CORBA 규격 준수 시험을 위한 분산 시험사례 실행 제어기

°정혜경*, 한재일*, 남궁한**
silk@cs00.kookmin.ac.kr, jhan@kmu.kookmin.ac.kr, nghan@etri.re.kr
*국민대학교 컴퓨터학부, **한국전자통신연구원

The Distributed Test Case Execution Controller for the CORBA Compliance Testing

H. K. Chung*, J. I. Han*, H. Namgoong**
*School of Computer Science, Kookmin University, **ETRI

요 약

대표적 분산 미들웨어인 OMG CORBA 는 이질적인 하드웨어와 소프트웨어가 다양한 매체를 통해 네트워크로 연결된 분산 컴퓨팅 환경에서 객체 지향적 시스템 통합 기반 환경을 제공한다. 그러나 CORBA 를 구현한 상용이나 연구용 제품은 CORBA 규격을 정의된 대로 지원하지 않아 OMG 에서 CORBA 를 통해 성취하려고 하는 중요한 목표인 상호운용성과 이식성이 보장되지 못하는 결과를 야기하고 있다. 이러한 문제의 해결은 CORBA 규격 준수 여부를 시험하는 시험 도구를 필요로 한다. CORBA 규격 시험 도구는 규격에 정의된 API 등을 시험하기 위한 많은 분산 시험사례와 이들의 실행을 자동으로 제어해 줄 수 있는 실행제어기가 기본적으로 필요하다. 본 논문은 CORBA 규격 시험을 위한 분산 시험사례를 자동으로 실행시키는 분산 시험사례 제어기의 설계와 구현에 대하여 기술한다.

1. 서론

객체의 재사용과 시스템 구성이 용이한 컴포넌트 모델은 이제 컴퓨터 전분야에 걸쳐 적용되면서 컴포넌트 프로그래밍 시대가 개막될 것을 예고하고 있다. 컴포넌트 개념은 클라이언트/서버 모델에 까지 적용되기에 이르렀고, 이기종 간의 클라이어트/서버 시스템을 더욱 효율적으로 통합 관리할수 있는 시스템을 요구하게 되었다. 현재 OMG 의 CORBA 는 이와 같은 객체 지향기술을 비롯한 여러 진보된 기능과 함께 시스템과 언어에 독립적인 구조를 제공하고 있는 대표적인 미들웨어로 자리잡고 있다. 그러나 CORBA 를 구현한 상용이나 연구용 제품은 CORBA 규격을 정의된 대로 지원하지 않아 CORBA 를 통해 성취하려는 중요한 목표인 이식성과 상호운용성을 보장하지 못하는 결과를 야기하고 있다[1, 2]. 이와 같은 문제의 근본적인 해결은 CORBA 규격에 정의된 기능의 완전한 지원 여부를 검증(verification)할 수 있는 CORBA 규격 검증 시험 도구를 개발하여 구현된 CORBA 제품에 대하여 상호운용성과 이식성을 보장하는 것이다[1].

CORBA 규격 시험은 규격에 정의된 API 등을 시험하기 위하여 많은 분산 시험사례와 이들의 실행을 자동으로 제어해 줄 수 있는 실행제어기가 필요하다[3].

본 논문은 CORBA API 기능 시험을 위한 분산 시험사례를 자동으로 실행시키는 분산 시험사례 제어기에 적합한 모델을 제시하고 구현에 대하여 논한다. 또한 분산 시험사례의 자동 실행을 위한 실행 제어기의 개발에 관련된 자동 시험 기술에 대하여 간략히 기술한다.

* 본 연구는 국민대학교 교내연구비와 한국전자통신연구원의 부분적인 연구비 지원으로 수행되었음.

본 논문의 구성은 다음과 같다. 2 장은 규격정의에 따른 자동 시험 기술과 CORBA 규격 시험에 관련된 연구에 대하여 설명하고 3 장에서는 CORBA 규격시험을 위한 분산 시험사례의 자동 실행제어 모델을 제시하며 구조를 설명한다. 4 장은 구현에 대하여 기술하였으며, 마지막으로 5 장에서는 본논문의 결론 및 향후과제를 제시한다.

2. 관련 연구

지난 70 년대에는 시험사례의 실행이 수동으로 이루어졌다. 그러나 80 년대에 프로그램의 크기가 커지고 개발된 프로그램의 계속적인 업그레이드에 따른 회귀시험(regression testing)으로 인해 효율적인 시험 수행 방법이 요구되어 시험의 자동화 방법에 대한 연구가 시작되었다. 규격정의에 따른 소프트웨어 시험은 다음과 같은 기능을 수행한다[3].

