안드로이드 애플리케이션 GUI 테스팅 도구 적용 및 사례연구

김태균, 권기현
경기대학교 컴퓨터과학과
e-mail : {bigvirus1809, khkwon} @kyonggi.ac.kr

The Android application GUI testing tool apply and case study

Taekyun Kim, Gihyun Kwon
Dept. of Computer Science, Kyonggi University

요 약

GUI 는 안정성과 견고성 그리고 사용성을 검증하기 위하여 반드시 테스트가 되어야 한다. 하지만 이러한 GUI 테스팅은 기존 소프트웨어 테스트 과정보다 복잡하고 많은 양의 테스트 작업을 요구하게 된다. 그리하여 많은 자동화된 테스트 도구가 개발 되어왔으며, 많은 산업체에서는 이러한 자동화된 테스트 도구를 사용하여 소프트웨어의 품질을 측정하고 있다. 하지만 모바일 소프트웨어의 GUI 를 테스트하기 위한 도구는 현재까지 개발 중이며 많은 연구가 선행 되고 있다. 본 논문은 이러한 모바일 애플리케이션의 GUI 를 테스트 하기 위하여 많이 사용되고 있는 오픈 소스 기반의 도구 3 개를 선정하여 실제 애플리케이션에 적용하여 보고 분석을 통하여 향후 테스트 도구 개발에 대한 방향을 제시 하고자 한다.

1. 서론

소프트웨어 테스팅은 프로그램의 구현 과정에서 만들어지는 소프트웨어의 에러를 고치기 위해 소프트웨어의 오류를 밝히고 소프트웨어의 품질을 보증하기 위한 것이다. 소프트웨어 생명 주기에서 테스팅은 프로젝트의 전체 시간의 약 50%를 차지 할 수 있다. 그 중 Graph User Interface(GUI)는 사용자가 소프트웨어를 보다 쉽게 작동할 수 있게 하며, 사용자와 소프트웨어 간 상호작용에 있어 널리 사용되는 인터페이스 중 하나이다. GUI 는 평균 애플리케이션 코드의 약 48%, 구현에 소요되는 시간의 약 50%를 차지한다고 추정할 수 있다. 이러한 GUI 는 안정성과, 견고성, 그리고 사용성을 검증하기 위하여 반드시 테스트가 되어야 한다.[1,4] 하지만 GUI 테스트는 단일 테스트와 GUI 작동에 의한 소프트웨어의 내부 동작 테스트가 병행되어야 하기 때문에 기존의 테스트 과정 보다 복잡하고 많은 양의 테스트 작업을 요구하게 된다.[3] GUI 를 테스트하는 것은 정적 또는 동적 분석을 통해 수행 될 수 있다. 정적 분석은 모델 체킹과 정형 검증과 같이 코드 리뷰와 formal analysis 에 의해서 이다. 동적 분석은 Application Under Test(AUT)를 실행하는 것으로 진행된다. 명세서가 정형 적일 경우 테스트 케이스의 구성과 실행은 자동화 될 수 있다.[1] 모바일 애플리케이션은 휴대폰이나 단말기에서 사용하기 위해 구현된 응용프로그램을 의미한다. 하지만 모바일 디바이스의 디스플레이는 일반 컴퓨터에 비해 매우 작기 대문에 가시성 및 가독성이 떨어지고 표현할 수 있는 정보의 양이 제한되는 문제가 있지만 사용자는 GUI 를 통해 애플리케이션과 커뮤니케이션을 하기 때문에 사용자에게 가장 많은 영향을 미치는 부분이다. 이러한 GUI 는 사용자가 직접 경험하는 품질 척도이기 때문에 중요한 요구 사항이 된다.[1, 5]

본 논문에서는 모바일 GUI 테스팅을 위하여 여러 모바일 플랫폼 중 구글에서 공개한 안드로이드 플랫폼으로 제한한다. 논문에서 사용된 애플리케이션은 하이브리드 방식으로 구현된 스마트 캠퍼스 애플리케이션이며, 자동화된 테스트를 하기 위하여 여러 오픈 소스 기반의 도구들 중 3 가지 도구를 선택하여 적용한다. 또한 테스트 결과를 통하여 기존 도구의 문제점을 분석 향후 개발될 도구 방향에 대하여 이야기하고자 한다. 논문의 구성은 2 장에서는 스마트 캠퍼스 애플리케이션과 오픈 소스 기반의 테스트 도구인 Robottium, MonkeyTooks, AndroidGUITAR 에 대하여 살펴보고, 3 장에서는 스마트 캠퍼스 애플리케이션을 통하여 각 도구들의 적용 결과를 이야기 하고자 한다. 4 장에서는 2 장에서 소개한 도구들의 분석 기준과 분

□ 본 연구는 경기도의 경기도지역협력연구센터사업의 일환으로 수행하였음.[2011-0215,모바일 플랫폼 기반의 콘텐츠 응용 소프트웨어 기술 개발]

석 결과에 대하여 이야기하며, 마지막 5 장에서는 결론 및 향후 연구를 기술한다.

