컴포넌트 테스팅을 위한 Web Service의 구현

박세희 [0] 진영택*
nm242m@hanmail.net ytjin@hanbat.ac.kr

An Implementation of Web Service for Component Testing

Sehui Park [0] Young Taek Jin*
*Hanbat National University

요 약

컴포넌트를 기반으로 하는 소프트웨어 개발에 대한 많은 관심은 컴포넌트가 요구 사항에 명시된대로 동작하는지를 테스팅 하는 일과 테스트된 컴포넌트 조립하여 어플리케이션을 개발하는 일에 주어지고 있다. 본 논문에서는 웹 서비스 기술을 이용하여 컴포넌트 구매자에게 후보 COM 컴포넌트를 편리하게 테스팅할 수 있는 테스팅 환경의 구현에 대해 서술한다. 이를 통해 컴포넌트 제공자 및 사용자는 컴포넌트의 신뢰성과 이해성을 증진시킬 수 있다.

1. 서 론

컴포넌트 기반 개발(CBD)은 단기간에 걸쳐 소프트웨어 제품을 제작할 수 있어 최근 각광받고 있는 개발 기술이다. 이러한 컴포넌트 기술에 힘입어 현재 많은 수의 컴포넌트 전문 판매 사이트가 운영되고 있으며, 수 많은 상용 컴포넌트가 이 곳에서 판매되고 있다[1]. 이러한 마케팅 환경에서 컴포넌트 판매자는 컴포넌트의 신뢰성 보증 및 판매 촉진을 위해 구매자가 원하는 컴포넌트를 정확하게 테스팅하고 소프트웨어에 적용시킬 수 있도록 할 수 있는 환경을 제공해야 한다. 현재 운영되고 있는 판매 사이트는 구매자에게 컴포넌트에 대한 명세서와 관련 문서, 그리고 구매하기 전 평가할 수 있는 데모 컴포넌트를 제공한다. 구매자는 원하는 컴포넌트를 검색한 후, 컴포넌트 명세서와 데모 컴포넌트를 구매자의 개발환경에 다운로드한다. 구매자는 명세서를 참조하여 데모 컴포넌트를 구동시키기 위해 클라이언트 프로그램을 작성하고 컴포넌트가 제대로 동작하는지를 테스팅한다. 구매자는 클라이언트 프로그램을 제작하거나 테스팅할 수 있는 환경을 갖춰야 한다. 또한 클라이언트 프로그램에서 컴포넌트를 생성하고 인터페이스 메소드를 호출하는 방법에 대해 알고 있어야 한다. 컴포넌트 테스팅 웹 서비스는 구매자가 컴포넌트 명세서를 웹 브라우저상에서 손쉽게 활용할 수 있고 클라이언트 프로그램 제작 과정 없이 웹브라우저 상에서 직접 컴포넌트를 테스팅할 수 있는 환경을 제공한다. 본 논문에서 제시하는 컴포넌트 테스팅 웹 서비스는 Microsoft사의 COM(Component Object Model)[2] 컴포넌트를 테스팅할 수 있다.

COM은 Microsoft사에서 내놓은 컴포넌트 기술이다. COM은 바이너리 표준을 따르므로 이식성이 강하며 플랫폼에 독립적으로 동작한다. 그리고 Stub과 Proxy을 사용하여 다른 곳에 있는 컴포넌트에 접근할 수 있어 위치 투명성을 보장한다. 또한 새로운 버전의 컴포넌트가 개발되더라도 이전 버전의 인터페이스는 새로운 컴포넌트가 모두 포함하므로 버전관

본 연구는 한국과학재단 목적기초연구 (R01-2001-000-00343-0(2003))지원으로 수행되었음

리의 문제점을 배제할 수 있다.

2. 웹 서비스

웹 서비스는 다양한 플랫폼에서 실행되는 서로 다른 어플리케이션들 간의 상호 운용성을 제공하는 소프트웨어 시스템이다[3]. 웹 서비스는 SOAP(Simple Object Access Protocol)[4]을 통하여 상호 운용성을 제공한다. SOAP은 XML기반으로서 서로 다른 어플리케이션 간에 쉽게 메시지를 전달할 수 있다. 그리고 HTTP, SMTP, TCP, JMS, IIOP 등 다양한 프로토콜을 지원하여 확장성이 뛰어나다. 웹 서비스는 그 자체로서 하나의 완전한 어플리케이션의 의미를 갖는게 아니라 대규모 솔루션의 한 기능을 담당하는 컴포넌트 역할을 한다. 이는 웹 서비스가 프로그램과 프로그램의 상호작용을 위한 목적에 의미를 둔다고 할 수 있다[5]. 이러한 장점으로 인해 서로 다른 언어로 작성된 웹 서비스 간에 통신이 가능할 뿐만 아니라 PC 그리고 PDA 같은 모바일 장치에서도 웹 서비스와 통신 할 수 있다. 이러한 웹 서비스의 다양한 특징 때문에 컴포넌트 구매자의 환경과 독립적으로 테스팅이 가능하다.