- 소프트웨어 규격을 정의한다. 이 정의를 규격정의(specification)로 부른다.
- 규격정의로부터 시험사례를 생성한다.
- 생성된 시험사례를 필요한 보조 실행 코드와 함께 실행된다.
- 출력된 시험 결과로부터 시험사례의 품질(test quality)과 소프트웨어 품질에 대하여 분석한다.

이와 같이 전체적인 시험의 자동화는 시험사례의 실행을 자동으로 실행 제어하는 것 뿐만 아니라 시험사례의 자동 생성과 시험 및 소프트웨어 품질에 대한 자동 평가를 포괄하여야 한다. 자동 시험 기술은 위의 각

절차에 대한 자동화를 요구하며 다음과 같이 크게 세 분야에 대한 연구가 진행되고 있다[3, 4].

- 소프트웨어의 기능을 명시하기 위한 언어의 정의 및 시험사례 자동 생성[7]
- 시험사례의 자동 실행 제어[8]
- 시험사례의 품질(test quality)과 소프트웨어 품질에 대한 자동 분석[9]

CORBA 규격 시험과 관련된 연구중 본 논문의 관심 대상인 CORBA 규격에 대한 시험도구는 Open Group 의 CORVAL(CORBA VALidation)프로젝트[1]에서 유일하게 연구되고 있다. Open Group 에서는 1997 년 CORVAL (CORBA VALidation) 프로젝트를 시작하여 1998 년 CORBA 2.0 규격에 대한 검증용 시험도구인 VSORB 와 VSJORB 를 개발하여 상용화 하였으며 현재 CORBA 규격 2.1 을 검증을 할 수 있도록 개정되었다. VSORB 는 CORBA 의 C/C++언어 매핑을 위한 것으로 Open Group 에서 개발한 규격정의 언어인 ADL(Assertion Definition Language)[5]과 자동 시험실행 도구인 TETware[6]를 사용하여 구현되었으며 제한된 정보가 공개되어 있다[2]. VSJORB 는 VSORB 의 Java 버전으로 VSORB 와 매우 유사하기 때문에 설명을 생략한다.

3. CORBA 규격시험을 위한 분산 시험사례 실행제어기

본 논문은 CORBA API 기능 시험을 위한 분산 시험사례의 자동 실행제어기를 설계한다. 이 실행제어기를 JDTC(Java Distributed Test-case Controller)로 부른다. JDTC 는 VSORB 에서 사용된 자동 실행도구 JETPack 의 주요 기능을 모두 제공하며 JETPack 이 TETWare 의 C/C++라이브러리 함수를 호출하여 사용하는 것과 달리 순수 Java 언어로 구현한다. [그림 1]은 JDTC 의 구조를 보이고 있다. JDTC 의 기본적인 기능은 다음과 같다.

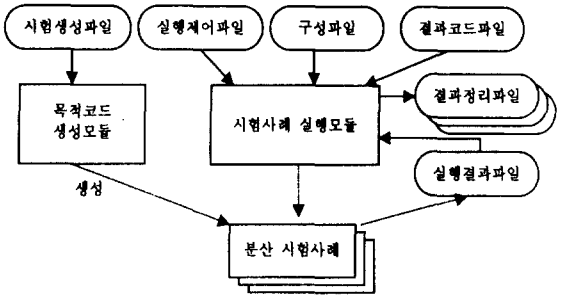
- 시험사례의 실행을 제어
- 결과 메시지 출력 등 시험사례 작성에 필요한 프로그래밍 인터페이스 제공
- 특정 시스템과 ORB(Object Request Broker)에 대한 정보를 바탕으로 시험을 위한 목적코드 생성

JDTC 는 두개의 모듈과 여섯 종류의 파일로 구성된다. 시험사례 자동실행제어기는 특정환경에서의 목적코드 생성을 위한 목적코드 생성모듈과 생성된 시험사례의 실행을 위한 실행모듈의 두 모듈로 구성된다. 기타 구성요소에 대한 설명은 [10]을 참조하기 바라며 지면 관계상 본 논문에서는 설명을 생략한다.

4. 실행제어기의 구현

본 장은 JDTC 를 구성하는 주요 클래스에 대해 그 기능과 역할을 간략히 기술하며 JDTC 객체간 대화흐름도를 보인다. JDEC 는 시험사례 실행모듈의 구현과 시험사례 작성을 위한 클래스의 두 그룹으로 나누어지며 각 그룹의 핵심적인 클래스는 다음과 같다.

