인터페이스 정보를 이용한 컴포넌트 테스팅

박세희 [0] 진영택* 황선명**
nm242m@hanmail.net ytjin@hanbat.ac.kr, sunhwang@dju.ac.kr

A Component Testing Method using Interface Information

Sehui Park [0] Young Taek Jin* Sun Myung Hwang**
*Hanbat national university
**Daejon university

요 약

컴포넌트를 기반으로 하는 소프트웨어 개발이 점차 확산됨에 따라 컴포넌트 기반 소프트웨어의 품질과 신뢰성을 보장하기 위한 컴포넌트 테스팅에 대한 필요성이 대두되고 있다. 본 논문에서는 컴포넌트 개발자가 제공해야 하는 인터페이스 정보의 내용을 제시하고 표현하며 그런 정보를 이용하여 단위 컴포넌트 테스팅과 통합 과정에 적용하는 과정을 사례를 통하여 제시한다.

1. 서 론

컴포넌트를 기반으로 한 소프트웨어 개발은 특히, 상용 컴포넌트(COTS: Commercial -Off -the-Shelf)를 구입하고 조립하여 단기간에 걸쳐 소프트웨어 제품을 만들 수 있는 유망한 방법이다. 컴포넌트의 설계 및 개발 못지 않게 중요한 분야는 컴포넌트 기반 소프트웨어 개발의 신뢰성 및 품질 보증을 위한 컴포넌트의 테스팅 분야이다[1]. 컴포넌트의 개발자는 컴포넌트가 소프트웨어가 요구 사항에 맞게 구현되었는지 또는 컴포넌트의 행위와 기능이 적합한 수준에 있는지를 테스팅할 수 있는 방법이 필요하다. 아울러 컴포넌트를 구입하여 사용하는 사용자 관점에서는 컴포넌트의 품질에 대한 인증과 컴포넌트가 적절히 동작한다는 것을 테스팅 하기 위한 기법이 요구된다. 컴포넌트는 보통 객체 지향 개념과 통합이 되고 또한 객체 지향 언어(C++, Java등)로 작성이 되기 때문에 기존의 객체 지향 테스팅 방법을 적용할 수 있다. 그러나 컴포넌트와 객체 사이에 각각 다른 특성이 존재하므로 효과적인 컴포넌트 테스팅을 위해서는 컴포넌트가 가지고 있는 여러 가지 특성을 파악해야 할 필요가 있다[2].

특히, 컴포넌트의 소스 코드는 보통 유용하지 않기 때문에 컴포넌트의 개발자는 컴포넌트에 대한 적절한 정보를 제공해야 한다. 그러한 컴포넌트의 명세는 컴포넌트의 합성과정에서 컴포넌트 기능의 부적절한 사용을 방지할 수 있으며 테스팅에도 이런 정보를 활용할 수 있다[3,4,5]. 컴포넌트의 사용과 관련된 가장 기본적인 정보는 컴포넌트 인터페이스에 대한 정보로서 인터페이스는 컴포넌트에 대한 액세스 포인트이다[6]. 인터페이스를 기반으로 한 컴포넌트 테스팅은 컴포넌트 통합 과정에서 발생하는 인터페이스 불일치를 테스트 한다. 이를 위해 인터페이스에 대한 기본 정보 및 인터페이스 사이의 종속성을 포함한 각종 정보가 요구된다. 본 논문에서는 컴포넌트의 테스팅을 위해 요청되는 컴포넌트 정보의 유형과 내용 및 이러한 정보를 컴포넌트 테스팅과 연결하기 위한 프레임웍을 제시한다.

본 연구는 한국과학재단 목적기초연구
(R01-2001-000-00343-0(2002))지원으로 수행되었음

2. 컴포넌트 정보 모델

2.1 컴포넌트 정보의 특성

소스 코드가 유용하지 않은 바이너리 컴포넌트에 대해 기능 테스팅을 수행할 때 제공되는 기본 정보는 사용자가 컴포넌트를 테스팅하는데 많은 도움을 준다. 그러나 정보의 종류가 많고 그러한 정보를 효과적으로 적용할 특정한 사용 배경을 정보 제공자가 모르기 때문에 정보의 제공 기준이 요구된다.

(1) 컴포넌트 정보의 용이성
컴포넌트 기반 시스템의 개발 과정에서 개발자가 편리하게 생성하고 쉽게 얻을 수 있는 컴포넌트 관련정보이어야 한다.

