상호운용 성숙도 모델을 이용한 상호운용성 측정 사례 연구

문소영○ 류동국* 서지연 김영철
홍익대학교 컴퓨터정보통신
국방과학연구소*
whit2@selab.hongik.ac.kr○

A case study about interoperability measurement using interoperability maturity model

SoYoung Moon○ DongKuk Ryu* JiYoun Seo R.YoungChul Kim
Dept. of Computer and Information Communication, Honglk University
Agency for Defense Development*

요 약

본 논문에서는 컴포넌트 기반 시스템 개발시 시스템간의 상호운용성을 높일 수 있는 방도를 제시하는데 있다. 다시 말하자면 다른 웹 서비스 방식과 분산 객체 방식의 컴포넌트 기반 시스템들의 상호운용 성숙 정도를 측정하려는 시도에 있다. 적용 사례에서는 클라이언트는 .NET 환경을 서버는 EJB 환경 상에서 개발된 EJB와 .NET 컴포넌트는 웹 서비스 방식과 IIOP.NET를 이용한 분산객체 방식 두 가지 방식으로 상호운용 하였다. 본 논문에서는 이 두 가지 상이한 상호운용 방식에 대하여 성숙도 모델인 LISI를 기반으로 상호운용 측정 기법을 적용하여 상호운용 능력을 측정하려 노력하였다. LISI 상호운용 능력을 측정한 결과 웹 서비스에 의한 방식이 분산 객체 방식보다 높은 상호운용 능력이 있음이 나타난다.

1. 서 론

규모가 크고 복잡한 소프트웨어는 일반적으로 다양한 소프트웨어 모듈들로 구성된다. 기존 컴포넌트(legacy component) 또는 서로 다른 언어들로 작성된 이종 컴포넌트들을 재상용하고자 할 때 다른 언어로 개발된 소프트웨어 컴포넌트들 사이에 상호 협력의 문제가 발생한다[1]. 이러한 이유로 개발자들은 상호운용성에 대해서 많은 관심을 가지게 되었다.

일반적인 상호운용의 형태는 시스템간의 단순한 정보 공유 및 교환이었다. 정보 기술 환경이 발전하고 복잡해짐에 따라, 상호운용의 형태도 분산 환경에서의 시스템간에 일부 기능을 공유하는 형태로 발전하고 있다.

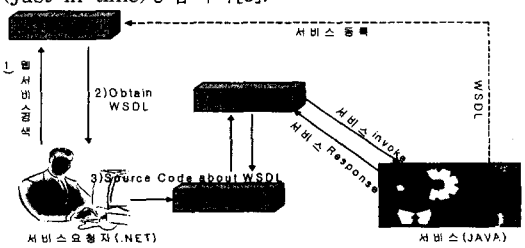
이종 응용 체계(heterogeneous application)의 상호운용성(interoperability)은 서로 다른 프로그래밍 언어와 플랫폼에서 운영되는 체계들이 다른 네트워크 상에서 통신 및 상호작용하는 능력을 말한다[2].

미들웨어(EJB, COM/DCOM)는 컴포넌트 인터페이스를 통해 컴포넌트의 기능을 접근하기위한 상호운용성 문제의 해결책을 제공한다 [2]. 최근 많은 정보 시스템이 웹 환경에서 EJB와 .NET과 같은 컴포넌트 모델을 사용하여 개발되는 추세이다. 또한 개발되는 시스템은 데이터나 일부 프로그램을 공유하여 개발되어 상호 유기적인 운용을 하는 경우도 있다. 본 논문의 2장에서는 관련연구로 웹 서비스를 이용한 상호운용과 분산 객체 수준에서의 IIPOP.NET을 통한 상호운용에 대해서 살펴보고, 3장에서는 웹 서비스와 분산객체를 이용한 적용 사례를 보일 것이며, 4장에서는 LISI 기반 상호운용 성숙도 모델을 이용하여 웹 서비스와 분산객체의 상호운용의 성숙도를 측정해 볼 것이다. 5장에서는 어떤 접근 방법이 상호운용에 적합한지를 논할 것이다.

