EJB 컴포넌트의 테스팅 사례연구

서예영, 신현정, 이남용
[yysuh, hjshin]@selab.soongsil.ac.kr,
nylee@computing.soongsil.ac.kr
소프트웨어공학 연구실
숭실대학교 대학원 컴퓨터학과

Contents

Research Purpose

❑ EJB 컴포넌트의 품질인 Functionality, Reliability, Performance, Stability를 보증하기 위하여 Rational QualityArchitect(RQA) 툴을 이용하여 효과적인 컴포넌트 테스팅 방법을 제안

- ➢ 툴을 이용하여 컴포넌트 및 인터페이스를 갖는 가상 컴포넌트을 작성하여 단위 테스팅 방법
- ➢ 테스트케이스 설계 및 테스트 스크립트 작성 방법
- ➢ 테스트데이터의 Regression Test시 재사용 방법
- ➢ RQA가 제공하는 테스트 스크립트를 특정 EJB서버에 커스터마이징하는 방법

❖ Quality Factors of EJB Components

- ✓ Can write scalable, reliable, and secure applications without writing your own complex distributed object framework
- ✓ Can quickly and easily construct server-side components in Java by leveraging prewritten distributed infrastructure provided by the industry
- ✓ Is designed to support application portability and reusability across any vendor's enterprise middleware services

Research Scope

❑ RUP Elaboration Phase와 UML Components의 Provisioning Workflow에서 컴포넌트 테스팅 방법 연구

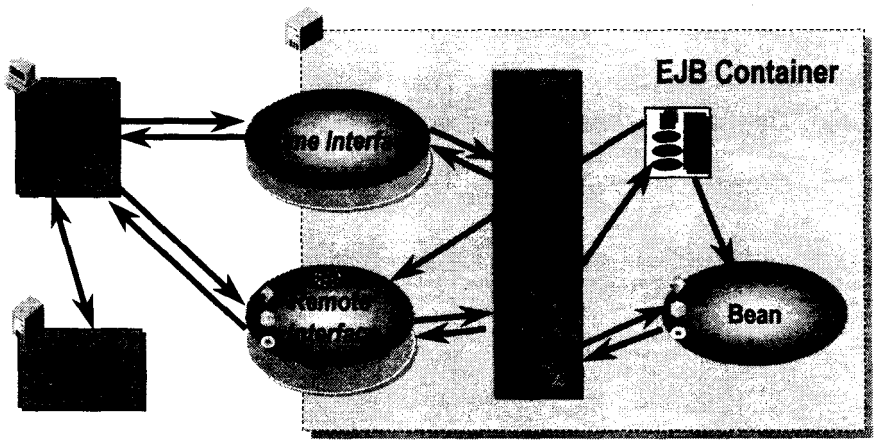
❑ RUP의 Design Workflow에서 산출된 Design Model로부터 EJB Component의 메소드 특성에 따라 Test Case, Test Script, Test Data, Test Source Code, Test Results, Test Log를 자동으로 생성방법 연구

❑ Rational QualityArchitect와 TestManager를 이용

➢ Test Case, Test script, Test Data, Test Source Code, Test Results, Test Log 등을 버전별로 자동 관리함

Research Method

- EJB Components



Research Method

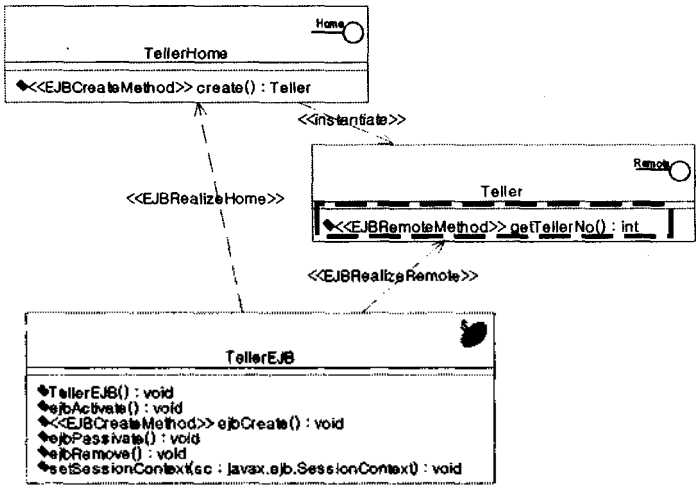
- Strategy for EJB Component Testing
  - Client applications interact directly with the home and remote interfaces but not with the implementation class.
  - Thus, we need to test only the methods defined in the home interface and the remote interface
  - EJB test scripts emulate the client applications.
- Test Design for testing an EJB Component
  - Unit Testing is to test the behavior of EJB Component in terms of the methods or the operations