2. 적용 애플리케이션 및 테스팅 도구

본 장에서는 테스트에 사용된 애플리케이션에 대한 소개와 적용한 테스팅 도구에 대하여 간략히 소개한다

2.1. 적용 애플리케이션

본 논문에서 적용된 애플리케이션은 총 34 개의 화면과 이벤트를 발생 시킬 수 있는 버튼, 메뉴 등은 46 개를 가지고 있다. 또한 하이브리드 애플리케이션 개발 방법을 따르고 있어 웹 애플리케이션과 네이티브 애플리케이션의 특징을 모두 가지고 있는 애플리케이션이다. 아래 그림 1 은 본 논문에서 사용된 애플리케이션의 화면을 보이고 있다.

a) 메인 화면 b)하위 화면

(그림 1) 애플리케이션 화면

아래 표 1 는 적용된 애플리케이션의 주요 구성 요소를 나타내고 있다.

<표 1> 애플리케이션 구성 요소

구성 요소	비고
액티비티 수	34
네이티브 액티비티	15
웹 액티비티	19
이벤트 컴포넌트	46
액티비티 간 전이 횟수	93

2.2. Robotium

Robotium 은 2010 년 1 월 버전 1.0 을 공개 하였으며, 현재 버전 3.1 을 제공하고 있다. Robotium 은 안드로이드 애플리케이션을 위해 black-box 테스트 케이스를 작성 할 수 있도록 만들어진 테스트 프레임워크이다. 개발자는 안드로이드 activity 의 기능에 대하여 시스템의 테스트 시나리오를 작성하여 사용할 수 있다. 그리고 액티비티, 다이얼로그, 토스트, 메뉴 와 컨텍스트 메뉴 등에 대하여 지원을 하고 있으며, 다음과 같은 유용성을 가지고 있다.[8]

- 테스트 애플리케이션에 대한 최소한의 지식을 가지고 강력한 테스트 케이스를 만들 수 있다.
- 여러 액티비티에 대하여 자동으로 실행하여 처리한다.
- 테스트 케이스의 가독성은 기존 instrumentation tests 에 비해 향상되었다.
- 매우 빠르게 테스트 케이스를 실행 할 수 있다.
- 테스트를 실행하기 위해 Maven 또는 Ant 로 통합될 수 있다.

2.3. MonkeyTools

Monkey 는 안드로이드 SDK 에서 제공하는 프로그램으로, 에뮬레이터 또는 디바이스에 대해 랜덤 한 이벤트 스트림을 발생 시킨다. 이것은 마친 원숭이가 기기를 작동하는 것처럼 랜덤 한 이벤트를 발생한다. 예를 들어 인간은 OK 버튼을 포함한 pop-up 메뉴가 나오면 OK 버튼을 클릭하지만, Monkey 는 이를 인지하지 못하고 아무 버튼 혹은 아무 곳이나 터치하게 된다. 스마트폰을 사용함에 있어, 인간은 선행 학습에 의해 디바이스와 상호작용을 하기 때문에 극단적인 사용이나 edge case 들에 대해 에러가 발생하는지를 확인하기 어렵다. Monkey 의 경우 랜덤 이벤트 스트림을 발생하기 때문에 예측 불허인 극단적인 상황도 테스트가 가능하다. 또한 프로그램 자체에서 이벤트의 발생 빈도, 이벤트의 발생 주기, 이벤트의 발생 종류들을 조절 할 수 있기 때문에 자원을 낭비하지 않고도 극단적인 상황을 쉽게 테스트 할 수 있으며, Monkey 가 실행 될 때 이벤트를 생성하고, 생성된 이벤트 정보를 시스템으로 전달 할 수 있다. Monkey 테스트는 보통 애플리케이션을 배포하기 전 edge case 들에 대한 저항성을 기르기 위해 사용된다.[7]