- 시험사례 실행모듈 클래스



[그림 1] JDTC 구조

public class DTCC

public class JdtcD
public class TestSession
public class Synchronizer
public class ScenarioParser

- 시험사례 작성을 위한 클래스

public abstract class TestCase
public abstract class TestClientFrame
public abstract class TestServerFrame

DTCC(Distributed Test-Case Controller)객체는 시험사례 실행제어 파일, 시나리오 파일에 명시된 시험사례의 목록을 만들어 순서대로 하나씩 시험사례 실행을 제어하며 시험 결과를 보여줌으로써 전체적인 시험사례의 실행을 관리한다. JdtcD 객체는 기본적인 시험사례 실행 환경을 초기화 하며 각 시험사례를 실행할 때 마다 새로운 쓰레드를 분기하여 TestSession 객체를 실행한다. TestSession 객체는 특정 시험사례의 실행에 필요한 시험환경을 초기화 하며 시험 클라이언트와 시험 서버가 기본적으로 필요로 하는 여러가지 기본 서비스와 통신을 담당한다. Synchronizer 객체는 분산 객체 사이의 동기화를 위해 임의의 사건(event)이름을 이용한 대기(wait)와 통지(notify)서비스, 즉 이벤트 서비스를 제공한다. ScenarioParser 객체는 주어진 시나리오 파일의 시험사례를 명시된 실행 순서대로 목록을 만들어 돌려준다. TestCase 클래스는 시험 클라이언트와 시험 서버가 TestSessioin 객체에게 보다 쉽게 서비스 요청을 할 수 있게 하는 공유 API 를 제공한다. TestClientFrame 클래스는 시험 클라이언트의 초기화, 시험 사례의 실행완료 통지, 그리고 시험사례의 시험그룹에 있는 시험항목을 실행해 주는 메쏘드를 제공한다. TestServerFrame 클래스는 시험 서버의 초기화 메쏘드를 제공한다.

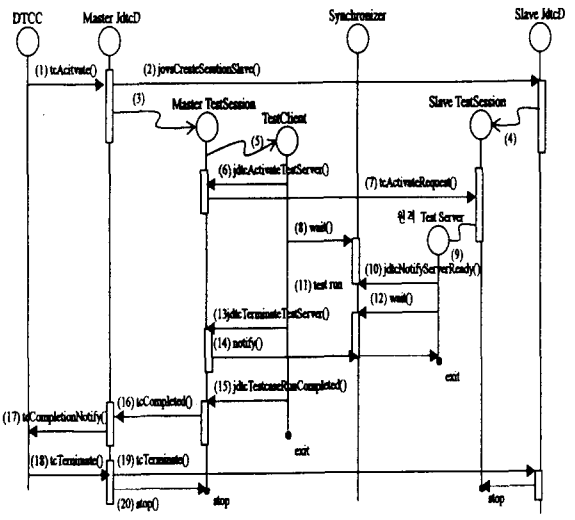
서로 다른 컴퓨터에서 시험 클라이언트와 시험 서버가 실행되는 경우 JDTC 의 작동은 다음과 같다. DTCC 객체는 시험사례 목록에 있는 시험사례를 실행하기 위해 JdtcD(JDTC Daemon)객체는 DTCC 객체로부터 시험사례 실행을 요청 받아 슬레이브 JdtcD 객체에게 슬레이브 TestSessioin 객체를 생성할 것을 요청하고(2), 자신의 프로세스에 쓰레드를 하나 분기하여 매스터 TestSession 객체를 실행한다(3). 슬레이브 JdtcD 객체도 쓰레드를 하나 분기하여 슬레이브 TestSession 객체를 실행한다(4). 매스터 TestSession 객체는 JdtcD 객체로 부터 시험사례의 시험 클라이언트에 대한 정보를 얻어

