이종 분산 컴포넌트 기반 시스템간의 상호운용 수준 측정 기법

류 동 국[†] · 김 영 철[††]

요 약

시스템간의 상호운용성은 국방, 금융, 행정 등 임무 중심적인 시스템의 개발 및 운용에 있어서 많은 관심과 연구가 되고 있는 분야이다. 특히 시스템들이 이종의 컴포넌트들을 기반으로 개발됨에 따라 이종 분산 환경에서 컴포넌트 기반 시스템간의 상호운용성 확보는 매우 중요한 문제로 부각되고 있다. 본 논문은 이러한 문제점을 해결하기 위하여 이종 분산 컴포넌트 기반 시스템간의 상호운용 수준 측정에 필요한 표준 컴포넌트 규격 정의, LISI 능력 모델 확장, 그리고 CBD 방법론의 개발 절차를 반영한 상호운용 질의서를 포함하고 있다. 또한 서로 다른 방식으로 개발된 두 프로토타입 시스템에 대하여 상호운용성 수준을 측정하고 결과를 비교 분석하여 본 본문의 상호운용 능력 모델 및 질의서를 검증하였다.

An Interoperability Level Measurement Mechanism between Heterogeneous Distributed Component Based Systems

Dong-Kuk Ryu[†] · R. Young-Chul Kim[††]

ABSTRACT

In this paper we describe interoperability maturity measurement, due to highlight a momentous issue on interoperability between heterogeneous distributed component based systems. it also suggests one of various ways for supporting interoperability between each other different component based systems, that is, using web services based mechanism and also revises "LISI (Level of information systems interoperability)" model to measure interoperability maturity. We also develop the interoperability questionnaire based on the interoperability factors considered at each phase of component based development process, which is possible to measure interoperability maturity between heterogeneous distributed component systems. Based on revised interoperability maturity model and questionnaire we measure each interoperability maturity on examples of two different implementations, that is, web services and distributed object based mechanism. As a result, we suggest an enhanced interoperability way based on analysing problems of these measured results.

키워드 : 상호운용성(Interoperability), 측정(Measurement), 컴포넌트(Component), 질의서(Questionnaire)

1. 서 론

인터넷과 같은 컴퓨터 통신망과 정보 시스템 개발 기술의 발달로 인하여 근래에 개발되는 정보 처리 시스템들은 타 정보 시스템과의 유기적인 상호운용이 매우 활발하다. 정보 시스템의 상호운용은 초창기에는 단순한 정보의 공유 및 교환이 일반적인 형태였다. 그러나 점차 많은 정보 시스템이 개발되면서 정보 시스템간의 유기적인 형태의 정보 교환 요구가 증대되고 인터넷과 같은 정보 통신 기반 환경이 구축됨에 따라 상호운용의 형태는 점차 복잡해지고 분산 환경에서 시스템의 일부 기능을 공유하는 형태로 발전하고 있다[1, 3].

정보 시스템의 상호운용성이 가장 중요시되는 분야는 임무 중심적인 기능을 수행하는 국방, 금융, 행정 등이다. 예를 들어 미군은 걸프전 및 이라크전을 수행하면서 컴퓨터 게임을 하듯이 전쟁을 수행하였다. 전투 지휘관은 컴퓨터 화면을 보면서 전 전장의 상황을 파악하고 컴퓨터 통신을 통하여 명령을 전달하였다. 이러한 전투 환경이 가능하게 된 것은 미군이 사용하는 여러 정보 시스템들이 유기적인 상호운용을 통하여 효율적인 정보 공유 및 교환이 가능하기 때문이다. 만약 정확하고 효율적인 상호운용이 가능하지 않고 상호운용 과정에 오류가 발생하여 잘못된 정보가 전달되게 된다면 무기 시스템의 오작동 등 심각한 문제점이 발생할 수 있다. 따라서 임무 중심적이고 여러 복합체계가 유기적으로 운용되는 환경에서는 정보 시스템의 상호운용성은 아주 중요한 고려

※ 본 논문은 2004년도 홍익대학교 학술연구조성비 및 국방과학연구소 위탁과제로 연구되었음.
† 정 회 원 : 국방과학연구소 선임연구원
†† 정 회 원 : 홍익대학교 컴퓨터정보통신공학과 교수
논문접수 : 2004년 6월 30일, 심사완료 : 2005년 1월 3일

요소가 된다[2].

그러나 정보 시스템의 개발 기술 및 운용 환경은 복잡하고 다양하여 상호운용은 점점 어려워지고 있다. 이러한 상호운용의 문제점을 인식하고 이를 해결하기 위해 많은 연구비를 지원하여 연구하고 있는 곳은 미국 국방성이다. 미군은 DoD AF(Architecture Framework)와 LISI(Level of Information System Interoperability)를 개발하여 정보 구조적 측면에서 상호운용성을 측정하는 평가 모델을 개발하여 사용하고 있다[2, 3]. LISI는 CMM(Capability Maturity Model)을 개발한 Carnegie Mellon 대학의 SEI 연구소에서 개발하였으며, CMM의 성숙도 측정 모델을 근간으로 정보시스템의 상호운용성 수준을 측정하는 평가 모델이다[5].

최근 들어 많은 정보 시스템들이 컴포넌트를 기반으로 개발되고 있다. 컴포넌트 기반 개발 방법은 표준화된 컴포넌트를 사용하므로 상호운용을 향상시키는데 많은 도움을 주고 있다. 그러나 이종 분산 컴포넌트 기반의 시스템 환경에서 컴포넌트간의 상호운용성 확보는 지금도 해결해야할 부분이 많이 남아있다[1, 6].

본 논문에서는 이종 분산 컴포넌트 기반 시스템에서의 상호운용성 수준을 측정하기 위하여 표준화된 컴포넌트 규격을 정의하고 LISI 능력모델을 컴포넌트 기반 시스템에 맞게 확장하였다. 그리고 상호운용성 수준 측정에 필요한 질의서를 CBD(Component Based Development) 방법론의 개발 절차를 반영하여 작성하였다. 미군의 LISI 질의서는 미군 정보 시스템의 정보 유출을 방지하기 위하여 미국내 관계자 이외에는 외부에 공개하지 않고 있다. 따라서 국내에서 개발되는 이종 분산 컴포넌트 기반 시스템의 상호운용성을 측정하기 위해서는 질의서의 개발이 필수적이다. 이를 이용하여 본 논문에서는 분산 객체 방식과 웹 서비스 방식의 컴포넌트 기반 시스템 프로토타입을 개발하여 상호운용 수준을 측정하고 결과를 비교 분석하여 질의서 및 개선된 능력모델을 검증하였다.