2.4. AndroidGUITAR

AndroidGUITAR 는 EFG(Event Flow Graph)기반 GUI 테스팅 도구인 GUITAR(Graphical User Interface Test Framework)를 확장하여 안드로이드 플랫폼의 GUI 의 테스팅을 단순화하기 위한 도구이다. EFG 는 GUI 의 모든 실행 가능한 이벤트 시퀀스를 모델링 한 그래프이다. AndroidGUITAR 는 Ripper 와 EFG Generation(EG), TestCase Generation(TG), Replay 등 총 4 개의 도구로 구성이 되어있다. [9]

- Ripper: 역 공학을 통하여 GUI 구성하고 있는 요소들을 분석하여 GUI 구조를 분석.
- EG : Ripper 도구를 통하여 분석된 GUI 의 구조를 이용하여 EFG 를 생성
- TG : EG 도구로부터 생성된 EFG 를 이용하여 TestCase 생성
- Replay: TG 도구로부터 생성된 테스트 케이스를 이용하여 실제 애플리케이션을 실행.

GUITAR 는 다음과 같은 유용성을 가지고 있다.

- 역 공학을 이용하여 자동으로 GUI 구조를 분석
- 모델 기반 테스트를 위한 EFG 를 분석하고 자동으로 구성
- 테스트 케이스 알고리즘을 이용하여 다양한 테스트 케이스를 자동으로 생성

3. 애플리케이션 적용 결과

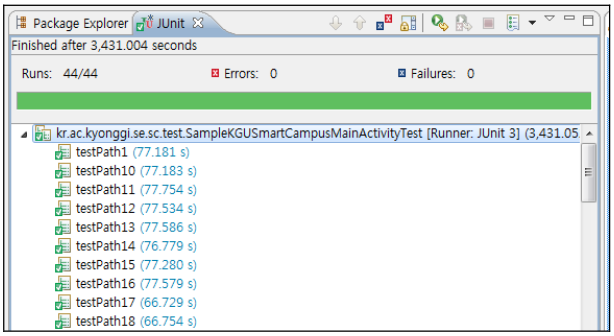
3.1. 애플리케이션 적용 결과

본 절에서는 2 절에서 소개한 3 가지 도구를 사용하여 스마트 캠퍼스 애플리케이션에 적용 하였다. 스마트 캠퍼스 애플리케이션을 테스트 하기 위하여 요구 사항으로부터 표 2 와 같이 테스트 검사 리스트를 작성하였다.

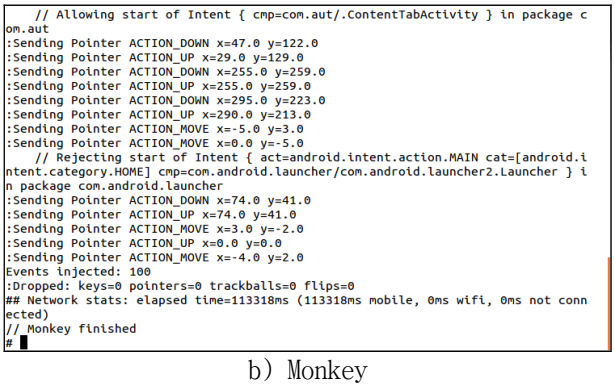
<표 2> 애플리케이션 테스트 검사 리스트

	기능명	입력 값	확인 사항
1	경기 소개	메인 화면에서 경기소개 아이콘을 클릭	• 총장 인사말 컨텐츠 확인
2	교육 이념	경기 소개 화면에서 교육이념 탭 버튼 클릭	• 교육 이념 컨텐츠 확인
…			
94	경기 발자취	경기 소개 화면에서 경기 발자취 탭 버튼 클릭	• 경기 발자취 컨텐츠 확인 • 검색 기능 확인

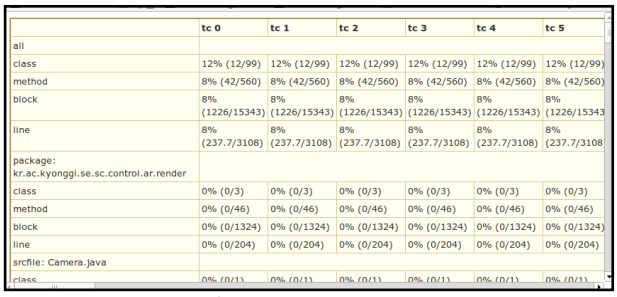
테스트 검사 리스트를 기반으로 Robottium 에 대해서만 테스트 케이스를 작성할 수 있었으며, 아래 그림 2 는 각 도구들에 대한 테스트 결과 화면이다.