2. 관련 연구

2-1. 웹 서비스를 이용한 상호운용성

웹 기술들이 발전하면서, 웹 서비스(web service)라는 개념들이 나타나게 되었다. 웹 서비스란 웹 기술과 XML을 이용한 분산 컴퓨팅(distributed computing)이라고 할 수 있다. 서비스는 애플리케이션이 될 수도 있고, 프로세스일 수도 있고, 하나의 컴포넌트일 수도 있다. 그러나 서비스를 이용하는 입장에서는 어떤 기술로 어떻게 만들어져 있는가는 중요하지 않다[3]. CORBA, RMI, DCOM과 같은 기존 기술들 대신에 웹 서비스를 이용하는 이유는 웹이 접근하기 쉽고, 어디에서나 사용할 수 있기 때문이다. 또한 웹 서비스의 목적은 이기종 플랫폼의 연동이다[4]. 웹 서비스의 장점은 플랫폼 및 언어의 독립성, JIT(Just-in-time)통합이다[5].



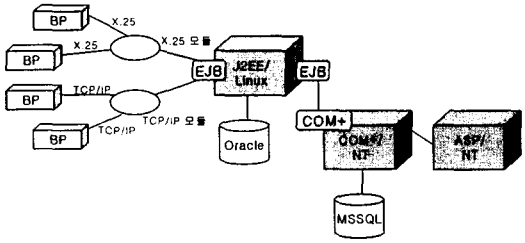
<그림 1> 웹 서비스를 통한 상호운용 절차

<그림 1>에서와 같이 우선은 서비스를 제공하는 EJB

는 UDDI에 서비스를 등록한다. 서비스 요청자 .NET은 UDDI에서 서비스를 검색한다. 제공받을 서비스를 검색한 후 WSDL을 얻어와 이에 맞는 언어로 소스코드를 작성한다. XML과 HTTP를 이용해서 어느 플랫폼이든지 해당 서버와 서비스에 접근할 수 있도록 해주는 SOAP(Simple Object Access Protocol)을 통하여 서비스를 요청하고, 응답을 받아올 수 있다.

2-2. 분산객체를 이용한 상호운용성

JAVA/EJB와 .NET/COM+ 같이 재사용 가능한 컴포넌트로 이뤄진 시스템들을 통합하려 한다. 또한 다른 플랫폼에서 개발한 컴포넌트를 사용하려할 때 각 시스템을 재구축하지 않고 재사용한다는 것은 시간과 노력 및 비용을 절약하게 된다. 이렇게 하는 방법이 바로 상호운용인데, 이전의 시스템에 RMI(Remote Method Invocation)와 CORBA(Commom Object Request Broker Architecture) 연동 부분이 있을 경우 IIOP(Internet Inter-ORB Protocol)에 대한 지원이 필요하다.



<그림 2> EJB 시스템과 COM+ 시스템 연결

EJB 시스템에서 .NET로 구축된 시스템의 COM을 직접 호출하고, 거꾸로 .NET에서 EJB를 호출할 필요가 있을 수 있다. <그림 2>는 EJB 시스템과 COM+ 시스템의 상호운용을 위한 상황을 도식화하여 보여주고 있다. 상호운용을 도와주기 위해서 <그림 4>의 IIOP.NET을 이용한다.

3. 적용 사례

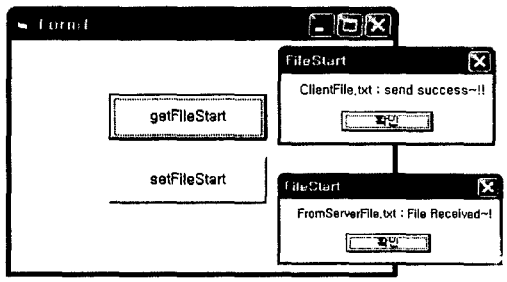
EJB와 .NET 간의 상호운용성을 웹 서비스와 분산객체를 이용하여 구현해보았다.
제한조건으로는 서버 환경에 EJB를 채택하였고, 클라이언트 환경은 .NET하에서 시도해본다.

