임베디드 소프트웨어의 테스트 케이스 리엔지니어링

서 광 익[0] · 이 동 근 · 최 은 만
동국대학교 컴퓨터멀티미디어공학과
{bradseo[0], dongkun, emchoi}@dgu.ac.kr

Reengineering Test Cases of Embedded Software

Kwang Ik Seo[0] · Dongkun Lee · Eun Man Choi
Department of Computer Multimedia Engineering, Dongguk University

요 약

소프트웨어를 동적으로 테스트 하려면 대상 소프트웨어에 적절한 데이터를 주어 실행해 보아야 한다. 효과적인 테스트가 되기 위해서 테스트 케이스의 선택뿐만 아니라 테스트 케이스가 어떻게 표현되었는가가 중요하다. 또한 정적인 테스트 작업에도 테스트를 위한 체크리스트가 어떻게 작성되었는지에 따라 테스트 작업의 효율성이 좌우된다. 이 논문에서는 비효율적이며 문제가 있는 테스트 케이스와 체크 리스트들을 리엔지니어링하는 방법을 제시하고 이를 실험 하였다. 임베디드 시스템의 일종인 디지털 방송수신 장치에 탑재된 소프트웨어를 대상으로 하여 이미 사용 중인 테스트 케이스의 효율성과 적합성을 따져보고 이를 리엔지니어링 하였다. 리엔지니어링 한 후의 테스트 케이스의 산출물이 얼마나 효과적인지를 살펴보았다. 또한 제품 계열 개념의 소프트웨어를 테스트하기에 적합하도록 테스트 케이스를 재사용 또는 restructuring하는 방법도 연구하였다.

1. 서 론

소프트웨어 테스트 작업은 소프트웨어 프로덕트가 예상하는 것만큼 잘 작동되는지를 확신할 수 있는 좋은 방법 중의 하나이다. 소프트웨어를 테스트 하려면 먼저 시험 대상 소프트웨어에 대한 요구를 잘 분석하여 어떤 테스트 데이터를 주었을 때 어떤 결과가 나와야 하는지를 찾아내야 한다. 이를 테스트 케이스라고 하는데 테스트 케이스를 어떻게 작성하는가는 테스트 작업의 효율성을 크게 좌우한다.

소프트웨어에 직접 입력을 주어 실행시키는 동적인 테스트 작업은 테스트 케이스가 얼마나 오류를 잘 발견할 수 있도록 최소화 되었는가가 중요한 관건이다. 입력 자료값을 주고 수행결과를 확인하는 동적인 테스트뿐만 아니라 소프트웨어의 외관과 실행시키기 위하여 사용자가 메뉴를 선택하고 여러 가지 정보가 전달되는 방법을 구체적으로 확인하는 작업도 매우 중요한 시험 작업이다. 이러한 측면에서 사용성(Usability)을 확인하기 위한 체크 리스트도 테스트 슈트라 볼 수 있다.

임베디드 시스템의 경우 소프트웨어와 하드웨어를 개발 과정에서 분리하여 테스트하기가 매우 어렵다. 서로 밀접히 관련되어 있어 같이 연동되는 부분이 많고 단위 시험에서 시뮬레이션 환경을 제공하여 테스트하였다 하더라도 타겟 환경에서 통합한 후 시스템 수준에서 기능과 사용성 측면의 테스트가 함께 이루어져야 컴포넌트들의 연동을 확인할 수 있다.

이론적인 테스트 기법에서는 테스트 케이스는 한번 사용하고 버리는 것으로 교육하고 받아들이는 경우가 많았다. 그러나 실제 개발 현장에서는 100% 새로운 요구가 아닌 유사한 부분이 많은 소프트웨어를 개발하고 있다. 특히 소프트웨어 제품 계열(Software Product Line) 성격의 시스템은 비슷한 구성요소들 중에 일부만을 바꾸어 새로운 소프트웨어, 또는 탑재 시스템으로 완성하는 경우가 많다. 이런 경우 테스트 케이스를 모두 다시 만드는 것은 비효율적이며 지루한 작업이다.