Research Method

- Templates for Unit Testing
  - Business/Finder/getter/Setter Methods: remote.template
  - Create Methods: home.template
- Templates for Stub
  - Session_Bean.template
  - Session_Home.template
  - Session_Remote.template
  - MethodBody.template, MethodBodyWithoutExceptions.template. MethodBodyWithoutLookUp.template, MethodBodyWithoutReturnValue.template

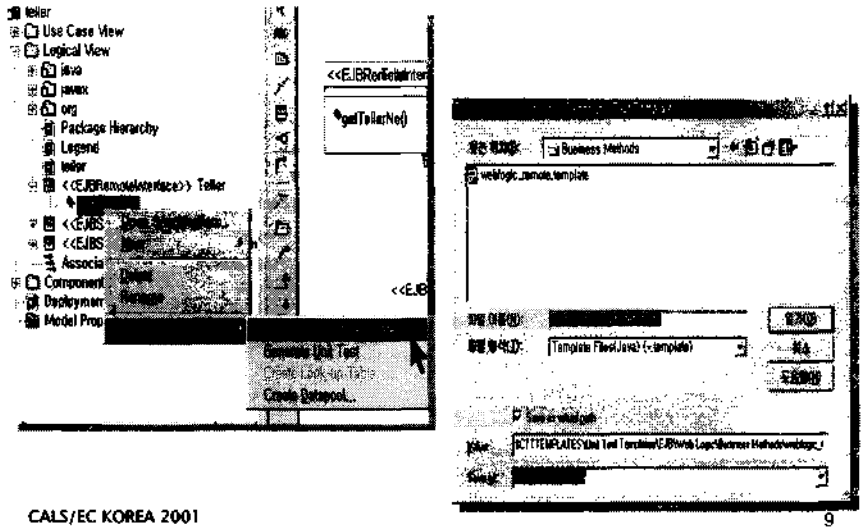
EJB Component Testing

- Identification of the targets associated with EBJ Component (TellerEJB)



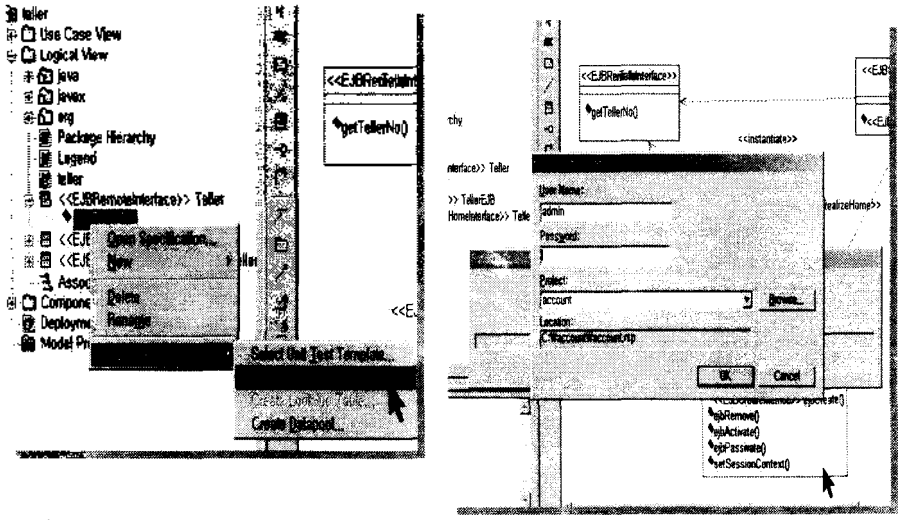
EJB Component Testing(Cont'd)

❑ Unit Test Code(Select Test Template) Generation for Teller Remote Interface with RQA



EJB Component Testing(Cont'd)

