model which supports systematically the web-based application-to-application interactions and the approach which can integrate the new distributed computing platform and the existing environments have been investigated. The web services has emerged as the solution to these needs.

This paper describes the background of the emergence of the web services and the definition and characteristics of web services. Then, after explaining the operation mechanism of web services, the core techniques which are essential to implement the web services and the implementation cases are examined. Lastly, the future of web services has been mentioned.

**KEYWORDS**

Web Services, Next Generation Internet Services, SOAP, UDDI, WSDL

## 1. 서 론

### 1.1 웹 서비스 발생 배경

웹 서비스는 웹 발전의 자연스러운 결과이다. 처음부터 글로벌하게 정보를 공유하고 배포하기 위한 하나의 방법으로서 그리고 하나의 거대한 효율적인 분산 콘텐츠 저장소로서 역할을 해온 웹은 점차 그 영역을 확장함으로써 더욱 복잡한 비즈니스 대 비즈니스 상호작용 등과 같은 클라이언트와 서버간에 더욱 고도화된 상호작용의 형식을 가능하게 하고 있다. 그러나 최근들어 진정한 웹 기반의 어플리케이션 대 어플리케이션 상호작용 모형들이 만들어지기 시작했는데 처음에는 전자 마켓플레이스와 자동화된 비즈니스 대 비즈니스 트랜젝션의 개발 부문에서 시작되었으며 최근들어서는 대규모 자원 공유를 가능케하는 부문에서도 적용되기 시작했다.

이에 따라 웹 기반의 어플리케이션 대 어플리케이션 상호작용을 체계적으로 지원하는 적절한 모형과 이러한 새로운 분산 전산 플렛폼을 기존 환경들과 효율적으로 통합할 수 있는 방안에 대한 요구가 발생했다. 즉, 모든 사람들은 동일한 운영체제, 프로그램 언어, 분산 객체시스템, 데이터베이스 등을 사용하지 않는다. 웹 서비스는 이러한 요구에 대한 해답을 제공하기 위한 하나의 시도로 발생하였다.

이러한 문제들에 대한 만족스러운 해결책을 제공하기 위해서는 매우 이질적인 분산 시스템을 효율적으로 다룰 수 있는 방안이 필요하다. 이와 같이 이질성의 문제를 다루는 것이 웹 서비스에 있어서 핵심적인 내용이 되는 이유는 웹이 다양한 플렛폼과 프로그램들이 존재하는 분산형 매체로서의 특성을 가지고 있고 기존 모형들과 끊김없

이 통합되기 위해 유연성이 필요하기 때문이다.

이질성 문제에 대한 웹 서비스의 접근 방법은 두 가지이다. 첫번째는 기본적인 상호운용성을 확보하기 위한 최소한의 표준 집합을 정의하는 것이다. 두번째는 이질적인 시스템들 내에서 어플리케이션들의 통합된 뷰를 제공함으로써 상호작용시 양쪽 모두에서 이용이 가능할 경우, 또다른 상호작용 모형들을 활용할 수 있도록 하는 것이다. 다른 분산 시스템들과 마찬가지로, 웹 서비스는 데이터 교환 프로토콜과 웹 어플리케이션의 기능적인 표현을 위한 일련의 메타데이터 등과 같은 두개의 기본적인 요소 위에서 구축된다(Curbera et al. 2001).

결론적으로 웹 서비스가 등장하게 된 배경으로는 단순하고 개별적으로 코딩된 어플리케이션으로부터 안정화되고 스크립트화된 컴포넌트로 발전하고자 하는 욕구를 들 수 있다. 또한 강 결합(tightly coupled) 시스템으로부터 약 결합(loosely coupled) 시스템으로 발전할 필요가 있으며 인터넷을 통해 비즈니스들을 연결시킬 수 있는 사용이 용이한 프로그램 모형의 필요성도 웹 서비스의 등장을 촉진시킨 원인이 되고 있다.

### 1.2 연구의 필요성 및 목적

최근들어 웹 서비스(Web Services)라는 용어가 정보 산업 분야에서 관심을 끌기 시작하였다. 이를 반영하듯 인터넷 분야 주요 조직체인 W3C에서도 웹 서비스 관련 워크샵(http://www.w3.org/2001/01/WSWS)을 개최하고 특히 IBM(http://www-106.ibm.com/developerworks/webservices/), 마이크로소프트(http://msdn. microsoft.com/vstudio/default.asp), SUN(http://java.sun.com/webservices/) 등과 같은 세계 대표적인 IT 기업들이 각기 자체의 웹 서비스 전략들을 발표하기 시작했다.