따라서 이 논문에서는 제품 계열 개념의 소프트웨어를 테스트 할 때 이미 개발되어 테스트한 경력이 있는 요소들을 비용, 오류 추출율이라는 측면에서 과거 테스트 케이스를 개선하고 재사용하는 방법을 제안하였다. 또한 과거 테스트 케이스와 비교 시험하여 개선된 테스트 케이스가 과연 효율적인지를 보였다.

2. 좋은 테스트 케이스

IEEE에서 발행한 소프트웨어 관련 용어집[1]에 의하면 테스트 케이스는 "특정 프로그램의 경로나 소프트웨어의 요구사항을 확인 등 일정한 목적을 위하여 개발된 테스트 입력, 실행 조건, 예상 결과의 집합체"라고 정의하고 있다. 그러나 실제 소프트웨어를 세밀하고 체계적으로 테스트하려면 입력과 예상 결과 이외에도 테스트가 이루어지기 전까지의 상태와 환경, 테스트 케이스가 커버하는 기능이나 품질 요소도 중요한 요소이다.

일반적으로 좋은 테스트 케이스란 테스트 목적에 맞는 오류를 잘 드러내고, 최소한의 비용과 노력으로 원하는 테스트를 커버하게 하여야 한다. 즉 오류추출율과 커버리지가 좋으면 테스트 케이스는 좋은 것이라고 이론적으로 알려져 있다. 그러나 실무 환경에서 실제 테스트 케이스를 작성해보면 이외에도 다른 여러 가지 속성들을 만족하여야 한다. 특히 테스트 케이스를 실행시키기 위하여 필요한 조건이나 환경뿐만 아니라 테스트 케이스가 어떻게 표현되었는가도 테스트 작업의 성과를 좌

우하는 중요한 측면이다.

테스트 케이스는 일종의 소프트웨어의 기능이나 성능에 대하여 던지는 일종의 질문이다[2]. 따라서 각 테스트 케이스는 테스트 엔지니어가 잘 이해하고 작업할 수 있도록 정리되어야 한다. 또한 임베디드 시스템과 같이 제품 계열 개념에 의하여 확장되거나 조금 다른 프로덕트를 테스트 할 때 쉽게 재구성할 수 있도록 구조화 되어야 한다.

[표 1] 좋은 테스트 케이스가 되기 위한 속성

오류추출성 (Readable)	테스트 목적에 맞는 오류를 잘 드러내야 한다.
커버리지 (Coverage)	테스트 하려는 타겟이 테스트 케이스에 의하여 전부 확인될 수 있어야 한다.
경제성 (Minimize cost)	테스트 작업을 실행하는 데 소요되는 노력의 최소화.
수정가능성 (Up-to-date)	테스트 슈트가 수정 가능해야 한다.
구조성 (Structural)	재사용을 위하여 수직적으로 분해하고 합성할 수 있도록 잘 체계화 되어야 한다.
문서화 (Documented)	테스트 케이스가 작업에 도움이 되도록 잘 설명되어 있고 정리되어야 한다.

3. 테스트 케이스 리엔지니어링 작업

테스트 케이스의 리엔지니어링 작업을 위해 먼저 해야 할 일은 두 가지 이해 작업이다[3]. 먼저 기존의 테스트 케이스의 구성 요소에 대한 분석과 테스트 대상 시스템을 이해하는 것이다. 이해 작업이 끝나면 새로운 테스트 케이스 서식을 설계하고 기존의 테스트 케이스를 제안된 테스트 케이스 서식으로 다시 작성한다. 그리고 새로 작성된 테스트 케이스에 대한 평가는 두 차례에 걸쳐 실무자들에 의해 점검이 되고 최후 수정하였다. 최종 수정된 테스트 케이스의 효율성과 실용성을 분석하기 위해 전체 기능에 대한 기존 테스트 케이스와 재 작성된 테스트 케이스를 모두 비교 수행한다.