a) Robottium



b) Monkey



	tc 0	tc 1	tc 2	tc 3	tc 4	tc 5
all						
class	12% (12/99)	12% (12/99)	12% (12/99)	12% (12/99)	12% (12/99)	12% (12/99)
method	8% (42/560)	8% (42/560)	8% (42/560)	8% (42/560)	8% (42/560)	8% (42/560)
block	8% (1226/15343)	8% (1226/15343)	8% (1226/15343)	8% (1226/15343)	8% (1226/15343)	8% (1226/15343
line	8% (237.7/3108)	8% (237.7/3108)	8% (237.7/3108)	8% (237.7/3108)	8% (237.7/3108)	8% (237.7/3108
package: kr.ac.kyonggi.se.sc.control.ar.render						
class	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)	0% (0/3)
method	0% (0/46)	0% (0/46)	0% (0/46)	0% (0/46)	0% (0/46)	0% (0/46)
block	0% (0/1324)	0% (0/1324)	0% (0/1324)	0% (0/1324)	0% (0/1324)	0% (0/1324)
line	0% (0/204)	0% (0/204)	0% (0/204)	0% (0/204)	0% (0/204)	0% (0/204)
srcfile: Camera.java						
class	0% (0/1)	0% (0/1)	0% (0/1)	0% (0/1)	0% (0/1)	0% (0/1)

c) AndroidGUITAR

(그림 2) 애플리케이션 화면

Robottium 의 경우 테스트 케이스가 많을 경우 테스트 도중 멈추는 현상이 발생하여 테스트 케이스를 분리하여 적용 하였다. 테스트 결과로 총 5 개의 오류를 발견 할 수 있었으며 발견된 오류를 분석 결과 다음 표 3 과 같은 결과를 구할 수 있었다.

<표 3> 발견된 오류 리스트

	오류명	오류 설명
1	DB 접근	데이터 베이스 접근 오류
2	화면	잘못된 화면으로 상태 전이
3	컴포넌트	화면상 컴포넌트 차이
4	네트워크 접근	네트워크 응답 시간 초과
5	성능 오류	화면 응답 시간

4. 도구 분석 기준 및 분석 결과

본 절에서는 2 장에서 소개한 도구를 분석하기 위한 기준과 분석 결과에 대하여 설명한다. 또한 분석 결과에 따른 해결 방안에 대하여 이야기 하고자 한다.

4.1. 도구 분석 기준

도구를 분석하기에 앞서 6 가지의 기준을 가지고 도구를 분석 하였다. 도구 분석 기준은 요구사항 반영 여부, 테스트 케이스 생성시 기준이 되는 커버리지, 요구 사항에 따른 테스트 케이스 생성, 테스트 자동 실행, 테스트 결과에 따른 리포트 기능과 마지막으로 사용환경을 기준으로 분석 하였다. 아래 표 4 은 각 기준에 대한 간략한 설명이다.

<표 4> 도구 분석 기준

	기준	기준 설명
1	요구사항 반영	개발 초기 과정 요구사항 명세서 및 테스트 케이스 명세서의 요구사항에 대한 반영 여부
2	커버리지	테스트 케이스 생성시 적용되는 커버리지 종류
3	테스트 케이스 생성	자동으로 테스트 케이스 생성 여부
4	테스트 실행	생성된 테스트 케이스를 자동으로 실행 여부
6	리포트	테스트 결과에 대한 리포트 기능
7	사용환경	테스트 환경

위의 표 3 과 같은 기준을 통하여 2 장에서 소개한 도구들에 대하여 분석을 하였다.

4.2. 도구 분석 결과

스마트 캠퍼스 애플리케이션을 테스트 하기 위한 사용한 도구 3 가지에 대한 도구 분석 결과는 다음 아래 표 4 와 같다.