인터넷이 이용자간 정보공유가 용이하여 정보의 바다로 여겨지며 널리 보급되면서 웹 기술은 e-비즈니스 분야에서 클라이언트 인터페이스의 표준으로 자리를 잡았다. 하지만 지금까지의 웹 기술은 클라이언트와 서버간의 자료전달에 치중하고 있어 인터넷은 서로 밀접히 연결된 정보의 결합체가 아닌 독립적인 정보를 가진 섬들의 모임에 지나지 않았다. 최근 IT 시장이 급속히 성장하면서 e-비즈니스 솔루션 개발 비용축소 및 개발기간 단축을 위해 어플리케이션의 통합과 업체간의 협력(eCollaboration) 등의 새로운 시도가 활발히 진행되고 있다. 이에 따라 웹 서비스가 등장하게 되었다. 인터넷이 제1의 물결, 웹이 제2의 물결이라고 한다면 웹 서비스는 제3의 물결로 이야기할 수 있다.

본 고에서는 먼저 웹 서비스의 발생 배경과 정의, 특징 등에 대해 언급을 한다. 그리고 웹 서비스가 작동하는 원리에 대해 상세히 설명을 한 후, 웹 서비스를 구현하는데 필요한 핵심 기술들과 실제 구현 사례들을 살펴본다. 마지막으로 웹 서비스의 향후 발전 방향에 대해 언급한다.

## 2. 웹 서비스 개요

### 2.1 웹 서비스 정의

웹 서비스를 개발하고 구현하는데 참여하고 있는 업체와 유관 기관들을 중심으로 웹 서비스에 대해 다양한 정의가 내려지고 있다. 본 고에서는 Oracle, IBM, UDDI 프로젝트 등에서 제시한 웹 서비스 정의를 살펴봄으로써 웹 서비스에 대한 이해를 증진시키고자 한다.

Oracle사는 기본적으로 웹 서비스를 하나의 표준화된 XML 기반의 인터페이스를 통하여 플랫폼에 독립적이고 프로그램 언어에 중립적인 방법으로 네트워크상에서 어플리케이션들이 엑세스할 수 있는 하나의 로직으로 간주한다. 이에 따라 웹 서비스들은 분산 전산 환경하에서 동적으로 등록되고 탐색되며 구동될 수 있도록 설계되어져 있다. 또한 웹 서비스는 인터넷상에서 어플리케이션들간에 프로그램에 의한 실시간 상호작용을 촉진시킴으로써, 기업들로 하여금 더욱 쉽게 정보를 교환할 수 있도록 해주며 정보자원의 활용을 강화시켜주고 비즈니스 프로세스를 통합시켜준다.

웹 서비스를 구현하기 위해서는 다음과 같은 세가지의 기본적인 요구사항이 있다. 첫째, 서비스들을 기술하기 위한 표준적인 방법으로서 기술(description)이 있으며 여기에는 최소한의 입/출력 인터페이스 규격과 저작권 사항, 버전, 최신 URL 등과 같은 몇가지의 메타 정보가 포함된다. 둘째, 원하는 특성들을 갖는 연관된 서비스들을 탐색하기 위한 표준적인 수단으로서 탐색(discovery)이 있다. 셋째, 서비스에 대한 요구를 보내고 응답을 받기위한 표준 메커니즘으로서 전송(transport)이 있다. 이러한 요구사항들을 충족시키기 위해 개발된 WSDL, UDDI, SOAP 등과 같은 표준들은 웹 서비스의 기본이 되고 있다(Oracle 2001).

IBM사는 웹 서비스를 새로운 웹 어플리케이션들 중의 하나로 바라본다. 이것은 독립적이며 모듈화된 어플리케이션으로서 웹 상에서 등록이 되고 탐색이 되며 실행이 가능하다. 웹 서비스에서는 여러 기능들을 수행시킬 수가 있는데 이들 기능들은 단순한 요청에서 부터 복잡한 비즈니스 프로세스에 이르기까지 어느것도 가능하다. 일단 웹 서비스가 설치되면, 다른 어플리케이션들(다른 웹 서비스도 포함)에서 구동된 웹 서비스를 탐색하여 구동시킬 수가 있다(http://www6.software.ibm.com/developerworks/education/wsbasics).

UDDI 프로젝트에서는 웹 서비스를 자기 서술적이고 독립적인 어플리케이션 로직의 모듈화된 단위로서 인터넷 접속을 통하여 다른 어플리케이션들에 대해 약간의 비즈니스 기능을 제공하는 것으로 정의를 한다. 이들 어플리케이션들은 각 웹 서비스가 구현되는 방법에 대해 걱정할 필요없이 HTTP나 XML 등과 같은 범용의 웹 프로토콜과 데이터 포맷들을 통해 웹 서비스에 엑세스를 한다. 규모가 더 큰 워크플로우나 비즈니스 트랜젝션을 구동시키기 위해 웹 서비스들을 다른 웹 서비스들과 혼합시키거나 일치시킬 수 있다(http://www.uddi.org/faqs.html).