본 논문의 구성은 다음과 같다. 제2장에서는 상호운용 수준 측정에 관련된 LISI 모델의 소개와 이종 분산 환경에서의 컴포넌트간 상호운용의 문제점, 그리고 웹 서비스를 활용한 상호운용성 향상 방안을 기술하였다. 제3장에서는 이종 분산 컴포넌트 기반 시스템의 상호운용성 수준 측정 기법을 설명하였다. 제4장에서는 컴포넌트 기반 시스템 프로토타입에 대하여 상호운용성 수준을 측정하고 결과를 비교분석하였다. 마지막으로 제5장에서 결론을 기술하였다.

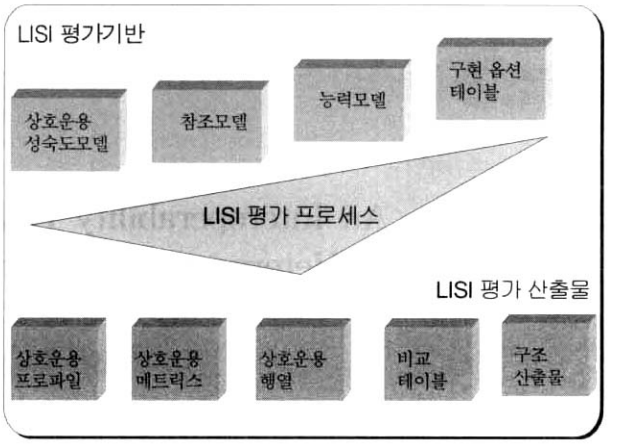
2. 관련연구

이 장에서는 LISI 모델을 소개하고 이종 컴포넌트 기반 시스템의 상호운용 문제점과 지금까지 연구된 어댑터 방식과 웹 서비스 방식의 상호운용성 해결방법을 기술한다.

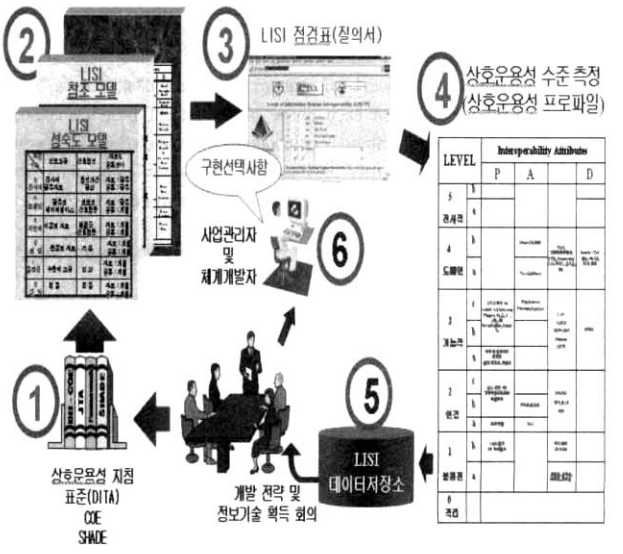
2.1 LISI

2.1.1 LISI 개요

LISI는 미군이 국방 정보 시스템의 상호운용성을 향상하기 위하여 사용하는 평가 모델이다. 1998년 최초로 개발되어 미군 국방 정보 시스템의 상호운용 수준 평가에 사용되고 있다. LISI는 상호운용 능력을 5단계의 수준으로 구분하고 있으며 상호운용성 평가에 필요한 평가 프레임워크와 프로세스를 포함하고 있다[3].



(그림 1) LISI 구성요소



(그림 2) 상호운용성 측정 프로세스

LISI는 (그림 1)과 같이 상호운용성 평가의 기반환경을 제공하는 LISI 평가 기반과 이를 활용하여 평가하는 평가 프로세스, 그리고 평가 결과에 해당하는 LISI 평가 산출물로 구성된다. LISI의 상호운용 평가 프로세스는 먼저 구조화된 상호운용 질의서를 이용하여 평가 대상 시스템의 정보를 수집한다. 질의서를 통하여 수집된 정보는 LISI 평가 기반에서 정의된 상호운용 성숙도 모델, 참조 모델, 능력 모델, 구현 옵션 테이블을 이용하여 LISI 평가 프로세스를

통하여 상호운용 프로파일, 상호운용 메트릭스, 상호운용 행열, 비교 테이블, 구조 산출물 등의 LISI 평가 산출물을 생성한다[3].

2.1.2 LISI 측정 프로세스

LISI의 상호운용성 수준 측정 프로세스는 (그림 2)와 같다. 1단계와 2단계는 평가 기반을 정립하는 단계이고 일반적인 측정은 3단계에서 6단계까지이다. 세부적인 프로세스는 다음과 같다.

① 상호운용에 대한 정의, 표준, 지침 제정하여 상호운용 수준 측정 환경을 정립한다.
② LISI 성숙도 모델, 참조 모델, 구현 옵션을 개발하여 상호운용성 수준 측정을 가능하게 한다.
③ 상호운용 질의서를 통하여 개발 시스템에 대한 구현 옵션을 입력한다.
④ 입력된 질의서를 바탕으로 개발 시스템의 정보를 종합한 상호운용 프로파일을 작성한다.
⑤ 체계의 정보를 수집한 상호운용 프로파일을 바탕으로 상호운용성을 평가한다.
⑥ 더욱 향상된 상호운용 수준의 확보를 위하여 시스템의 구현 옵션을 조정하고 시스템 개발에 반영한다.

2.2 이종 분산 컴포넌트의 상호운용 기술

이 절에서는 먼저 컴포넌트 개발 모델을 소개하고 이종 분산 컴포넌트에서 발생하는 상호운용 문제점을 기술한다. 그리고 지금까지 연구되어온 이종 컴포넌트의 상호운용 기법인 어댑터와 웹서비스 방식을 기술하고 이종 분산 컴포넌트에 대한 상호운용 수준 측정의 필요성에 대하여 설명한다.

2.2.1 컴포넌트 개발 모델

현재 일반적으로 사용되는 대표적인 컴포넌트 모델은 .Net과 J2EE이다. .Net은 마이크로소프트에서 개발한 컴포넌트 모델로 개발언어의 제약은 없으나 윈도우즈 계열의 운영체계에서만 동작한다. 반면 J2EE는 썬 마이크로 시스템즈에서 개발한 컴포넌트 모델로서 운영체계의 제약은 없으나 개발언어는 자바에 국한된다. 이 두 컴포넌트 개발 모델은 컴포넌트 개발 시장을 양분하면서 광범위하게 사용되고 있다. 그러나 개발사의 이해관계에 따라 두 컴포넌트 모델의 상호운용을 위한 표준화 작업은 현재 이루어지고 있지 않아 이종 컴포넌트간의 상호운용이 어려운 현실이다[8, 9]

<표 1>은 .Net과 J2EE의 요소 기술을 비교한 표이고, <표 2>는 계층별로 요소 기술을 분류한 것이다. <표 2>와 같이 3계층 구조로 컴포넌트 기반 시스템을 개발함에 있어서 각각의 계층별로 .Net이나 J2EE를 사용하는 방법은 가능하나 서로 다른 모델을 연결하여 주는 추가적인 미들웨어나 어댑터가 필요하여 시스템의 성능을 저하시키는 문제점이 발생한다.