시험 클라이언트를 독립된 프로세스로 실행시킨다(5). 시험 클라이언트는 시험 서버가 필요한 경우 매스터 TestSession 객체에게 시험 서버의 실행을 요청하며(6) 매스터 TestSession 객체는 슬레이브 TestSession 객체에게 시험 서버를 독립된 프로세스로 실행시킬 것을 요청한다(7). 시험 클라이언트는 시험 서버가 서비스 할 준비가 될 때 까지 Synchronizer 객체에서 제공하는 대기기능을 사용하여 기다리며(8) 시험서버가 Synchronizer 객체에게 서비스 준비완료 통지를 하면(10) 다시 실행을 계속한다. 슬레이브 TestSession 객체는 매스터 TestSession 객체의 요청을 받아 시험 서버를 독립된 프로세스로 실행하며(9), 시험 서버는 Synchronizer 객체에게 서비스 준비가 됐음을 알린다(10). 시험 클라이언트와 시험 서버 객체는 상호 협조하여 필요한 시험을 수행하며 시험 수행 과정에 대한 메시지를 매스터 TestSession 객체에게 보내 현재 실행하고 있는 시험 사례의 시험 수행 기록 및 시험 항목과 실행 결과를 로그 파일에 저장하도록 요청한다(11). 시험 서버는 시험 사례의 실행이 완료 되면 시험 클라이언트로부터 시험 완료 통지를 받기 위해 Synchronizer 객체에게 대기 요청을 하며(12) 시험 클라이언트(실제로는 매스터 TestSession 객체)로부터 통지가 오면 임시로 생성한 파일 등을 삭제하고 종료한다(14). 시험 클라이언트는 시험 사례 실행이 완료되면 매스터 TestSession 객체에게 시험 서버의 종료를 요청하며(13) 매스터 TestSession 객체는 Synchronizer 객체를 통해 시험 서버에게 시험 완료 통지를 한다(14). 그 후 시험 클라이언트는 매스터 TestSession 객체에게 시험 완료 사실을 알린 후 종료한다(15). 시험 완료 메시지를 받은 매스터 TestSession 객체는 매스터 JdtcD 객체에게(16), 매스터 JdtcD 객체는 DTCC 객체에게(17) 이 사실을 전파한다. DTCC 객체는 시험 클라이언트로부터 시험완료 메시지가 전달되면 매스터 JdtcD 에게 현 시험사례의 실행을 위해 생성되었던 매스터/슬레이브 TestSession 객체를 소멸시킬 것을 요청한다(18). 매스터 JdtcD 객체는 DTCC 객체의 TestSession 객체 소멸 요청이 오면 슬레이브 JdtcD 객체에게 슬레이브 TestSession 객체를 소멸시킬 것을 요청하고(19) 매스터 TestSession 객체를 소멸 시킨다(20). 슬레이브 JdtcD 객체는 매스터 JdtcD 객체의 요청을 받아 슬레이브 TestSession 객체를 소멸시킨다.

5. 결론 및 향후 과제

CORBA 가 제공하는 잇점은 이식성과 상호운용성을 보장하는데 있다[2]. 그러나 현재의 CORBA 제품들이 CORBA 규격을 정의된 대로 지원하지 않아 CORBA 의 목적을 실현하지 못하는 상태이다. 이 문제의 해결은 CORBA 규격 검증을 필요로 한다.

CORBA 규격 시험은 규격에 정의된 API 등을 시험하기 위하여 많은 분산 시험사례를 필요로한다. 또한 많은 분산 시험사례의 실행을 효율적으로 수행하기 위해 분산 시험의 수행을 자동으로 제어해 줄 수 있는 실행 제어기가 필요하다. 본 논문은 CORBA API 기능 시험을 위한 분산 시험 사례를 자동으로 실행시키는 분산 시험 사례 제어기에 적합한 모델을 제시하였고 구현에 대하여 설명하였다.



[그림 2] JDTC 객체간 대화 흐름도

JDTC 는 CORBA 규격에 정의된 기능시험을 위한 분산 시험사례의 자동 실행을 주 목표로 개발되었으나 일반 분산 시험사례의 실행에도 사용될 수 있다. JDTC 는 앞으로 사용자의 편의를 위한 GUI 인터페이스의 개발, 규격정의 언어의 지원, 최근 활발히 연구되고 있는 안정성(safety)와 생존성(liveness)을 보장할 수 있는 기능 등을 추가하여 신뢰성 있는 범용 분산 시험사례 자동실행 제어기로 확장되어야 할 것이다.

참고문헌

[1] The Open Group, CORBA Validation(CORVAL), URL: http://www.opengroup.org/vsorb/corval/

[2] The Open Group, VSORB Test Suite Specification Release 1.0.0, X/Open Company Ltd., April 1997

[3] R.M.Poston, Automatic Specification-based Software Testing, IEEE Computer Society Press, 1996

[4] P. Stocks and D.Carrington, "A Framework for Specification-Based Testing," IEEE Transactions on software Engineering, Vol. 22, November 1998

[5] The Open Group ADL Translation System User's Guide: Getting Started With ADLT Release 2.0, X/Open Compan Company Ltd., Feb 1998

[6] The Open Group, TETware Design Specification Revision 1.0, X/Open Company Ltd., April 1996

[7] W.E.Howden, "The Theory and Practice of Functioinal Testing," IEEE Software, Vol. 17, No. 7, July 1991

[8] G.Duval, "Specification and Verification fo an Object Request Broker," Proceeding of the 1998 International Conference on Software Engineering. , April 1998

[9] J.D.Musa and A.F.Ackerman, "Quantifying Software Validation: When to Stop Testing?," IEEE Software, Vol. 6, No. 3, May 1989

[10] 정혜경외 6 명, "ORB 기능의 CORBA 규격 준수 시험", 한국정보과학회 추계학술대회 논문지, 제 26 권 2 호, pp. 629-631, 1999