Journal of Korea Multimedia Society Vol. 12, No. 7, July 2009(pp. 883-892)

이종 객체로 구성된 분산시스템을 위한 WSDL기반 SOAP Bridge

박철우[†], 박세명[††]

요 약

분산서비스 객체들의 효율적인 연동을 제공하기 위해 제시된 WSDL기반의 SOAP메시지 방식이 제시되었으나, 이 방식에서 사용하는 SOAP 메시지와 객체참조문서(WSDL)는 해당 객체의 개발환경에 의존적이므로 서로 다른 개발환경에서 개발된 객체들의 통합을 위해서는 SOAP 메시지와 객체참조문서(WSDL)의 개발환경 의존성을 해결하여야 한다. 본 논문에서는 이종 분산객체 환경에서 객체들의 통합을 지원하기 위해 SOAP Bridge, WBSB (WSDL Based SOAP Bridge)를 제안, 구현하였다. 서비스 요청 메시지를 WSDL 저장소의 WSDL문서를 참조하여 해당 서비스 객체에 적합한 SOAP 메시지로 변경한 후, 이를 서비스 객체에게 전달하는 기능을 WBSB에서 수행함으로써 사용자는 사용하고자 하는 객체의 개발환경을 고려하지 않고도 해당 서비스를 이용할 수 있다.

WSDL-based SOAP Bridge(WBSB) for Distributed System with Heterogeneous Distributed Objects

Park, cheol Woo[†], Park, Se Myung[††]

ABSTRACT

SOAP using WSDL(Web Service Definition Language) was proposed for effective integration of distributed objects. But the SOAP message and WSDL document automatically generated from development tool are fully dependent on the development environments. So the dependency of SOAP messages and WSDL documents for objects to the development environment is still a big problem to be solved for the integration of heterogeneous distributed objects. In order to support integration of heterogeneous distributed object, we implemented SOAP Bridge called WBSB(WSDL Based SOAP Bridge). As service request message for a particular service object is translated into appropriate SOAP message for the selected service object by WBSB using WSDL repository, a client can use service object without considering the development environment of the service object.

Key words: SOAP(SOAP), XML(XML), WSDL(WSDL), Distributed Service Object(분산서비스객체), CORBA(CORBA), COM(COM)

1. 서 론

현대 사회에 들어서 인터넷이 보편화되고 하드웨어 기술이 발전하면서 일반 컴퓨터의 성능이 점차 강력해지고, 가격이 저렴해짐으로써, 고비용의 중앙 집중형 컴퓨팅 환경에서 분산컴퓨팅 환경으로 변화되고 있다. 이런 분산 환경 하에서 객체 간의 통신을 위해 사용되는 대표적인 프로토콜로서는 OMG(Object

※ 교신저자(Corresponding Author) : 박세명, 주소 : 김해시 어방동 인제대학교 컴퓨터공학부(621-749), 전화 : 055)320-3276, FAX : 055)322-3107, E-mail : smpark@cs.inje.ac.kr
접수일 : 2008년 9월 10일, 완료일 : 2009년 4월 13일

[†] 정회원, 인제대학교 대학원 전산학과
(E-mail : bluew00@naver.com)
[††] 종신회원, 인제대학교 컴퓨터공학부 교수
※ 본 연구는 2004년 인제장학재단 연구비 지원에 의한 것임.

Management Group)의 CORBA(Common Object Request Broker Architecture)[1]프로토콜인 IIOP와 마이크로소프트(Microsoft)의 DCOM(Distributed Component Object Model), 그리고 선마이크로시스템의 RMI(Remote Method Invocation)을 들 수 있다[2].

그러나 CORBA, DCOM, RMI 같은 기존의 대표적인 분산객체 기술들은 대부분 방화벽을 통과하지 못하여 보안이 강화된 인터넷상에서 객체 기반의 분산 컴퓨팅이 어려우며, 다른 프로토콜과의 상호운용에 있어 문제를 가지고 있으므로[3-5], 분산객체 기술과 Web과의 호환성 문제를 해결하기 위하여 XML을 이용하고자 하는 연구[6,7]가 시도되었으나 이 또한 XML문서 전달 및 처리를 위해 사용하는 기술에 따른 플랫폼과 프로그래밍 언어에 종속적이라는 문제점도 내포하고 있다. 이기종간 분산객체들의 상호연동을 제공하기 위하여 메타데이터를 정의하는 연구[8,9]가 수행되었으나 메타모델간의 호환성 문제로 인해 분산객체간의 통합이 어렵다는 문제점은 여전하다.

따라서 최근 Web기반으로 분산객체들 간의 상호연동을 지원하기 위해 분산객체의 표준화된 명세화를 위한 WSDL(Web Services Description Language)[10]을 기반으로 SOAP (Simple Object Access Protocol)[11]을 이용하는 방식[3,4,12-15]이 제안되어 객체를 기술하는 방식과 서비스를 요청하는 방식의 표준화를 시도함으로써 객체 통합에서 발생했던 많은 문제점을 해결하고자 하였다. 그러나 위 방식은 WSDL에 의해 표준화된 객체 명세화 는 명세화된 서비스를 누구나 이용할 수 있게 한다는 목적은 달성하였지만 사용할 서비스가 개발된 환경에 따라 상이한 WSDL문서가 작성되고 이를 기반으로 작성된 SOAP 메시지의 차이로 인해 여전히 이종 서비스 객체간의 서비스 통합에는 문제점을 가지고 있다. 따라서 이종 서비스객체에 의해 제공된 명세화된 서비스를 통일된 방법으로 이용할 수 있는 방안이 반드시 필요하다.

