사용자 요구사항 기반의 테스트 케이스 추출 설계 기법

송유진*, 이은주**
경북대학교 전기전자컴퓨터공학부
e-mail:{ yujinkor*, ejlee** }@knu.ac.kr

The Design of Modeling Testcase Abstraction based on User Requirement

Yu-Jin Song*, Eun-Joo Lee**
School of Electrical Engineering and Computer Science
Kyungpook National University

요 약

인터넷의 발달로 인해 최근 Software 분야는 복잡해지고 많은 분야에서 결정적인 산출물로 다양한 어플리케이션이 개발되고 있다. 또한 Software의 품질과 보증을 목적으로 테스팅의 규칙적인 방법이 요구되어진다. 본 논문에서는 다양한 Software Application의 테스팅을 위한 개발 방법으로 확장성과 융통성 그리고 재사용성을 위하여 테스트 케이스 추출을 위한 메타모델을 시각화하고 테스팅 단계별 요구되어지는 모델을 Unified Modeling Language를 이용한 개발 관점을 테스트 설계 모델과 테스트 제어 모델로 구분하여 제시하고 특정 시스템 도메인을 활용하여 테스트 실행 환경과 테스트 프로세스 설계에 대한 방법을 제안한다.

1. 서론

소프트웨어 개발에 있어서 소프트웨어 테스팅은 매우 중요한 단계이다. 또한 많은 조직에서 시스템 개발 비용 중 30-50%가 테스팅에 소요된다[1]. 소프트웨어 테스팅은 소프트웨어의 결함을 나타내기 위한 시도로서, 기존의 소프트웨어 테스팅은 개발 과정 중 마지막 단계에서 소프트웨어 개발 생명주기의 전통적인 방법을 적용하게 된다. 그렇지만 마지막 단계에서 적용하는 테스팅은 소프트웨어 개발에서 추가비용이 발생할 수 있고 결함을 수정하는데 비용이 발생하게 된다[2][3]. 따라서 개발 단계에서 발생할 수 있는 추가 비용을 최소화하고 각 개발 단계에서 산출된 결과물에 대한 재사용성과 확장성 그리고 융통성을 높이기 위한 방법으로, 객체지향 소프트웨어 개발에서 소프트웨어 테스팅은 시스템의 각 개발 단계별 요구사항을 분석하고 설계단계에서부터 기능적 요구사항과 비기능적 요구사항을 구분하여 설계하고 테스트 케이스를 추출하도록 제안한다.

본 논문에서는 테스트 케이스 추출 방법을 메타모델로 시각화하기 위하여 요구사항 모델링으로 나타내고 이를 바탕으로 테스팅 계획단계에서의 모델을 테스트 설계 모델과 테스트 제어 모델로 구분한다. 본 논문의 구성은 다음과 같다. 테스팅에 대한 관련연구에 대해 설명하고 본론으로 시스템의 요구사항과 UML 모델의 두가지 관점을 기준으로 정적 뷰를 테스팅 설계 모델로 동적 뷰를 테스팅 제어 모델로 분류하고 테스트 케이스 추출을 위한 시스템에 대해 플랫폼에 독립적인 관점에서의 설계 프로세스를 제시하고 마지막으로 결론 및 향후 연구를 제시한다.

2. 관련연구

2.1 RIM의 Current Testing Process

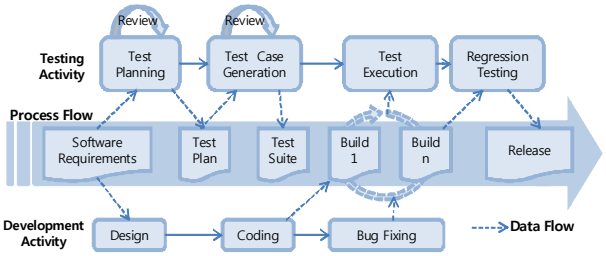
테스팅 프로세스를 개발하는 가장 큰 목적은 구체적 형태를 갖도록 요구사항을 바탕으로 테스팅 활동을 자동화하는 것이다. 따라서 넓게 적용되는 테스팅 접근 방법으로 요구사항 명세를 통한 품질향상과 개발 라이프 사이클을 통한 테스팅이다. RIM(research in motion)[1]의 소프트웨어 검증과 확인에 대한 테스팅 활동은 두가지 목적을 갖는다.