그림 1은 이러한 작업 과정을 보이고 있다.

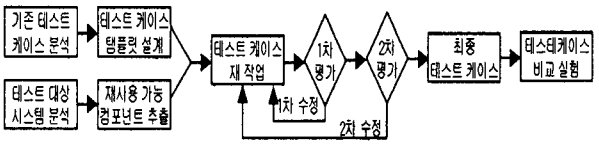

그림 1 테스트 케이스 리엔지니어링 절차

4. 테스트 케이스 재사용을 위한 전략

테스트 케이스의 재사용을 위해 재사용 단위를 결정하는 것이 필요하다[4]. 재사용을 할 수 있는 단위는 두 가지로 나누어 볼 수 있다. 첫 번째는 특정 기능 단위로 재사용 하는 것이다. 예를 들면 시스템의 기본 설정을 변경하기 위해 접근 할 경우 비밀번호를 묻게 된다. 이러한 일부 기능이 테스트 케이스 재사용의 단위가 될 수 있다. 그리고 이보다 더 큰 단위로 특정 모델 또는 제품이 테스트 케이스의 재사용 단위가 될 수 있다. 예를 들면 개발된 여러 모델들이 개발 초기 제품의 특정 모델을 기반으로 하는 경우이다. 그림 2와 같이 본 연구에서는 두 가지 경우를 모두 고려하여 테스트 케이스를 설계하였다. 그림 2의 (A)는 제품을 구성하는 재사용 가능한 단위를 표현하고 있고, (B)는 각 모델의 기본이 되는 제품을 표현하고 있다. 그림 (C)는 재사용이 가능한 기능 단위를 포함하고 있는 기본 모델이다. 즉 본 연구에서 재사용을 위한 단위의 의미는 두 가지로써 필수 기능과 이를 포함하고 있는 기본 모델이다.

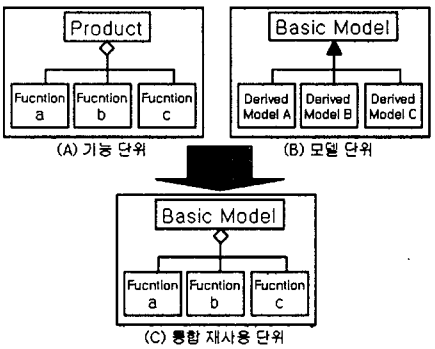

그림 2 테스트 케이스 재사용 단위

5. 기존 테스트 케이스의 문제점

테스트 대상 시스템을 시험할 때 사용하는 기존의 테스트 케이스의 구조성과 정의된 테스트 명세의 기술 방법에 대한 문제점 몇 가지를 발견하였다.

첫 번째로 기존 테스트 케이스는 현재 실무에서 실제 사용하고 있다. 그리고 가장 기본적인 기능에 대한 시험 항목을 포함하고 있어 여러 제품 라인을 시험 할 때 커널과 같이 공통적으로 사용하고 있다. 그러나 기존 테스트 케이스는 변경된 시스템의 기능을 충분히 반영하고 있지 못하고 있다.

두 번째로 서로 다른 테스트 요소들이 있음에도 이들이 혼재되어 있다. 예를 들면 하나의 테스트 체크리스트에 기능 테스트와 UI 테스트 그리고 경계값 테스트와 실시간 테스트 등이 중복되어 있다. 이러한 경우 만약 오류가 발견 된다면 오류를 기록할 때 시험 항목의 내용과 발견된 오류의 매핑이 불일치하게 된다.

세 번째로 동일한 명칭이나 기능임에도 불구하고 테스트 케이스에서 쓰이는 용어가 서로 다르다. 예를 들면 문자를 입력하는 작업 창을 'Text Edit 창' 또는 '키보드 창' 또는 'Edit 창' 등으로 불린다. 이는 시험자에게 혼란을 초래할 수 있다.