<표 5> 도구 분석 결과

	Robotium	Monkey	Android GUITAR
요구사항	가능	불가능	불가능
커버리지	가능	불가능	불가능
테스트 케이스 생성	불가능	가능	가능
테스트 실행	가능	가능	가능
리포트	가능	가능	가능
사용환경	Eclipse	Command Line	Command Line

위의 표 5 를 보듯이 Monkey 와 AndroidGUITAR 는 고객의 요구사항에 대하여 반영이 불가능 하였다. Monkey 의 경우 테스트를 하기 위하여 이벤트를 랜덤하게 발생 시키면서, 시스템의 멈춤 현상 혹은 시스템의 에러를 발견하게 된다. AndroidGUITAR 는 시스템의 GUI 구조를 분석하여 EFG 를 생성하여 EFG 로 부터 노드 pair 인 테스트 케이스를 자동 생성하여 테스트를 실행하기 때문에 시스템의 요구사항을 충분히 반영하지를 못한다. Robotium 은 테스트 케이스 생성 시 사용자가 직접 관여 하기 테스트 케이스를 생성하기 때문에 시스템의 요구사항을 충분이 반영하여 테스트 케이스를 생성 할 수 있다. 하지만 Robotium 은 테스트 케이스를 자동적으로 생성하지 못하므로 테스트 케이스 생성에 많은 노력이 필요로 한다.
3 가지 모든 도구에서 리포트 기능은 보유하고 있었지만 지원하는 부분은 매우 비약 하였다. 그 중 AndroidGUITAR 는 안드로이드 SDK 에 포함 되어 있는 EMMA 라이브러리를 사용하여 HTML 로 테스트의 결과를 보여 줌으로써 사용자에게 보다 편한 테스트 결과 리포트를 제 공 하고 있었다. 하지만 규모가 큰 시스템일 경우 GUI 구성 요소를 가지고 오는 과정에서 모든 GUI 구성 요소를 가지고 오지 못하는 경우가 발생하였다.
본 논문에서 사용된 테스트 도구들 대부분은 요구사항을 반영하지는 못하였다. 반영이 가능한 Robotium 역시 테스트 케이스 생성에 많은 노력이 필요하게 되었다.

5. 결론 및 향후 연구

본 논문에서는 모바일 애플리케이션 테스팅 도구를 개발하기 위하여 안드로이드 애플리케이션에 대해 GUI 테스팅을 하였다. GUI 테스팅에 사용된 도구는 오픈 소스 기반의 테스팅 도구 3 개를 적용 하여 보았으며, 2 개의 도구에서는 고객의 요구사항 및 시스템의 요구사항을 반영하기 어려운 문제점을 가지고 있었으면, 요구사항을 반영할 수 있는 도구 역시 테스트 케이스를 생성하는데 많은 노력이 필요하였다. 이러한 문제점을 해결하기 위하여 GUI 전이 관계에 해당하는 요구사항을 LTL 로 표기하고 Synthesis 를 통하여 요구사항에 부합하는 오토마타를 생성하는 LTL Synthesis 기법이 필요하다. 이러한 LTL Synthesis 로부터 생성된 오토마타는 요구사항이 올바르다면 Synthesis 로부터 만들어진 모델은 항상 시스템의 행위를 만족한다고 볼 수 있다. LTL Synthesis 를 통하여 요구사항에 부합하는 애플리케이션의 모델을 생성하고 이러한 모델을 이용하여 테스트 케이스를 생성, 테스트 케이스를 자동으로 실행 하여 주는 도구가 필요하다.
향후 연구로는 GUI 전이 관계에 대하여 LTL Synthesis 를 통하여 요구사항에 부합하는 모델을 생성하는 것에 대한 연구를 진행 할 것이며, 또한 이렇게 생성된 모델을 이용하여 테스트 케이스를 자동으로 생성하는 알고리즘과 모바일 애플리케이션에 특화된 테스팅 도구를 개발 할 것이다.

참고문헌

[1] Xuebing Yang. “Graphic User Interface Modeling and Testing Automation”
[2] Paul Amman, Jeff Offutt. “Introduction to software testing”
[3] 문중희, 이남용. “소스코드기반의 GUI 테스트 자동화 기법의 구현” 정보과학회논문지: 소프트웨어 및 응용 제 36 권 제 9 호 2009
[4] Antti Jaaskelainen, Mika Katara, Antti Kervinen, & Mika Maunumaa. “Automatic GUI Test Generation for Smartphone Applications an Evaluation”, ICSE’09, May 16-24, 2009, Vancouver, Canada
[5] Cuixiong Hu, Iulian Neamtiu. Automating GUI Testing for Android Applications. AST ’11, May 23-24, 2011, Waikiki, Honolulu, HI, USA
[6] Martin Kropp, Pamela Morales. “Automated GUI Testing on the Android Platform”
[7] “http://developer.android.com/guide/developing/tools/monkey.html”
[8] “http://code.google.com/p/robotium/”
[9] “http://sourceforge.net/apps/mediawiki/guitar/index.php?title=GUITAR_Home_Page”