- 정확하고 완전하게 모호하지 않도록 테스트 할 수 있는 요구사항들을 확인
- 개발한 프로덕트는 사전의 확인된 요구사항과 결합시켜 증명하는 테스트 케이스를 설계

(그림 1)은 병렬적인 구조로 프로젝트가 개발되는 RIM의 Testing Process Model을 나타낸다[3]. 첫 단계에서 소프트웨어의 요구사항을 준비하고 그 요구사항을 바탕으로 프로젝트가 설계되고 그 설계를 기초로 테스트 계획을 수립하면서 테스트 행위를 시작한다.

본 논문에서는 테스팅 프로세스 모델의 소프트웨어 요구사항을 사용자가 요구하는 기본 시스템의 기능적인 부분과 비기능적인 부분으로 나누어 설계되어지는 관점을 크게 두가지 뷰로 나누어 세부 상세 모델링을 제안한다.

1) http://www.rim.com



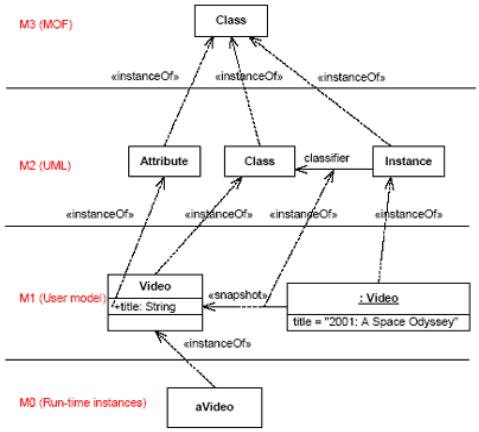
(그림 1) RIM의 Testing Process Model[3]

2.2 Software Testing

응용 프로그램의 동작과 성능 그리고 안전성이 요구하는 수준을 만족하는지 확인하기 위해 개발되어지는 모든 과정에서 테스팅 활동은 결함을 발견하기 위한 활동이다. 개발과 테스팅이 단계별로 어떻게 대응되는지를 보여주는 V모델은 단위테스트, 통합테스트, 시스템테스트, 인수테스트 4단계의 테스트 레벨로 구성된다. 그중 단위 테스팅은 테스트 가능한 최소단위로 분리된 소프트웨어 즉 모듈, 프로그램, 객체, 클래스 내에서 결함을 찾고 그 기능을 검증하는 것이다. 단위 테스팅은 코드를 중심으로 수행되며 결점은 발견될 때마다 수정되며, 원시 코드를 시험 대상으로 주된 테스트 방법은 화이트 박스 테스트이다. 단위 테스팅의 목적은 기본적 경로를 확인하고 모든 오류 처리 경로를 확인하는데 제어 흐름 테스팅 기법이 활용된다. 테스트 프로세스는 테스트 수명주기 활동을 중심으로 모든 관련 구성요소와 관련된 사항과 활동을 포함하는 개념이다[2].