〈표 1〉 .Net과 J2EE의 기술비교

기 능	.Net	J2EE
GUI	WinForms	SWING/AWT
웹	ASP/ASP.Net	JSP
웹 서버 모듈	ISAPI,HttpHandler, HttpModule	서블릿, 필터
비즈니스 로직 컴포넌트	COM+ 컴포넌트	EJB 세션빈
데이터 컴포넌트	ADO.Net	EJB 엔터티빈
네이밍	ADSI	JNDI
원격호출	.Net Remoting	RMI/RMI-IIOP
메시징	MSMQ	JMS
데이터 접근	ADO.Net	JDBC,SQL/J
트랜잭션	COM+/MTS	JTA

〈표 2〉 .Net과 J2EE의 계층별 주요 기술

계 층	.Net	J2EE
프리젠테이션	ASP.Net	JSP/서블릿
비즈니스 로직	COM+ 컴포넌트	서블릿, EJB 컴포넌트
데이터베이스	ADO.Net, SQL 서버	JDBC, 오라클

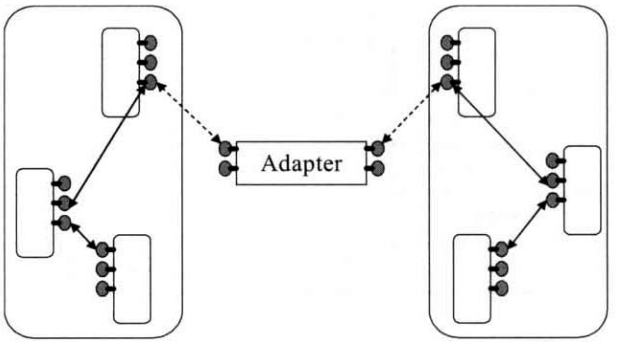
2.2.2 상호운용 어댑터 개발

현재 .Net과 J2EE와 같은 컴포넌트 모델은 서로 다른 컴포넌트 모델과의 상호운용을 지원하지 않고 있다. 따라서 분산 환경에서 이종 컴포넌트간의 상호운용은 (그림 3)과 같은 어댑터를 사용하여 일반적으로 해결한다. 어댑터는 서로 다른 프로그래밍 언어, 개발 플랫폼, 네트워크 환경의 차이점을 보완해주는 일종의 미들웨어의 역할을 수행하며 다음과 같은 기능을 제공한다[7-9].

- 이종 컴포넌트간의 개발 및 운용 환경 차이점 보완
- 컴포넌트간의 인터페이스 불일치 조정
- 컴포넌트 재사용 지원
- 기존 컴포넌트 관리 및 정보 제공

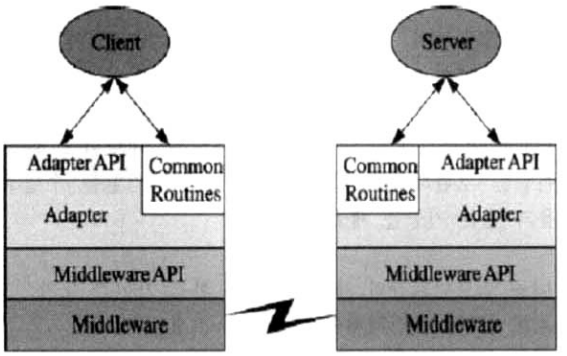
상호운용 어댑터의 개발 방식은 미들웨어를 사용하는 [8]의 방식과 내부의 세부적인 사항을 제어하는 [9]의 방식이 연구되고 있다. [8]에서 제안하는 방식은 이종 컴포넌트의 상호운용을 위하여 (그림 4)에서와 같이 CORBA와 같은 미들웨어를 사용하여 이종 컴포넌트간에 존재하는 미세한 차이점을 해결하는 방식이다. 이 방식은 미들웨어가 운용 환경의 차이점을 모두 보정하여 주므로 어댑터를 간단하게 구현할 수 있으나 미들웨어가 추가됨에 따른 부하가 가중되어 전체적으로 시스템의 성능이 저하된다. [9]에서 제안하는 방

식은 [8]의 단점을 보완하는 방법이다. 미들웨어를 사용하지 않고 어댑터가 내부적으로 불일치 부분을 보정하는 기능을 가지고 있다. 이 방식은 시스템의 효율성을 높이는 방식이기는 하나 어댑터 개발에 많은 비용이 든다.



(그림 3) 어댑터를 이용한 상호운용

어댑터를 사용하여 개발하는 방식은 컴포넌트 모델 개발사가 이종 컴포넌트간의 상호운용을 지금까지는 지원하지 않기 때문에 일종의 미들웨어로서 문제점을 해결한 것이다. 그러나 이러한 어댑터는 표준화되어 있지 않고 컴포넌트의 개발과 함께 부가적으로 복잡하고 구현 난이도가 높은 어댑터를 추가적으로 개발하는데 많은 노력이 추가된다는 또 다른 문제를 내포하고 있다. 따라서 어댑터 방식은 핵심 기능을 수행하는 컴포넌트들이 이종 분산 환경으로 구현될 경우에만 제한적으로 사용하여야 한다.



(그림 4) 어댑터를 이용한 이종 컴포넌트간의 통신

2.2.3 웹 서비스를 이용한 상호운용

웹 기술이 발전하여 웹 서비스(Web Service)의 개념으로 발전하고 있다. 웹 서비스는 어댑터를 사용하는 기존 방식의 문제점을 해결하고 이종 분산 컴포넌트의 상호운용을 효과적으로 지원한다.