(2) 컴포넌트 정보의 필수성

컴포넌트의 행위를 테스팅 하기 위해 인터페이스를 구성하는 각 오퍼레이션에 대한 사전/사후 조건 및 불변 조건(invariant)을 검사하는 것이 필요하다. 이런 정보는 정적, 동적 테스팅에 필수적이며 UML과 OCL(Object Constraint Language)[7]과 같은 표준 명세를 사용하여 제공되어야 한다.

(3) 컴포넌트 정보의 연관성

컴포넌트의 설계시 이용되는 설계 패턴은 테스팅 시에 이와 유사한 테스트 패턴으로 적용될 수 있으며 테스트 패턴을 활용하게 되는 경우 테스트 케이스의 생성을 효과적으로 수행할 수 있다.

2.2 인터페이스 정보의 추출 및 내용

컴포넌트와 관련한 정보는 컴포넌트 개발 기술에 따라 다양한 방법을 사용하여 인터페이스에 대한 기본 정보를 얻을 수 있다. COM/DCOM 모델에서는 타입 라이브러리와 MIDL을 통해서 정보를 얻을 수 있다. 타입 라이브러리는 COM객체 또는 COM 인터페이스의 바이너리 표현이며 COM 컴포넌트와 함께 채택된다. 또한 *IUnknown* 인터페이스는 컴포넌트 인터페이스에 대한 정보를 얻을 수 있도록 한다. [표 1]은 대표적인 컴포넌트 모델에서 인터페이스와 이벤트에 대한 정보를 제시하고 있다.

표1 인터페이스정보

모델 / 항목	COM/DCOM	CORBA	EJB
도구	MIDL Type library	IDL	Introspection Beaninfo class
정보 내용	인터페이스 특성, 오퍼레이션,인수	인터페이스 특성, 오퍼레이션,인수	인터페이스 특성, 오퍼레이션,인수

2.3 인터페이스 정보의 표현

인터페이스 관련 정보는 전술한 바와 같이 컴포넌트 사용자에 의해 추출될 수 있지만 이 정보 이외에 컴포넌트 개발자에 의해 제공되야 하는 정보로서 다음과 같은 내용을 나타내며 인터페이스 정보를 표현하기 위해 XML[8]을 이용하였다.

1) 인퍼페이스 서명(signature)
   i) 특성: 외부에서 측정 가능한 구조적 요소로서 컴포넌트를 이해하거나 변경하여 채택하는데 사용된다. 어떤 특성은 읽기만 하고 변경할수 는 없다
   ii) 연산: 컴포넌트가 제공하는 서비스 또는 기능을 표현한다.
   iii) 사건: 컴포넌트가 생성하는 사건으로서 다른 컴포넌트가 응답할 수 있다.
2) 인터페이스 제약 사항
   i) 서명에 나타난 각 요소들과 관련한 제약 사항: 사전/사후 조건 및 특성에 대한 범위 제약
   ii) 각 요소들 사이의 관계에 대한 제약 조건: 한 연산은 다른 연산의 호출 후에 호출, 특정한 특성 값이 어떤 범위에 있을 때 호출되는 연산

3) 인터페이스 팩키징 또는 형상
   i) 컴포넌트가 사용되는 문맥에 따라서 적용되는 연산이 각각 다를수 있기 때문에 그런 상황에 따라 인터페이스를 팩키징 하며 각 인터페이스 형상에 대한 사용 시나리오가 제공된다. 이러한 XML로 표현된 컴포넌트 기본 정보 및 인터페이스 정보를 추출하는 과정은 배포된 컴포넌트 내에 컴포넌트 정보를 개발자가 쉽게 접근하여 참조할 수 있는 형태로 변환하여 주는 컴포넌트 정보 변환 컴포넌트를 [그림 1]에 제시된 바와 같이 포함시킨다. 이 컴포넌트는 XML로 표현된 컴포넌트 정보를 Microsoft XML Parser(MSXML 3.0)를 사용하여 DOM 구조(트리)로 변환하여 메모리에 저장한다. 개발자는 이 컴포넌트가 제공하는 메소드를 통하여 DOM 구조로 된 각각의 컴포넌트 정보 노드들에 쉽게 접근하고 검색할 수 있으며, 텍스트 형태로 출력할 수 있다.

3. 컴포넌트의 테스팅

컴포넌트 시스템은 개별 컴포넌트의 합성으로 작성되며 소프트웨어 컴포넌트는 계약적으로 명시된 인터페이스를 갖는 합성 단위이다. 개별 컴포넌트의 테스팅을 토대로 본 논문에서는 여러 컴포넌트가 서로 상호 작용하고 종속 관계를 가질 때에 수행되는 테스팅에 초점을 맞추며 다음과 같은 인터페이스 오류를 탐색한다.