❑ Unit Test Code Generation for Teller Remote Interface with RQA



EJB Component Testing(Cont'd) - Test Code

```
//package unittests.;
// Date: 2001-03-06 오후 9:21:58  Author: <author_name>
// This script tests int getTellerNo() method of .Teller.

import com.rational.test.ct.*;
import com.rational.test.tss.*;
import com.rational.test.vp.*;
import com.rational.test.vp.ui.*;
//import .*;
import java.util.*;
import java.math.*;
import javax.naming.*;
import javax.rmi.PortableRemoteObject;

public class TellergetTellerNo extends com.rational.test.tss.TestScript {
   /**
   * Gets initialContext.
   * This method uses weblogic.jndi.WLInitialContextFactory to obtain an
   *  initial naming context.
   *
   * @return InitialContext The initial naming context.
   */
   private InitialContext getInitialContext() throws NamingException {
      InitialContext  initialContext  = null;
      Properties p                    = new Properties();
      String contextFactory           = "weblogic.jndi.WLInitialContextFactory";
      String url                  = "t3://localhost:7001";
      String portNumber               = null;
      String hostName                 = "localhost";
```

```
set CLASSPATH=%JAVA_HOME%\lib\tools.jar;%RL_HOME%\lib\weblogic_sp.jar;
%RL_HOME%\lib\weblogic.jar;
%RL_HOME%\lib\rational_ct.jar;%RL_HOME%\lib\rttsoajaoa.jar;
%RL_HOME%\lib\rttssjava.jar;
%RL_HOME%\lib\swingall.jar;%CLASSPATH%
```

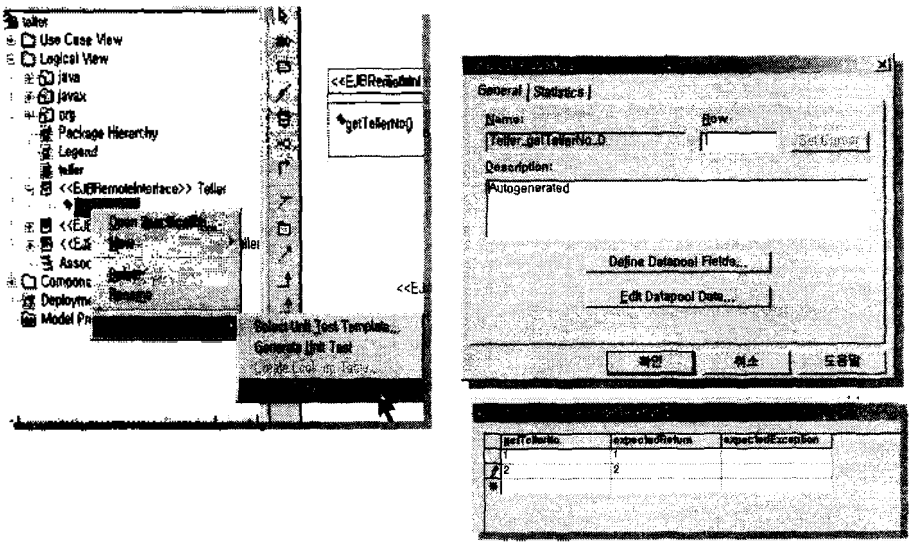

EJB Component Testing(Cont'd) - Test Code

```
      // !!! Add Port number and host name to use specific URL.
      // !!!

      // Use the port number and host name if provided by the user
      if (portNumber != null) {
         url = "t3://" + hostName + ":" + portNumber;
      } // endif

      // add the URL for the naming context
      p.put(javax.naming.Context.PROVIDER_URL, url);

      // add the context factory for the naming context
      p.put(javax.naming.Context.INITIAL_CONTEXT_FACTORY, contextFactory );

      try {          // Get the initial context
         initialContext = new InitialContext(p);
      } catch (NamingException e) {
         throw(e);
      } // endtry

      return initialContext;
   }

   public static void main(java.lang.String[] args) {
      try
      {  TellergetTellerNo script = new TellergetTellerNo();
         script.testMain(null);
      } catch (Exception e) {
         System.out.println("From getMessage() : " + e.getMessage());
         e.printStackTrace();
      }

      System.exit( 0 );
   }
```