3-1. 웹 서비스를 이용한 구현

웹 서비스를 이용해서 서버로부터 파일을 얻어오고, 클라이언트가 서버에게 파일을 전송해주는 시나리오로 구현하였다.

```
// 서버 EJB 소스의 일부분
public void setFile(DataHandler dh){
        .....
  try {
    fos = new FileOutputStream("FromClientFile.txt");
    dh.writeTo(fos);
    fos.flush();
  } catch .....
}
```

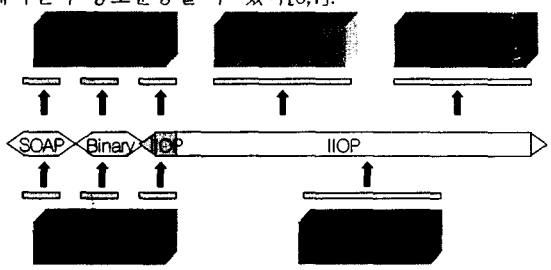
```
'클라이언트 .NET 소스의 일부분
WScript.Echo ("Create Client Succeeded
============" )
 'wsdl과 wsml을 이용하여 SoapClient 객체를 초기화한다.
call client.mssoapinit(wsdl, "", "", wsml)
        .....
WScript.Echo ("Init Client Succeeded ----------" )
 Set ReceivedAttach = Client.getFile()
 FileSaveName = "FromServerFile.txt"
 ReceivedAttach.SaveToFile FileSaveName, True
```



<그림 3> 웹 서비스를 이용한 파일전송 실행 화면

3-2. IIOP.NET를 이용한 분산 객체 구현

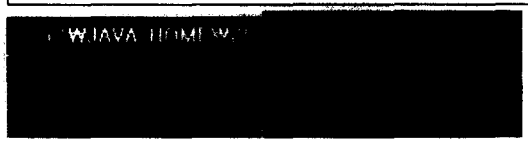
IIOP.NET은 CORBA의 핵심 부분인 IIOP 기반의 닷넷 리모팅 채널과 IDL(Interface Definition Language)과 CLS(Common Language Specification)간 변환 툴을 제공한다. 자바에는 이미 RMI/IIOP가 제공되고 있으므로 IIOP.NET을 이용하면 자바의 RMI/IIOP 기반의 애플리케이션과 상호운용할 수 있다[6,7].



<그림 4> IIOP.NET 개요

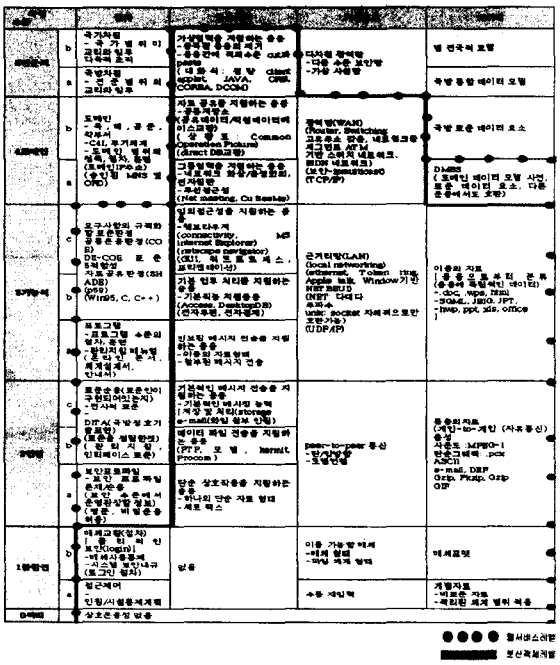
```
//서버 EJB 소스의 일부분
        .....
AdderImpl adder = new AdderImpl();
// publish the reference with the naming service:
Context initialNamingContext = new InitialContext();
initialNamingContext.rebind("adder", adder);
System.out.println("Server Ready...");
        .....
```

```
// 클라이언트 .NET C# 소스의 일부분
IiopClientChannel channel = new IiopClientChannel();
ChannelServices.RegisterChannel(channel);
                    .....
CorbaInit init = CorbaInit.GetInit();
NamingContext nameService =
        init.GetNameService(nameServiceHost, nameServicePort);
double result = adder.add(sum1, sum2);
Console.WriteLine("result: " + result);
```