본 논문에서는 이를 위해 SOAP Bridge, WBSB (WSDL Based SOAP Bridge)를 구현하였다. 본 논문에서 제시, 구현된 WBSB는 이용 가능한 서비스를 수집한 후(WSDL 저장소) 사용자에게 사용가능한 서비스 목록을 제시한다. 서비스 사용자는 간단한 XML 문서를 이용하여 해당 서비스의 요청을 브리지에게 전달하면 브리지는 사용자의 요청에 해당하는 서비스 객체의 WSDL을 참조하여 해당 객체에 적합한 SOAP 메시지를 생성한 후, 이를 해당 서비스에 전달(작업 요청)한다. 본 논문에서 제안된 WBSB는 서비스 사용자에게는 사용하고자 하는 서비스 객체의 개발 환경에 무관하게 준비된 모든 이종 객체를 활용할 수 있는 편의를 제공하며, 개발자에게는 개발자에게 익숙한 개발도구를 이용, 서비스 객체를 구현할 수 있도록 지원한다.

본 논문의 구성은 다음과 같다. 2장에서는 분산객체들의 연동에 관한 방법에 대한 기존연구의 문제점을 보이고, 3장에서는 본 논문에서 제안한 SOAP Bridge, WBSB의 구성을 설명한다. 4장에서는 간단한 동작실험을 통해 제안된 WBSB를 테스트하고, 5장에서는 결론과 향후 과제를 기술한다.

2. 관련연구

CORBA, DCOM, RMI 같은 기존의 대표적인 분산객체시스템들은 동일한 프로토콜 환경에서 실행되는 분산 응용프로그램을 구현하는 데에는 아무런 문제가 없다. 그러나 이런 분산객체시스템들은 연결지향(connection-oriented)적인 프로토콜을 사용하므로 연결된 상태에서 Client 또는 Server의 갑작스런 끊김이 발생하면 적절히 대응하지 못한다. 그리고 수천 개의 Client가 Server로 연결되어 있는 경우, 비 활동 중인, Client로 인하여 Server측의 자원이 낭비 될 수도 있으며, 대부분 방화벽을 통과하지 못하여 보안이 강화된 인터넷상에서 객체 기반의 분산 컴퓨팅이 어렵다. 더욱이 프로토콜의 구조가 각각 다르며 복잡하여 다른 프로토콜 환경과의 상호운용에 있어 문제를 가진다[3-5]. 특히 기존의 분산 객체 기술은 Web과의 호환성을 직접 가지지 못하기 때문에 Web을 통하여 데이터 교환이 필요한 어플리케이션에서는 데이터 교환을 위한 별도의 프로토콜을 지원하여야 했으므로 이러한 문제를 해결하기 위하여 Web server를 이용한 이기종 분산객체의 연동방식이 제안되었으며, 이러한 사례로 XML과 분산 객체 기술인 CORBA의 상호연동에 관한 연구를 들 수 있다[7].

Web과 분산객체시스템을 연동하기 위하여 CGI

를 이용하면 구현이 쉽고 모든 언어로 구현 할 수 있는 장점을 가지나 낮은 수행능력과 큰 오버헤드, 그리고 빈약한 CGI간 통신능력 등 여러 가지 단점을 가지고 있다. Servlet을 이용하면 Web Server의 부하를 쓰레드를 사용하여 줄일 수 있기 때문에 초기화에 따른 오버헤드가 없고 사용자 수가 증가하더라도 시스템 성능이 비례적으로 감소하는 현상이 없는 장점을 가지나 Servlet을 지원하는 Web Server가 적고, Web Server에서 Servlet를 인식할 수 있도록 하기위한 Servlet 엔진이 필요하다는 단점을 가지고 있다. Web API(NSAPI, ISAPI)를 이용하면 CGI보다 더 빠르게 실행되는 Web Server 프로그램을 구현할 수 있는 장점을 가지나 표준이 아니기 때문에 Web Server가 바뀔 경우 구축된 프로그램을 다시 구현해야하는 단점을 가진다. Applet을 이용하면 CORBA 모듈을 다운로드하기 때문에 어떤 플랫폼이라도 CORBA 기반의 통신을 할 수 있는 장점을 가지나 CORBA 모듈을 다운로드 하는 큰 부하가 발생하는 단점을 가진다. 메타데이터를 이용하면 모든 객체를 관리하기 편한 장점을 가지나 서로 다른 메타데이터간의 호환성을 지원하지 않는 단점을 가진다[7-9]. 이러한 문제를 해결하기 위해 SOAP 응용프로그램을 이용한 연동 구조가 제시되었으며, 그림 1은 SOAP 응용프로그램을 이용한 이기종 분산객체의 연동 구조이다.

위 그림에서 SOAP은 단순히 텍스트 형식의 메시지만 주고받을 뿐 직접적인 서비스를 제공하지 못하고, 오류를 직접 처리하지 못하는 단점을 가지지만 플랫폼에 무관하게 모든 분산객체들과 연동 할 수 있으며, 또한 Web을 통한 통신도 지원한다. 이러한 기능으로 인해 SOAP은 서로 다른 분산컴퓨팅 프로토콜을 사용하는 객체들 간에 중계자 역할을 함으로써 그림 1과 같이 이질 객체간의 연동을 지원하는 분산 응용프로그램의 구현을 가능하게 하였다[12-14]. 즉 기존에는 CORBA, RMI, DCOM을 사용하는 분산객체들을 연동하기 위한 분산 응용프로그램을 구현하기가 어려웠으나, 그림 1과 같이 각각의 분산객체 사이에 SOAP 응용프로그램만 추가하면, 분산객체 간에 통신이 가능하게 되었다. 그러나 XML 기반의 SOAP 메시지처리 기능이 모든 분산객체에 추가되는 문제점을 해결하기 위해 분산객체의 Client와 Server 사이에 SOAP을 직접 이용하지 않고 아래의 SOAP Client를 이용한 연동 구조가 제시되었다. 그림 2는 SOAP client를 이용한 분산객체의 연동구조이다.

