X-Internet 환경에서 X-Forms 기반 UI 소프트웨어의 효율적인 테스트 케이스 작성을 위한 연구

이승혁, 이창섭
㈜ 컴스퀘어
e-mail : opseung@comsquare.co.kr

A Study on Designing Test Case For X-Forms Base UI Software with X-Internet Environment

Seung-Hyuk Lee, Chang-Sub Lee
Corporation of Comsquare

요 약

X-Internet 과 X-Forms 기반의 UI 소프트웨어 시스템은 개발이 완료되어 운영되는 과정에도 요구사항의 변경, 성능 향상, 개발 과정 중의 오류 등의 다양한 이유로 변경될 수 있다. 기능 변경은 테스트 케이스 작성 시간과 비용의 증가로 이어진다. 본 논문은 소요되는 시간과 노력을 최소화하기 위한 테스트 케이스 작성 기법을 제안한다. 이를 위해 테스트할 기능을 분석하고 기존의 테스트 케이스에 새로운 기능을 접목하여 작성할 수 있는 테스트 케이스 재사용 알고리즘을 제안한다. 또한 오류가 전체 영역에 분포하지 않고 특정 영역에 분포하는 속성을 이용하여 모든 테스트 케이스를 생성, 실행하는 것이 아니라 오류를 잘 찾을 수 있는 테스트 케이스를 선정하여 테스트하는 기법의 효율성을 증명한다.

1. 서론

소프트웨어 테스트는 시험 대상 소프트웨어에 대한 요구를 분석하고 주어진 테스트 데이터에 따른 결과를 찾아내는 것이다. 이러한 테스트 작업은 소프트웨어에 직접 입력을 주어 실행시키는 데이트 케이스가 얼마나 오류를 잘 발견할 수 있는가 하는 것이 관건이다. 오류는 어떠한 소프트웨어라도 잠재적으로 가지고 있으며, 소프트웨어 비용에 큰 역할을 차지한다. 따라서 오류를 찾아내기 위한 테스트 기술의 발전이 필요하다. 미국표준기술연구소(NIST)의 보고서에 따르면 소프트웨어 오류를 50% 줄이면 117 억 달러를 절약할 수 있다고 한다[1]. 테스트 케이스를 어떻게 작성하느냐에 따라 소프트웨어 품질과 개발 효율성을 크게 좌우한다[2].

웹 기반 애플리케이션은 전자상거래와 사이버교육 등의 다양한 서비스를 제공한다. 그러나 사용자의 눈높이에 따른 요구사항이 많아지면서 풍부한 사용자 인터페이스를 웹으로 제공받길 원하고 있다. 이러한 시대의 흐름에 탄생한 것이 X-Internet 과 X-Forms 이다. 2000 년 IT 조사기관인 포레스터리서치가 처음 고안해 낸 개념으로 실행 가능한(eXcutable), 확장 가능한(eXtended) 인터넷이라는 의미로 실시간 양방향 통신, 유비쿼터스 컴퓨팅, 클라이언트 서버(CS) 환경에서 풍부한 사용자 인터페이스를 제공하는 장점이 있다. 이와 관련된 국내외 업체는 X-Internet 에서 사용자의 풍부한 요구사항을 제공할 수 있는 X-Forms 기반의 UI 개발 소프트웨어들을 속속 개발 및 출시하고 있다. X-Forms 기반의 UI 소프트웨어는 그 중요도에 비해 테스트에 대한 연구는 부족한 실정이다[3]. 이미 개발이 완료된 소프트웨어라도 사용자의 요구사항이 수시로 반영되어야 하기 때문에 그 기능 또한 수시로 추가될 수 있다. 따라서 테스트를 하기 위한 테스트 케이스 작성은 매우 복잡하고 많은 시간이 투자되는 작업이

며, 이러한 기술의 도입으로 테스트와 품질 관리가 점점 어려워지고 있는 실정이다. 또한 기능별 컴포넌트 분할이 쉽지 않고, 링크, HTML 호환 오류, 동적인 부분과 정적인 부분을 동시에 테스트 하는 기술이 부족하다. 테스트 비용은 크게 두 부분으로 구분하여 측정한다. 하나는 테스트를 수행하는데 필요한 시간이고 다른 하나는 테스트 케이스를 만들고 분석에 필요한 시간이다. 따라서 테스트 케이스의 복잡성과 그 수를 줄이는 것은 테스트에 소요되는 노력과 시간을 크게 줄일 수 있다[4].