### 2.2 웹 서비스 특징

웹 서비스는 다음과 같은 특징들을 가지고 있다(Curbera, et al., 2001). 첫째, 약결합된 상호작용 모형이다. 약 결합은 플렛폼들의 이질성에 기인하며 상호운용성 프로토콜의 특성에 반영이 되고 있다. 둘째, 유연한 통합을 제공한다. 서비스 기술에 기반하여, 상호작용의 양 끝단에서 IIOP (Internet Inter-ORB Protocol) 등과 같은 더욱 효율적인 프로토콜을 지원하는 상황들을 웹 서비스가 탐지하는 것이 가능하다. 따라서 두 어플리케이션들간의 통합 수준은 공통의 프로토콜에 의해 지원되는 수준까지 확장될 수 있다. 셋째, API 대신 메시지를 제공한다. 공통의 인터페이스를 가정하는 대신, 이 모형은 서비스 기술과 산업체 표준화를 위한 기반으로서 메시지와 문서 포맷을 이용하는데 의존한다. 이러한 환경하에서 메시지 브로커링과 번역은 서비스 상호운용성 제공을 위한 핵심 메커니즘이 된다.

웹 서비스는 장점과 단점을 가지고 있는데 웹 서비스의 장점은 다음과 같다. 첫째, 플렛폼에 독립적이기 때문에 모든 플렛폼으로부터 여러곳에서 특정 서비스에 엑세스할 수 있다. 둘째, 어플리케이션 구성요소들 사이의 진정한 분리를 통해 상이한 비즈니스 요구사항들에 대응할 수 있다. 따라서, 비즈니스 요구사항들의 변화로 인하여 발생하는 구성요소의 변화에 의해 특정 시스템이 영향을 덜 받게된다. 셋째, 프로그램 언어에 독립적이며 개발자들이 읽을 수 있는 형태로 서비스들이 선언된다. 따라서 개발자들이 보다 쉽게 이해할 수가 있다. 넷째, 구현이 비교적 간단하고 이질적인 어플리케이션과 환경들을 서로 연결시켜주는 방법을 제공하므로 기존 전산 자원들을 최대한 활용할 수 있게 된다.

웹 서비스의 단점으로는 주요 웹 서비스 프로토콜에 내장형 보안이 거의 없다는 점이다. 또한 웹 서비스 프로토콜에는 QoS(Quality of Service)나 트랜잭션 어플리케이션 기능이 거의 없다. 따라서 웹 서비스는 데이터를 중계하고 원격 질의를 가능하게 하는데는 뛰어나지만 데이터가 일부만 손실되어도 막대한 피해를 입을 수 있는 어플리케이션들을 지원하는데에는 그리 뛰어나지 못하다. 즉, 웹 서비스는 엔터프라이즈급 어플리케이션들을 구축하는데 필요한 보안과 QoS, 트랜잭션 기능 등이 미비하다.

## 3. 웹 서비스 관련 기술 및 작동 원리

### 3.1 관련 기술

웹 서비스는 다양한 하드웨어와 소프트웨어의 장벽을 해결하기 위해 개방형 표준을 채택하고 있으며 현재 웹 서비스에서 채택하고 있는 개방형 표준으로는 SOAP, UDDI, WSDL 등이 있다.

#### 3.1.1 SOAP

2000년 4월 IBM, 마이크로소프트, Userland, DevelopMentor가 공동으로

SOAP 1.1을 발표하였다. 2001년 2월에 ebXML은 SOAP 1.1에 기반을 둔 TRP (Transport, Routing, Packaging) 규격을 발표하였으며 2001년 7월에 W3C는 SOAP 1.2의 초안을 발표하였다.

SOAP은 분산 환경하에서 정보의 상호교환을 위한 단순한 프로토콜이다. 이것은 XML 기반의 프로토콜로서 다음과 같은 네가지 구성요소로 이루어진다. 첫째, 메시지내에 담겨있는 내용과 이 내용물을 처리하는 방법을 기술하기 위한 프레임워크를 정의하는 객체(envelope), 둘째, 기반 프로토콜을 이용한 메시지 상호교환을 위한 전송 바인딩 프레임워크, 셋째, 어플리케이션에서 정의된 데이터 타입들의 인스턴스들을 표현하기 위한 인코딩 규칙들(encoding rules)의 집합, 넷째, 원격 프로시듀어 콜(RPC)들과 응답들을 표현하기 위한 규칙(convention)이다. SOAP은 다른 프로토콜들과 결합하여 사용될 수도 있다(http://www.w3.org/TR/ SOAP/).

SOAP은 마이크로소프트사와 IBM사에 의해 급속하게 보급되고 있는 표준으로 클라이언트의 작업 요청과 시스템의 반응을 XML 문자열로 포장하고 전송 프로토콜로 HTTP를 사용한다. 즉, SOAP은 HTTP와 XML의 결합이라 볼 수 있다.