3. 컴포넌트 테스팅 시스템의 구성

전체 시스템 구성도는 [그림1]과 같다.

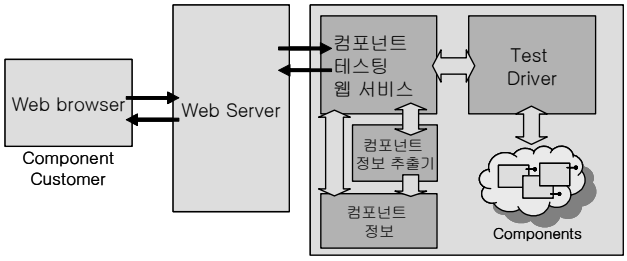

그림1 전체 시스템 구성도

구매자는 웹 브라우저를 통해 사용할 후보 컴포넌트에 대한 명세서 참조 및 검색을 수행하고 기능성을 테스팅 하기 위해 웹 서비스를 이용한다. 웹 서버는 컴포넌트 구매자와 웹 서비스의 중간에 위치하며 구매자의 요청을 웹 서비스로 전달하고 웹 서비스로부터의 처리결과를 구매자에 제공한다. 웹 서비스는 웹 서버가 수신한 클라이언트의 모든 요청을 실제로 처리한다. 처리 과정에서 웹 서비스는 컴포넌트의 등록에서 부터 컴포넌트 정보를 추출하고 관리하며 테스트 드라이버를 구동하여 컴포넌트를 테스팅한다. 테스트 드라이버는 컴포넌트 개발자가 제공한 테스트 케이스 정보를 가지고 컴포넌트 객체를 생성하고 인터페이스 메소드를 호출한다. 컴포넌트 테스트 결과가 컴포넌트 구매자에게 제공되며 아울러 이 과정에서 컴포넌트에 대한 이해가 증진될 수 있다.

3.1 테스팅 웹 서비스

1) 컴포넌트 제작자는 컴포넌트를 판매하기 전에 웹 서비스에 판매할 컴포넌트를 등록하기 위한 환경을 제공하며 등록 과정에서 컴포넌트의 이용과 관련한 부가 문서를 함께 등록한다. 부가 문서는 구매자가 컴포넌트에 대해 이해하기 쉬운 형태로 된 텍스트 파일 혹은 웹 페이지로써 컴포넌트가 포함하고 있는 각각의 인터페이스들에 대한 자세한 설명과 메소드가 수행하는 기능, 그리고 컴포넌트를 사용하는 간단한 예제들을 함께 포함한다. 그리고 컴포넌트 개발과정에서 얻어진 컴포넌트 내부 정보를 바탕으로 실제 컴포넌트 테스팅에서 필요한 테스트 케이스 정보 또한 가지고 있다. 웹 서비스는 등록 과정에서 컴포넌트 개발자가 작성한 컴포넌트 명세의 내용과 컴포넌트 구현간의 불일치를 탐지하기 위해 컴포넌트로부터 정보를 추출한다. 이 정보는 컴포넌트 생성 및 메소드 호출을 위해 CLSID, ProgID, InterfaceID 등이며 XML 형식으로 표현된다[그림2].

2) 컴포넌트 구매자가 컴포넌트 명세서를 요청하면 요청 메시지는 웹 서버를 통해 웹 서비스로 전달된다. 웹 서비스는 요청 메시지를 분석하여 컴포넌트 명세의 내용과 컴포넌트 구현간의 불일치가 없을 경우 등록된 컴포넌트 명세서를 웹 서버에 전달한다.

3) 컴포넌트 구매자가 컴포넌트를 테스팅하고자 하는 경우 개발자가 제공한 XML 형식으로 변환된 테스트 케이스는 테스트 드라이버로 전달되며 [그림3] 테스트 드라이버가 테스트 케이스를 바탕으로 컴포넌트 메소드를 호출하여 생성된 결과값은 테스트 케이스와 같이 XML형식으로 변환된다[그림4].