2.3 OMG의 메타모델

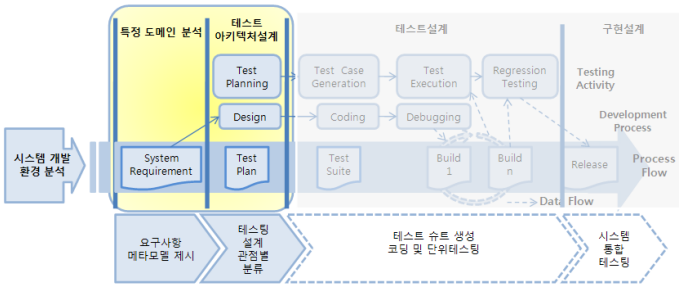
복잡한 시스템을 모델링하고 이해하기 위해서는 시각적 모델링 개념을 제공하는 표준을 사용하게 된다. 따라서 가장 널리 알려진 객체 지향 모델링 개념을 가지고 시스템의 주요 기능을 쉽게 이해할 수 있는 표준화된 Unified Modeling 표기법을 사용하게 된다. 따라서 주요기능으로 첫째, 고급 자동화를 지원하고 모델의 모호함과 부정확성을 없애고 프로그램이 모델을 변형 및 조작이 가능하도록 정확한 언어를 정의하고 둘째, 사용자가 쉽게 접근할 수 있고 모듈식의 향상된 언어 구조를 제공하며 셋째, 유연한 새로운 계층 기능으로 규모가 큰 시스템의 소프트웨어 모델링을 지원한다. 넷째, 확장메커니즘을 이용하여 도메인 스팩의 특성화를 지원한다. 다섯째, 단순하고 일관성 있는 언어로 중복된 개념을 제거하고 다양한 모델링 개념들의 정리가 쉬워진다. (그림 2)에서 OMG 4Layer를 나타낸다. M1은 시스템 분석가들이 일반적인 모델링 케이스 도구를 통해 특정 도메인 시스템을 설계한다고 했을 때의 메타모델 수준이 사용자 모델을 도식하게 된다. M2는 UML기반의 설계를 위해 Class, Attribute, instance 등과 같은 모델 요소를 정의하는 메타모델이며 M3는 Meta Object Facility는 M2에 속한 메타 모델을 정의하는 메타메타 모델이다[8].



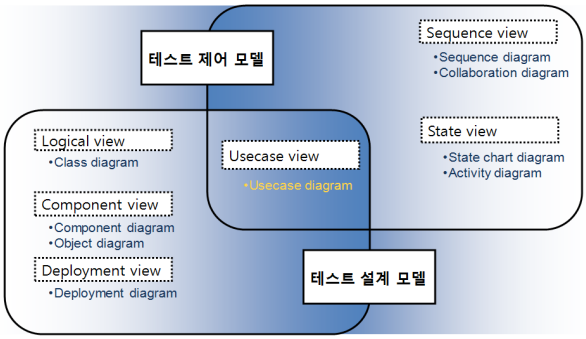
(그림 2) OMG 4 Layer 메타모델

3. 테스트 케이스 추출 모델링 식별

사용자 요구사항에 대해 메타모델링을 이용하여 제안 시스템의 설계를 시각화하고 사용자 관점에서의 테스트케이스를 추출하기 위해, 관점을 테스팅 설계 모델과 테스팅 제어모델의 두가지로 분류한다. 첫째, 테스팅 설계 모델은 사용자의 요구사항에 대한 내용을 정적 뷰로 접근한다. 둘째, 테스팅 제어모델은 사용자의 요구사항을 시간적 혹은 공간적 흐름에 따라 결과를 동적 뷰로 접근한다. (그림 3)은 테스트 프로세스 단계의 산출물을 나타낸다. (그림 4)는 테스트 설계 모델과 테스트 제어 모델에 대한 식별을 나타낸다. 본 논문에서는 간단한 동전 교환 시스템을 이용하여 테스트 설계 모델과 테스트 제어 모델에 대한 분류를 하고 이에 대한 상세 다이어그램으로 각 시스템 뷰에 대한 모델링한다.