이러한 특징을 갖는 SOAP은 다음과 같은 장점을 갖는다. 첫째, 프로토콜 자체가 매우 가벼운데 HTTP 패킷을 전송하고 수신하는 기능과 XML을 처리할 수 있는 기능 등과 같은 두가지의 기본적인 기능만을 요구한다. 둘째, SOAP은 개방형 표준이기 때문에 점점 더 많은 개발자들과 벤더들이 SOAP을 지원하고 있다. 더 많은 벤더들이 SOAP 제품과 서비스들을 제공함에 따라 SOAP 사용의 장점은 더욱 커진다.

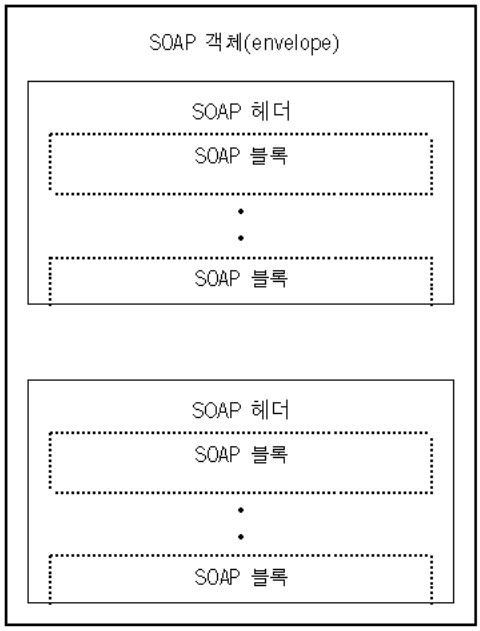
셋째, 진정한 분산형 상호운용성을 촉진시킨다. 넷째, 방화벽 제한이 없다.

이에 반해 SOAP은 다음과 같은 단점도 가지고 있다. 첫째, HTTP 기반이기 때문에 stateles하고 요청/응답 아키텍쳐 기반이기 때문에 콜백 기능이 없다. 둘째, SOAP 요청/응답을 파싱하기 위해 XML 프로세서가 매번 로드되어야하며 필요한 정보를 추출하기 위해 SOAP 요청/응답을 파싱해야하기 때문에 성능이 약간 저하된다. 셋째, 현재는 값에 의한 파라미터 직렬화만을 지원한다. 넷째, 만약 XML 스키마 확장이 SOAP 메시지를 포맷하는데 이용되기 위해서는 완벽한 스키마 파싱을 지원하는 XML 프로세서들을 요구하게 된다.

SOAP 노드에는 발신자(sender), 수신자(receiver), 중개자(intermediaries), 실행자(actors) 등이 있다. SOAP 메시지는 객체(envelope), 실행자 속성과 must-Understand 속성 등으로 구성된 헤더(header), 본문(body), VersionMismatch, MustUnderstand, DataEncodingUnknown, Client, Server 등과 같은 오류 코드로 구성된 오류(fault) 등으로 구성되어 있다. <그림 1>은 SOAP의 개념적인 구성도를 나타낸다.

### 3.1.2 UDDI

오늘날 웹 서비스 레지스트리를 위한 실질적인 표준 규격으로 기능을 하고 있는 UDD는 Ariba, IBM, 마이크로소프트사 등




<그림 1> SOAP의 개념적인 구성도

에 의해 프로젝트 형태로 추진되고 있다. UDDI 프로젝트는 2000년 9월에 시작되었기 때문에 비록 역사는 오래되지 않았지만 매우 빠른 속도로 발전을 하고 있으며 2001년 8월까지 UDDI 비즈니스 레지스트리에는 5,200개 이상의 비즈니스와 7,800개의 서비스가 등록되어 있으며 매일 새로운 비즈니스와 서비스들이 등록되고 있다. UDDI 버전 2 레지스트리에서는 서비스 탐색이 쉽고 비용대 효과면에서 우수하여 웹 서비스를 증진시키는 장점이 있다. e-비즈 레지스트리를 더욱 정교하고 효율적으로 만들기 위해 UDDI 버전 3를 위한 작업들이 현재 진행중이며 여기에서는 고도화된 질의 기능, 캐쉬, 보안, 세계화 등과 같은 계획들이 포함되어 있다.

UDDI의 목표는 웹 상에서 이용가능한 전자 서비스들에 대하여 분산형 웹 기반 글로벌 레지스트리를 생산하는 것이다. 실제, 이것은 전자 서비스들에 대하여 일종의 전자 전화번호부를 제공하는 것과 같다. UDDI를 개발하게된 직접적인 동기는 연관된 이질적인 복잡한 시스템, 인터페이스, 프로토콜들을 인식하지 않은 상황에서 중개 소프트웨어와 중개자들을 통해 이용자들이 서비스들을 탐색하고 상호작용할 수 있는 전자 마켓플레이스의 개념을 구현하는데 있다. UDDI에서는 최종 이용자가 직접 사용하는 것 보다는 개발자와 컴퓨터 시스템들이 사용할 수 있도록 설계가 된 간단한 탐색 인터페이스를 갖는 디렉토리를 제공한다. 또한, UDDI는 특정 클라이언트 어플리케이션이 이러한 서비스와 통신하기 위해 필요로 하는 컴퓨터 인터페이스들을 기술할 수 있는 메커니즘을 제공한다. 중개자들이 이용자를 위해 연관된 전자 서비스들을 찾기 위해 그리고 이용자들을 대신하여 이들 서비스들과 상호작용하기 위하여 이러한 디렉토리 서비스를 이용할 것으로 기대가 된다.