3.2 테스팅의 구현

테스트 드라이버는 웹 서비스로부터 XML 형태의 테스트 케이스를 전달받아 컴포넌트 객체를 생성하고 인터페이스 메소드를 호출하며 테

스트 환경의 구성은 [그림5]와 같다.

```
<?xml version="1.0" encoding="euc-kr" ?>
<cominfo>
 <coclasses>
  <coclass name="Calculator" uuid="
{0FC1D312-B533-4B0E-830F-7423C7C89AAC}" />
  </coclasses>
  <enums />
  <structs />
  <unions />
 <interfaces>
 <interface name="ICalculator" uuid="
{EEE36036-8AAC-469C-A2FB-AAD4075A2D71}"
                dual="true"
                helpstring="ICalculator Interface">
 <function name="Calculate" rettype="HRESULT">
  <parameter name="strExpression" type="BSTR" flag="in" />
  </function>
 </interface>
</cominfo>
```

그림2 컴포넌트로부터 추출된 컴포넌트 정보

```
<?xml version="1.0"?>
<invokeinfo>
  <coclass name="Calculator" uuid=
        "{0FC1D312-B533-4B0E-830F-7423C7C89AAC}"/>
 <interface name="ICalculator" uuid=
          "{EEE36036-8AAC-469C-A2FB-AAD4075A2D71}">
   <function name="Calculate">
     <parameter value="3 * ( 4 - 2 ) - 5" type="BSTR"
      flag="in"/>
   </function>

   <function name="Calculate">
     <parameter value="3 * ( 4 - 2 - 5" type="BSTR"
      flag="in"/>
   </function>
 </interface>
</invokeinfo>
```

그림3 테스트 드라이버로 전달되는 테스트 케이스

```
<?xml version="1.0"?>
<invokeResult>
  <coclass name="Calculator" uuid=
        "{0FC1D312-B533-4B0E-830F-7423C7C89AAC}"/>
 <interface name="ICalculator" uuid=
          "{EEE36036-8AAC-469C-A2FB-AAD4075A2D71}">
   <functionResult name="Calculate">
     <return>S_OK</return>
   </functionResult>
   <Exception>NOEXCEPTION</Exception>

   <functionResult name="Calculate">
     <return>S_FAIL</return>
   </functionResult>
   <Exception>E_INVALID_EXPRESSION</Exception>
 </interface>
</invokeResult>
```

그림4 테스트 드라이버로부터 생성된 결과

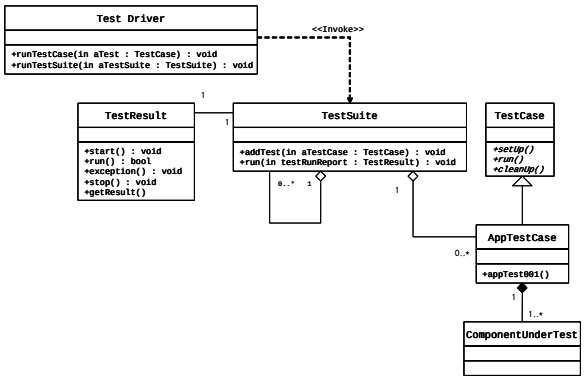

그림5 테스팅 프레임웍의 구성도

테스팅 환경은 Incremental Testing Framework[10]에 제시된 테스팅 패턴을 기본으로 하여 구현되었다. 테스트 드라이버는 테스트 케이스의 집합인 테스트 슈트(test suite)를 구동한다. 이 테스트 슈트는 테스트 결과를 보고하기 위한 테스트 결과 객체를 생성하며 추상 클래스인 테스트 케이스로부터 상속을 받은 어플리케이션 테스트 케이스 객체를 추가하거나 생성한다. 이 객체는 테스트할 컴포넌트 객체를 생성하기 위한 셋업, 평가 및 클린업 메소드를 가진다. 테스트 대상인 COM 컴포넌트는 C++ 언어뿐만 아니라 Visual Basic과 같은 스크립트 언어에서도 호출될 수 있도록 IDispatch 인터페이스를 제공한다. C++ 에서는 직접 컴포넌트 객체를 생성하고 호출하는 반면 스크립트 언어에서는 IDispatch 인터페이스 객체를 얻어 IDispatch 인터페이스가 제공하는 GetIDsOfName 메소드와 Invoke 메소드를 이용하여 간접적으로 컴포넌트 메소드를 호출할 수 있다. 어플리케이션 테스트 케이스 객체는 특정 COM 컴포넌트만을 테스팅하는 것이 아니고 여러 종류의 COM 컴포넌트를 테스팅할 수 있어야 하기 때문에 이 IDispatch 인터페이스를 이용하여 컴포넌트 객체를 생성하고 인터페이스 메소드를 호출한다. 테스트 케이스 객체는 컴포넌트 객체를 생성하고 인터페이스 메소드를 호출하는데 필요한 정보인 CLSID, InterfaceID, 메소드 명, DISPID 등을 포함하고 있으며 CLSID를 이용해 CoCreateInstance 메소드를 통해 컴포넌트 인스턴스를 생성하고 메소드를 호출하기 위해 IDispatch 인터페이스를 이용한다. 테스트 드라이버는 새로운 컴포넌트가 등록될 때마다 새로 컴파일 될 필요가 없기 때문에 Invoke 메소드를 이용하여 컴포넌트 메소드를 실행 시에 호출할 수 있다. [그림6] 에서는 어플리케이션 테스트 케이스 객체가 CoCreateInstance API로 컴포넌트 객체를 생성하고 Invoke 메소드를 이용해 컴포넌트 메소드가 호출하는 과정을 보여준다.