(그림 3) 테스트 프로세스 단계의 메타모델과 모델링 분석



(그림 4) 테스트 설계 모델과 테스트 제어 모델 식별

4. 테스트 케이스 추출에 대한 설계

4-1. 요구사항 정의 및 모델링

기능적인 요구사항은 시스템의 정상적인 동작을 통하여 얻어지는 결과를 제시할 수 있는 기본적인 조건이 된다. 비기능적인 요구사항은 시스템의 정상적인 동작을 할 수 있도록 제시되어지는 제약사항이 된다. 따라서 요구사항 모델링 단계에서 시스템에 대한 비기능적인 부분을 하드웨어측면 혹은 소프트웨어측면에서의 필수조건으로 분류하고 기능적인 요구사항을 정상적인 결과를 도출하기 위해 수행되어져야하는 작업순서로 성공시나리오로 분류하고 어떤 결과를 도출하기 위한 선택적인 조건을 선택적 시퀀스로 기술하고 사후조건에 대한 내용을 명세화하여 시스템의 개발과 변경에 따른 신뢰성을 확보한다. <표2>에서는 사용자의 요구사항을 분류하여 필수조건과 선택조건 그리고 사후조건으로 분류하고 기본 이벤트 흐름과 선택적 흐름을 통해 결과를 도출하는 시나리오를 구분하여 요구사항에 대해 모델링을 하게 된다. 따라서 본 논문에서는 요구사항에 대한 시나리오에서 추출할 수 있는 개발 시스템의 필수조건, 선택적 시퀀스, 사후조건 형태를 추출하는 요구사항 메타모델로 동전교환 이벤트 흐름에 대한 시나리오를 나타낸다.

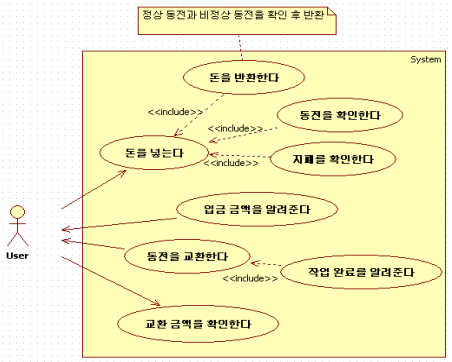
<표 2> 동전교환 이벤트 흐름에 대한 시나리오

필수조건
i) 동전교환기 시스템은 정상적으로 동작을 한다. ii) 동전교환기는 다른 동작은 하지 않고 대기상태이다. iii) 동전교환기의 동전 여유분은 충분히 있다.
성공시나리오
i) user는 교환할 지폐 혹은 동전을 선택한다. ii) 시스템은 삽입된 지폐 혹은 동전을 확인한다. iii) 시스템은 삽입된 지폐 혹은 동전의 금액을 알려준다. iv) 시스템은 삽입된 금액에 맞게 동전으로 교환한다. v) 시스템은 교환한 동전을 내보낸다. vi) user는 교환된 동전을 확인한다.
선택적 시퀀스
i) A1 : 시스템은 삽입된 지폐 혹은 동전을 확인한다. ii) 시스템은 user에게 삽입된 지폐 혹은 동전의 금액을 알려준다. iii) A2 : 시스템은 삽입된 지폐 혹은 동전 인식이 불가능할 경우 iv) 시스템은 user에게 삽입된 지폐 혹은 동전을 반환한다.
사후조건
i) 동전 교환의 정상적인 완료를 user에게 알려준다.

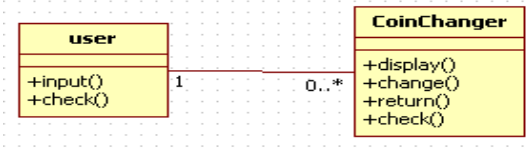
4-2. 테스트 케이스 추출에 대한 모델링 분석

4-2-1. 테스팅 설계 모델

요구사항에 대한 세부적인 부분을 사용자 관점에서 시스템 전반적인 기능부분에 대해 쉽게 이해하고 시각화할 수 있다. 따라서 사용자 요구사항에 대한 모델링을 상세하게 필수조건과 사후조건 그리고 성공적인 시나리오를 수행하기 위한 선택적인 시퀀스가 발생하게 된다. 이러한 부분을 테스팅 설계 모델 범주에 포함되는 유즈케이스 다이어그램과 클래스 다이어그램으로 시스템에 대한 기능을 나타낼 수 있다. (그림5)에서 동전교환 시스템의 유즈케이스 다이어그램을 나타내고 (그림6)에서는 클래스 다이어그램으로 나타내고 있다.