UDDI의 활용 예는 다음과 같다. 특정 이용자가 책이나 CD 등과 같은 특정 물품을 구매하기를 희망한다고 가정하자. 이 경우, 해당 물품을 팔거나 제공하는 회사를 찾기 위해 글로벌 UDDI 디렉토리를 탐색한다. UDDI 디렉토리에서는 이러한 회사들을 탐색한 후, 이용자에게 이름, 주소, 전화번호, 웹 페이지와 기타 서비스 등과 같은 상세 연락처 정보를 제공할 뿐만 아니라 해당 물품을 전자적으로 구매하는 방법에 관한 몇가지 기술적인 정보도 제공한다. 어떤 회사는 웹 양식 기반의 접근방법을 제

공하여 HTTP 접속 상태에서 구매자의 상세정보를 제공함으로써 해당 물품을 구매할 수 있다. 다른 회사는 EDI를 이용할 수 있고 또 다른 회사는 공개된 SOAP API를 이용할 수 있다. 이론적으로, 해당 소프트웨어가 이들 서비스들과 상호작용할 수 있도록 자동으로 조정되어 사용되기 위해서는 UDDI를 통해 충분한 정보가 제공되어야 한다. 이용자 측면에서 보면, 비록 서로 다른 회사들이 서로 다른 인터페이스를 통해 유사한 서비스들을 제공하지만 특정 물품의 구매는 투명하게 발생한다.

3.1.3 WSDL

WSDL은 웹 서비스의 IDL(Interface Definition Language) 버전이며 특정 웹 서비스의 방법과 프로토콜, 데이터 포맷들을 더욱 상세하게 정의하는 일종의 스크립트이다. WSDL은 XML 포맷으로 구성되고 HTTP를 통해서 전달될 수 있으며 인터페이스를 정의하는 IDL에 해당한다. 즉, 특정 서비스가 어떤 메소드, 어떤 속성을 가지며 어떤 인수로 호출해야 하고 어떤 방식의 리턴값을 제공하는지를 알려주는 것이다. 이 내용을 알게 되면 클라이언트에서는 알아낸 인터페이스 규약에 맞추어 호출하고 서비스를 사용할 수 있게 된다.

2000년 9월에 WSDL 버전 1.0이 발표되었으며 2001년 3월에 WSDL 버전 1.1이 W3C에 제출되었다. 현재는 실질적인(de facto) 산업표준으로 자리잡고 있으며 산업계에서 표준화된 서비스 인터페이스를 정의할 수 있도록 해준다. UDDI 레지스트리와 함께 사용되어 호환되는 서비스들의 동적인 검색과 바인딩을 가능하게 해준다. 또한 이질적인 어플리케이션들에 대해 정규화된 기술을 할 수 있도록 해준다.

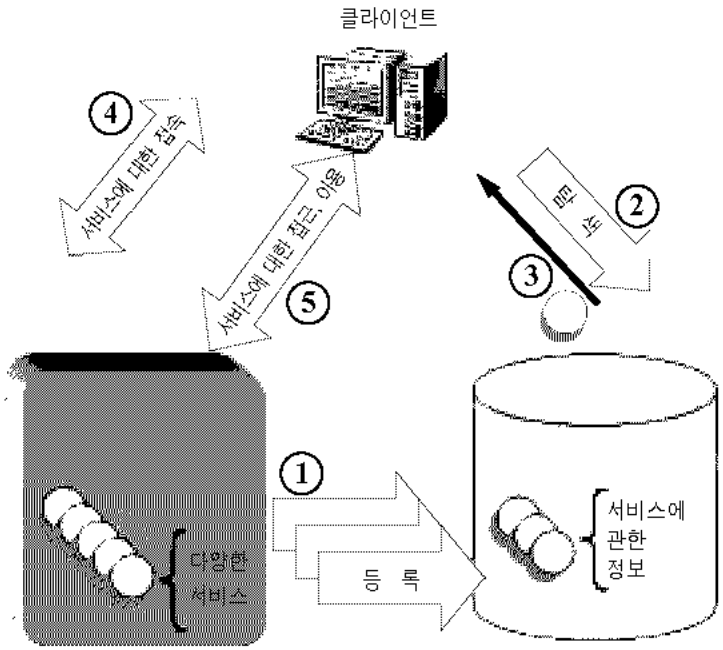
WSDL 문헌은 여러 섹션들로 구성되어 있다." 스키마 섹션"에서는 서버와 통신하기 위해 사용되는 데이터의 유형과 구조를 정의한다. 예를 들어, 여행 서비스에서는 비행기 편명, 여행 시간, 제공되는 식사 유형 등을 포함하는 XML 데이터 구조를 정의할 수 있다. WSDL에서는 모든 데이터 기술 언어를 허용하지만, 대부분이 XML 스키마를 이용한다. 다른 WSDL 섹션들에서는 메시지 흐름, 서버가 수행할 수 있는 작업 리스트, 네트워크 프로토콜과 위치(예, 호스트, 포트, HTTP를 이용할 경우 URL)에 관한 정보 등을 정의함으로써 클라이언트들이 서버와 통신을 할 수가 있게 된다. 이러한 정보와 단일의 문헌을 결합시킴으로써 웹 서비스 서버에 접속할 수 있는 방법을 완벽하게 기술할 수 있는 방법을 제공한다. 여기에서는 메시지의 문법만을 기술하고 그 의미를 기술하지는 않는다. WSDL을 받아서 자동으로 클라이언트 혹은 서버 코드를 생성시키는 투울 킷이 나와 있다.

