지능적 웹 서비스 테스트 프레임워크

김상균[O] 이규철
충남대학교 컴퓨터공학과
{skkim[O], kclee}@ce.cnu.ac.kr

Intelligent Web Services Testing Framework

Sang-Kyun Kim[O] Kyu-Chul Lee
Dept. of Computer Engineering, Chungnam National University

요 약

웹 서비스는 SOAP,WSDL,UDDI와 같은 표준을 통해 서비스를 기술하고 발견,호출할 수 있는 인터페이스들을 정의하고 있지만 서비스에 대한 테스트 방법들은 정의하지 않고 있으며 현재 대부분 웹 서비스 포탈을 통해 단순한 메타데이터 기반의 테스트가 수행되고 있다. 하지만 실제 웹 서비스를 이용한 e-비즈니스가 수행되기 위해서는 사용자들이 정말 자신이 원하는 서비스인지 테스트하는 것이 매우 중요하다. 따라서 본 연구에서는 웹 서비스 테스트를 보다 지능적으로 수행할 수 있도록 웹 서비스의 입력과 출력 정보에 시맨틱 정보를 추가하는 테스트 프레임워크를 제안한다. 이러한 시맨틱 정보에 기반한 웹 서비스의 정확한 테스트를 통해 웹 서비스가 활성화되고 진정한 웹 서비스 기반의 e-비즈니스가 이루어질 수 있을 것이다.

1. 서 론

웹 서비스(Web Services)는 표준화된 XML 메시지를 통해 네트워크 상에서 접근 가능한 연산들의 집합을 기술하는 인터페이스로 정의된다.[1] 이러한 웹 서비스는 XML을 기반으로 한 공개 표준을 사용해서 CORBA, 자바 RMI, DCOM등 기존 분산 컴퓨팅 환경에서 문제점이었던 이기종 시스템간의 상호 운용성 문제를 해결하였으며, 웹 기반의 시스템이라면 웹 서비스를 쉽게 구축할 수 있는 특징을 가진다. 또한 Microsoft, IBM, Sun, Oracle등을 비롯한 세계의 주요 IT업체들의 지원을 받으면서 업체중심의 표준화와 시스템, 개발도구들이 나오기 때문에 다른 e-비즈니스 프레임워크들보다 현실적인 프레임워크로 인식되고 있다.

이러한 웹 서비스는 SOAP, WSDL, UDDI 이렇게 세 가지의 공개 표준을 기반으로 사용된다. SOAP(Simple Object Access Protocol)은 웹 서비스간의 통신을 위해 웹 서비스의 요청 및 응답에서 사용되는 메시지 형식을 정의하고 있으며 HTTP프로토콜을 기반으로 하기 때문에 기존 웹 환경을 확장한 형태로 분산 컴퓨팅 환경을 구현할 수 있는 이점을 가진다.

WSDL(Web Services Description Language)은 기존의 CORBA나 COM의 IDL(Interface Definition Language)과 유사한 역할을 수행하는데 웹 서비스 설명서와 같이 모든 웹 서비스를 접근하고 실행하기 위한 서비스 정보, 메시지 포맷, 입출력 연산등과 같은 정보를 포함한다. 또한 이러한 정보를 통해 서비스 제공자와 사용자의 사용자들간의 통신과 관련된 자세한 정보를 정의함으로써 웹 서비스 테스트를 자동화할 수 있도록 한다.

UDDI(Universal Description, Discovery and Integration)는 웹 서비스에 대한 디렉토리 서비스를 지원하기 위해 개발된 표준으로서 웹 서비스를 등록하고 검색/발견하기 위한 데이터구조와 API들을 제공한다. UDDI는 웹 서비스 정보를 관리하기 위해서 기본적으로 white pages, yellow pages, green pages와 같이 세 가지 컴포넌트를 사용하는데 white pages는 비즈니스의 이름,설명,연락처,식별자 정보등을 가지며, yellow pages는 표준 분류법에 따른 산업적인 분류 정보를 가진다. 또한 green pages는 웹 서비스 이용에 관련된 기술적 정보를 포함하며 여기에 WSDL에 대한 참조 정보도 포함된다.

하지만 이러한 표준들을 기반으로 하는 웹 서비스는 UDDI를 통해 단순한 키워드 기반의 메타데이터 정보를 검색하고 WSDL을 통한 호출 매커니즘만 정의하고 있으며 웹 서비스가 어떻게 동작하는지 알기 위한 테스트 방법들은 정의하지 않고 있다. 하지만 웹 서비스 테스트는 실제 웹 서비스 기반의 e-비즈니스를 위해서는 반드시 필요한 기능 중에 하나이다.