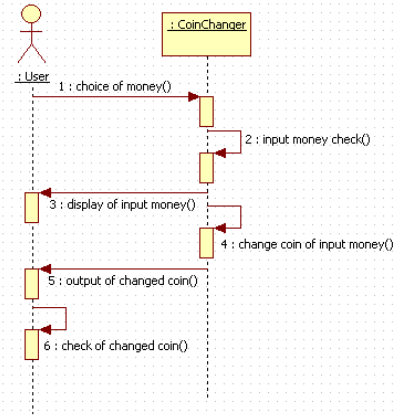
(그림 5) 동전 교환 시스템의 유즈케이스 다이어그램



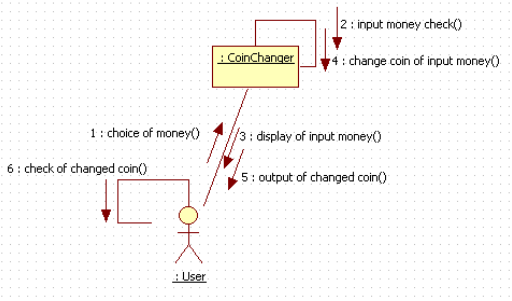
(그림 6) 동전 교환 시스템의 클래스 다이어그램

4-2-2. 테스팅 제어 모델

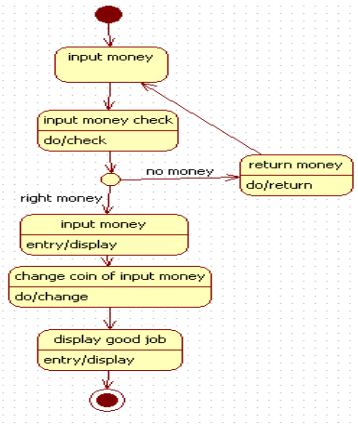
객체 지향 시스템을 설계하고 구현하기까지 기능적인 요구사항과 비기능적인 요구사항에 따라 상세하게 설계되고 그 설계에 따라 시스템을 개발하게 된다. 그중에서 기능적인 요구사항에 맞게 시스템이 운영되기 위해 성공적인 시나리오를 이용하여 선택적인 시퀀스 흐름대로 오류없이 수행되어져야 한다. 따라서, 시스템의 테스팅 설계 모델에서 제안된 다이어그램 중에서 유즈케이스 다이어그램을 설계할 때 선택적으로 발생될 수 있는 이벤트를 나타낼 수 있다. 동전교환 시스템의 시퀀스 다이어그램을 (그림7)에서 나타내고 시퀀스 다이어그램을 (그림8)에서 협동다이어그램으로 이벤트에 대한 흐름을 나타낸다.



(그림 7) 동전교환 시스템의 시퀀스 다이어그램



(그림 8) 동전교환 시스템의 협동 다이어그램



(그림 9) 동전교환 시스템의
상태 다이어그램

또한 실제 이벤트의 흐름을 분석하기 위해 테스트 설계 모델의 하나인 유즈케이스 다이어그램에서 스테레오타입으로 정의되어진 유즈케이스를 바탕으로 선택조건 분기가 발생할 수 있는 이벤트를 추출하여 오브젝트들 사이에 발생할 수 있는 이벤트를 상태 다이어그램으로 나타낸다. (그림9)에서 동전 구별 이벤트에 대한 흐름을 상태 다이어그램으로 나타낸다. 동전 혹인 지폐를 확인하고 만약 동전이라면 정상 동전인지 불량동전인지를 구별하여 선택분기를 통해 불량동전이라면 삽입한 동전을 반환하고 정상 동전이라면 실제 입력된 동전의 금액을 알려주고 잔돈으로 교환되어 올바른 작업의 종료를 알려준다.