이를 해결하고자 본 논문이 제안하는 내용은 다음과 같다. 첫째, X-Forms 를 분석하여 개발과정에서 생성된 테스트 케이스를 분류하고 정적인 부분과 동적인 부분을 테스트할 수 있는 알고리즘을 구성한다. 둘째, 비용과 효율성을 제공하는 테스트 케이스 작성 방법으로 과거 실행 및 오류 검출 정보를 활용하여 오류 발생이 많은 테스트 케이스를 선택하고 이를 이용하여 테스트하는 기법을 제안한다. 이렇게 제한된 방법이 테스트 케이스 작성 노력을 감소할 수 있다는 결과를 얻을 수 있다.

2. 본론

UI 소프트웨어는 개발이 완료되어 운영되는 과정에도 요구 사항의 변경, 성능 향상, 개발 과정 중의 오류 등에 다양한 이유로 일련의 변경 과정을 거친다. 프로그램의 변경은 변경된 기능만 영향을 주지 않고 다른 기능들도 영향을 미칠 수 있다는 점에서 신중하게 이루어진다. 변경된 기능에 의해 다른 기능이 영향을 받아 발생할 수 있는 오류를 찾기 위해 복잡하고 다양한 테스트 케이스를 작성해야 한다. 이로 인해 발생되는 시간과 비용은 X-Internet 기반 UI 소프트웨어 개발에 큰 걸림돌로 작용된다. 기존의 테스트 방법은 개발 과정에 작성된 전체 테스트 케이스를 모두 수행하는 것이다. 이 방법은 변경된 기능이 작을수록 비용면에서 매우 비효율적이다. 또한 이러한 테스트는 이미 운영중인 소프트웨어에 행해진다는 점에서 빠른 처리가 필수적이다[5]. 따라서 기능 변화에 따른 테스트 케이스 작성을 효율적으로 할 필요가 있다. 이를 위해 테스트할 기능을 분석하고 기존의 작성된 테스트 케이스에 새로운 기능을 접목하여 작성하므로 재사용성을 높여 소요되는 시간을 절감하는 방법을 제안한다. 일반적으로 오류는 전체 테스트 케이스에 골고루 분포되지 않고 특정 영역에 집중되는 Pareto 형상을 따른다. 따라서 모든 테스트 케이스를 실행하지 않고 과거 실행 및 오류 검출 정보를 활용하여 가장 오류를 잘 찾을 수 있는 테스트 케이스를 선정하고 실행한다. 선정된 테스트 케이스로 테스트한 결과 오류가 발생하였을 경우에는 가장 간단한 기능을 담고 있는 테스트 케이스를 생성하여 실행하므로 보다 빠른 오류 검출을 할 수 있다.

2.1 재사용을 위한 테스트 케이스 분류

테스트 기법에서는 테스트 케이스를 한번 사용하고 버리도록 교육받고 받아들여졌다[6]. 그러나 완전 새로운 요구 사항이 아닌 이상 테스트 케이스를 모두 재작성하는 것은 비효율적인 작업이다. 따라서 재사용성을 높이기 위해 블랙박스 테스트를 위한 테스트 케이스 생성 기법에 대해 논한다. UI 소프트웨어는 잘못된 구현이나 인터페이스, 다중 프로세스, 스레드, 인터럽트 등의 동적인 부분에 대한 에러가 많다. 따라서 단위 테스트 규모를 벗어난 시스템 수준의 테스트에서는 화이트 박스 테스트보다 블랙박스 테스트 기법을 주로 사용한다[7]. 따라서 본 논문에서 테스트 케이스 작성은 블랙박스 테스트 기법에 맞게 작성한다.

테스트 케이스의 재사용성을 위해 요구명세를 분할하고 여러 종류의 테스트 케이스 표현법과 테스트 케이스 단위를 고려하여 기능 위주로 기존 테스트 케이스를 분류한다. X-Internet 기반의 UI 소프트웨어는 기본적으로 X-Forms 스팩을 따른다. 따라서 X-Forms 을 기준으로 기능별 분류 작업을 한다. 분류된 기능에 따라 기존의 테스트 케이스 나눈 것이 <표 1>이다. 이렇게 나누어진 테스트 케이스는 새로운 기능이 추가될 때 테스트 케이스를 선택하는 기준이 된다. 선택된 테스트 케이스를 재사용하여 새로운 테스트 케이스를 생성하므로 전체 테스트 케이스를 생성하는 것보다 그 수를 현저히 감소할 수 있다.