3.2. 작동원리

웹 서비스는 SOAP, UDDI, WSDL 등과 같은 개방형 표준 기술을 기반으로 서비스 제공자(Service Provider)와 서비스 저장소(Service Registry), 그리고 서비스 요청자(Service Requester) 등이 유기적으로 결합하여 구현된다. 첫째, 서비스 제공자는 비

즈니스 관점에서는 서비스 소유자이며 아키텍쳐 입장에서는 서비스를 제공하는 플렛폼을 말한다. 즉, 서비스 제공자는 e-비즈니스 서비스를 제공하는 단체나 기업을 말하며 서비스 저장소에 WSDL 방식을 통해 작성된 서비스 내용을 등록할 수 있다. 둘째, 서비스 저장소는 서비스 제공자가 제공하는 해당 서비스의 상세 정보를 저장하고 있으면서 검색의 대상이 되며, 객체 바인딩의 정보를 제공하기도 한다. 즉, 서비스 저장소는 등록된 서비스 내용을 UDDI 기술을 이용하여 전화번호부 책자처럼 목록화하거나 서비스 내용을 정리해 둔다. 셋째, 서비스 요청자는 비즈니스 관점에서는 서비스의 기능을 원하는 측이며 아키텍쳐 입장에서는 서비스와의 접속을 시도하고 초기화하는 주체이기도 하다. 즉, 서비스 요청자는 서비스 저장소를 통해 원하는 서비스를 WSDL 체크를 통해 확인한 후, 서비스 공급자로부터 SOAP 기술을 통해 직접 지원을 받게 된다.

<그림 2>는 웹 서비스가 작동하는 원리를 개념적으로 나타내고 있다. 먼저 서비스 제공자는 자신이 제공하는 서비스에 대한 목록 정보를 서비스 저장소(Service Registry)에 등록(publish)을 한다. 여기에서 "등록"이란 서비스 요청자가 탐색을 하여 원하는 서비스를 이용할 수 있도록 상세정보와 함께 저장소(registry)에 등록하는 행위를 말한다. 다음으로 서비스를 요청하는




<그림 2> 웹 서비스 작동 개념도
(http://www.onjava.com/lpt/a//onjava/2001/08/07/webservices.html)

프로그램에서는 해당 서비스의 정보를 수집하기 위해 인터넷 디렉토리 시스템을 사용한다. 보통 URL을 통해 웹 자원을 탐색(find)하듯 이러한 디렉토리 시스템은 인터넷 곳곳에 산재해 있으며 이를 UDDI 레지스트리라고 부른다. 여기에서" 탐색"이란 서비스 요청자가 원하는 서비스의 내용을 탐색하고 필요한 정보를 획득하는 동작을 말한다. UDDI 레지스트리를 통하여 원하는 서비스의 존재 여부와 위치, 그리고 호출에 필요한 각종 정보를 얻은 서비스 요청자는 서비스 제공자와 직접적인 통신을 시도한다. 따라서 서비스 제공자가 서버를 다른 기계로 옮기든 서비스의 로직이 바뀌든 간에 서비스 요청자 입장에서는 서비스의 내용만 이해하고 있다면 소스의 수정없이 프로그램 스스로 상호 통신이 가능하게 된다.