네 번째로 테스트 케이스는 시험 내용과 그에 대한 예상 결과를 함께 기술해야 한다. 하지만 그림 3에서 보는 바와 같이 테스트 케이스의 서식이 테스트 결과를 구별하여 구체적으로 기술하는 곳이 없다.

다섯 번째로 테스트 케이스가 기능 위주로 기술되어 있고 그 기능에 접근하는 절차나 과정에 대한 시험 내용은 없다. 특정 기능이 오류 없이 잘 동작한다 하더라도 그 기능의 시작점까지 접근하는 과정에 오류가 발생할 수 있다.

이 밖에도 테스트 케이스의 다양한 부가 정보나 발견된 에러에 대한 관리를 지원하는 정보가 부족하다. 이러한 문제점을 보완하기 위해 본 연구에서는 테스트 케이스 서식을 새롭게 설

계하고 테스트 작업 내용을 작성하는 방법에 대해 제안하였다.

Item / No	B. 1. 4 Installation		Result
	Test Segment	Test Description	
B. 1. 4. 4	Manual Search	1. Transponder에서 User Define 선택 시 Edit Transponder Menu로 이동 되어야 한다.	
		2. Transponder에서 User Define 선택 시 Edit Transponder Menu로 이동 Frequency 항목에 커서 이동 Arrow Right/OK Key 누르면 입력 창 출력되어야 하며, 숫자 키 입력과 동시에 다음 자리에 커서 입력이 되어야 한다.	

그림 3 기존 테스트 케이스

그림 3은 기존에 사용하던 테스트 케이스의 한 부분이다. 테스트 케이스에는 관련 기능과 번호 그리고 그 결과를 적게 되어 있으나 테스트 데이터나 환경, 전제 조건 등에 대한 내용은 찾아 볼 수 없다.

6. 리엔지니어링한 테스트 케이스

새로 작성한 테스트 케이스는 5절에서 언급했던 문제점들을 모두 해결하도록 수정하였다. 먼저 4절에서 재사용 단위가 가능한 기본 모델의 모든 기능들에 대해 테스트 케이스를 작성하였다. 그리고 하나의 체크 리스트는 하나의 최소 기능을 설명하는 목적어 하나와 서술어 하나를 기본으로 하였고 UI 시험 같은 경우 하나의 아이콘이나 하나의 아이템을 체크리스트의 단위로 정의했다. 그리고 각 체크리스트가 기능 시험인지 또는 UI 시험인지 각 특성을 분류할 수 있도록 구별하였다. 또한 테스트 케이스에서 사용되는 용어들을 통일하여 정리하였고 다양한 부가 정보들을 포함 할 수 있도록 테스트 케이스 서식을 설계하였다. 그리고 기능의 동작과 그 기능에 접근하는 과정 및 초기 상태에 대한 시험을 구별하여 시험 범위를 확장하였다. 이 외에도 테스트 케이스의 고유 번호를 통해 테스트 케이스의 시험 내용을 직관적으로 식별할 수 있도록 구성했다.

그림 4는 새로 작성한 테스트 케이스의 항목들을 간략히 보인 것이다.