SOAP을 이용한 CORBA와 XML의 연동을 보면, SOAP Server에서 CORBA Client로 요청을 보내므로 CORBA 환경의 수정 사항 없이 연동할 수 있음을 그림 2를 통하여 볼 수 있으며 기존 분산객체 기술이 가지는 단점을 보완하고 있음을 알 수 있다[4]. 그리고 SOAP을 이용하지만 전체 요청처리과정은 연결지향적이지 않아 Client와 Server의 갑작스런 끊김과 Client의 비활동으로 인한 Server측의 자원낭비가 없고, 방화벽을 통과할 수 있다는 장점을 가진다. 또한 SOAP 메시지는 복잡하지 않으므로 상호연동에 있어 아주 유용하다[11]. 이처럼 SOAP을 이용한 연동구조는 Web을 통한 통신과 상호연동에만 초점을 맞추면 아주 완벽하나, 분산객체들의 확장과 통합에 있어서는 많은 수정사항이 따르게 된다. 예를 들어 기존 CORBA 분산객체 연동구조에 DCOM 분산객체를 확장하고자 한다면 DCOM 분산객체 환경에 맞는 SOAP Server와 SOAP Client를 각각 구현하여야 한다. 그리고 이를 이용한 서비스를 이용하기 위해서는 Web Client에서 모든 분산객체의 구현방식를 알고 있어야 한다는 심각한 문제점을 가지며, 이

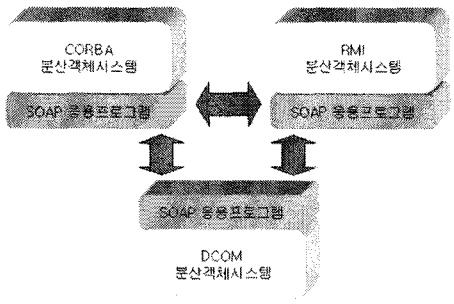

그림 1. SOAP기반 이종 분산객체 연동 구조

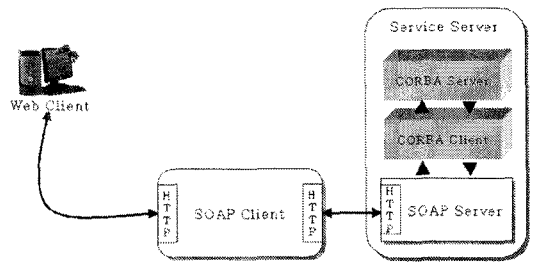

그림 2. SOAP Client를 이용한 연동 구조

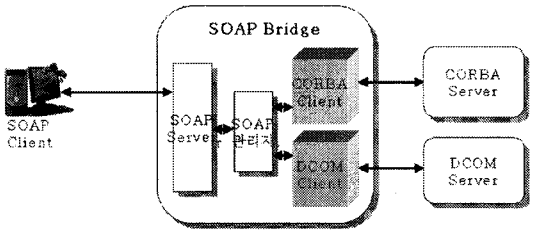

그림 3. SOAP Bridge를 이용한 연동구조

를 위해 SOAP Bridge를 이용한 분산객체 연동 구조가 제시되었다[3]. 그림 3은 SOAP Bridge를 이용한 분산객체 연동구조를 보이고 있다.

SOAP Bridge를 이용한 분산객체 연동방식은 Web을 통한 통신이 가능하고 다른 분산객체들과 상호연동을 제공한다. 그림 2에서 SOAP을 이용한 연동구조와는 달리 그림 3에서는 분산객체마다 SOAP Client를 각각 두는 것이 아니라 SOAP Bridge를 둠으로써 분산객체들을 보다 효과적으로 통합, 관리할 수 있음을 보여주고 있다[3]. 그러나 SOAP Bridge를 이용한 연동방식도 분산객체의 확장, 즉 분산객체의 수의 증가에 따라 각 Server 객체별로 구현되는 객체 Client의 증가에 따라 SOAP Bridge의 크기가 커질 뿐 아니라 최종 서비스 이용자는 자신이 사용하고자 하는 분산객체들에 대한 정보를 알고 있어야 한다는 문제점을 내포하고 있다.

이처럼 분산된 서비스 객체를 하나의 표현방식인 WSDL[10]로 표현하고, 메시지전달을 위해 SOAP을 이용하는 방식은 제시된 문제점외 실제 시스템을 구축하는 과정에서는 각 서비스 객체의 개발환경에 따라 객체를 기술하는 WSDL의 표현의 차이는 여전히 서비스 통합에 장애로 남아있음을 확인할 수 있다. 따라서 통합 서비스 환경의 구축을 위해서는 WSDL과 SOAP 메시지의 표현 차이를 극복하는 방안이 제시되어야 한다.

3. WBSB(WSDL Based SOAP Bridge) 제안

본 장에서 제안하는 WBSB의 핵심 기능의 필요성을 위해 각 개발환경에서 생성하는 서비스 객체를 기술하는 WSDL의 차이점과 그로인해 발생하는 SOAP 메시지의 차이점을 살펴보고, 차이점으로 인한 문제점을 해결하는 제안된 WBSB를 통한 분산객체들의 연동구조를 제안하였다.