## 4. 웹 서비스 구현 사례

### 4.1 마이크로소프트사

마이크로소프트사는 2000년도 TechED 행사에서 현재의 웹 기술의 한계를 극복할 수 있는 한 방편으로 차세대 인터넷(Next Generation Internet)으로 닷넷(.Net)이라는 새로운 개념을 내놓았다. 닷넷에서는 PC 중심의 클라이언트 환경에서 벗어나 태블릿(tablet) PC, 스마트폰, 핸드헬드(handheld) 컴퓨터, 게임 콘솔 등의 단말기에서도 스마트 디바이스를 통해 인터넷 서비스를 쉽게 이용할 수 있게 한다. 또한, 웹 서비스라는 새로운 개념으로 독립적으로 제공되고 있는 인터넷 서비스들을 통합 및 재가공하여 새로운 인터넷 서비스를 생성할 수 있는 새로운 플랫폼을 제공하고 있다.

닷넷에서 모든 어플리케이션은 XML로 데이터를 주고받고 SOAP을 통해 객체를 생성하고 관리한다. 이렇게 데이터와 객체에 관련된 규약을 표준화함으로써 개발자와 사용자 모두에게 도움이 되고 있다.

2002년 2월 13일 공개된 Visual Studio.-Net은 마이크로소프트사의 개발자들이 현재 사용하고 있는 Visual Studio 도구를 대체할 개발자용 도구이다. 새 도구는 현재 마이크로소프트사 개발자들이 가장 많이 사용하고 있는 비주얼 베이직과 비주얼 C++ 언어외에 자바와 비슷한 C#언어 등으로 이루어져 있다. 또한, 코드가 이미 들어있어 개발자들에게 시간을 절약하게 해주고 복잡한 프로그래밍 인터페이스를 단순화해주며 오류들을 제거해주는 닷넷 프레임워크도 들어있다. 이외에도 복잡한 웹 기반 비즈니스 시스템을 손쉽게 구축하는 MS의 개발 소프트웨어인" 액티브 서버 페이지"의 새 버전" ASP 닷넷"도 포함되어 있다.

### 4.2 썬마이크로시스템사

SUN ONE은 솔라리스 운용체계를 사용한 중대형 컴퓨터(서버)로 유명한 미국 썬 마이크로시스템사의 인터넷 서비스 전략이다. ONE은 Open Net Environment의 약어로서 썬은 마이크로소프트의 인터넷 서비스 전략인 닷넷에 대응해 이를 추진하고

있다.

SUN ONE은 Open Net이라는 약어에서 알 수 있듯이 특히 개방형의 인터넷 환경을 중요시하고 있다. 또한, 마이크로소프트의 닷넷 전략이 소비자 중심에 맞춰져 있는 반면, SUN ONE은 기업의 네트워크 환경이 인터넷과 보다 쉽고 다양하게 접목되도록 하는데 초점을 두고 있다. 즉, 닷넷이 노트북, 이동전화, PDA 등 소위 이동형 단말기들이 언제 어디에서나 인터넷에 접속해 여행, 금융, 레저 등의 서비스를 이용하도록 하는데 주안을 두지만 SUN ONE은 기업의 업무생산성을 높이기 위해 각종 어플리케이션과 인터넷 네트워크망과의 효율적인 연계에 중점을 두고 있다.

### 4.3 IBM사

IBM사는 인터넷 기반의 웹 서비스 인프라 소프트웨어 전략을 발표했는데 이 전략의 주요 골자는 웹 스피어 제품군과 티볼리, 로터스, DB2 등 전체 미들웨어 라인을 중심으로 개방형 인터넷 표준들을 적극 수용한다는 것이다. 이를 위해 IBM사는 웹 서비스 솔루션을 웹 스피어, DB2, 티볼리, 로터스 등과 같은 4대 핵심 분야로 구분하고 관련 솔루션을 지속적으로 출시하고 있다. 웹 서비스를 위한 주요 제품으로 웹 스피어 어플리케이션 서버 4.0과 웹 스피어용 스튜디오 테크놀로지 프리뷰, 웹 스피어 비즈니스 인터그레이터, DB2/XML 익스텐더, 티볼리 웹 서비스 매니저, 티볼리 시큐어웨어 폴리시 디렉터, 로터스 웹 서비스 인에블먼트 킷 등이 있다.

IBM WebSphere Application Server 버전 4.0은 J2EE(Java 2 Enterprise Edition) 표준 플렛폼 환경을 가지고 있으며 이러한 엔터프라이즈 자바 운영체제를 기반으로 웹 서비스 서버기능들을 모두 갖추고 있다. 즉, 고객의 입장에서 자바 서블릿이나 EJB 등의 J2EE 플렛폼으로 응용프로그램을 구성하여 시스템을 구축할 수 있으며 메인프레임이나 ERP 시스템 등 여타의 엔터프라이즈 자원들과의 통합 처리가 가능한데 이를 통해 자연스럽게 기업환경에서의 웹 서비스 수행이 이루어진다.