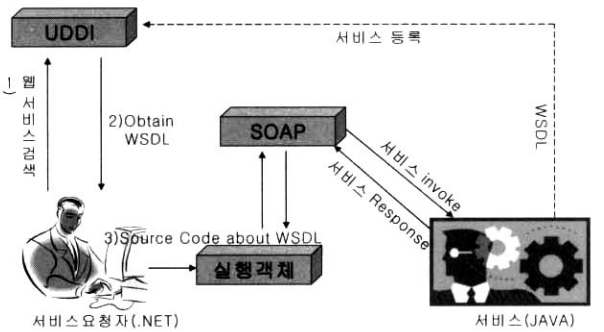
2.2.3.1 웹 서비스의 구성

웹 서비스는 웹 기술의 발전된 형태이다. 웹 서비스는 웹 기술에 XML을 이용한 분산 컴퓨팅이라 할 수 있다. 웹 서비스에서 제공하는 서비스는 어플리케이션, 프로세스 또는 컴포넌트 등이 제약 없이 될 수 있다. 그러나 서비스를 이용하는 측면에서는 서비스의 내부가 어떠한 기술로 구현되었는지는 고려대상이 아니다[10]. CORBA, RMI, DCOM과 같은 기존의 분산 컴퓨팅 기술 대신에 웹 서비스를 사용하는 이유는 웹이 이용하기 편리하고 어디에서나 손쉽게 사용할 수 있기 때문이다. 또한 웹 서비스는 이종 플랫폼간의 연동을 지원하고 운용 플랫폼 및 개발 언어에 독립적이다. 웹 서비스의 세부 요소 기술은 다음과 같다[10, 11].

- SOAP : XML에 기반을 둔 플랫폼 독립적이며 간단한 통신 프로토콜
- WSDL : 웹 서비스의 인터페이스 정보와 연결점 정보를 기술하는 언어
- UDDI : 서비스 검색을 위한 등록 디렉토리

2.2.3.2 웹 서비스의 운용

웹 서비스의 운용 절차는 (그림 5)와 같다. 우선 서비스를 제공하는 컴포넌트를 UDDI에 등록한다. 그리고 서비스를 요청하는 컴포넌트는 UDDI에서 서비스를 검색하여 WSDL로 서비스 컴포넌트의 수행에 필요한 정보를 제공받는다. 제공받은 WSDL를 수행하면 SOAP을 통하여 서비스 컴포넌트를 수행하고 결과를 받는다.



(그림 5) 웹 서비스 운용 절차

2.2.3.3 웹 서비스를 이용한 이종 컴포넌트 상호운용

웹 서비스는 플랫폼 및 개발 언어의 제약 없이 WSDL을 통하여 실행 객체의 인터페이스를 정의하고 XML 기반의 SOAP을 이용하여 통신한다. 따라서 기존의 어댑터를 사용하는 방식과는 다르게 편리하게 이종 분산 컴포넌트를 사용한 시스템을 개발할 수 있다. 기존 상호운용 어댑터의 가장 큰 단점은 표준화가 어렵다는 점이다. 복잡하고 다양한 기존 시스템들의 운용환경을 지원하는 어댑터의 개발은 현실적으로 불가능하다. 그러나 웹 서비스는 UUDI를 이용하여 서비스를 표준화 한다면 이종 컴포넌트간의 상호운용성을 보다 쉽게 확보할 수 있다.

2.2.4 이종 분산 컴포넌트 상호운용 수준 측정 필요성

현재 국방부 및 정보통신부 등 많은 정부 기관에서는 핵

심 컴포넌트를 개발하여 컴포넌트 기반 시스템의 개발을 시도하고 있다. 이러한 컴포넌트들을 효율적으로 활용하기 위해서는 분산 환경에서 이종 컴포넌트를 조합하여 새로운 시스템을 개발하여야 한다. 현재 컴포넌트에 대한 다양한 품질 측정 기법이 제시되고는 있으나 컴포넌트 자체에 대한 품질 평가에 한정되어 있고 컴포넌트간의 상호운용성 측정에 대한 연구는 미진한 상태이다[4]. 이종 분산 환경의 컴포넌트 기반 시스템 개발에 있어서 이종 컴포넌트간의 상호운용성 문제가 항상 발생하므로 체계적인 관리가 필요하다. 이종 컴포넌트간의 상호운용성을 체계적으로 관리하고 증진하기 위해서는 컴포넌트의 상호운용 수준 측정이 선행되어야 한다. 컴포넌트 또는 컴포넌트 기반 시스템의 상호운용 수준 측정을 통하여 이종 분산 컴포넌트 기반 시스템의 상호운용 문제점을 파악하고 대안을 도출하여 시스템의 상호운용성을 향상할 수 있다.

3. 이종 분산 컴포넌트 기반 시스템 상호운용 수준 측정

이 장에서는 기존 LISI 모델의 문제점을 기술하고 이종 분산 컴포넌트 기반 시스템의 상호운용 수준 측정을 위한 표준 컴포넌트 규격 정의, LISI 능력 모델 확장, 컴포넌트 상호운용 질의서에 대하여 기술한다.

3.1 기존 LISI 모델의 문제점

미 국방성이 상호운용성 수준을 측정하는 모델로 사용하고 있는 모델인 LISI는 이종 분산 컴포넌트 기반 시스템의 상호운용성을 측정하기 위해서는 다음과 같은 문제들이 해결되어야 한다.

● 일반 정보처리 시스템 평가에 부적합

LISI 모델은 1998년 미군에 의해서 개발되었다. LISI 구현옵션 및 질의서는 미군 정보 시스템의 평가에 적합한 형태로 구성되어 일반에서 사용하는 정보 처리 시스템의 상호운용성 평가에는 부적합한 면이 있다. 일반 정보 처리 시스템의 상호운용성 평가에 적합한 형태로 LISI 모델의 평가 기반 및 평가 프로세스의 개선이 필요하다.

● 최신 정보기술 및 방법론 반영 미흡

LISI 모델을 기반으로 시스템의 상호운용성을 평가하기 위해서는 먼저 평가 대상 시스템의 정보를 수집하는 질의서가 필요하다. 그러나 현재 미군 LISI 모델 질의서는 민간에 공개되어 있지 않아 활용하기 힘들고 [12]의 질의서 또한 단순한 표준의 존재여부만을 확인하는 기초적인 수준이다. 따라서 최신의 정보 기술과 개발 방법론이 반영된 질의서가 작성되어야 한다.