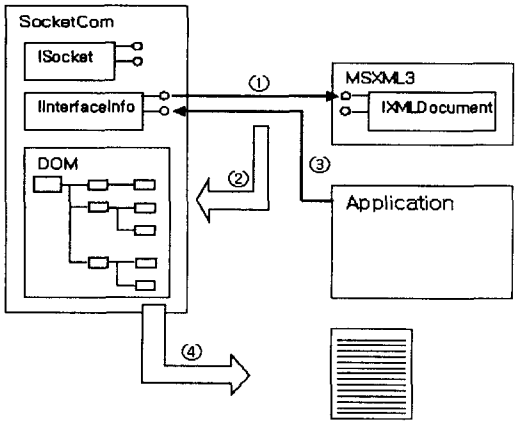

그림1 XML로 표현된 컴포넌트 정보의 활용

i) 클라이언트 컴포넌트 X가 서버 컴포넌트 Y의 사전 조건을 위반하는 메시지를 보내는지의 여부

ii) 클라이언트 컴포넌트 X가 서버 Y의 순차적인 제약조건을 위반하는 메시지를 보내는지의 여부

iii) 메시지에 잘못된 인수 또는 부정확한 인수값을 제공하는지의 여부.

3.1 테스트의 수행

컴포넌트를 통합 테스트 하기 이전에 단위 컴포넌트의 테스팅은 카테고리 분할 방법을 적용하여 테스트 케이스를 생성한다. 컴포넌트와 함께 패키징 되는 정보를 활용하여 컴포넌트와 같은 테스트 클래스를 생성하고 컴포넌트의 인터페이스에 주어진 제약 조건을 검사한다. 검사가 완료된 후에 실제의 작업은 [그림 2]에 제시된 바와 같이 테스트할 컴포넌트에 위임이 되고 수행결과를 테스트 오라클과 비교한다.

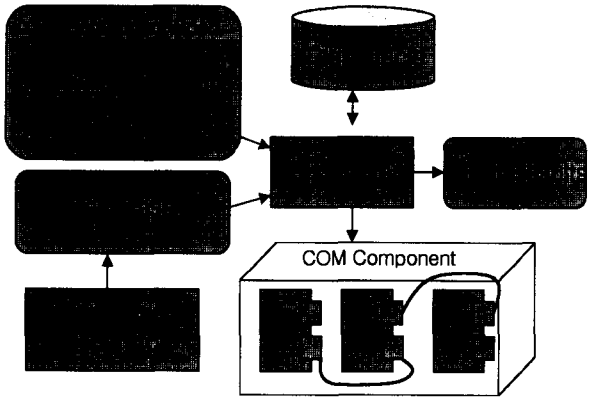

그림2 컴포넌트 테스팅의 수행 과정

4. 결론 및 향후 과제

컴포넌트를 기반으로 한 소프트웨어 개발은 미리 작성된 컴포넌트를 재사용하여 특정한 요구에 맞게 채택하여 사용한다. 제삼자에 의한 개발된 컴포넌트를 사용하는 경우 각 컴포넌트가 수행할 작업에 대한 신뢰 및 품질이 보증되어야 한다. 본 연구는 컴포넌트 테스팅 모델을 토대로 테스팅의 방향을 설정하기 위해 컴포넌트 테스팅을 위한 인터페이스 정보의 유형과 범위를 제시하였다. 향후 과제로서 메타데이터와 아키텍처 명세를 토대로 컴포넌트 통합 테스팅을 위한 테스트 케이스를 자동으로 생성하고 관리하기 위한 기법을 연구한다.

참고 문헌

1. Jerry Gao,Testing component based software, Technical Report, San Jose State university.

2. John D. McGregor and Davis A. Sykes, A Practical guide to testing object-oriented software, Addison -Wesley 2001.

3. Allessandro Orso, Mary Jean Harrold and David Rosenblum. Component metadata for software engineering tasks, proceedings 2nd International Workshop on engineering distributed objects, Nov.2000.

4. J.Han. An approach to software component specification, proceedings of the 1999 international Workshop on component-based software engineering, May 1999.

5. J.Han. A comprehensive interface definition framework for software components, proceedings of the 1998 Asia-Pacific software engineering conference, pp.110-117, IEEE Computer society press.

6. Klaus Bergner et al, A Formal Model for component ware, Foundations of component-based systems, Cambridge University Press, pp.189-210, 2000.

7. Jos Warmer and Anneke Kleppe, The Object Constraint Language -Precise Modeling with UML, Addison-Wesley, 1999.

8. Junichi Suzuki and Yoshikazu Yamamoto, Managing the software design documents with XML,