현재 이러한 기능을 지원하기 위해서 여러 웹 서비스 솔루션 업체들은 다양한 웹 서비스 개발도구를 만들고 웹 서비스 포탈들을 구축해서 웹 서비스에 대한 자동화된 런타임 환경을 만들어 줄 뿐만 아니라 추가적인 메타데이터 정보들을 제공하고 있다. 하지만 이러한 방법들은 웹 서비스를 이용하는데 자동화 기능에 초점을 맞추고 있으며 실제 웹 서비스를 테스트할 때 필요한 입력과 출력 메시지에 대한 설명을 메타데이터에 의존하고 있어서 정확한 테스트를 할 수 없다.

예를 들어 날씨정보 웹 서비스를 이용하기 위해서 UDDI에서 웹 서비스를 검색해서 WSDL파일을 가져온 후 클라이언트 프로그램을 자동으로 구현해 주는 웹 서비스 개발 도구를 이용해 테스트하는 경우를 고려해 보자. 이 경우 웹 서비스 테스트를 위해 입력값을 넣어야 하지만 어떤 입력값을 넣어야 하는지 즉, 오늘 날짜를 넣어야 하는지 지명을 넣어야 하는지 또는 날짜는 어떤 포맷으로 입력해야 하는지에 대한 명확한 시맨틱 정보가 따로 정의되지 않고 있다.

따라서 본 연구에서는 이러한 문제를 해결하기 위해서 웹 서비스 각 연산의 입력과 출력에 대한 시맨틱 정보를 이용하는 테스트 프레임워크를 제안한다. 이를 통해 사용자들은 단순히 메타데이터 정보에 의존한 검색뿐만 아니라 시맨틱 정보를 이용한 웹 서비스의 지능적인 테스트를 통해 정말로 자신이 원하는 서비스를 발견할 수 있으며 이를 통해 진정한 웹 서비스 기반의 비즈니스가 이루어질 수 있을 것이다.

2. 관련 연구

UDDI는 사용자들이 전화번호부에서 전화번호를 찾듯이 호출하고자 하는 웹 서비스를 등록해 놓고 검색할 수 있는 API들을 제공한다. 또한 원하는 웹 서비스를 찾은 후에는 검색된 서비스에 대한 WSDL을 통해 웹 서비스를 호출해서 실제로 사용할 수 있다. 하지만 UDDI는 서비스에 대한 간단한 메타데이터 정보만을 제공하며 실제 호출시에 사용해야 하는 WSDL도 단순히 URL만 명시하고 있어서 실제 비즈니스에서 웹 서비스를 이용하기에는 서비스에 대한 정확한 설명이 부족하다.

이러한 문제를 해결하기 위해서 최근 XMethods[2]와 Salcentral[3]과 같은 여러 웹 서비스 포탈 사이트들이 등장하였다. 이들은 웹 서비스를 등록하고 검색할 수 있는 기본적인 기능 외에 포탈 안에서 웹 서비스가 제대로 동작하는지 실행,테스트하고 서비스를 실제 구매할 수 있도록 할뿐만 아니라 자신이 등록한 웹 서비스가 몇 번 호출되고 사용되었는지에 대한 통계정보와 같은 웹 서비스 관리 기능도 제공한다.

하지만 이러한 포탈 사이트는 UDDI위에서 동작하지 않으며 웹 서비스에 대한 정보를 별도의 저장소에 따로 구축하고 서비스를 하고 있다. 또한 테스트에 필요한 입력과 출력 정보는 서비스 기술에서 입력된 메타데이터 정보만을 사용할 뿐 입력과 출력 메시지에 대한 시맨틱 정보를 따로 관리하지 않는다.

3. 지능적 웹 서비스 테스트 프레임워크

본 연구에서는 웹 서비스 사용자들이 웹 서비스가 어떻게 동작하는지 정확하게 테스트할 수 있도록 웹 서비스의 각 연산마다 입력과 출력에 대한 시맨틱 정보를 추가함으로써 보다 지능적인 테스트를 할 수 있도록 한다.

그림 1은 웹 서비스의 연산에 시맨틱 정보를 추가하는 3가지 방법을 설명한 그림으로 각각의 방법들은 하위 절에서 설명한다.