3.1 WSDL(Web Services Description Language) 분석

WSDL은 IBM, Microsoft 등의 회사들이 Web 시스템 기능을 명세화하는 방법을 표준화하기 위해 만들어 졌다. WSDL은 Web 시스템에서 제공하는 기능들을 외부에서 이용할 수 있도록 사용법을 알려주는 인터페이스 언어로 다른 분산 컴퓨팅 기술인 CORBA의 IDL (Interface Definistion Language)에 해당한다. WSDL문서는 Web의 기능, 즉 프로시저의 이름, 인자의 종류, 반환형의 종류, 전송 프로토콜의 종류, 그리고 종점 URL 등으로 구성되며, XML로 작성되고 특정 프로그래밍 언어로 변환되어 Client 응용프로그램에서 사용된다. WSDL 문서가 활용되는 방식은 아래와 같다.

그림 4에서 보는 것과 같이 WSDL 문서에서 생성된 코드는 Web 시스템에 바인딩되어 SOAP 메시지를 생성하고, 이를 전송하는 스텁 역할을 한다. WSDL 문서는 프로그래머가 직접 작성할 수도 있으나, 보통 관련된 API나 Tool을 이용하여 작성된다. 즉 C#으로 구현된 객체는 .NET 시스템 자체에서 직접 작성되고, C, VC++ 그리고 VB로 작성된 객체는 IBM에서 제공하는 SOAP ToolKit을 이용, 작성되며, Java로 구현된 객체는 JAX-RPC를 이용, 작성된다. 이처럼 구현되는 환경에 따라 다른 방식으로 WSDL 문서가 작성되므로 작성된 문서는 기본적으로 동일한 구조를 따르기는 하나 다르게 작성되며, 이처럼 서로 다르게 작성된 WSDL 문서는 이를 기반으로 작성되는 SOAP 메시지의 차이를 유발, 통합환경 구축의 문제점으로 지적되고 있다.

그림 5는 C#으로 구현되어 .NET 시스템에서 작성

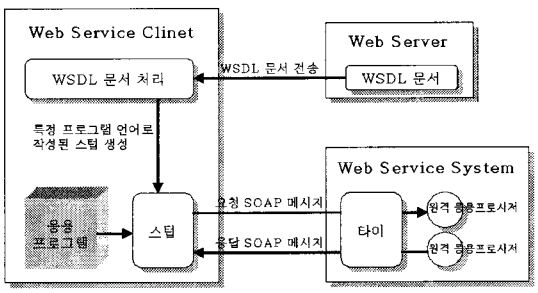

그림 4. WSDL 문서의 활용 예

된 WSDL문서의 일부분이고, 그림 6은 VB로 구현되어 IBM의 SOAP ToolKit3.0을 이용하여 작성한 WSDL문서의 일부분이며. 그림 7은 JAVA로 구현되어 JAX-RPC를 이용하여 작성한 WSDL문서의 일부분이다. 그림 5, 6, 7에서 알 수 있듯이 각각 다른 환경에서 구현되어진 분산객체를 위한 WSDL은 유사한 서비스(덧셈, 뺄셈, 곱셈)를 제공하는 객체라 하더라도 서로 다르게 작성됨을 확인할 수 있다.(밑줄친 부분 참조)

3.2 생성된 SOAP 메시지 분석

분산객체와 그를 이용하는 Client 구현 시, 개발자가 사용할 객체 정보를 고려하여 적합한 SOAP 메시지를 작성하여야 하므로 SOAP을 이용함으로써 얻을 수 있는 장점들을 최대한 활용하기 위해서는

```
- <definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:soap="http://schemas.xm
  xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:s0="http://bluew00.net/Addition/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/" xmlns:mime="http://schemas.xmls
  targetNamespace="http://bluew00.net/Addition/" xmlns="http://schemas.xmlsoap.org/wsdl/">
 - <types>
  - <s:schema elementFormDefault="qualified" targetNamespace="http://bluew00.net/Addition/">
   - <s:element name="add">
    - <s:complexType>
     - <s:sequence>
       <s:element minOccurs="1" maxOccurs="1" name="para1" type="s:double"/>
       <s:element minOccurs="1" maxOccurs="1" name="para2" type="s:double"/>
- <portType name="AdditionSoap">
 - <operation naem="add">
   <input message="s0:addSoapIn"/>
   <output message="s0:addSoapOut"/>
  </operation>
 </portType>
- <binding name="AdditionSoap" type="s0:AdditionSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
- <operation name="add">
  <soap:operation soapAction="http://bluew00.net/Addition/add" style="document"/>
- <service name="Addition">
 - <port name="AdditionSoap" binding="s0:AdditionSoap">
   <soap:address location="http:localhost/soapTest/Addition.asmx"/>
  </port>
 </service>
```

그림 5. C# 분산객체의 WSDL 문서

```
- <definitions name="Subtraction" targetNamespace="http://bluew00.net/subtraction/"
  xmlns:wsdlns="http://bluew00.net/subtraction/" xmlns:typens="http://bluew00.net/sub
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org
- <types>
 - <schema targetNamespace="http://bluew00.net/subtraction/" xmlns
   xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
   elementFormDefault="qualified">
- <message name="sub.subtrac">
  <part name="para1" type="xsd:double"/>
  <part name="para2" type="xsd:double"/>
 </message>
- <message name="sub.subtracResponse">
  <part name="Result" type="xsd:double"/>
 </message>
- <portType name="subSoapPort">
 - <operation name="subtrac" parameterOrder="para1 para2">
   <input message="wsdlns:sub.subtrac"/>
   <output message="wsdlns:sub.subtracResponse"/>
  </operation>
 </portType>
- <binding name="subSoapBinding" type="wsdlns:subSoapPort">
  <stk:binding preferredEncoding="UTF-8"/>
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
- <operation name="subtrac">
  <soap:operation soapAction="http://bluew00.net/subtraction/sub.subtrac"/>
- <service name="Subtraction">
 - <port name="subSoapPort" binding="wsdlns:subSoapBinding">
   <soap:address location="http://localhost/testWeb/Subtraction.ASP"/>
  </port>
```