EJB Component Testing(Cont'd) - Test Code

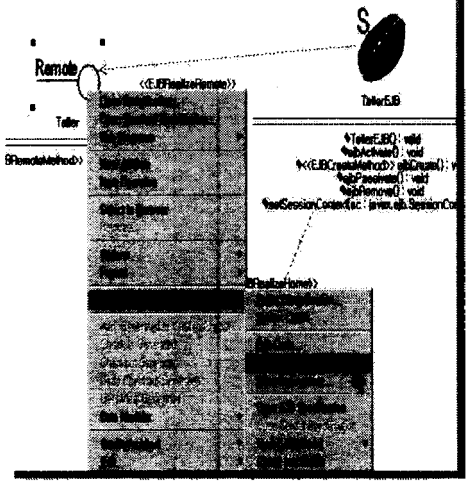
```
public void testMain(String[] args) {
    // System.out.println("1 pass");
    boolean fRetval = false;
    TSSDatapool dp = new TSSDatapool();
    int iDPCount = 0;

    try
    {
        // Initialize test services
        tms.startTestServices();

        // Initialize arguments, to int getTellerNo() method
        int expectedReturn = 0;
        int actualReturn = 0;
        String sExpectedException = "";

        // Contact the bean home through JNDI
        InitialContext initContext = getInitialContext();

        // TODO: the generated JNDI context name is defaulted to be the Remote Interface Name.
        // Please replace with appropriate JNDI context name for your needs, if you have deployed
        // the EJB in some other JNDI subcontexts.
        // TellerHome home = (TellerHome)initContext.lookup("Teller");

        //Context ctx = new InitialContext();
        Context initial = getInitialContext();

    //look up jndi name
    Object ref = initial.lookup("TellerHome");
```


EJB Component Testing(Cont'd) - Test Data Generation



EJB Component Testing(Cont'd) - Source Code Generation

❑ Generation of the source code for Teller Remote Interface



Teller.java

```
//
// -- Java Code Generation Process --

// Import Statements
import java.rmi.RemoteException;
import javax.ejb.*;

public interface Teller extends javax.ejb.EJBObject
{

    /**
     * @rosewld 387348E20342
     * @J2EE_METHOD -- getTellerNo
     */
    public int getTellerNo    0 throws
java.rmi.RemoteException,javax.ejb.EJBException;
}
```


EJB Component Testing(Cont'd) - Source Code Generation

❑ Generation of the source code for TellerHome Interface and TellerBean

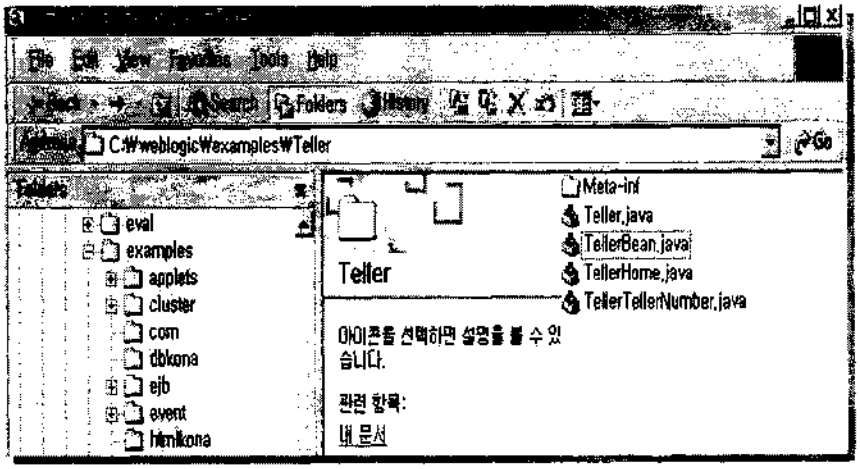
TellerHome.java

```
//
// -- Java Code Generation Process --

// Import Statements
import java.rmi.RemoteException;
import javax.ejb.*;

public interface TellerHome extends javax.ejb.EJBHome
{

    /**
     * @rosewld 387348E20348
     * @J2EE_METHOD -- create
     * Called by the client to create an EJB bean instance. It
requires a matching pair in
     * the bean class, i.e. ejbCreate(...).
     */

    public Teller create    0
            throws
javax.ejb.CreateException,java.rmi.RemoteException;
}
```