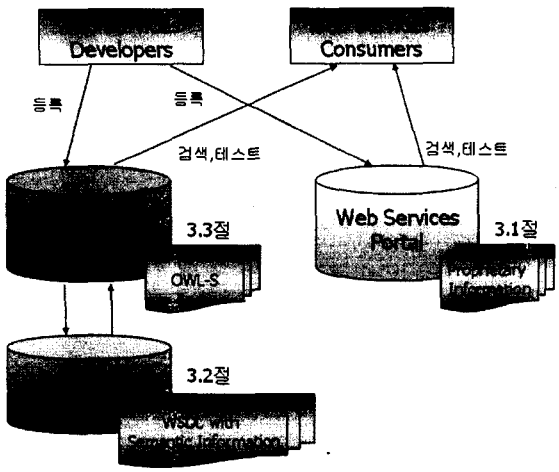

그림 1 지능적 웹 서비스 테스트 프레임워크

3.1 Proprietary 정보

웹 서비스 개발자가 자신이 개발한 웹 서비스들을 XMethods나 Salcentral과 같은 웹 서비스 포탈에 등록하는 경우 웹 서비스의 입력과 출력 정보에 시맨틱 정보를 추가하고자 하려면 웹 서비스 포탈에서의 웹 서비스 관리 정책 또는 포탈을 구현한 방법에 따라 표준에 기반한 방법 대신 별도의 저장소를 통해 등록할 수 있다. 현재 대부분의 웹 서비스 포탈 사이트가 이와 비슷한 방법을 사용하지만 시맨틱 정보를 입력 받아 관리하지는 않는다.

3.2 WSDL의 documentation 엘리먼트

현재 웹 서비스 상호 운용성 기구인 WS-I[4]에서 권고하는 WSDL 버전은 1.1이다. 하지만 WSDL 1.1은 입출력 인자의 데이터 타입만 기술하게 되어 있으며 입출력에 시맨틱 정보를 부여할 수 없다. 따라서 WSDL의 스키마를 그래도 유지하면서 현재의 WSDL에 입출력에 대한 시맨틱 정보를 추가할 수 있도록 다음과 같이 웹 서비스의 입력과 출력 메시지를 나타내는 message 엘리먼트 안에 사람이 읽고 해석할 수 있는 정보를 담기 위해 만든 documentation 엘리먼트를 이용한다.

```
<wsdl:message name="getForecastRequest">
  <xsd:documentation>
    <annotation>도시 이름을 입력</annotation>
    <examples>대전</examples>
  </xsd:documentation>
  <wsdl:part name="in0" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getForecastResponse">
  <xsd:documentation>
    <annotation>섭씨온도 출력</annotation>
    <examples>20도</examples>
  </xsd:documentation>
  <wsdl:part name="getForecastReturn" type="apachesoap:Element"/>
</wsdl:message>
```

시맨틱 정보

그림 2 WSDL의 시맨틱 정보 예제 (일부)

위 예제는 날씨 검색 웹 서비스 중 getForecast-Request라는 연산에서 사용되는 입력과 출력 메시지 엘리먼트들을 보인 예제이다. 본래 message 엘리먼트에는 메시지 이름, 데이터 타입 그리고 인자 정보를 가지는 part 엘리먼트만 둘어가기 때문에 이 웹 서비스를 테스트하려는 사람은 이 연산의 입력에 무엇을 넣어야 하고 출력으로 어떤 결과가 나오는지 알지 못한다. 따라서 본 연구에서는 다음과 같이 시맨틱 정보를 나타내주는 두 개의 엘리먼트를 새로 정의하였으며 documentation 엘리먼트에서 삽입해 사용하도록 하였다.

- annotation : 입력과 출력 메시지에 대한 설명
- examples : 입력과 출력 메시지 예제

documentation 엘리먼트는 WSDL 스키마에서 정의된 엘리먼트로서 문서의 어떤 엘리먼트 안에도 삽입될 수 있으며 또한 documentation 엘리먼트는 mixed 내용 모델[5]을 사용하기 때문에 안에 어떤 엘리먼트나 텍스트가 혼합되어서 삽입되어도 에러가 발생하지 않는다.

웹 서비스 개발자가 위와 같은 엘리먼트를 추가한 WSDL을 UDDI에 등록했다면 웹 서비스 사용자들은 그림 1에서 정의한 Semantic Web Services Provider를 통해서 자신이 원하는 웹 서비스를 검색,테스트할 수 있게 된다.

또한 이러한 테스트를 위해서 Semantic Web Services Provider는 웹 서비스 테스트를 수행할 때 WSDL에서 위와 같은 엘리먼트를 처리할 수 있어야 하며 이 방법은 시맨틱 정보를 저장하기 위해 따로 저장소를 마련하지 않아도 되는 장점이 있다.