그림 6. VB 분산객체의 WSDL 문서

```
- <definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://bluew
  xmlns:xsd="http://www.w4.org/2001/XMLSchema" xmlns:soap="http://schem
  targetNamespace="http://bluew00.net/multiplication/multiply">
- <message name="multiplF_mul">
  <part name="double_1" type="xsd:double"/>
  <part name="double_2" type="xsd:double"/>
 </message>
- <portType name="MultiplF">
 - <operation name="mul" parameterOrder="double_1 double_2">
   <input message="tns:MultiplF_mul"/>
   <output message="tns:MultiplF_mulResponse"/>
  - <output>
    <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/enco
     namespace="http://bluew00.net/multiplication/multiply">
    </output>
   <soap:operation soapAction=""/>
  </operation>
 <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style=
</binding>
- <service name="Multiply">
 - <port name="MultiplFPort" binding="tns:MultiplFBinding">
   <soap:address xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
   location="http://localhost:8080/multiply/multiply"/>
  </port>
 </service>
```

그림 7. Java 분산객체의 WSDL 문서

WSDL 기술도 함께 사용되어야 한다. 그런데 3.1절에서 설명한 바와 같이 WSDL 문서는 분산객체들의 구현 환경에 따라 다르게 작성되므로 WSDL과 SOAP을 이용하는 장점을 활용하고 있지 못하다. 예를 들어 Java 언어를 사용하고 Linux 환경에서 구현되어진 분산객체와 동일한 객체를 VB 언어를 사용하여 Windows 환경에서 구현한 후, 분산객체 개발환경에서 제공하는 도구를 활용하여 WSDL 문서를 작성하면 같은 기능을 나타내지만 서로 다르게 작성되며, 이를 기반으로 작성된 SOAP 메시지 또한 다르게 작성된다. 그림 8은 각각 C#과 Java로 구현된 분산객체에 서비스를 요청하는 SOAP 메시지이다.

그림 8의 왼쪽메시지는 C#언어로 구현되어진 분산객체에 보내는 SOAP message이며, 그림의 오른쪽 메시지는 Java언어로 구현된 분산객체에 보내는 SOAP message이다. C#과 Java 분산객체에 요청하기 위해 만들어진 SOAP 메시지를 보면 C#에 요청하는 SOAP message는 SOAPAction이 HTTP 헤더에 포함하고 있고, 파라미터 엘리먼트에 type이 기술되지 않았으나, Java에 요청하는 SOAP message는

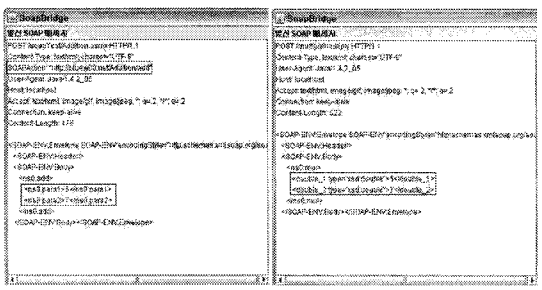

그림 8. SOAP 메시지 비교

SOAPAction이 기술되지 않았으며, 파라미터 엘리먼트에 type이 기술되어져 있고, 엘리먼트 이름도 바뀌어져 있는 것을 볼 수 있다. 이런 차이점은 WSDL 문서에 따라 각각의 SOAP message가 다시 작성되어야 함을 보여준다.

3.3 효과적인 연동구조를 지원하는 WBSB

SOAP Bridge는 단지 여러 분산객체들을 처리 하기위하여 분산객체들에 대한 정보를 바탕으로 Client들의 요청에 따라 SOAP 메시지를 중계하는 역할을 한 것에 지나지 않는데 반해, 제안된 WBSB는 .Client의 요청 SOAP메시지를 분석하여 서비스 제공자인 분산객체에 맞게 다시 SOAP 메시지를 작성, 전달하고, 분산객체에서 처리된 결과 SOAP 메시지를 받아 다시 Client에게 전달한다. 즉 본 논문에서 제안한 WBSB는 WSDL저장소를 두고 객체생성 환경에 따라 조금씩 다르게 작성되어지는 WSDL문서를 수집, 분석하여 SOAP message를 작성시 필요한 정보, 즉 soapAction, targetNamespace, methodName, parameterName, parameterType, destinationAddress, messageName를 획득하고, 이를 이용하여 분산객체에 적합한 SOAP message를 작성함으로써 개발환경에 무관하게 서비스 객체들의 상호 연동을 보다 효과적으로 지원할 수 있게 된다.

제안된 SOAP Bridge는 WSDL이라는 명세화된 문서를 동적으로 참조하여 SOAP 메시지를 작성하며, WSDL저장소를 활용하므로 Client는 서비스제공자인 분산객체의 구현과 무관하다. 본 논문에서 제시된 분산객체간의 연동방식은 그림 9와 같다.