4. 실행 과정

본 논문에서는 괄호를 포함하는 수식을 계산하기 위한 공학용 계산기 컴포넌트를 테스트 대상으로 하여 실험하였다. 계산기 컴포넌트는 입력된 수식을 파싱하고 우선순위에 맞게 수식을 계산하여 계산 결과 값을 반환하며 실험에서 계산기 UI는 배제하였

다.

```
hr = ::CoCreateInstance(clsid, NULL, CLSCTX_ALL, IID_IUnknown,
            (void **)&pUnknwn);
hr = pUnknwn->QueryInterface(IID_IDispatch, (void **)&pDisp);

pUnknwn->Release();

szMember = bstrMethodName;

hr = pDisp->GetIDsOfNames(IID_NULL, &szMember, 1,
            GetUserDefaultLCID(), &dispid);

varg.vt = VT_BSTR;
varg.bstrVal = (BSTR)Params[0];

params.cArgs = 1;
params.cNamedArgs = 0;
params.rgdispidNamedArgs = &dispIDParam;
params.rgvarg = &varg;

hr = pDisp->Invoke(dispid, IID_NULL, GetUserDefaultLCID(),
            DISPATCH_METHOD, &params, NULL, NULL, NULL);
```

그림6 Invoke를 이용한 컴포넌트 메소드 호출 과정

(1) 컴포넌트 구매자가 웹브라우저를 통해 웹 서버에 접속한다. 웹 서버는 클라이언트에 컴포넌트 목록을 송신하기 위해 웹 서비스로부터 컴포넌트 리스트를 요청한다. 웹 서버의 요청에 따라 웹 서비스는 컴포넌트 정보 레지스트리로부터 등록된 컴포넌트들의 목록을 작성한다. 작성된 목록은 웹 서버에 전달된다. 컴포넌트 목록을 전달받은 웹 서버는 구매자에 컴포넌트 목록을 송신한다.

(2) 구매자는 명세서에 명시된 인터페이스 정보와 메소드 정보를 참조하여 컴포넌트를 테스팅하기 위한 인터페이스와 메소드 및 개발자가 제공한 테스트 케이스를 선택하고 웹 서버에 전송한다.

(3) 테스팅 요청을 받은 웹 서버는 구매자가 입력한 내용을 웹 서비스에 전달한다. 웹 서비스로부터 테스트 정보를 전달받은 테스트 드라이버는 각 테스트 케이스에 대해 테스트를 수행하고 결과 값을 다시 XML 형식으로 변환하여 웹 서비스로 전달한다. 웹 서비스는 테스트 드라이버로부터 전달받은 결과 값을 다시 웹 서버에 전달하고 웹 서버는 결과 값을 구매자에 전송한다.

5. 결론 및 향후 과제

컴포넌트 기반 개발을 통한 소프트웨어 생산성 및 품질 향상은 소프트웨어 개발에 대한 새로운 패러다임의 도입과 컴포넌트 판매 시장을 형성하는 결과를 유발하였다. 이러한 마케팅 환경에서 컴포넌트 판매자는 컴포넌트의 신뢰성 보증 및 판매 촉진을 위해 구매자가 원하는 컴포넌트를 정확하게 테스팅하고 소프트웨어에 적용시킬 수 있도록 할 수 있는 환경을 제공해야 한다.

본 논문에서는 웹 서비스 기술을 이용하여 컴포넌트 구매자에게 후보 COM 컴포넌트에 대한 이해 촉진과 편리하게 테스팅할 수 있는 테스팅 환경의 구현에 대해 기술 하였다. 향후 과제로서 이기종 컴포넌트(EJB, CORBA)들 간의 인터액션을 테스트할 수 있는 방법과 이를 웹서비스에서 효과적으로 전달할 수 있는 방법에 대해 연구한다.

참고 문헌

1. http://www.componentsource.com
2. http://msdn.microsoft.com
3. http://www.w3.org/2002/ws
4. http://www.w3.org/TR/2003/REC-soap12-part0-20030624/
5. Adam Freeman, Allen Jones. .NET XML Web Services - Microsoft Press 2003.
6. Julian Templeman. John Paul Mueller. COM Programming with Microsoft.NET - Microsoft Press 2003.
7. Ted Faison. Component Based Development with Visual C# - M&T Books.
8. Jerry Gao,Testing component based software, Technical Report, San Jose State university.
9. John D. McGregor and Davis A. Sykes, A Practical guide to testing object-oriented software, Addison -Wesley 2001.
10. Robert V.Binder. Testing Object-Oriented Systems - Addison Wesley.