● 컴포넌트 기반 시스템에 대한 고려 미흡

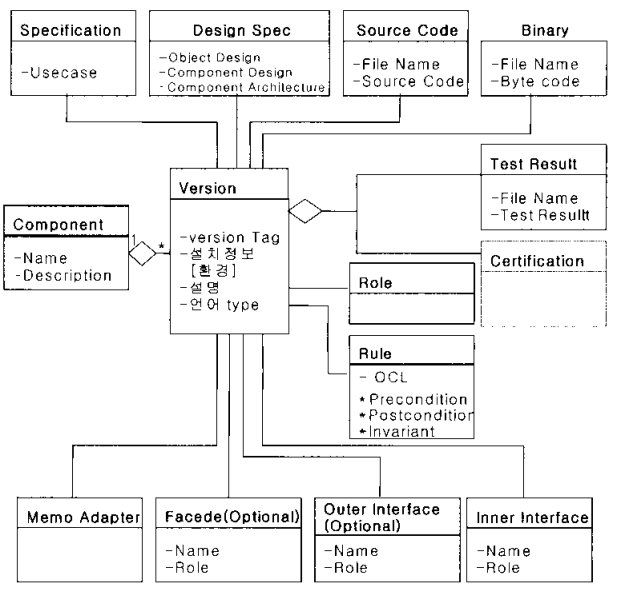
컴포넌트 기반 시스템은 서로 다른 여러 개발자들이 다른 개발환경에서 개발한 컴포넌트들을 조립하여 새로운 시스템을 개발하기 때문에 컴포넌트간에 상호운용 문제가 발생할 수 있으나 기존 LISI 모델에서는 이에 대한 고려가 없다.

기존 LISI 모델의 문제점을 해결하고 이종 분산 컴포넌트 기반 시스템의 상호운용성을 확보하기 위해서는 먼저 컴포넌트에 대한 표준화된 규격의 정의가 필요하다. 표준화된 컴포넌트를 바탕으로 LISI 모델에서 컴포넌트를 평가할 수 있도록 능력모델을 확장하여 컴포넌트의 상호운용성 수준 측정이 가능하도록 하여야 한다. 그리고 최신 기술 및 방법론을 적용한 질의서가 작성되어야 실질적인 평가가 가능하게 된다.

3.2 이종 분산 컴포넌트 기반 시스템의 상호운용 수준 측정 기법

3.2.1 표준 컴포넌트 규격 정의 및 분류

이종 분산 컴포넌트의 상호운용성을 측정하기 위해서는 먼저 컴포넌트에 대한 표준화된 규격이 필요하다. 여러 CBD 방법론에서 컴포넌트에 대하여 다양한 형태로 규격의 표준화가 진행되고 있으며 지금까지 국내표준으로 제시된 것은 없다고 볼 수 있다[15]. 본 논문에서는 아직까지 모호한 컴포넌트를 정확하게 식별하여 컴포넌트에 대한 구체적인 정보를 바탕으로 컴포넌트간의 상호운용을 할 수 있도록 UML을 사용하여 컴포넌트의 규격을 (그림 6)과 같이 정의하였다[15, 16].



(그림 6) UML을 이용한 컴포넌트 규격

또한 정의된 컴포넌트를 바탕으로 컴포넌트를 분류할 수 있도록 <표 3>와 같이 개발된 컴포넌트를 분류할 수 있도록 목록화하였다. <표 3>는 국방컴포넌트 분류체계를 기본으로 하여 상호운용 관점에서 필요한 요소(인터페이스, 어댑터)등을 추가한 것이다[17].

〈표 3〉 컴포넌트 분류 표

컴포넌트 분류 코드(ID) : Ge-Gr-Pl-Ab (XXXX-XX-XXXX-XXX)	
컴포넌트 이름	
컴포넌트 도메인 스코프 정보	-환경정보(설치실행지침서)
	-configuration
	-language type
	-컴포넌트 설명 * contact 정보 * developer 정보
컴포넌트 인터페이스 정보	Inner Interface
	Outer Interface
	Facede (High Level Interface)
Role	
Rule (Constraint Language)	
Adapter (Memo Style Adapter)	
요구 문서 (Requirement Spec.) 분석 문서 (Analysis Document) 설계 문서 (Design Document) 구현 문서 (Implementation Document) 테스트 문서 (Test Document)	

3.2.2 LISI 능력 모델 확장

현재의 LISI 모델에서는 컴포넌트 기반 시스템에 대한 고려가 부족하여 LISI 능력모델의 확장이 필요하다. 기존 LISI 능력모델은 상호운용 수준을 0단계에서 4단계까지 정의하고 각 단계에 대하여 P(절차), A(응용 체계), I(기반구조), D(데이터)로 세부적으로 분류하여 수준을 정의한다[3]. 좁은 의미에서의 컴포넌트는 일반적으로 A(응용)에 속한다. 그러나 컴포넌트는 작게는 시스템 개발에서 비즈니스 로직을 구현한 컴포넌트(A, 응용 체계)에서부터 시스템의 운영체계나 통신 프로토콜(I, 기반 구조), 그리고 데이터베이스(D, 데이터)를 모두 포함하는 개념이다[13]. 즉, 컴포넌트는 A, I, D의 속성 모두에 포함된다. 따라서 컴포넌트 기반 시스템의 상호운용 능력을 측정하고 평가하기 위해서는 컴포넌트에 대한 정의와 LISI 능력 모델에 컴포넌트와 관련된 요소에 대한 평가를 위한 측정항목의 추가가 필요하다[14]. 본 논문에서는 컴포넌트의 상호운용 수용 능력 모델을 <표 4>와 같이 정의하였다. 컴포넌트의 상호운용 성숙도를 의존, 독립, 상호운용, 도메인, 통합 컴포넌트로 5단계의 수준으로 분류 하였다. 각 단계는 세부적인(a,b) 단계로 다시 분류된다. 컴포넌트에 대한 수준별 정의는 다음과 같다.

- 의존 컴포넌트(0) : 가장 상호운용 수준이 낮은 단계이다. 의존 컴포넌트는 개발 언어, 운용환경 등에 의존적인 컴포넌트로 특정 개발언어, 운영체계, 통신 인프라에서만 동작한다.
- 독립 컴포넌트(1) : 독립 컴포넌트는 EJB나 .Net과 같이 특정 컴포넌트 모델이 정의되어 있어 모델에서 정의하는 규격을 만족하는 컴포넌트간에 상호운용이 가능한 컴포넌트이다. 그러나 서로 다른 모델에서 개발된 컴포넌트간에는 상호운용이 불가능하다.
- 상호운용 컴포넌트(2) : 상호운용 컴포넌트 단계에서는 컴포넌트 모델에 제약이 없이 모든 컴포넌트가 컴포넌트 기술적으로는 상호운용이 가능한 단계이다. 그러나 각 컴포넌트간의 상호운용에 대한 세부적인 규약이나 절차는 정의되어 있지 않는 단계이다.
- 도메인 컴포넌트(3) : 도메인 컴포넌트는 컴포넌트를 공유하는 도메인에서 컴포넌트 표준을 개발하여 제공하므로 도메인 내부에서는 상호운용이 가능한 단계이다.
- 통합 컴포넌트(4) : 통합 컴포넌트는 컴포넌트의 표준이 정의되고 모든 컴포넌트 사용자에게 표준으로 개발되어 각각의 도메인에서 공통으로 사용가능한 단계이다.