3.4 WBSB의 기능

그림 9에서 제시된 WBSB를 기능별로 나타내면 그림 10과 같으며, 이는 XML관리자, SOAP관리자,

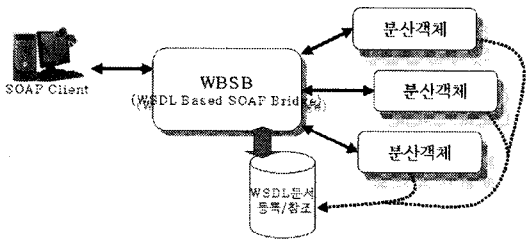

그림 9. WBSB에서 분산객체들의 연동

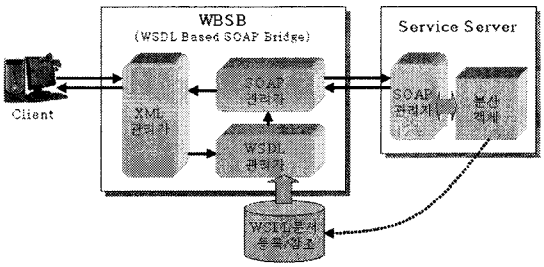

그림 10. WBSB와 분산객체들의 연동

WSDL관리자로 구성된다.

제안된 WBSB와 분산객체 연동 흐름은 다음과 같다. 먼저 Service Server는 WSDL 문서를 작성하여 WSDL 저장소에 등록한다. 그 후 Client와 WBSB 그리고 Service Server간에 서비스를 요청하고 응답한다. Client는 간단한 XML 요청 메시지를 WBSB에 보낸다. 이 요청 메시지를 WBSB의 XML관리자가 받아서 메시지를 분석하여 그 정보를 WSDL관리자에게 전달한다. WSDL관리자는 정보에 맞는 WSDL 문서를 검색하고 SOAP 관리자에게 검색결과를 전달한다. SOAP 관리자는 전달받은 정보를 바탕으로 Service Server에 요청할 SOAP 메시지를 작성하고 Web을 통하여 요청한다. Service Server의 SOAP 관리자는 요청을 전달받으면 SOAP 메시지를 분석하고 해당 분산객체에 서비스를 요청한다. 분산객체에서 처리된 결과를 기반으로 SOAP 관리자가 SOAP 메시지를 작성하여 응답한다. 그러면 다시 WBSB가 받아 응답 SOAP 메시지를 분석하고 결과만을 추출하여 Client에게 XML 응답 메시지를 보내게 된다. 구현된 WBSB의 모듈중 XML관리자는 XML 문서 분석/작성기능을, WSDL관리자는 WSDL 문서의 분석 및 WSDL 문서 정보를 관리하며, 마지막으로 SOAP관리자는 SOAP 메시지를 분석, 작성하는 기능을 갖는다.

4. 제안된 SOAP Bridge 구현 및 평가

본 논문에서 제안하는 SOAP Bridge를 테스트하기 위하여 SOAP Bridge와 Client를 구현하였으며, 서비스제공자인 Server 객체는 각각 다른 환경 하에서 서로 다른 방식으로 구현하여 제안된 WBSB의 동작을 평가하였다.

4.1 평가 환경

SOAP Bridge는 Client의 서비스 요청을 분석하여 서비스제공자인 분산객체에게 요청 SOAP message를 만들어 서비스를 요청하며, 분산객체에서 처리된 결과인 응답 SOAP message를 받아 분석하여 그 결과를 Client에게 전달한다. 실험을 위해서는 SOAP Bridge의 동작을 확인할 수 있도록 SOAP Bridge의 동작의 핵심인 SOAP message의 흐름과 그 내용을 가시화하였다. 즉 SOAP Bridge의 기능에 있어 가장 중요한 부분인 WSDL을 분석하여 동적으로 작성한 요청 SOAP message와 Service Server로부터의 응답 SOAP message를 출력하게 하여 이를 비교할 수 있게 평가용 인터페이스를 구현 하였다. 그림 11은 SOAP Bridge의 GUI이며, 발신 SOAP message 창은 분산객체에게 요청하는 SOAP message를, 수신 SOAP message 창은 분산객체로부터 받은 응답 SOAP message를 보여주도록 구성하였으며, Java를 이용하여 XML과 SOAP 메시지 분석 및 전송 기능을 갖도록 구현하였다[5]. WBSB 개발 환경은 표 1과 같다.

Client는 Server와 연결한 후 Server 객체에 서비스를 선택, 요청할 수 있으며, 요청은 HTTP 프로토콜을 이용하여 XML 문서로 요청을 전달하도록 구현하였다. 그림 12는 본 논문에서 실험을 위해 구현한 Client GUI이며, VB(Visual Basic)을 이용하여 XML 분석과 전송 기능과 Addition, Subtraction, 그리고 Multiplication 3개의 서비스를 요청할 수 있다.

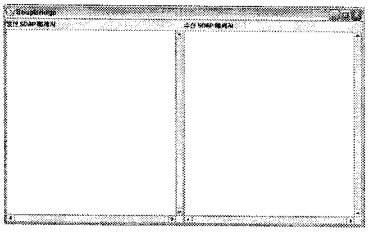

그림 11. SOAP Bridge GUI

표 1. SOAP Bridge 구현 환경

구현환경 \ Service	구분
O/S	Windows 2000 Server
프로그래밍 언어	Java
개발 툴	J2sdk 1.4.2

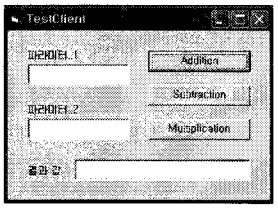

그림 12. Client GUI

표 2. Client 구현 환경

구현환경 \ Service	구 분
O/S	Windows 2000 Server
프로그래밍 언어	Visual Basic, ASP
개발 툴	visual studio 6.0

Client 구현환경은 표 2와 같다.

서비스객체는 덧셈, 뺄셈, 곱셈의 기능을 제공하며, 표 3의 환경에서 구현하였다. 서비스객체는 HTTP를 통하여 통신하기 위하여 각각 환경에 맞는 Web Server를 이용하여 Web상에 노출 하였으며, Web에 등록과 WSDL 작성은 각 객체의 개발환경에서 수행하였다.