<표 1> 재사용을 위한 테스트 케이스 분류

기능	분류된 테스트 케이스의 수
Bind	21
Body	32
Bool	27
Browser	15
Button	58
Calendar	8
Caption	11
Case	34
Checkbox	29
Combo	35
Datagrid	101
Group	21
Image	11
Import	14
Input	42
Instance	23
Iviewer	24
Listbox	21
Mutilinegrid	58
Output	19
Radio	15
Scrollbar	35
Switch	21
Textarea	26
Treeview	25
합계	726

2.2 재사용 알고리즘

X-Internet 기반의 UI 소프트웨어는 X-Forms 스팩을 따르기 때문에 사용자들이 이러한 소프트웨어를 이용하여 프로그래밍 할 때 HTML 을 기본 프로그래밍으로 사용한다. 또한 HTML 을 확장한 XHTML 이나 UI 데이터를 구성하는 XML 을 이용한다. X-Forms 스팩을 따르는 control 들을 정적으로 구현하기 위해 XHTML 을 사용하고 동적으로 구현하기 위해 JavaScript 나 VBScript 를 사용한다. 이러한 기반의 테스트를 위해 테스트 케이스를 생성하기 위한 알고리즘을 제안한다.

제안한 알고리즘은 새로운 기능의 추가에 대응하기 위하여 확장성을 최우선으로 고려하여 구성한다. <표 2>는 재사용을 높이기 위한 테스트 케이스 알고리즘이다. UI 에서 실질적인 데이터 역할을 하는 부분은 <xhtml:head>와 </xhtml:head>에 구현하므로 X-Forms 의 어떠한 컨트롤에서도 사용할 수 있다. 정적인 기능의 컨트롤 선언 시 컨트롤의 속성으로 선언하고 동적인 기능의 컨트롤 제어는 CDATA 형태로 컨트롤에 event 가 발생할 때 실행하도록 한다. 또한 동적인 제어를 하기 위해 X-Internet 에서 제어가 가능한 JavaScript 와 VBScript 를 사용한다. 재사용 알고리즘을 이용하면 X-Forms 컨트롤에 새로운 기능이 추가될 때 확장성이 용이하면서 테스트 케이스 생성 시간을 최소화 할 수 있다.

<표 2> 재사용 알고리즘

	<xhtml:html> <xhtml:head> <xhtml:title> 재사용 알고리즘 </xhtml:title>
XML 데이터 선언	<model> <instance> <root> <!--UI 에서 사용하는 데이터(XML)--> </root> </instance> </model>
	</xhtml:head> <xhtml:body>
정적 기능 구현	<controlname attribute="<!--정적 control 제어-->" Style="<!--정적 control 제어-->"

동적 기능 구현	<script type ="javascript"
	event="<!--동적 control 제어-->
	<![CDATA[
	// 동적 control 제어를 위한 javascript 구문
	]]>
	</script>
	<script type ="vbscript"
	event="<!--동적 control 제어-->
	<![CDATA[
	// 동적 control 제어를 위한 vbscript 구문
	]]>
	</script>

	</xhtml:body> </xhtml:html>

2.2 테스트 케이스의 선택

2.1 에서 기능별로 분류된 테스트 케이스를 이용하여 새로운 요구 사항이 발생하였을 때 기능에 따른 테스트 케이스만 테스트 하는 방법을 제시하였다. 하지만 기능별로 분류하였다고 하더라도 하나의 기능에 따른 테스트 케이스의 수는 수십 가지이며 이를 모두 테스트 하는 것은 비효율적이다. 따라서 많은 테스트 케이스 중에서 과거 실행 정보와 오류 검출 정보를 이용하여 몇 가지 테스트 케이스만 선정하여 테스트하는 방법을 제안한다. 일반적으로 UI 소프트웨어는 정적인 부분과 동적인 부분으로 분류할 수 있다. 정적인 부분은 control 의 모양에 해당하고 동적인 부분은 동작에 해당한다. 이러한 control 을 테스트하기 위해 테스트 케이스의 수는 <표 3>과 같다.