〈표 4〉 컴포넌트 상호운용 능력 모델

상호운용수준		C (Component)			설명
범위	컴퓨팅환경				
(4) Enterprise	Universal	통합 (Enterprise)	b	전 세계적 상호운용	-통합 환경의 컴포넌트 상호운용 가능
			a	국가적 상호운용	
(3) Domain	Integrated	도메인 (Domain)	b	도메인간 상호운용	-도메인 컴포넌트 상호운용 가능
			a	도메인 내부 상호운용	
(2) Functional	Distributed	상호운용 (interoperable)	b	컴포넌트 상호운용 기술적 방안 제시	-컴포넌트간 상호운용 가능 -상호운용 규약 및 컴포넌트 설치 및 운용 절차 부재
			a	컴포넌트 규약 및 절차 정의	
(1) Connected	Peer to Peer	독립 (independent)	b	운용환경 독립	-컴포넌트 모델 종속
			a	개발 언어, 운용환경 의존	
(0) Isolated	Manual	의존 (dependent)	b	운용환경 의존	-개발환경 및 운용환경에 의존
			a	개발환경 의존	

3.2.3 컴포넌트 상호운용성 질의서

3.2.3.1 기존 LISI 기반 상호운용 평가의 문제점

LISI 모델은 질의서를 이용하여 평가 대상 시스템의 상호운용에 관련된 정보를 수집하여 평가한다. 그러나 LISI 모델은 미군이 사용하는 모델이어서 평가 프로세스는 공개되었으나 세부적인 평가대상 정보를 수집하는 질의서나 구현옵션은 공개되지 않은 상태이다. 따라서 LISI 기반의 평가를 하기 위해서는 정보 시스템의 상호운용 특성을 효과적으로 추출할 수 있는 질의서가 필요하다. 현재 국내에서 가용한 질의서는 [12]에서 작성한 질의서가 존재하나 단순한 표준에 대한 존재여부를 검사하는 초보적인 수준이다. 컴포넌트 기반 시스템이나 분산 환경에 대한 고려가 미비하여 발전된 정보 시스템 환경에 맞는 질의서가 필요하다.

3.2.3.2 컴포넌트 상호운용성 질의서

본 논문에서는 컴포넌트 기반 시스템의 상호운용성 측정을 위한 질의서를 Check List 형식으로 작성하였다. 먼저 컴포넌트 기반 시스템의 상호운용성을 측정하기 위하여 CBD 방법론에서 정의하는 컴포넌트 개발 단계별로 질의서를 작성하였다. CBD 방법론은 [17]을 사용하였다. 질의서는 CBD 방법론에서 정의하는 요구사항 분석, 설계, 상세설계, 구현 및 시험, 설치의 단계별로 질의가 구성되어 있어 사용자는 개발의 각 단계별로 상호운용성을 평가하고 평가 결과를 개발에 반영할 수 있다.

상호운용의 평가 방식에 따라 개별 시스템의 상호운용 능력을 평가하는 질의와 시스템간의 상호운용성을 평가하는 질의를 포함하여 모두 1,179개의 질의를 작성하였다[18]. 검사의 형식 또한 평가자의 주관적인 평가를 지양하기 위하여 구체적인 항목을 점검하는 Check List형식으로 변환하여 작성하였다. <표 5>는 상호운용 질의서의 한 예를 보여준다. 본 질의서는 CBD 방법론의 절차에 따라 질의가 진행된다. 따라서 CBD 방법론의 개발 단계별로 개발과 병행하여 상호운용성 평가가 가능하게 된다. 질의서의 구성에 CBD 방법론에서 정의한 관련 산출물을 추가하여 평가자는 질의에 관련된 문서의 세부항목을 참조하여 평가하면 되므로 평가에 필요한 정보를 쉽게 얻을 수 있다. 그리고 평가의 이해를 높이기 위하여 질의서의 구성에 절차흐름을 포함하였다.

4. 상호운용 수준 측정 사례 분석

이 장에서는 본 논문에서 제시한 컴포넌트 기반 시스템 상호운용 측정 기법의 적합성을 검증하기 위하여 프로토타입 형태의 컴포넌트 기반 시스템을 개발하여 평가하고 결과를 비교분석하였다.

4.1 프로토타입 개요

본 논문에서는 EJB와 .Net으로 두개의 컴포넌트를 개발하였다. 그리고 개발된 컴포넌트를 조합하여 웹 서비스와 IIOP.Net을 이용한 컴포넌트 기반 시스템을 각각 개발하였다. 즉 동일한 컴포넌트들을 사용하였으나 시스템 개발 방식을 달리하였다. 이렇게 개발된 두개의 시스템 각각에 대하여 본 논문에서 제시하는 상호운용성 평가 질의서를 이용하여 상호운용성 수준 평가를 하였다.

4.2 프로토타입 구조

EJB와 .Net 컴포넌트간의 상호운용성을 웹 서비스와 분산객체를 이용하여 구현하였다. 서비스를 제공하는 역할을 하는 서버 컴포넌트와 서비스를 제공 받는 클라이언트 컴포넌트의 구현기술은 다음과 같다.

- 서버 컴포넌트 : EJB 기반
- 클라이언트 컴포넌트: .Net 기반

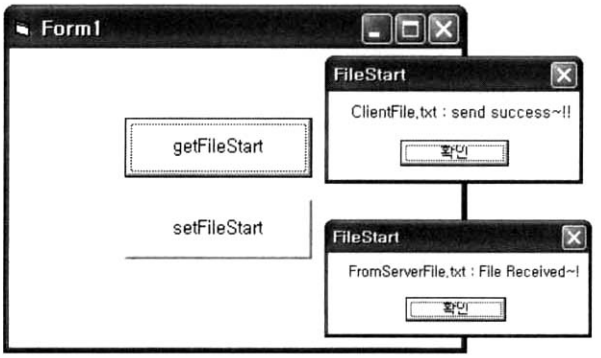
4.3 웹 서비스를 이용한 구현

웹 서비스를 이용해서 서버 컴포넌트와 클라이언트 컴포넌트가 파일을 전송하는 시나리오로 구현하였다. (그림 7)은 웹 서비스를 이용한 EJB 서버 컴포넌트와 .Net 클라이언트 컴포넌트의 소스코드와 실행화면이다.