Addition 서비스객체는 .Net기반으로 SOAP을 지원하며 WSDL 생성이 자동으로 지원되는 C#을 이용하여 Web 서비스 프로그램으로 구현하였고[16], Multiplication 서비스객체는 java.rmi.reomte 인터페이스를 확장하여 서비스를 구현하였으며, SOAP을 지원하기 위한 WSDL 생성은 J2EE 1.4 SDK에 패키지 된 wscompile툴을 이용하였다[5,17]. 마지막으로 Subtraction 서비스객체는 VB를 이용하여 COM 객체로 서비스를 구현하고, SOAP을 지원하기 위한 WSDL 생성은 SOAP ToolKit 3.0을 이용하였다[18,19].

표 3. 서비스제공 객체 구현 환경

구현환경 \ Service	Addition (덧셈)	Subtraction (뺄셈)	Multiplication (곱셈)
O/S	Win XP	Win 98	RedHat Linux 8.0
프로그래밍 언어	C#	Visual Basic	Java
개발 툴	visual studio.net	visual studio 6.0	J2sdk 1.4.2
Web Server	IIS 5.0	PWS 4.0	Apache Tomcat 5.0

4.2 제안된 SOAP Bridge WBSB 테스트

본 논문에서 제안하는 SOAP Bridge를 테스트하기 위하여 Client, SOAP Bridge 그리고 3가지의 서비스객체를 아래 그림 13과 같은 실험환경을 구축하였다.

아래 그림 14, 15, 16은 Addition, Subtraction, Multiplication 3가지의 서비스객체를 실행한 후, SOAP Bridge를 이용하여 Client를 실행한 결과를 나타낸다.

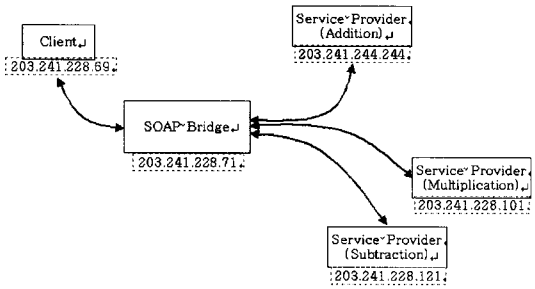

그림 13. SOAP Bridge Test 환경

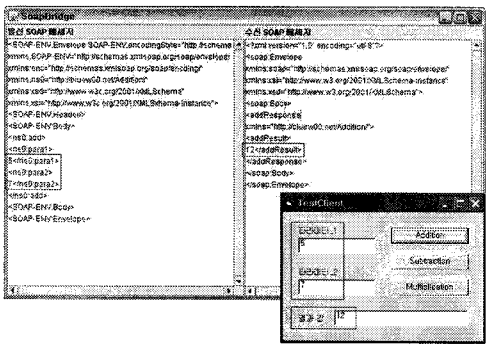

그림 14. Addition 실행 결과

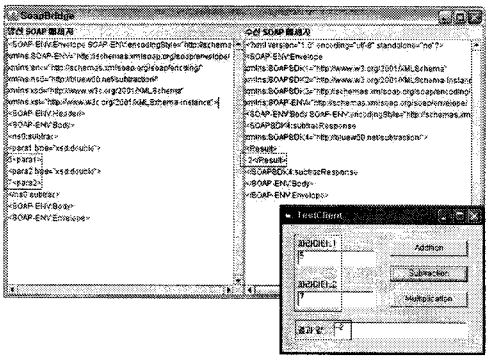

그림 15. Subtraction 실행 결과

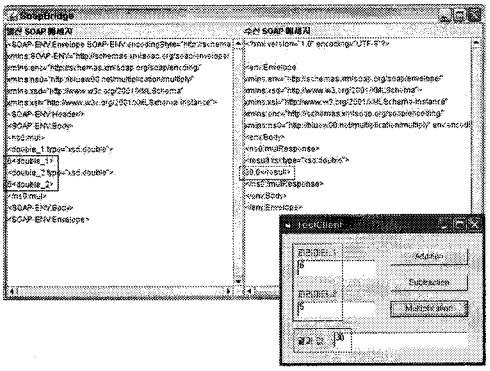

그림 16. Multiplication 실행 결과

이와 같은 결과를 통해서 본 논문에서 구현된 SOAP Bridge는 표 3과 같이 이종의 운영체제 하에서 서로 다른 프로그래밍 언어를 사용하는 개발 환경에서 구현되어진 여러 서비스 객체들을 하나의 Client, 즉 동일한 접근방식으로 준비된 서로 다른 서비스들을 사용할 수 있음을 보여주었다. 즉 서로 다른 환경의 호스트 상에 구현, 준비된 이종 서비스 객체들을 SOAP 브리지를 통하여 하나의 Client 인터페이스를 이용하여 호출할 수 있도록 지원함으로써 사용자에게는 사용할 객체가 어떤 환경에서 구현되었는지에 대한 지식을 요구하지 않으며, 서비스제공자들에게는 개발한 서비스를 등록하는 것을 요구할 뿐, 특별한 개발환경에 종속되기를 요구하지 않는다는 점에서 본 논문에서 제안한 SOAP Bridge는 이종객체들을 효과적으로 통합할 수 있음을 보여주고 있다.