<표 3> control 을 테스트하기 위한 테스트 케이스의 수

```
테스트 케이스의 수 : C
정적인 기능(속성) : A
정적인 기능(스타일) : S
동적인 기능 : E
C = (A1· A2· A3 …) + (S1· S2· S3 …) + (E1· E2· E3 …)
```

정적인 기능과 동적인 기능이 추가될 때 마다 테스트 케이스의 수는 급격하게 증가한다. 이를 해결하기 위해 단위 기능에 대한 테스트 케이스를 제외하고 모든 기능이 복합적으로 적용된 테스트 케이스만을 선정한다. 선정된 테스트 케이스에 새로 추가되는 기능을 넣어 테스트한다. 그 결과 에러가 발생하지 않으면 pass 처리하고 테스트를 종료한다. 하지만 테스트 결과 에러가 발생하면 새로 추가된 기능만 재사용 알고리즘을 이용하여 테스트 케이스를 생성하여 확인하여 에러 발생의 원인을 찾는다. 이러한 방법은 테스터들에게 모든 테스트 케이스에 새로운 기능을 적용하여 테스트할 때와 다른 불안감을 심어줄 수 있다. 하지만 <표 4>에서 보듯 하나의 기능이 추가되었을 때 전체 테스트 케이스를 테스트할 때와 본 논문에서 제시한 모든 기능이 집중된 테스트 케이스만을 테스트 하는 기법이 같은 개수의 에러를 발견하였음을 알 수 있다. <표 4>는 7 개의 기능이 추가되었을 때 개발과정중에 생성한 모든 테스트 케이스로 테스트 한 결과와 재사용 알고리즘에 적용하고 과거 실행 정보와 오류 검출 정보를 이용하여 선택된 테스트 케이스로 테스트 한 결과를 보여주고 있다. 두 테스트 기법 모두 테스트의 4 개의 에러를 찾을 수 있음을 보여주고 있다.

<표 4> control 을 테스트하기 위한 테스트 케이스의 수

	Every TestCase Tesing		Selected TestCase Testing	
	TestCase number	Error discovery	TestCase number	Error discovery
기능 1	27	1	2	1
기능 2	38	0	3	0
기능 3	29	0	4	0
기능 4	15	0	3	0
기능 5	58	1	4	1
기능 6	12	0	2	0
기능 7	46	2	6	2
합계	225	4	24	4

3. 결론

웹 기반의 애플리케이션은 다양한 서비스를 제공한다. 하지만 사용자의 요구사항이 날로 높아짐에 따라 X-Internet 기반의 X-Forms 가 탄생하게 되었다. 이러한 시대의 흐름에 많은 업체들은 X-Forms 스팩을 준수하는 UI 개발 소프트웨어를 앞 다투어 출시하고 있다. 본 논문은 UI 소프트웨어의 효율적인 테스트 기법으

로 테스트 케이스를 선정하고 선정된 테스트 케이스만 테스트하므로 오류 발견의 노력을 줄 일 수 있는 기법을 제안한다. X-Forms 를 분석과 동시에 개발 과정에서 생성된 테스트 케이스를 분석하여 정적인 기능과 동적인 기능을 동시에 테스트 할 수 있는 테스트 케이스 알고리즘을 제시한다. 많은 테스트 케이스 중에서 재사용 알고리즘을 사용하고 과거 실행 정보와 오류 검출 정보를 이용하여 기능이 복합적으로 적용된 테스트 케이스를 선정한다. 선정된 테스트 케이스에 새로운 기능을 추가하여 오류가 없으면 pass 처리하고 테스트를 종료한다. 오류가 발생하면 재사용 알고리즘을 이용하여 단위 테스트 케이스를 생성하여 오류 발생 원인을 찾는다. 이 기법은 기존의 모든 테스트 케이스를 생성, 실행하는 것보다 적은 시간과 비용을 제공하면서 테스트 케이스의 가장 큰 목적인 오류 발견 건수를 동일한 수준으로 유지할 수 있음을 증명하였다.

향후 UI 소프트웨어뿐만 아니라 X-Internet 전반에 걸친 소프트웨어에 대한 분석과 테스트 케이스 작성 기법에 대한 연구가 지속되어야 한다.

참고문헌

[1] Tassey, G., “The Economic Impacts of Inadequate Infrastructure for Software Testing: Final Report”, National Institute of Standards and Technology, 2002.

[2] 서광익, 최은만, “블랙박스 테스트 케이스의 리엔지니어링”, 정보처리학회논문지 D 제 13-D 권 4 호, 2006.

[3] 한국 소프트웨어진흥원, http://www.software.or.kr

[4] 박은영, “테스트 케이스 작성 방법”, Sten Journal, 2003.

[5] 소선섭, 채의근, “회귀 테스팅의 테스트 케이스 우선 순위화 기법의 실험적 연구”, 정보처리학회논문지 D 제 12-D 권 제 2 호, 2005.

[6] 한국정보통신기술협회, “소프트웨어 테스트 전문기술”, 2006.

[7] H. Gomma, “Designing Software Product Lines with UML”, Addison Wesley, 2004.