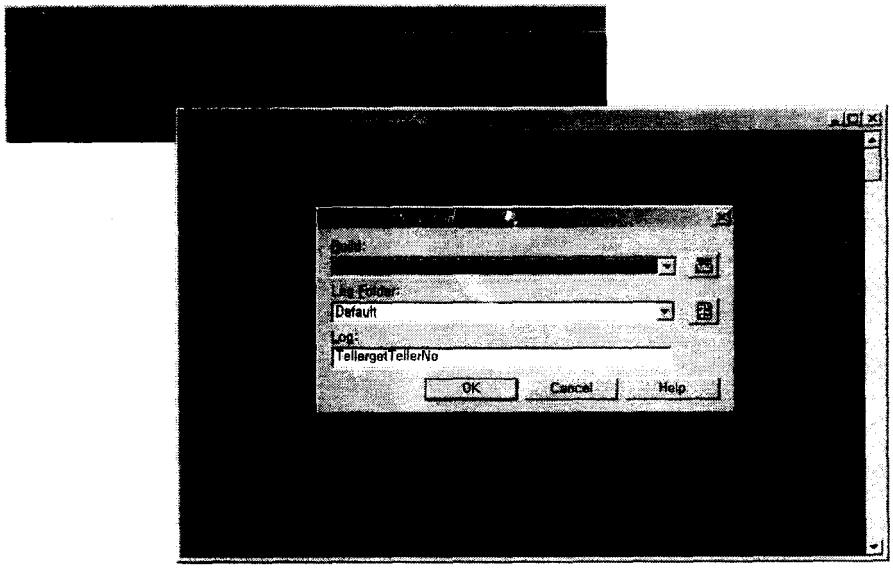
TellerBean.java

```
//
// -- Java Code Generation Process --

// Import Statements
import java.rmi.RemoteException;
import javax.ejb.*;

public class TellerEJB implements javax.ejb.SessionBean
{

    public void TellerEJB    0 throws javax.ejb.EJBException
    {
    }

    public void ejbActivate    0 throws
java.rmi.RemoteException
    {
    }

    public void ejbCreate    0 throws
javax.ejb.CreateException,java.rmi.RemoteException
    {
    }

    public void ejbPassivate    0 throws
java.rmi.RemoteException
    {
    }
```

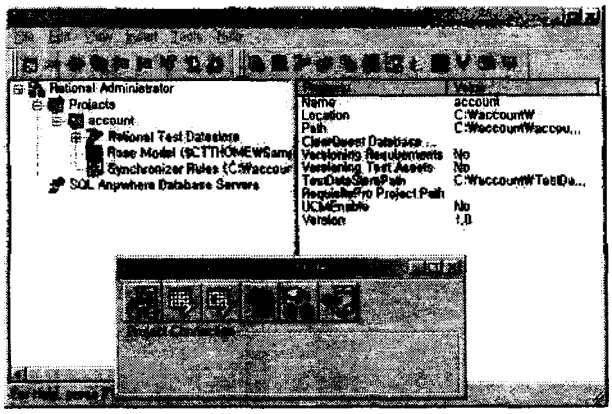
EJB Component Testing(Cont'd) - Importing Test Assets



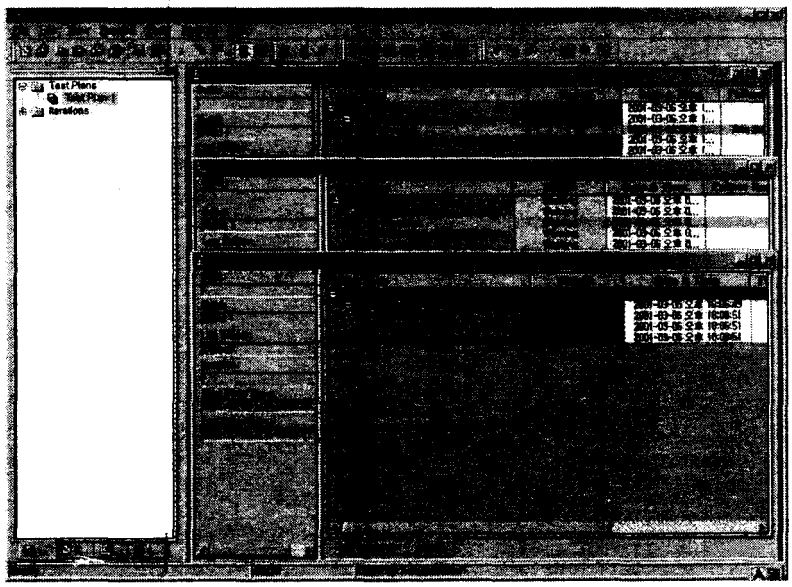
EJB Component Testing(Cont'd) - Test Script Execution