〈표 5〉 질의서의 구성 예

개발 단계	관련 문서	개발 절차 흐름	관련 문서 세부항목	질 의	LISI 특성	측정값
설계 (명세 단계)	컴포넌트 명세서	인터페이스 명세 ↓ 인터페이스 정보 모델링 ↓ 컴포넌트 명세	인터페이스 명세서	-식별된 컴포넌트와 인터페이스의 표준 명세화가 되었는가?	P(1)	
			컴포넌트 명세서	-국가 정보 기반 계획레벨에서의 표준 컴포넌트 명세화가 되었는가?	P(5b)	
			컴포넌트 명세서	-전사적 표준과 국방 정보 기술 구조 레벨에서의 표준 컴포넌트 명세화가 되었는가?	P(2b/c)	
			컴포넌트 명세서	-DBMS와 같은 응용 어플리케이션을 통해 공통의 데이터 접근에 대한 컴포넌트 아키텍처를 명세화 하였는가?	A(5)	
			컴포넌트 명세서	-공통의 레파지토리를 통해 다른 응용과 데이터를 공유에 대한 컴포넌트 아키텍처를 명세화 하였는가?	A(4b)	

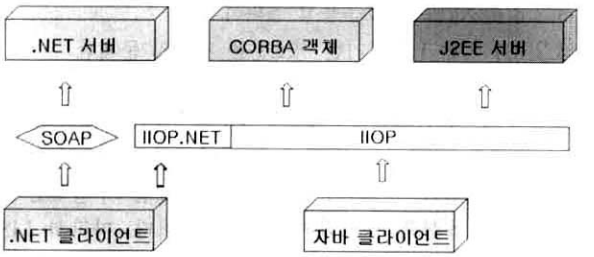
```
// 서버 EJB 소스의 일부분
// 클라이언트가 서버에게 파일을 전송한다.
// 전달되는 파일의 타입은 DataHandler이다.
public void setFile(DataHandler dh){
            .....
  try {
    fos = new FileOutputStream("FromClientFile.txt");
    dh.writeTo(fos);
    fos.flush();
  } catch
            .....
}
'클라이언트 .Net 소스의 일부분
WScript.Echo ("Create Client Succeeded ---" )
'wsdl과 wsml을 이용하여 SoapClient 객체를 초기화한다.
call client.mssoapinit(wsdl, "", "", wsml)
            .....
WScript.Echo ("Init Client Succeeded --------  --" )
'서비스를 호출한다.
Set ReceivedAttach = Client.getFile()
FileSaveName = "FromServerFile.txt"
'호출결과 반환받은 객체를 파일로 저장한다.
ReceivedAttach.SaveToFile(FileSaveName, True)
```



(그림 7) 웹 서비스를 이용한 코드 및 실행 화면

4.4 IIOP.Net을 이용한 구현

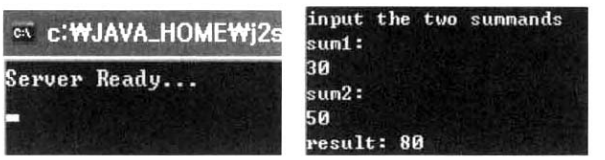
IIOP.Net은 CORBA의 핵심 부분인 IIOP 프로토콜과 .Net 리모팅 채널간의 변환 툴을 제공한다. 자바에는 이미 RMI/IIOP가 제공되고 있으므로 IIOP.Net을 이용하면 자바의 RMI/IIOP 기반 컴포넌트와 .Net 기반 컴포넌트 간에도 상호운용할 수 있게 된다[19]. (그림 8)은 IIOP .Net을 이용한 이종 컴포넌트의 연결 구조를 표시한 것이다. .Net이나 자바 클라이언트에서 SOAP, IIOP, IIOP.Net 등의 프로토콜을 이용하여 이종 서버 컴포넌트를 호출할 수 있다. (그림 9)는 IIOP.Net을 이용한 EJB 서버 컴포넌트와 .Net 클라이언트 컴포넌트의 소스코드와 실행화면이다.



(그림 8) IIOP.Net을 이용한 이종 컴포넌트 연결

```
//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);
NameComponent[] name = new NameComponent[] { new
                    NameComponent("adder", "") };
Adder adder = (Adder)nameService.resolve(name);
double result = adder.add(sum1, sum2);
Console.WriteLine("result: " + result);
```



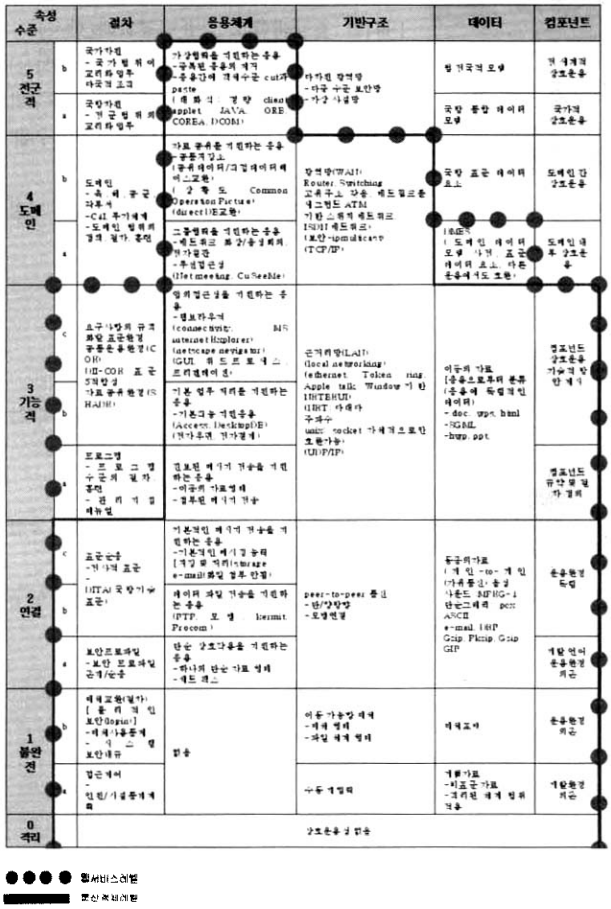
(그림 9) IIOP.Net을 이용한 코드 및 실행 결과

4.5 측정 결과

본 논문에서 제시하는 상호운용 수준 측정 기법을 이용하여 두 가지 프로토타입 컴포넌트 기반 시스템에 대하여 상호운용 수준을 측정하였다. (그림 10)은 웹 서비스 방식을 이용하여 개발한 컴포넌트 기반 시스템과 IIOP.Net을 이용하여 개발한 컴포넌트기반 시스템에 대하여 본 논문에서 개선한 LISI 상호운용 능력 모델에 적용하여 측정한 결과이다. 웹 서비스 기반 시스템의 상호운용성 측정 결과를 요소별로 살펴보면 절차 3c, 응용체계 5b, 기반구조 4b, 데이터 4a, 컴포넌트 3b이다. 상호운용 수준의 측정 결과 값은 가장 낮은 부분이 시스템의 상호운용 능력을 대표하므로 웹 서비스 기