3.3 OWL-S의 Service Model[6]

OWS-S는 DAML-S Coalition에서 시맨틱 웹 서비스를 위해 만든 언어로 2003년에 W3C의 표준으로 채택되었다. OWL-S는 웹 서비스에 시맨틱 정보를 기술하기 위해 서비스가 무엇을 하는지를 기술하는 Service Profile, 서비스가 어떻게 동작하는지를 기술하는 Service Model, 에이전트가 어떻게 서비스를 접근할 수 있는지를 기술하는 Service Grounding들을 제공하는데 본 연구에서 해결하려는 서비스의 입출력에 대한 시맨틱 정보는 그림 3과 같이 Service Model의 hasInput과 hasOutput 엘리먼트에 담기게 된다. 또한 이 엘리먼트는 parameterType이라는 엘리먼트를 통해 각각 입력과 출력 파라미터에 온톨로지를 참조하게 되는데 이 온톨로지를 통해 입출력에 대한 시맨틱 정보를 추가할 수 있다.

그림 3은 이와 같은 OWL-S를 통해 날씨 검색 웹 서비스의 입출력 정보에 시맨틱 정보를 부여한 예제이다. 이 예제에서는 각각의 입출력 정보에 대한 온톨로지를 간단한 URI로 표현했으며 이 URI에서 입출력에 대한 정보들을 가지게 된다.

또한 웹 서비스 테스트 환경에서 OWL-S를 사용하는 경우 지능적인 웹 서비스 테스트를 수행할 수 있을 뿐만 아니라 본래의 OWL-S의 목적인 지능적인 웹 서비스의

```
<process:AtomicProcess rdf:ID="WeatherForecastProcess">
  <process:hasInput>
    <process:Input rdf:ID="WeatherForecastInput">
      <process:parameterType rdf:resource="#Temperature"/>
    </process:Input>
  </process:hasInput>
  <process:hasOutput>
    <process:ConditionalOutput rdf:ID="WeatherForecastOutput">
      <process:parameterType rdf:resource="#NameOfCity"/>
    </process:ConditionalOutput>
  </process:hasOutput>
</process:AtomicProcess>
```

그림 3 OWL-S에서의 시맨틱 정보 예제 (일부)

발견과 Composition 기능도 일부분 지원할 수 있는 장점이 있다. 하지만 OWL-S 파서가 추가적으로 제공되어야 하고 각 웹 서비스에 대한 온톨로지가 추가적으로 구축된 후에 그림 1의 Semantic Web Services Provider와 같은 곳을 통해 제공되어야 한다.

4. 결론

본 연구에서는 보다 지능적인 웹 서비스 테스트를 위해 웹 서비스에 시맨틱 정보를 추가하는 프레임워크를 제안하였다. 실제 웹 서비스 기반의 e-비즈니스가 수행되기 위해서는 웹 서비스 포탈이나 다른 웹 서비스 제공자들을 통해 사용자들이 정말로 자신이 원하는 웹 서비스인지 테스트할 수 있는 시스템이 구축되어야 한다. 하지만 현재의 웹 서비스는 이러한 테스트 방법들을 정의하고 있지 않으며 또한 현재 이와 비슷한 환경을 제공하고 있는 웹 서비스 포탈들은 입출력에 대한 명확한 시맨틱 정보를 제공하지 않는다. 따라서 본 연구에서 제안한 프레임워크를 웹 서비스 테스트 환경에 적용한다면 보다 지능적인 테스트를 통해 실제 웹 서비스가 거래되는 e-비즈니스 환경이 조성될 수 있을 것이다.

5. 참고문헌

[1] Heather Kreger, " Web Services Conceptual Architecture(WSCA 1.0)", IBM Software Group, http://www-306.ibm.com/software/solutions/webservices /pdf/WSCA.pdf, May 2001

[2] XMethods, http://www.xmethods.com

[3] Salcentral, http://www.salcentral.com/salnet/webserviceswsdl.asp

[4] WS-I, Basic Profile Version 1.0a Final Specification, http://www.ws-i.org/Profiles/Basic/2003-08/BasicProfile-1.0a.html, Aug. 2003

[5] W3C Recommendation, XML Schema Part 0: Primer, http://www.w3.org/TR/xmlschema-0/, May 2001

[6] W3C Recommendation, OWL-S: Semantic Markup for Web Services, The OWL Services Coalition, Nov. 2003