5. 결 론

본 논문에서는 WSDL문서 참조를 통한 동적 SOAP메시지 작성 기능을 바탕으로 분산객체들 간의 상호연동을 지원하는 WBSB(WSDL Based SOAP Bridge)를 구현하고 평가하였다. 실험결과는 본 논문에서 구현한 WBSB는 서비스 객체의 개발환경에 무관하게 이종 분산객체를 동일한 방법으로 사용할 수 있도록 지원함을 보여준다. 이러한 결과는 사용자는 사용하고자 하는 서비스 객체의 개발 환경에 무관하게 준비된 모든 이종 객체를 활용할 수 있으며, 개발자는 자신에게 익숙한 도구를 이용, 서비스 객체를 구현할 수 있음을 보여준다.

분산 컴퓨팅 환경에서 서비스기반 컴퓨팅 환경으로 변화해가고 있는 현실의 상황을 고려해볼 때 객체가 제공하는 서비스의 식별은 다양한 처리객체를 효과적으로 통합, 이용할 수 있는 환경을 제공하기위해 필수적으로 해결되어야 할 것으로 판단되며, 또한 효과적인 WSDL저장소 구현에 대해서는 보다 많은 연구가 필요할 것으로 판단된다. 그리고 실험에서는 전용프로그램을 이용하여 서비스를 요청하였으나 서비스에 대한 사용자 접근성을 향상시키기 위해서는 Web API를 통해 서비스에 접근할 수 있게 지원하는 등 서비스 이용시 사용편의성을 증대시키는 추가 연구가 필요할 것으로 사료된다.

참 고 문 헌

[1] Kurt Wallnau, "Common Object Request Broker Architecture," CORBA v2.3, http://www.sei.cmu.edu/str/descriptions/corba_body.html, SEI(Software Engineering Institute | Carnegie Mellon), 11 Jan. 2007.

[2] Roger Sessions, ".NET vs JAVA 2 Enterprise Edition(J2EE) 비교 - eBusiness를 향한 두 개의 비전," http://www.bitclub.co.kr/download/tech_down/NETvsJ2EE.doc, ObjectWatch, Inc, Cutter Consortium 11/12. 2000.

[3] 박성은, 김신우, 이용규, "SOAP 브리지를 이용한 분산객체시스템의 연동," 정보처리학회 논문지A, 제10-2호, pp. 141-144, 2003.

[4] 이호섭, 홍충선, "웹 응용서버와 SOAP을 이용한 CORBA와 XML의 연동구조," 멀티미디어 저널, 제1권. 제2호, Dec. 2001.

[5] 신민철, "XML 웹 서비스," FREELEC, 서울, 2003.

[6] "Extensible Markup Language (XML) 1.0 (Fourth Edition)," W3C Recommendataion, http://www.w3.org/TR/2006/REC-xml-20060816/, 16 Aug. 2006.

[7] Stven Vermeulen, Bart Bauwens, Rrans Westerhuis, Rudi Broos, "XML and CORBA, Synergistic or Competitive?," IS&N 2000, Proceeding, pp. 155-168, 2000.

[8] "XMLDOM : DOM/Value Mapping Updated Submission," OMG Document orbos,/00-05-01, http://www.omg.org/docs/orbos/00-05-01.pdf, 01 May 2000.

[9] Mark Elenko and Mike Reinerstsen, "XML & CORBA," CORBA, XML And XMI Resource Page, http://www.omg.org/docs/orbos/00-05-01.pdf, Sep. 1999.

[10] "Web Providers Description Language (WSDL) Version 2.0 Part 1: Core Language," W3C Recommendation, http://www.w3.org/TR/2007/REC-wsdl20-20070626/, 26 June 2007.

[11] "SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)," W3C Recommendation, http://www.w3.org/TR/2007/REC-soap12-part1-20070427/, 27 Apr. 2007.

[12] "Simple CORBA Object Access Protocol (SCOAP)," OMG TC Document orbos/00-09-03, ftp://ftp.omg.org/pub/docs/orbos/00-09-03.pdf, 12 Sep., 2000.

[13] Dave Winer, "The XML RPC Seecifications" XML-RPC.com(UserLand Software Inc), ftp://ftp.omg.org/pub/docs/orbos/00-09-03.pdf, 15 June 1999.

[14] 이희권, 서희석, 김희완, "컴포넌트를 기반으로 한 SOAP 구조" JOURNAL OF THE KOREA SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS, KSIAM IT series Vol.10 No.1, pp. 71-82, Aug., 2006.

[15] 윤인숙, "웹 서비스를 통한 시스템 통합의 구현," eBizKorea. 통권77호, pp. 4-7, Oct. 2005.

[16] 문건웅 역, "C# Web Service," 정보문화사, 서울, 2002.

[17] "Designing Web Services with the J2EE(TM) 1.4 Platform : JAX-RPC, SOAP, and XML Technologies," The Java BluePrints team, http://java.sun.com/blueprints/guidelines/designing_webservices/html/, May, 2005.

[18] Rick Strahl, "Using Microsoft's SOAP Toolkit for remote object access," West Wind Technologies, http://www.west-wind.com/presentations/soap/, Nov., 2001.

[19] Keith Ballinger, "Real World XML Web

Services: For VB and VB .NET Developers," Addison-Wesley Publishing Co., 2002.

박 철 우

1996년 3월~2003년 2월 인제대학교 컴퓨터공학과 학사
2003년 3월~2008년 8월 인제대학교 전산학과 석사
2005년 2월~2007년 3월 (주)지란지교소프트 연구원
2007년 3월~현재 (주)지란지교소프트 선임연구원
관심분야 : 분산처리, 클라우드 컴퓨팅, 영상처리, 이미지 프로세싱, 패턴인식

박 세 명

1994년 8월 경북대학교 전자공학과, 전산공학 전공(Ph.D)
1990년 3월~현재 인제대학교 컴퓨터공학부 교수
관심분야 : 분산처리, 유비쿼터스 컴퓨팅, Grid컴퓨팅 지능형홈시스템