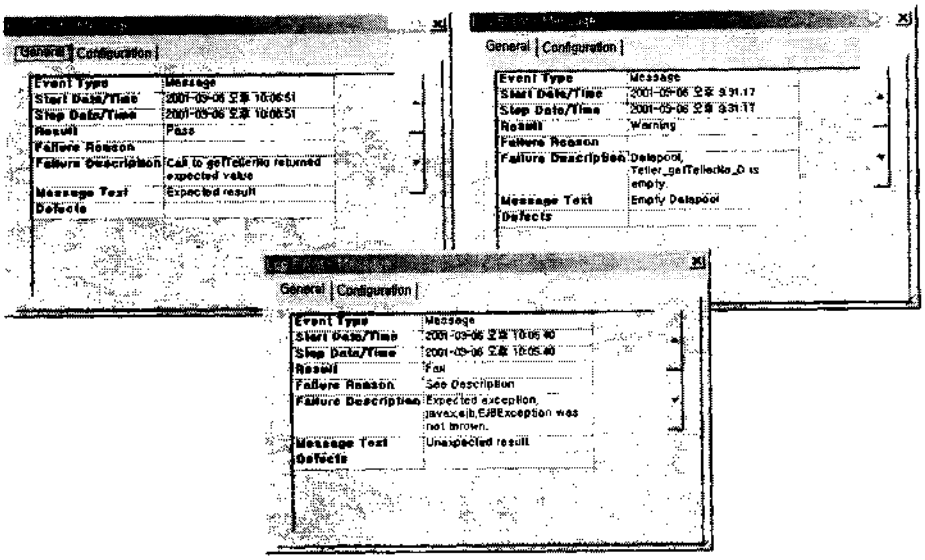
EJB Component Testing(Cont'd) - Execution Results



EJB Component Testing(Cont'd) - Execution Results: Test Log



EJB Component Testing(Cont'd) – Execution Results: Properties of Log Event



Findings and Conclusion

- 시스템개발 초기단계에서 컴포넌트의 테스팅이 가능하며, 이렇게 함으로써 조기에 Defect를 발견하고 fixing이 가능하여 프로젝트의 위험을 감소할 있음
- 비쥬얼 모델로부터 컴포넌트 테스트를 자동으로 생성 가능하며, 설계에 따른 구현을 검증할 수 있음
- Data Pool에 Test Case 각각에 대한 Input, Output을 포함하며, 컴포넌트의 변경에 대한 Regression Testing을 지원
- 테스팅 수행 결과를 가시적이며 체계적으로 다섯 단계에 걸쳐 보여줌
  - Test Log: Computer Start, Script Start, Message, Script End, Computer End
  - Result: Completed, Fail, Informational, Not Run, Pass, Stopped, Unevaluated, Warning
- 연관성 있는 메소드 간의 변경에 따른 Test Script의 재 생성은 물론 메소드 각각에 재 테스팅이 필요함
- RQA가 생성한 Test Script를 WebLogic 상에서 수행할 때에 WebLogic에 맞게 Test Script의 재 수정이 필요함
- Unit Testing이 완료된 컴포넌트를 새로운 환경에 재사용할 때에 컴포넌트의 Compatibility에 대한 추가적인 테스팅이 필요하며, 이에 대한 연구가 필요함

References

1. Brain Bryson, Quality by Design: Enabling Cost-Effective Comprehensive Component Testing, Rational Software 2001
2. Rational Software Corporation, Component Testing with Rational QualityArchitect, Rational Software White Paper, TP - 191, 2001