〈표 6〉 질의서 측정 결과

NO	개발단계	관련문서	개발 절차 흐름	관련 문서 세부항목	질의	LISI 특성	측정값 (웹 서비스)	측정값 (분산객체)
1	소프트웨어 아키텍처 정의(1R21)	소프트웨어 아키텍처 정의서	소프트웨어 아키텍처 정의 ↓ 표준 지침 수립	1. 아키텍처 제약 및 요구 사항	공통운용 환경에 기반한 표준 아키텍처 정의는 가능한가?	P(3c)	P(3c)	P(2c)
2				2. 상위 요구사항	공통운용환경 표준 적합성을 만족할 수 있는가?	P(3c)	P(3c)	P(2c)



(그림 10) 측정 결과

반 시스템은 3b에 해당한다 [3]. 그리고 IIOP.Net을 이용한 컴포넌트 기반 시스템의 상호운용성 수준은 절차 2c, 응용체계 5b, 기반구조 4b, 데이터 3c, 컴포넌트 3b의 측정 결과가 나왔다. 따라서 전체적인 상호운용 수준은 2c가 된다. <표 6>의 질의서 측정 결과와 같이 절차 부분에서 웹 서비스 방식이 분산 객체 방식보다 보다 높은 상호운용 수준 측정 결과를 보여주었다. 결과적으로 IIOP.Net을 이용한 상호운용 기법은 절차적 부분의 상호운용 능력이 저하되어 전체 시스템의 상호운용 수준이 낮게 나옴을 알 수 있다. 상호운용 수준을 높이기 위해서는 절차적인 부분의 보완이 필요한 것을 사례연구를 통하여 알 수 있다.

5. 결 론

기 개발된 컴포넌트를 조립하여 새로운 정보 시스템을 개발하는 CBD 패러다임은 앞으로 점차 일반화되어 많은 정보 시스템이 컴포넌트 기반으로 개발될 것이다. 본 논문에서는 이종 분산 컴포넌트의 상호운용성 수준을 측정하고 평가하기 위하여 표준 컴포넌트 규격을 정의하고, 기존의 LISI 모델을 수정 확장하여 컴포넌트 기반 시스템의 상호운용성 측정이 가능한 능력 모델을 제시하였다. 그리고 이종 분산 컴포넌트 시스템의 상호운용성 측정에 필요한 질의서를 작성하여 컴포넌트 기반 시스템의 상호운용성 측정을 용이하게 하였다. 끝으로 이종 분산 환경의 컴포넌트 프로토타입을 개발하여 본 논문에서 제시한 평가 방안을 적용하여 측정하였다. 향후 평가 시간의 단축과 정확한 평가를 지원하기 위하여 자동화된 평가 도구의 개발이 지속적으로 연구되어야 한다.

참 고 문 헌

[1] Wegner, P., "Interoperability", ACM Computing Surveys, Vol.28, No.1, March, 1996.

[2] "DoD Architecture Framework version 2.1", DoD Architecture Framework Working Group, 2000.

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

[4] 오기성, 이남용, 류성열, "소프트웨어 품질특성에 의한 상용 컴포넌트 선정 방법에 관한 연구", 정보처리학회 논문지 D, 제9-D권 제5호, pp.897-902, 2002.

[5] Paulk, M. C., Weber, C. V., Curtis, B. and Chrissis M. B., "The Capability Maturity Model Guidelines for Improving the Software Process", Carnegie Mellon University, Software Engineering Institute, Addison-Wesley Publishing Company, 1995.

[6] Garlen, D., Rovert, A. and John, O., "Architectural Mismatch or Why it's hard to build systems out of existing parts", Proceedings of the 17th International Conference on Software Engineering IEEE, 1995.

[7] Matzel, K. and Schnorf, P., "Dynamic Componenet Adaptation", Ubilab Tec. Rep. 97 6-1, Union Bank of

Switzerland. 1997.
[8] Chiang, C., "The Use of Adapters to Support inter-operability of Components for Reusability", Elsevier Science B. V., pp.149-156, 2002.
[9] Rine, D., Nader, D. and Khaled, J., "Using Adapters to Reduce Interaction Complexity in Reusable Component-Based Software Development", Symposium on Software Reusability Proceedings of the 1999 symposium on Software reusability, 1999.
[10] Francisco C., Matthew, D., Rania, K. and William, N., "Unraveling the Web Services Web an Introduction to SOAP, WSDL and UDDI", IEEE Internet Computing, 2002.
[11] Orth, G., "The Web Services Framework : A Survey of WSDL, SOAP and UDDI", Master Thesis, Vienna University of Technology, 2002.
[12] "국방정보체계 상호운용성 수준(LISI) 업무편람", 국방부, 2002.
[13] "Defense Information System(DII) Common Operating Environment(COE) Integration and Runtime Specification 3.1 (I&RTS)", US DoD, 1998.
[14] 류동국, 김영철, "컴포넌트 기반 시스템 상호운용성 측정 및 평가를 위한 상호운용 능력 모델 개발", 정보과학회 2004 춘계학술대회 발표논문집, 2004.
[15] George, T. H. and William, T. C., "Component Based Software Engineering", Addison-Wesley, 2001.
[16] 김윤정, "워크플로우메카니즘을 통한 소프트웨어 컴포넌트 식별 방법론에 관한 연구", 홍익대학교 일반대학원 전자전산전공 석사학위논문, 2004.
[17] "국방 컴포넌트 기반 방법론 지침", 국방과학연구소, 2004.
[18] 김영철, "컴포넌트 기반 체계 상호운용 적합성 평가 및 인증 기술 연구", 국방과학연구소 위탁과제, 홍익대학교, 2004.
[19] Patrik, R., "Building a Distributed Object System with .NET and J2EE Using IIOP.NET", The Code Project, 2003. 7., http://www.codeproject.com/csharp/dist_object_system.asp.

류 동 국

e-mail : dkryu@lycos.co.kr
1994년 중앙대학교 컴퓨터공학과(학사)
1996년 중앙대학교 컴퓨터공학과(공학석사)
1996년~1999년 국방정보체계 연구소 연구원
1999년~현재 국방과학연구소 선임연구원
관심분야 : 상호운용성, 소프트웨어 성숙도 모델, 컴포넌트 시험 및 평가 등

김 영 철

e-mail : bob@hongik.ac.kr
2000년 Illinois Institute of Technology(박사)
2000년~2001년 LG 산전 중앙연구소 Embedded System 부장
2001년~현재 홍익대학교 컴퓨터정보통신공학과 교수
관심분야 : 소프트웨어 성숙도 모델, Use Case 방법론 및 도구 개발, 정보 구조, 컴포넌트 시험 및 평가