## 5. 향후 발전 방향

최근 들어 웹 서비스 개발에 있어서의 일관성을 확보하기 위한 노력의 일환으로 IBM, 마이크로소프트, BEA사, HP, Intel, Oracle, SAP, Fujitsu 등이 웹 서비스 상호운용성 협회(Web Services Interoperability Organization)로 불리우는 새로운 인터넷 컨소시움을 결성하였는데 이 조직에서는 개발자들에게 웹 서비스 구축 교육을 제공하기 위한 노력을 할 뿐만 아니라 SOAP, UDDI, WSDL 등과 같은 각 표준들의 일관성을 장려할 예정이다 (http://www.infoworld.com/articles/hn/xml/02/02/05/020205hnwebconsortium.xml).

특히, 이 그룹은 트랜젝션 관리시스템, 보안, 식별, 권한부여 등과 같은 기본적인 기능들을 다루게될 미래의 웹 서비스 표준의 일관성을 적극 장려하게 될 것이다.

또한, 이 그룹은 새로운 표준들을 개발

하기 위한 노력은 하지 않을 것이며 대신 SOAP과 UDDI 등과 같은 기존 표준들의 상호운용성을 강화하는데 노력을 할 것이다. 예를들어 SOAP 1.2 와 1.3이 서로 상이할 경우, 단지 이들이 서로 통신할 수 있는 이상의 더욱 폭넓은 상호운용성을 갖을 수 있는 방법을 모색할 것이다. 이들은 자바와 닷넷이 더욱 향상된 상호운용성을 갖도록하는 것과 같은 획기적인 해결책을 원하지는 않는다.

## 6. 결 론

최근 들어 정보의 바다로 불리우는 인터넷상에서 원하는 정보를 손쉽게 구하지 못하고 비효율적인 작업을 하는 경우가 많다. 그 이유는 필요한 정보가 어디에 있는지 정확히 파악하기 어렵고 비록 정보의 위치를 파악하더라도 이를 구하기 위해서는 네트워크와 어플리케이션 연동 등과 같은 복잡한 과정을 거쳐야 하기 때문이다. 이러한 문제들을 해결하기 위해 언제, 어디서나 무엇으로든 원하는 정보를 손쉽게 주고 받고 이용할 수 있는 웹 서비스(Web Services)가 등장했다.

웹 서비스에서는 프로그램에 의해 자동적으로 트랜잭션이 발생하고 굳이 브라우저를 사용할 필요도 없다. 그리고 등록(publishing) 방식을 취하기 때문에 상대방 고객이 서비스 내용을 쉽게 해독, 이를 이용할 수 있다. 또한 분산된 전산 환경에서 동적으로 서비스를 제공할 수 있다. 이러한 웹 서비스의 잠재력 때문에 서비스가 본격화되면 e-비즈니스에 새로운 혁신의 바람이 불게 될것이며 향후 e-비즈니스의 가장 주도적인 패러다임으로 자리잡게 될 것으로 예측된다.

최근 들어 국내에서도 웹 서비스 구축을 위한 노력들이 시작되고 있다. 즉, 국내 특성에 맞는 업무 프레임워크와 한국형 UDDI를 개발하는 등 각종 표준을 제정함으로써 전세계 기업들의 통합된 웹 서비스가 보편화되었을 때, 국내 기업들이 주도권을 행사할 수 있을 것이다. 이미 미국의 경우 IBM, 썬마이크로시스템즈, 마이크로소프트사 등 유수의 IT 기업들이 uddi.org를 결성해 독자적인 웹 서비스 등록시스템을 구축해놓은 상태이다. 따라서 한국형 UDDI가 없으면 한국 기업들의 웹 서비스가 이들 외산 기업의 등록시스템을 사용해야 하고 이 경우 웹 서비스 시장에서 국내 기업들이 주도권을 상실할 우려가 있기 때문에 한국형 표준 프레임워크 및 UDDI 개발이 필수적이다.

## 참고문헌

Curbera, Francisco, Nagy, William A. and Weerawarana, Sanjiva. 2001. " Web Services: Why and How." <http://www.research.ibm.com/people/b/bth/OOWS2001/nagy.pdf>

Oracle. 2001. " Developing, Deploying, Managing Web Services with Oracle9i: An Oracle White Paper." <http://otn.oracle.com/products/dyna

mic_services/htdocs/ds_wp_deploy_and_manage/ds_wp_deploy_and_manage_1.html#_Toc516988388>

SOAP. <http://www.w3.org/TR/SOAP/>

UDDI. <http://www.uddi.org>

WSDL. <http://www.w3c.org/TR/wsdl>

<http://java.sun.com/webservices/>

<http://msdn.microsoft.com/vstudio/default.asp>

<http://www.onjava.com/lpt/a//onjava/2001/08/07/webservices.html>

<http://www.uddi.org/faqs.html>

<http://www.w3.org/2001/01/WSWS>

<http://www-106.ibm.com/developerworks/webservices/>

<http://www6.software.ibm.com/developerworks/education/wsbasics>