5. 결론 및 향후연구

소프트웨어 테스팅은 소프트웨어의 결함을 식별하기 위한 시도로 정의되는데 사용자 관점에서 시스템이 개발되고 요구사항에서 기능적 혹은 비기능적인 사항이 식별되지 않으면 소프트웨어 개발 구현단계에서 개발단계부터 다시 수정해야하는 어려움과 추가비용이 발생하게 된다[3]. 다양한 소프트웨어 어플리케이션의 테스팅을 위한 테스트케이스 추출방법으로 UML을 이용하여 개발 시스템의 테스팅 모델링 관점을 설계부분과 제어부분으로 나누어 소프트웨어의 확장성과 융통성을 향상시키고 이미 개발되어진 시스템 모델링 방법을 재사용하여 비용절감의 효과를 볼 수 있다.

본 논문에서는 개발하려는 시스템의 기능적 요구사항과 비기능적인 요구사항을 추출하여 필수 조건과 선택 조건 그리고 사후조건으로 분류하고 시각적으로 나타내기 위해 메타모델로 제안하였다. 그리고 요구사항에 대한 메타모델을 이용하여 개발 시스템의 테스트케이스 추출을 위해 모델링 관점을 테스팅 설계 모델과 테스팅 제어 모델로 분류한다. 또한 시스템 개발 관점에 따른 모델링을 통하여 간단한 어플리케이션을 모델링하였다. 현재 동전교환 시스템 모델링을 통해 보였듯이 사용자 관점의 요구사항 모델링을 분석하고 시스템 테스트와 인수 테스트에 대응하여 분석할 수 있다. 향후 연구로 사용자관점에서의 요구사항을 기반으로 모델링을 이용한 테스트 추출 시스템을 구현하고자 한다.

참고문헌

[1] 권원일, “소프트웨어 질 향상을 위한 소프트웨어 테스팅의 필요성”, http://www.fkii.or.kr/new/bbs/pdf-eng/2008_Autumn_19.pdf

[2] 권원일외, “소프트웨어 테스팅 실무”, Software Testing Alliances, p.38-67, 2008.

[3] 박경민, “가능성과 확장성을 품고 등장한 UML 2.0”, http://www.zdnet.co.kr/ArticleView.asp?artice_id=00000039134178, 2005.

[4] Lbrahim, R. Saringat, M.Z. Ibrahim, N. Ismail, N., "An Automatic Tool for Generating Test Cases from the System's Requirements", Computer and Information Technology, 2007. (CIT2007), 7th IEEE International Conference, Vol.16 p.861-866, Oct. 2007

[5] Mirarab, S.; Ganjali, A.; Tahvildari, L.; Shimin Li; Weining Liu; Morrissey, M.,“A requirement-based software testing framework: An industrial practice”,Software Maintenance, 2008. ICSM 2008. IEEE International Conference on, p.452-455, Sept. 28-Oct. 4 2008

[6] Simons, Anthony J.H., Griffiths, Neil, Thomson, Christopher,"Feedback-based specification, Coding and Testing with JWalk", Practice and Research Techniques, 2008. TAIC PART "08. Testing: Academic & Industrial Conference, Vo.l p.69-73, 29-31 Aug. 2008.

[7] M. Whalen, A. Rajan, M. Heimdahl, and S. Miller., "Coverage metrics for requirements-based testing", In Proceedings of ISSTA, p.25-36, 2006.

[8] M. Grindal, J. Offutt, and S.F. Andler., "Combination testing strategies : a survey", Softw.Test., Reliab., 15(3), p.167-199, 2005.