<그림 5> 서버와 클라이언트 실행

4. 성숙도 모델을 이용한 상호운용성 측정



<그림 6> LISI의 상호운용성 레벨

사실 LISI[10,11]는 미국 DoD 정보시스템간의 상호운용성을 측정하기위한 상호운용 성숙도 모델이라서 상용 컴포넌트 시스템간의 성숙도를 측정하려는 시도가 적절하지 않을 수있다. 그러나 본 논문의 의도는 컴포넌트 기반 시스템간의 상호운용성은 현재 중요한 이슈가 되고 있어 우리 Lab에서는 시도로써 LISI 그 자체로 측정하였다. <그림 6>와 같이 상호운용 성숙도 모델을 이용하여 두 가지 상호운용 기법에 대하여 상호운용 능력을 측정하였다. <그림 6>은 LISI 모델을 기반으로 상호운용 능력을 성숙도에 기반하여 수준을 정의한 것이다[8,9]. <그림 6>을 보면, 웹서비스의 상호운용성을 측정한 결과 절차는 3c, 응용체계는 5b, 기반구조는 4b, 데이터는 4a이다. 결과적으로 IMM을 이용한 웹서비스의 상호운용성은 레벨은 3c임을 알 수 있다. 그리고 분산객체를 이용한 상호운용성을 측정한 결과 절차는 1b, 응용체계는 5b, 기반구조는 4b, 데이터는 3c 로써 IMM을 이용해 살펴본 전체적인 레벨은 1b에 머무름을 알 수 있다.

5. 결론 및 향후 과제

4장에서 결과를 볼때 상호운용 능력을 측정한 결과 웹서비스에 의한 방식이 분산 객체 방식보다 보다 높은 상호운용 능력이 있음을 알 수 있다. 분산 객체 방식의 경우 상호운용 절차부분에서 웹 서비스에 비교하여 낮은 평가를 보였다. 웹 서비스는 표준화된 절차와 규격을 통하여 높은 상호운용 능력을 보여준다. 그리고 시스템 확장과 상호운용성을 생각하면 항상 느슨한 연결(loosely coupling)과 표준(standardization) 기반으로 시스템을 구성하는 것이 좋다. 성숙도 모델을 이용하여 상호운용성을 측정한 결과 비교적 비슷했지만, 표준기술을 사용한다는 의미에서 웹 서비스를 이용하는 것이 상호운용성에 도움이 된다는 것을 알 수 있다. 그러나 웹 서비스는 트랜잭션 처리 관련 규약이 미흡하다. 또한, 성능이나 튜닝에 대한 더 구체적인 방법들이 벤더들의 책임만 으로 되어 있고, 표준화에 대해서는 미흡한 편이다. 웹 서비스는 XML 기술처럼 계속 발전되고 있는 기술이다. 웹 서비스 기술은 만능이 아니며 비동기라는 측면 대문에 단순하지만 오히려 복잡해지기도 한다[3]. 이러한 문제로 향후엔 Memo 스타일의 어댑터를 개발하여 비동기적인 측면을 해소하고자 하고, 상호운용 성숙 모델을 상용 컴포넌트 기반 시스템의 상호운영성 측정을 할 수 있도록 보완해야 할것이다,

참고 문헌

[1] Barrett, Kaplan, Wiledn, "Automated Support for Seamless Interoperability in Polylingual Software Systems", SIGSOFT 1996

[2] C.C. Chiang, "The use of adapters to support interoperability of components for reusability", 2002

[3] 이승준, 마이크로소프트웨어, "웹 서비스와 상호운용성", 2003. 12

[4] 이영석, 마이크로소프트웨어, "현실로 다가온 웹 서비스, MS와 자바 플랫폼의 연동", 2003. 10

[5] 최종명, 유재우, 최재영 공저, "자바 개발자를 위한 XML", , 2002.5

[6] 이승준, 정균옥, 김경윤, 마이크로소프트웨어, "J2EE.NET 분산 객체 상호운용성", 2003. 11

[7] http://iiop-net.sourceforge.net/

[8] "Level of Information System Interoperability (LISI)", C4ISR Archtecture Working Group, 1998.

[9] "국방정보체계 상호운용성 수준(LISI) 업무편람", 국방부, 2002.

[10] 국방과학연구소, "컴포넌트 기반 체계 상호운용 적합성 평가 및 인증 기술 연구", 2004

[11] 국방과학연구소, "국방정보체계 상호운용 수준(LISI)", 2000.12