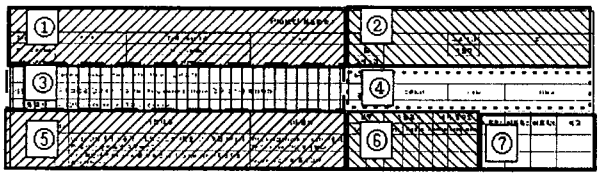

그림 4 리엔지니어링 한 테스트 케이스

새로 제안한 테스트 케이스는 표 작성의 용이성과 향후 테스트 케이스 재사용 자동화를 위해 DBMS와 연동이 가능한 스프레드시트 응용 프로그램을 사용했다. 테스트 케이스의 각 부분을 설명하자면 ①번에 해당하는 내용은 '시험 대상 시스템 정보'로써 시험 대상 모델번호와 테스트 케이스 식별번호, 소프트웨어 버전과 하드웨어 버전 그리고 로드 프로그램의 버전과 시스템 아이디의 정보를 기술한다. ②번 내용은 '시험자 정보'로써 시험 일자와 소요시간 그리고 수험자의 소속과 이름을 기록한다. ③은 '시험 내용 정보'로써 테스트 케이스의 이름과 테스트 케이스에 대한 설명 그리고 접근 절차에 대해 기술한다. ④번은 '시험 조건 정보'로써 시험 환경과 전제조건 그리고 시험 데이터와 테스트 케이스의 중요도에 대해 기술한다. ⑤번 내용은 '시험 내용'으로써 구체적인 시험 항목 즉, 체크리스트와 그에 대한 기대 결과를 기술한다. ⑥번은 '시험 종류 및 결과 정보'로써 각 체크리스트가 시험하는 종류가 어떤 것인지 구별할 수 있도록 했으며 시험 결과와 중요도를 평가할 수 있도록 하였다. ⑦번은 '품질 정보'로써 ISO/IEC 9126에서 정의한 품질 요소와 관계있는 메트릭을 기술함으로써 시험이 종료된 이후 품질을 평가할 때 사용할 수 있는 정보를 기록하도록 하였다.

또한 비밀번호 인증이나 경고문을 알리는 팝업창과 같이 중복되어 사용되는 기능은 모두 음영처리를 통해 재사용되고 있는 테스트 케이스 임을 표시하였다.

7. 결론 및 향후 과제

연구 결과 제안된 테스트 케이스는 기존의 테스트 케이스에 비해 더 많은 정보의 기록이 가능해 지면서 좀 더 완벽한 테스트가 가능하도록 하였다. 기존의 테스트 케이스의 개수는 100개였다. 하지만 개선한 테스트 케이스는 471개로 확장되어 기능의 초기 상태와 접근 절차의 시험이 가능해 졌고 테스트 종류를 구분할 수 있도록 하였다. 제안된 테스트 케이스의 양이 많지만 단위 기능의 테스트 케이스를 재사용했기 때문에 테스트 케이스의 개발 기간은 개수에 비해 짧다고 할 수 있다. 재사용된 테스트 케이스는 '일반-비밀번호' 인증이 40개, '알림-비밀번호' 인증이 25개, Group 기능이 62개, 예약기능이 7, 텍스트 입력 창이 12개, 세부 정보가 2개로써 모두 148개 이다. 다시 말하면 약 32% 정도가 재사용 되었다. 따라서 테스트 케이스를 재사용함으로 인해 32% 정도의 테스트 케이스 작성 시간을 감소시킬 수 있다.

현재 연구의 진행 상태는 2차 평가를 앞두고 있다. 2차 평가가 끝난 후 테스트 케이스를 최종 수정할 것이다. 그리고 특정 기능에 대해 제안된 테스트 케이스와 기존 테스트 케이스를 사용하여 비교 시험하고 결과를 정량적으로 제시할 계획이다. 그리고 도출된 결과를 통해 제안된 테스트 케이스의 재사용율과 효율성을 입증할 것이다.

참고문헌

[1] IEEE Std 610. 12-1990, IEEE Standard Glossary of Software Engineering Terminology, 1990.

[2] C. Kaner, "What is a Good Test Cases?", Software Testing Analysis & Review Conference East, May 12-16, 2003.

[3] A. von Mayrhauser, R. Mraz, J. Walls, P. Ocken, "Domain Based Testing : Increasing Test Case Reuse", Computer Design:VLSI in Computer and Professors, ICCD 94. Proceeding, IEEE, OCT 10-12, 1994

[4] R. S. Arnold, "A Road Map Guide to Software Reengineering Technology", Software Reengineering, IEEE Computer Society, 1993

[5] W.T. Tsai, "Scenario-Based Test Case Generation for State-Based Embedded Systems", Performance, Computing, and Communication Conference, IEEE, April 2003